Recognizing Ethnic Groups from Speech Using Machine Learning

Ro'a Nafi

Department of Electrical and Computer Engineering, Birzeit University, Palestine

1201959@student.birzeit.edu

Abstract

This project focuses on creating a simple system to recognize the ethnic group of British speakers in Birmingham as either 'Asian' or 'White' based on their speech. Using common speech processing features such as Energy, Zero-Crossing Rate, Pitch Frequency, and Mel-Frequency Cepstrum Coefficients (MFCCs) with deltas and delta-delta, we experimented with various machine learning models to achieve the best accuracy.

Our approach involved preprocessing the data, extracting features, and testing different models and parameters. The best results were achieved using the Gaussian Mixture Model (GMM) with 6 components and spherical covariance type, yielding an accuracy of 77.5%. The k-Nearest Neighbors (KNN) model, with k=1, weighted by uniform, and the Euclidean metric, achieved an accuracy of 62.5%. GMM demonstrated stronger performance overall, particularly in classifying Class 0, although it struggled with misclassifications in Class 1. Despite limitations, the project demonstrates the potential of combining basic features and machine learning techniques for ethnic group recognition. Future work aims to improve accuracy through larger datasets and advanced methods.

1. Introduction

Speech recognition and classification play an important role in modern technology, from voice assistants to understanding linguistic diversity. This project focuses on identifying the ethnic group of British speakers in Birmingham as either 'Asian' or 'White' based on their speech. By extracting key features like Energy, Zero-Crossing Rate, Pitch Frequency, and Mel-Frequency Cepstrum Coefficients (MFCCs), we apply machine learning models such as k-Nearest Neighbors (KNN), and Gaussian Mixture Models (GMM) to classify the speakers.

The "Voices across Birmingham" dataset provides real-world conversational audio for this task. The project involves processing audio data, extracting relevant features, and fine-tuning the models to achieve reliable results. While the task presents challenges, this work showcases how speech and machine learning techniques can be combined to address meaningful classification problems, paving the way for future improvements.

2. Background

2.1. Mel-Frequency Cepstral Coefficients (MFCC)

MFCC (Mel-Frequency Cepstral Coefficients) is a key feature extraction method in speech and audio processing. It transforms the audio signal into the frequency domain using the Discrete Fourier Transform (DFT), applies the mel-scale to mimic human hearing, and computes cepstral coefficients. MFCCs highlight important speech features, making them ideal for tasks like speech recognition, speaker identification, and emotion detection [1].

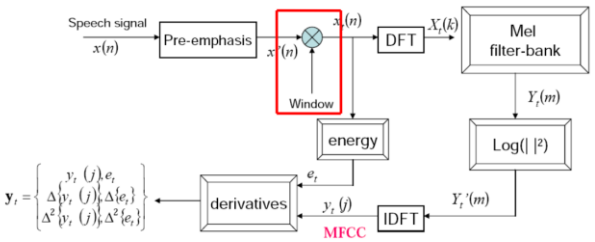

Figure 1: Roadmap of MFCC extraction [2]

The process of calculating MFCCs involves several steps to transform the raw audio signal into meaningful features:

- **Pre-Emphasis:** Pre-emphasis boosts higher frequencies in audio, which are often weaker, to improve phoneme recognition by making the signal more balanced for the model**.**
- **Windowing:** MFCC extracts features from audio for phoneme detection. Audio signals are divided into shorter segments for easier analysis. This segmentation allows for more accurate identification of individual phonemes within the speech.
- **Discrete Fourier Transform (DFT):** The signal is converted from the time domain to the frequency domain by applying the DFT transform, which simplifies analysis and reveals the frequency components of the signal and their intensities.
- **Mel Filter Bank:** Humans perceive sound differently than machines, with higher sensitivity to lower frequencies. The Mel scale mimics human hearing by mapping frequencies accordingly. By incorporating the Mel scale, speech recognition models can better represent audio and improve their accuracy.
- **Logarithm:** The output from the filters is transformed using a logarithmic scale to align with the way humans perceive sound intensity, with

higher gradients at lower input values and lower gradients at higher input values**.**

- **Discrete Cosine Transform (DCT):** The logarithmic mel-scaled coefficients are then decorrelated using the DCT, producing a set of compact coefficients. These coefficients, typically the first 12–13, represent the spectral envelope of the signal and are the core MFCC features.

