Efficient English Accent Classification Using Feature Engineering and Probabilistic Models

Seo Young Kim
Department of ECE
Carnegie Mellon University
Pittsburgh, PA 15213
seoyoung@andrew.cmu.edu

Siqi Li
Department of ECE
Carnegie Mellon University
Pittsburgh, PA 15213
siqili@andrew.cmu.edu

Hanzhi Yin
College of Fine Arts, School of Music
Carnegie Mellon University
Pittsburgh, PA 15213
hanzhiy@andrew.cmu.edu

Xiaoying Li
Department of ECE
Carnegie Mellon University
Mountain View, CA 94035
xiaoyin3@andrew.cmu.edu

1 Introduction

Despite the tremendous progress in automatic speech recognition (ASR) system in recent years that gives rise to applications such as conversational interactive voice responses (IVR's), automated car environment, and operating system voice assistant [5], the problem of identifying the speaker's accent from a given utterance still has a huge impact on ASR system's performance. Based on statistical studies, accent is one of the most critical factors of speaker variance [3]. By identifying accents, we can build more robust systems that are inclusive to speaker variability. Therefore, in our research, we aim at solving a multi-class English accent identification problem. There are two parts of our research. Given that accents mainly consist of subtle differences in a number of phonemes [6], we first design various feature engineering methods to examine the audio feature performance. Next, we implement an unsupervised Gaussian Mixture Model (GMM) and a Hidden Markov Model (HMM) for classification and cross compare their multi-class accent identification performance. To further innovate our design, since gender is another critical factor of speaker variability, we extend our research with gender classification using pitch estimation. By using various feature engineering methods and accent classification models, we believe our proposed method helps to serve as a preliminary step in the ASR pipeline in order to make the ASR system achieve better performance.

2 Related Work

There are a lot of research done in the English accent identification field. Our main baseline project is an AccentDB research [1]. This work adopts the Mel-Frequency Cepstrum Coefficients (MFCC) approach for feature extraction and deep neural network models for accent classification. For audio feature extraction, we base our original methods on the MFCC approach suggested by the AccentDB paper. For classification models, we innovate based on the two major directions of research that focus on accent classification. The first approach is a GMM [2], and the other is a HMM [4]. In our method, we reference these work to design classification models in order to achieve comparable accent identification performance as that of the AccentDB project.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

3 Dataset

Our input data contain speech audio files provided by an AccentDB extended dataset [1]. The dataset consists of wave audios produced by multiple speakers with nine different English accents, including 4 native ones of American, Australian, British, and Welsh and 5 non-native Indian English accents of Bangla, Malayalam, Odiya, Telugu, and Indian. It is worth mentioning that the American, Australian, British, Welsh, and Indian accents are machine generated instead of being recorded by a real person, so the dataset is relatively free of real-world noise and outliers. There are 16,894 audio samples for training and testing, all of which are recordings of speakers saying the same set of sentences, and each audio recording is pre-splitted to sentence long and trimmed to around 5 seconds. Hence, the dataset is complete (no further processing needed) and we use it as our direct input to extract features. The outputs of our feature extraction procedure are NumPy binary files containing 2D NumPy vectors.

4 Experiment

As seen in Figure 1, we experiment with feature engineering, then classify accents using different models.

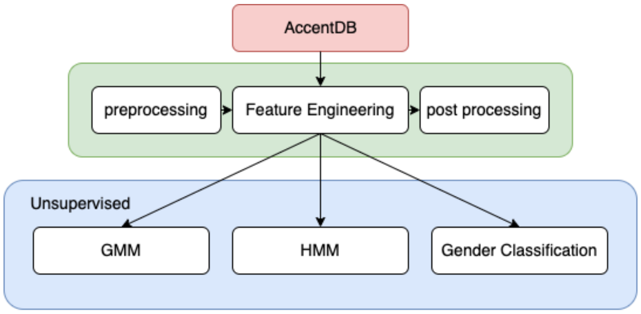


Figure 1: Our Proposed Method Flowchart.