First derivatives (deltas) and second derivatives (delta-deltas) of the MFCCs are computed to capture the temporal changes in the audio. Deltas measure the rate of change of the MFCCs over time, while delta-deltas represent the acceleration of these changes. Together, they provide richer temporal information, which is particularly useful for tasks that involve dynamic speech characteristics.

2.2. Other Features

The system has advanced features to support accurate audio signal analysis. Energy measures the power of the signal, allowing precise detection of significant changes. Zero-Crossing Rate (ZCR) calculates the rate at which the waveform changes sign, giving insights into signal dynamics. Pitch identifies the fundamental frequency, helping detailed analysis of tonal characteristics. These features together improve the system's performance, matching the project's goal of providing strong and efficient audio processing capabilities.

2.3. k-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is an easy supervised machine learning method. It is used for classification and regression tasks. It predicts the output for a new data point based on the closest training points in the feature space. Unlike many algorithms, KNN is a "lazy learner". This means it does not need a training phase, instead it stores all training data and makes predictions by calculating distances, like Euclidean or Manhattan distances. For classification, KNN uses majority voting among neighbors. For regression, it averages their values [3].

The performance of KNN depends on important factors like choosing the value of k. A small kk can cause overfitting by being too sensitive to noise, while a large k might cause underfitting by smoothing out local patterns. It is important to properly scale features to make sure no single feature dominates the distance calculations**.**

2.4. Gaussian Mixture Model (GMM)

A Gaussian Mixture Model (GMM) represents a dataset as a combination of multiple Gaussian distributions, each defined by its mean and covariance. This structure allows GMMs to model complex, multimodal data distributions effectively. Key components of a GMM include the means (centers of the distributions), covariances (which define the shape and orientation), and mixture weights (indicating the proportion of each Gaussian component in the overall model). GMMs are flexible for modeling different data distributions and provide probabilistic clustering. However, they can be sensitive to initial conditions and computationally expensive for high-dimensional data [4][5].

The Expectation-Maximization (EM) algorithm estimates parameters in GMMs by alternating between assigning data points to components (Expectation step) and updating parameters to maximize likelihood (Maximization step). While effective, it can be computationally intensive and sensitive to initialization.

2.5. Grid Search

Grid Search is a method used to find the best settings for machine learning models like k-Nearest Neighbors (KNN). It systematically tests different combinations of parameters to see which one works best for the model. For example, in KNN, it could test multiple values of k (the number of neighbors) and distance metrics. By evaluating each combination, Grid Search helps in selecting the optimal parameters that improve the model's performance [6].

3. Methodology

3.1. Data preprocessing

The audio data was read from the corresponding sub-directories. Silence was removed from the audio signals using the Librosa Python library with a specified decibel threshold (top_db=30), ensuring that only meaningful speech segments were retained. Additionally, Librosa, PyDub, and NumPy were utilized for feature extraction and in preparing the data for machine learning methods**.** Also, we standardize feature after extraction them**.**

3.2. Extracting Feature

Feature extraction involves transforming raw audio data into a set of measurable characteristics that can be used to train models. In this project, we focused on four primary features:

3.2.1. Energy

Energy measures the loudness of the audio signal. It is calculated as the mean of the squared amplitudes of the signal, providing a representation of its overall power. The energy EEE is defined as:

$$E = \frac{1}{N} \sum_{n=1}^{N} x[n]^2 \quad [7]$$

where x[n] is the amplitude of the signal at sample n, and NNN is the total number of samples.

3.2.2. Zero-crossing rate

The Zero-Crossing Rate (ZCR) calculates the rate at which the signal changes its sign (from positive to negative or vice versa). It is useful for identifying noisiness in the signal. The ZCR is defined as:

$$ZCR = \frac{1}{N-1} \sum_{n=1}^{N-1} 0.5 \, |sign(x[n]) - sign(x[n-1])| \, [2]$$

3.2.3. Pitch frequency

The pitch frequency is calculated as the average of the strongest frequencies in the spectrum. The pitch period, which is the inverse of the fundamental frequency, represents the duration of one vocal cord vibration cycle and is relevant only

for voiced speech frames. Estimation techniques include time-domain methods, like autocorrelation, and frequency-domain methods. This project focused on frequency-domain estimation using the librosa.piptrack method for accurate pitch feature extraction.