4.1 Feature Engineering

Raw audios cannot be directly utilized as dataset, so a transformation is necessary. The process can be pipe-lined as four steps. The first step is loading raw audios into memory. Reading sample rate can be an adjustable parameter. The second step is preprocessing audio data. For preprocessing, we trim out silence [1] or fix audio data to a same length, or apply both methods. The third step is using MFCC algorithm [2] to transform audio data into spectra. The number of MFCC features can be an adjustable parameter. The final step is postprocessing MFCC spectra. Since MFCC spectra are two dimensional, with one dimension being the time and the other being the number of MFCC features, transforming 2D spectra to 1D vectors is essential. For postprocessing, we either directly flatten the 2D array [1] or take the average along the time domain.

To summarize, there are four parameters to control the generated dataset's shape and behavior, which are audio loading sample rate, preprocessing methods (trimming silence, fixing length, or firstly trimming silence and then fixing length), number of MFCC features, and postprocessing methods (flattening array or taking average along the time domain). Note that the combination of trimming silence (preprocessing method) and flattening array (postprocessing method) cannot be combined together. Since trimming silence cannot ensure each audio to have the same length which yields uneven feature space, averaging along the time domain is the only feasible step. Regarding the preprocessing and postprocessing methods, there are only 5 available combinations. Also, we made an assumption that accents have stable properties along the time domain. If such assumption holds, the postprocessing method, taking average along the time domain, is feasible. Detailed illustrations on different preprocessing and postprocessing methods are shown in the Appendix A. In our experiment,

[1] the absence of ambient audible sound

[2] `python_speech_features`

the combination of preprocessing method to trim silence and postprocessing method to take average along time axis performs the best. More details on the experiment is presented in Section 5.1.

4.2 Multi-class Classification

We implement GMM and HMM to classify English accents. There is a class imbalance in our dataset and we carefully considered this imbalance. In the dataset, the American English accent takes around one-third of the whole dataset, while the other accents have similar sizes. In order to take this into account, instead of splitting the whole dataset into train and test data, we split data for each accent into train and test data.

4.2.1 GMM

In order to train GMM, we assume that 13 MFCC features are a mixture of Gaussian distributions. We train 9 GMMs, with each model corresponding to one accent. For each test data point, we feed it into all 9 GMMs and compute the log likelihood, and then classify the data point as the accent that gives the highest probability.

4.2.2 HMM

HMM has the same workflow as GMM, but it has a different input assumption. We assume that the MFCC features have similar properties as time sequences.

5 Results

For this experiment, all 9 accents are used for training and testing. Our evaluation metric is simply the classification accuracy. We split the data into 2 parts, with 80% being train data, and 20% being test data. After we get the classifiers from the training phase, we apply the classifiers on the test data and calculate the accuracy based on the prediction and true label.

5.1 Feature Engineering Results

All experiments are being conducted by stress testing classification using GMM 100 times. We use box plots to illustrate how training accuracy distributes.

We conduct several experiments to explore how different feature engineering schemes can affect training accuracy using GMM. We firstly explore how different audio loading sample rates affect accuracy. The preprocessing method is fixed to be fixing length to 7 seconds, the postprocessing method is fixed to be flattening 2D array to 1D vector, and the number of MFCC features is fixed to 13. Figure 2 shows the overall accuracy. We experiment with how different sample rates affect the accuracy and specifically chose to compare the results the of the 8 kHz and 16 kHz. The ordinary audio sampling rate is 8 kHz, and for high fidelity audio communication, the current standard is 16 kHz. Generally, in our experiment, 8 kHz audio sample rate performs better.

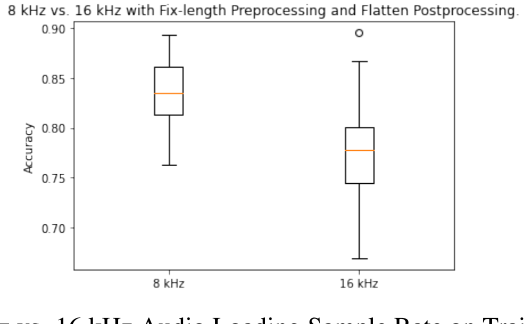


Figure 2: 8 kHz vs. 16 kHz Audio Loading Sample Rate on Training Accuracy.

We then explore how different combinations of preprocessing and postprocessing methods affect accuracy. The audio sample rate is fixed to be 8 kHz, and the number of MFCC features is fixed to

13. Figure 3 shows the overall accuracy. In our experiment, the combination of preprocessing method to trim silence and postprocessing method to take average along time axis performs the best.