3.2.4. MFCCs with its delta and delta-delta

We used the librosa library in order to calculate the MFC.
Here are a couple of equations that are used to describe the MFCC extraction process which is depicted in Figure 1:

- Pre-emphasis:

$$y[n] = x[n] - \alpha x[n-1] \quad [2]$$

- Framing and Windowing

$$w(n)=x(n)\cdot w(n) \quad [2]$$

- Fast Fourier Transform (FFT)

$$|x(k)|^2 = \sum_{n=0}^{N-1} |x_w(n) e^{-\frac{j2kn\pi}{N}}|^2 \quad [2]$$

- Mel-filterbank:

$$H_m(f) = \begin{cases} 0 & if\ f < f_{m-1} \\ \frac{f - f_{m-1}}{f_m - f_{m-1}} & if\ f_{m-1} \le f < f_m \\ \frac{f_{m+1} - f}{f_{m+1} - f_m} & if\ f_m \le f < f_{m+1} \\ 0 & if\ f \ge f_{m-1} \end{cases} \quad [2]$$

- Discrete Cosine Transform (DCT):

$$MFCC(n) = \sum_{k=0}^{K-1} log(x(k) cos(\frac{\pi n(k+0.5)}{K})) \quad [2]$$

3.3. KNN Model

The features were fed to the KNN model, and the parameters for the model were automatically chosen using the grid search technique:

```
Best Parameters: k = 1, weights = uniform, metric = euclidean
choose KNN best Parameters Done
```

Figure 2:best parameters for KNN models

Now the model was ready to evaluate in test set.

3.4. GMM Model

We followed the same steps for this model too and were able to determine the best parameters of the GMM:

```
Best Parameters: n_components: 6, covariance_type: spherical
choose GMM best Parameters Done
```

Figure 3:best parameters for GMM models

4. Experiments and Results

4.1. Dataset

The dataset has speech recordings from two ethnic groups: Asian and White, split into training and testing sets. Each audio sample was used to get features like MFCCs, energy, and pitch. These features were then put into classification models like KNN and GMM. This dataset allowed the analysis and classification of speech patterns for each group.

4.2. Training Process

In our system, we extract 39 features from audio signals, including energy, zero-crossing rate (ZCR), pitch, and Mel-Frequency Cepstral Coefficients (MFCCs). We extract 12 coefficients from the MFCCs, along with their delta and delta-delta features, resulting in a 39-dimensional feature vector. These features are then standardized for classification. We evaluate the performance of two models are K-Nearest Neighbors (KNN) and Gaussian Mixture Model (GMM) using grid search to optimize hyperparameters such as the regularization number of neighbors for KNN, and components for GMM. The best-performing model is selected for classification.

4.3. Testing Process: After training both models, we used the testing dataset to assess the system. After extracting all features, the trained model was used to predict the ethnic class for every test file. The F1-score, accuracy, precision, and recall metrics were used to evaluate the classification performance. A confusion matrix was also created to offer a comprehensive picture of the classification outcomes.

4.4. Results

The test dataset was used to evaluate the system performance, and these are the findings.

The evaluation of the two models on the test set showed different levels of performance. GMM (Gaussian Mixture Model) had an accuracy of 77.5%, with higher precision and recall for Class 0, but it had problems with Class 1, where its recall dropped to 55%, as seen in Fig. 4. KNN (K-Nearest Neighbors) had an accuracy of 62.5%, with a more balanced performance between the two classes, but overall precision, recall, and F1-score were lower than GMM as seen in Fig. 5. The confusion matrices in Fig. 4 and Fig. 5 show the areas where both models made mistakes, with GMM performing better overall but still having problems, especially in Class 1 classification.

These results show that while GMM was better than KNN in accuracy and overall classification metrics, both models need more optimization to reduce mistakes and improve their ability to generalize well.

```
Evaluate GMM on the Test set...
Accuracy: 77.50%
Classification Report:
              precision    recall  f1-score   support

         0.0       0.69      1.00      0.82        20
         1.0       1.00      0.55      0.71        20

    accuracy                           0.78        40
   macro avg       0.84      0.78      0.76        40
weighted avg       0.84      0.78      0.76        40


Confusion Matrix:
```