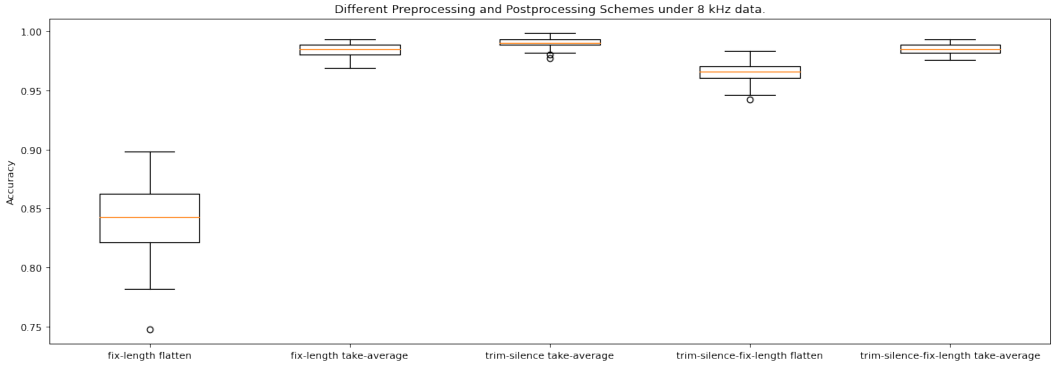


Figure 3: Different Preprocessing and Postprocessing Methods on Training Accuracy

Finally, we explore how different numbers of MFCC features affect accuracy. The audio sample rate is fixed to be 8 kHz, the preprocessing method is fixed to be trimming silence, and the postprocessing method is fixed to be taking average along the time axis. Figure 4 shows the overall accuracy. In our experiment, as the number of MFCC features increases, the accuracy increases. Though it may seem that the more the better, the trade off between the training speed and the accuracy should be considered. So far, we believe that having number of MFCC features to be 13 should be decent, as it marks the edge of accuracy saturation.

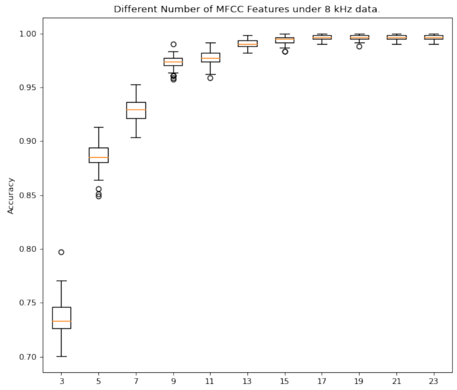
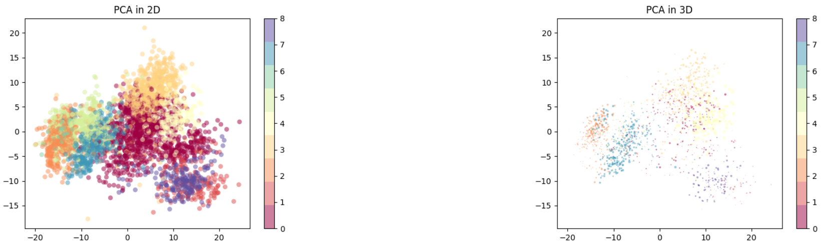


Figure 4: Different Number of MFCC Features on Training Accuracy

5.2 GMM and HMM results

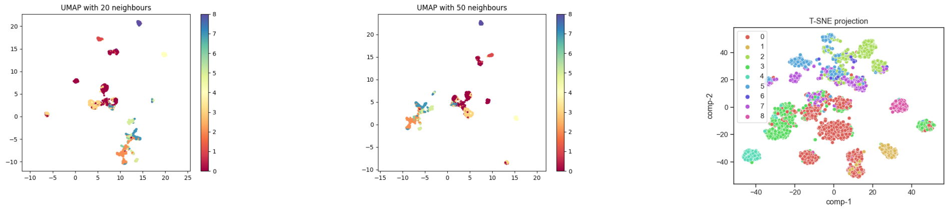
For both GMM and HMM, the same combination of preprocessing method to trim silence and postprocessing method to take average along time axis is used. For GMM and HMM, the classification accuracies are 98.5% and 71.2%, respectively. We visualize the data with different dimensionality reduction methods, namely PCA, T-SNE, and UMAP. For GMM and HMM, the same visualization method is used. GMM visualization results are reported in Appendix B. Detailed implementations of the methods are included in Appendix as well. As shown in Figure 5, the dataset clusters pretty well in general. As evident in the classification accuracy, the HMM model does not perform as well as GMM. This indicates that our initial assumption for MFCC features might be wrong. MFCC features might not have enough inner connection to make classification under HMMs.