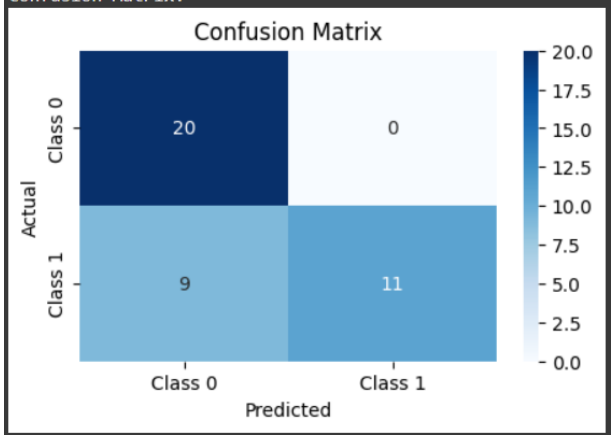

Figure 4:confusion matrix, and relevant metrics scores for GMM

```
Evaluate KNN on the Test set...
Accuracy: 62.50%
Classification Report:
              precision    recall  f1-score   support

         0.0       0.62      0.65      0.63        20
         1.0       0.63      0.60      0.62        20

    accuracy                           0.62        40
   macro avg       0.63      0.62      0.62        40
weighted avg       0.63      0.62      0.62        40
```

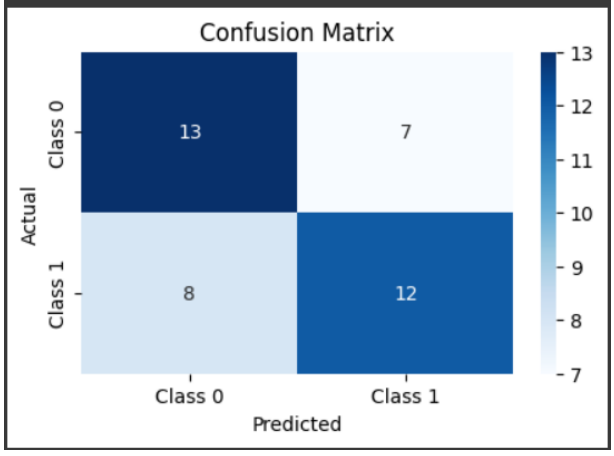

Figure 5: confusion matrix, and relevant metrics scores for KNN

5. Conclusion and future work

This project studied how to recognize ethnic groups from speech using machine learning. It focused on identifying Asian and White British speakers in Birmingham by using features like Energy, Zero-Crossing Rate, Pitch Frequency, and Mel-Frequency Cepstrum Coefficients (MFCCs). We tested two models, k-Nearest Neighbors (KNN) and the Gaussian Mixture Model (GMM). The GMM model performed better, with 77.5% accuracy, but both models had some problems with misclassification. This shows that combining basic audio features with machine learning can work well, but there are still challenges. The main issue was the limited dataset, which made it harder for the models to work well for all speakers.

In the future, the project can be improved in many ways. A larger dataset with more speakers and accents could help the models perform better. Using advanced features or deep learning models like Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) might also give better results. Further, improving the models through better tuning and combining different methods could reduce mistakes and make the system stronger.

6. References

[1] Analytics Vidhya, "MFCC Technique for Speech Recognition," [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/mfcc-technique-for-speech-recognition/. [Accessed: Jan. 9, 2025].

[2] "Spoken Language Processing Course Slides," 2024–2025.

[3] IBM, "K-Nearest Neighbors Algorithm," [Online]. Available: https://www.ibm.com/think/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20(KNN)%20algorithm%20is%20a%20non,used%20in%20machine%20learning%20today. [Accessed: Jan. 9, 2025].

[4] CmdLineTips, "Gaussian Mixture Models with Scikit-learn in Python," [Online]. Available: https://cmdlinetips.com/2021/03/gaussian-mixture-models-with-scikit-learn-in-python/?utm_source. [Accessed: Jan. 9, 2025].

[5] Applied AI Course, "Gaussian Mixture Model in Machine Learning," [Online]. Available: https://www.appliedaicourse.com/blog/gaussian-mixture-model-in-machine-learning/?utm_source. [Accessed: Jan. 9, 2025].

[6] Scikit-learn, "Grid Search," [Online]. Available: https://scikit-learn.org/stable/modules/grid_search.html. [Accessed: Jan. 9, 2025].

[7] Wikipedia, "Energy (Signal Processing)," [Online]. Available: https://en.wikipedia.org/wiki/Energy_(signal_processing). [Accessed: Jan. 9, 2025].

7. Appendix

The Code is on my Colab Notebook:
Recognizing Ethnic Groups