(a) Clustering after reducing the dimension to 2D

(b) Clustering after reducing the dimension to 3D

Figure 5: Visualization of HMM classification using PCA for dimensionality reduction



(a) Clustering using UMAP with 20 neighbors

(b) Clustering using UMAP with 50 neighbors

(c) Clustering using T-SNE

Figure 6: Visualization of HMM classification using UMAP and T-SNE

5.3 Gender Classification

To further extend our experiment, we perform gender classification since it has a huge impact on automatic speech recognition. Gender classification is a technique that aims to determine the gender of a speaker through speech signal analysis. This has several potential applications such as (1) as part of an automatic speech recognition system to enhance speaker adaptation, and (2) as part of automatic speaker recognition systems. It is well known that the major difference between male and female speech is the pitch. Generally, women have higher pitch than men. This discrimination qualifies pitch as an effective feature for gender classification. In our classification system, the modified Harmonic Product Spectrum [7] is used to estimate the pitch of segmented speech. Because our dataset does not have gender labels, we could not evaluate how well the model has classified gender. So, we test out the pitch detection algorithm on a different dataset, The Free ST American English Corpus dataset (SLR45)[3], that has gender labels to evaluate the effectiveness of the algorithm. This dataset contains utterances from 10 speakers (5 females and 5 males). Each speaker has about 350 utterances and we did not use any preprocessing or postprocessing techniques. The method has achieved about 80% accuracy on classifying female and male on this dataset. On the AccentDB dataset, the model has classified 5819 females, 11,494 males out of 16,894 audio samples. The visualization of the gender classification results is shown in Figure 7.

6 Discussion and Analysis

Our approach is based on the AccentDB paper [1] as mentioned earlier. In the paper, the authors use the similar feature engineering techniques without experimenting different preprocessing and postprocessing methods. For the classification method, they use deep neural architectures like Convolutional Neural Networks (CNNs). With this deep learning model, they achieve 99.3% accuracy and 99.5% (with attention) on the AccentDB dataset. Although deep learning models are known to have high performance, the major downside of deep learning is its computational intensity, which requires high-performance computational resources and long training times. In our method, we experiment with different preprocessing and postprocessing techniques in order to reduce this bottleneck, and improve the overall performance up to about 10%. In addition to the combinations of different preprocessing and postprocessing methods, we implement GMM as our classification

[3] http://www.openslr.org/45/

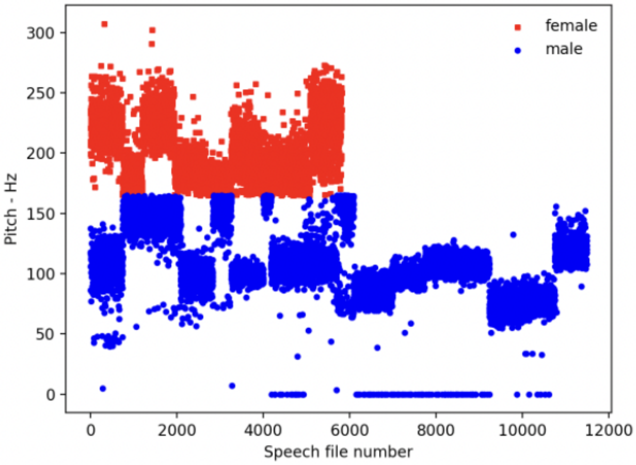


Figure 7: Gender classification results

model and achieve 98.5% accuracy, which is comparable to AccentDB results. Our method improves the efficiency of the paper we are basing on by reducing the required computation and resources, while achieving comparable results. In addition to this, we further extend our research by performing gender classification. Although gender classification results do not have a direct impact on accent classification results, it gives more insights on the overall dataset. Therefore, we have optimized the baseline paper by reducing the bottleneck of computationally-intensive models, while achieving similar performance with more insights about the speakers' accents.

It is worth noting that some raw audios from AccentDB are machine generated, which may have several drawbacks. Since machine generated audios are rigid and have easily discernible aspects, it may leave quantitative trails in the dataset and let machine learning algorithms identify them quickly. Additionally, there is a limitation on testing how well our classification model will perform in real-world scenarios, because the dataset does not have any noise. Having no noise in some audios may be easy for our trained classifiers, but in reality, noise may impede classification, and so far given by the AccentDB dataset, there is no way to testify that.

One limitation of the HMM is that the model can only take in two dimensional data, while we have three dimensional data after applying MFCC to the raw data. We have to either flatten or average the data along the time axis, which could eliminate some properties among the MFCC features. One possible solution to this problem is to modify the existing HMM algorithm so that it takes in three dimensional data. However, dimensionality reduction is more widely used in machine learning based models, and directly training on three dimensional data is rare. Another possible way is to average along the MFCC feature axis, which will still eliminate some properties of the original data, which potentially can be another limitation.

7 Conclusion

In our proposed method, we optimize the baseline paper by reducing the bottleneck of a computationally-intensive model, while achieving a similar performance with more insights about the speakers' accents. We explore various feature engineering methods to transform audio data into computable datasets, by pipelining the process to four steps: load, preprocess, transform, and postprocess. We also explore two unsupervised learning methods, GMM and HMM. As gender also affects automatic speech recognition, we extend our research to cover gender classification. Future scope of our work includes experimenting how gender classification can be incorporated in our accent identification model, applying accent identification in real-time, and testing our approach on an extremely noisy dataset. Additionally, we may experiment with using all the MFCC vectors in a recording, assuming them to be a set of IID (Independent and Identically Distributed) vectors and consider an alternate approach such as using i-vectors to perform factor analysis on GMM features since they may be more accurate than neural features.

References

[1] Afroz Ahamad, Ankit Anand, and Pranesh Bhargava. Accentdb: A database of non-native english accents to assist neural speech recognition. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5353–5360, Marseille, France, May 2020. European Language Resources Association.

[2] Too Chen, Chao Huang, E. Chang, and Jingehan Wang. Automatic accent identification using gaussian mixture models. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2001. ASRU '01.*, pages 343–346, 2001.

[3] Chao Huang, Tao Chen, and Eric Chang. Accent issues in large vocabulary continuous speech recognition: Special double issue on chinese spoken language technology. *International Journal of Speech Technology*, 7, 01 2004.

[4] Kanchan Naithani, V. M. Thakkar, and Ashish Semwal. English language speech recognition using mfcc and hmm. In *2018 International Conference on Research in Intelligent and Computing in Engineering (RICE)*, pages 1–7, 2018.

[5] Pratiksha Raut1 and Seema Deoghare. Automatic speech recognition and its applications. In *International Research Journal of Engineering and Technology*, pages 2368–2371, Maharashtra, India, May 2016. International Research Journal of Engineering and Technology.

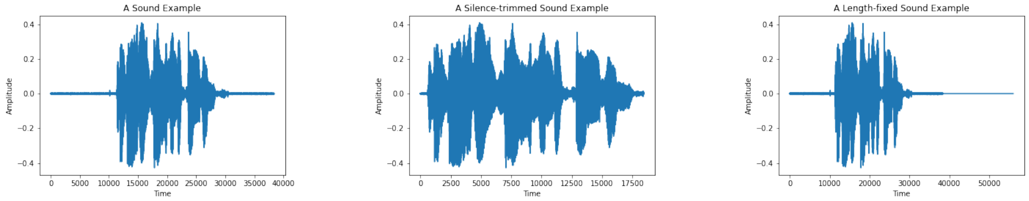
[6] Tingyao Wu, Dirk Van Compernolle, Jacques Duchateau, Qian Yang, and jean-pierre Martens. Improving the discrimination between native accents when recorded over different channels. pages 2821–2824, 01 2005.

[7] Yu-min Zeng, Zhen-yang Wu, Tiago Falk, and Wai-yip Chan. Robust gmm based gender classification using pitch and rasta-plp parameters of speech. In *2006 International Conference on Machine Learning and Cybernetics*, pages 3376–3379, 2006.

A Feature Engineering

A.1 Preprocessing Methods

See Figure 8.

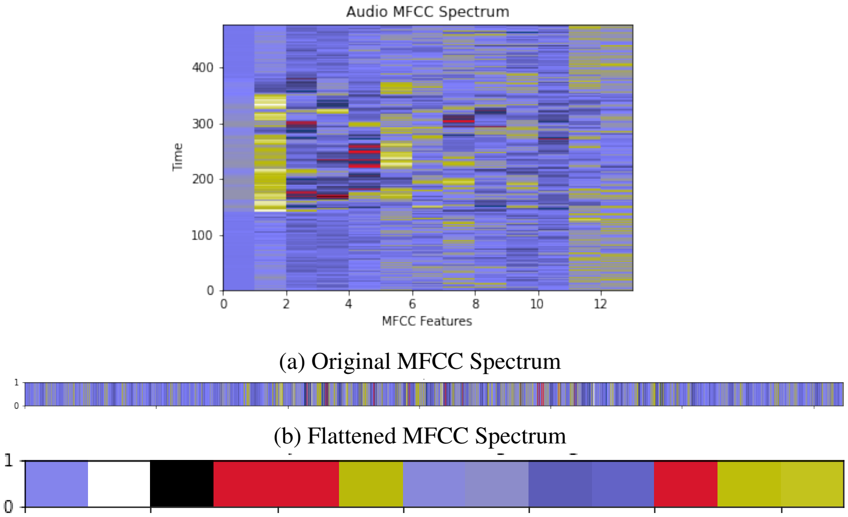


(a) Original Audio (b) Audio with Silence Trimmed (c) Audio with Length to be Fixed

Figure 8: Different Preprocessing Methods Applied to Raw Audio

A.2 Postprocessing Methods

See Figure 9.



(a) Original MFCC Spectrum

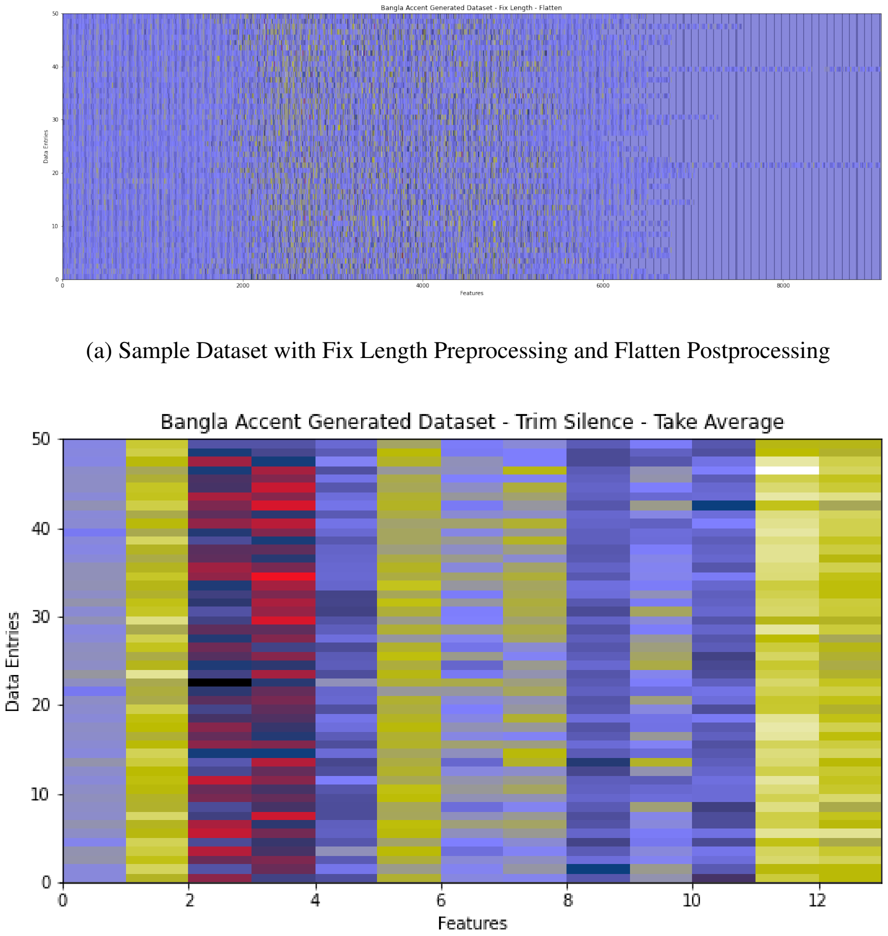
(b) Flattened MFCC Spectrum

(c) MFCC Spectrum with Time Axis Averaged

Figure 9: Different Postprocessing Methods Applied to MFCC Spectrum

A.3 Sample Datasets

See Figure 10.



(a) Sample Dataset with Fix Length Preprocessing and Flatten Postprocessing

(b) Sample Dataset with Trim Silence Preprocessing and Take Average Postprocessing

Figure 10: Two Sample Partial Dataset with 8 kHz Sample Rate and 13 MFCC Features

B GMM, HMM, and Gender Classification

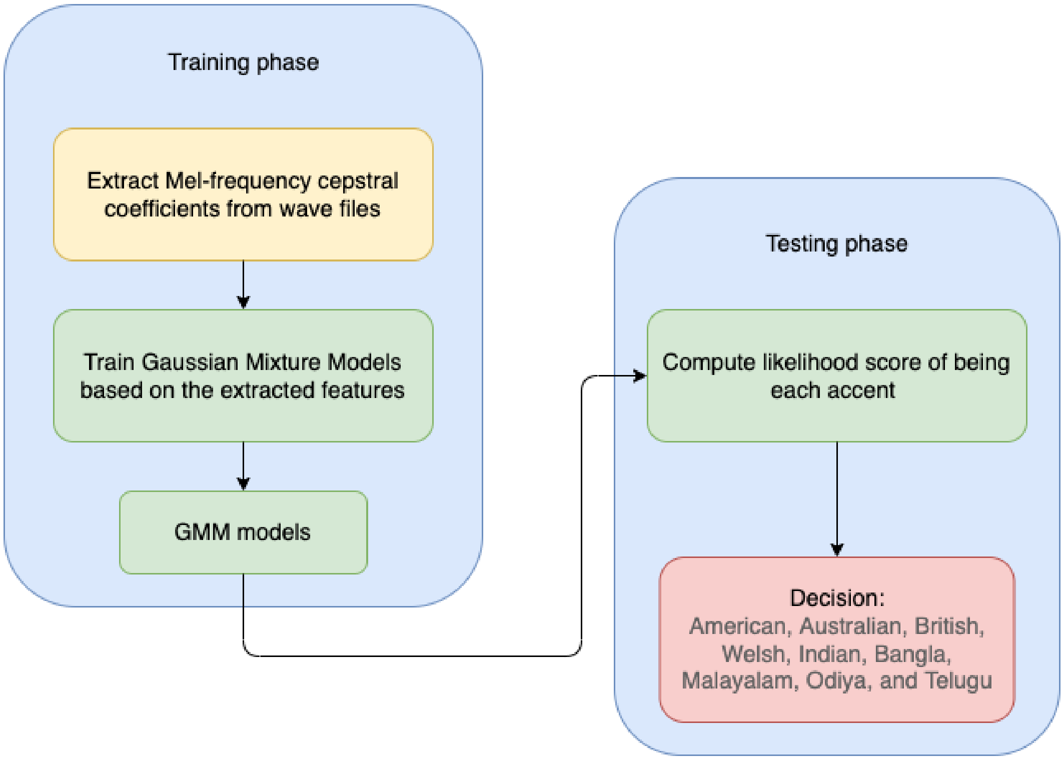

Training phase
Extract Mel-frequency cepstral coefficients from wave files
Train Gaussian Mixture Models based on the extracted features
GMM models
Testing phase
Compute likelihood score of being each accent
Decision:
American, Australian, British, Welsh, Indian, Bangla, Malayalam, Odiya, and Telugu

Figure 11: Flowchart for GMM

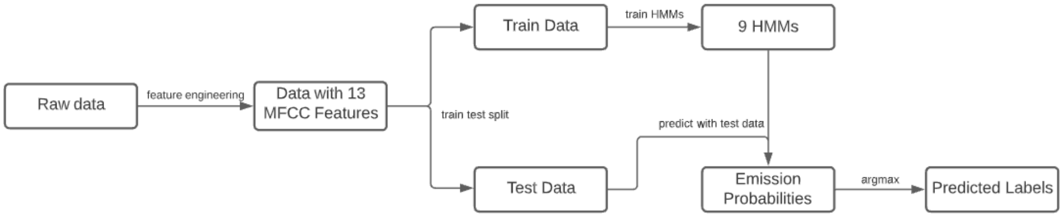
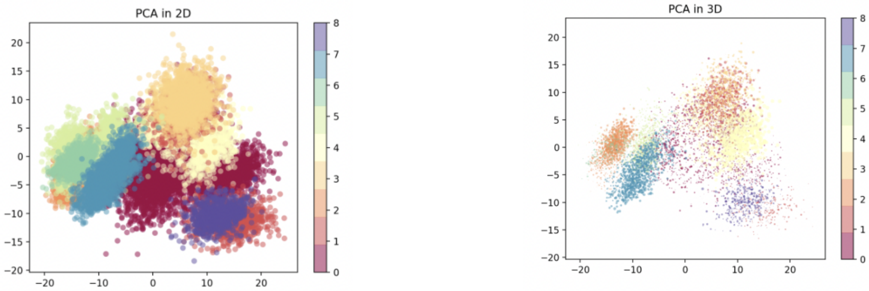


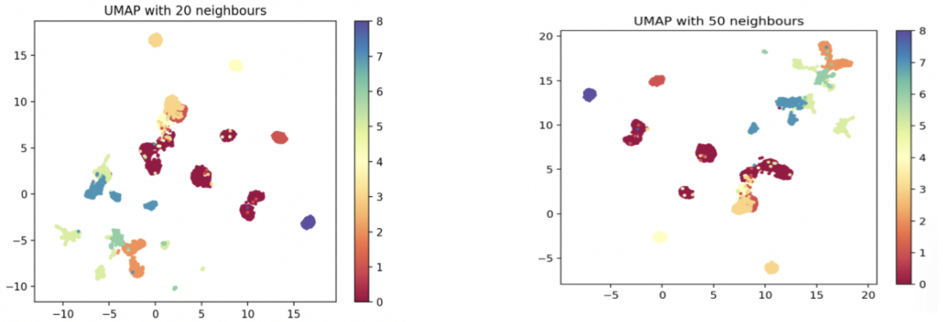
Figure 12: Flowchart for HMM



(a) Clustering after reducing the dimension to 2D

(b) Clustering after reducing the dimension to 3D

Figure 13: Visualization of GMM classification using PCA for dimensionality reduction



(a) Clustering using UMAP with 20 neighbors

(b) Clustering using UMAP with 50 neighbors

Figure 14: Visualization of GMM classification using UMAP

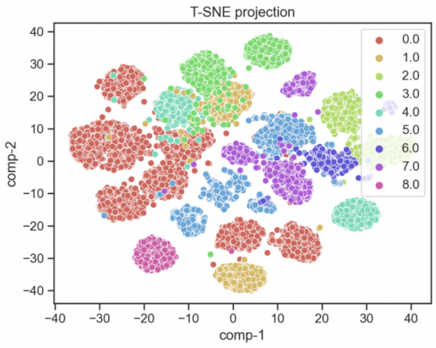
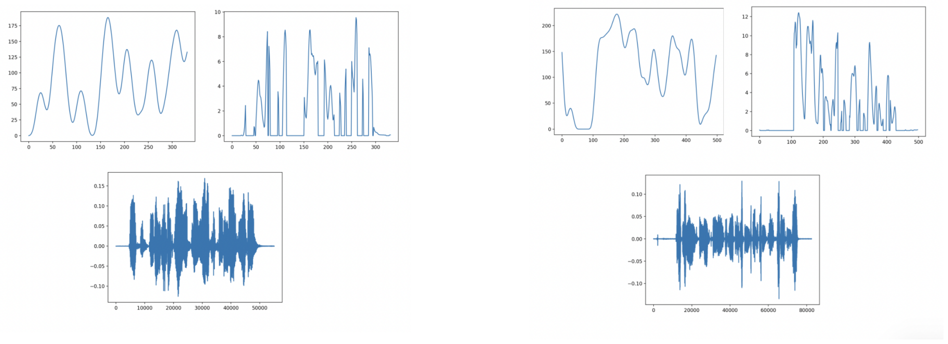


Figure 15: Visualization of GMM classification using T-SNE



(a) Smoothed out frequency, magnitude, and raw audio visualization of a male audio

(b) Smoothed out frequency, magnitude, and raw audio visualization of a female audio

Figure 16: Visualization of male and female audio