# English Accent Detector
# Final Project Report

Mariam Elgamal (me1553), Yeojin Jung (yj1254)

*Tandon School of Engineering, New York University*

**Abstract**

In this report we explain the rationale, implementation and results of our English Accent Detector. The aim of our project is to predict the country or region of origin of the speaker from their English pronunciation and accent. Many applications of this work include language teaching, video game development and advanced speech recognition. Previous work in the literature focused on classifying two English accents using different preprocessing and machine learning techniques. Our work proposes different preprocessing approaches to classify around 16 different English accents using Mozilla's Open source speech dataset, Common Voice. We then optimize for different neural network models to outperform the accent classification reported in the literature by around 20% increase in accuracy.

## 1. Introduction

Speech recognition has been a highly active research area. Part of this work includes accent recognition to facilitate for better performing and more inclusive automatic speech recognition (ASR), commonly known as Apple Siri and Amazon's Alexa. We believe one's English accent is largely influenced by their mother tongue, especially from the frequently used pronunciations, intonations, and syllables. Our work focuses on utilizing a relatively new dataset in the area of speech recognition to improve the performance and accuracy of multi-accent classifiers, an underserved area of research.

## 2. Literature Review

There are many factors that contribute to differences in the English accent between natives and non-natives, as well as between different non-native accents. In previous research, links were drawn between word duration and accentedness [3]. Where native English speakers produce shorter word duration with less variance as well as exhibit stronger function word reduction and frequency effects than non-natives. Other factors and limitations include spontaneity, effect of globalization on non-native accents, fluency and the variety of accents in our training datasets.

There are many approaches to speech accent classification in the literature from classical Hidden Markov Models (HMMs) to modern machine learning methods using Support Vector Classifier (SVC) and deep learning methods using different types of neural networks. We found that most of the literature were mainly invested to differentiate between two accents per classification at most, particularly British English and American English. They scored relatively high accuracies including Hernandez et. al. who achieved a 71% test accuracy over two accents in a commercial video game in 2018 [4]. The latest includes differentiating between English accents from UK and Mexico using Long Short-Term Memory neural network, and achieved a classification accuracy of 94.47% in 2019 [11].

Literature attempting to classify three accents or more tends to be scarce and is a less active area of research when compared to classifying two accents through different preprocessing methods. We attribute this to the relatively low accuracies currently reported, which do not exceed a 40% classification accuracy as of the latest communicated research project in 2017 [9, 10, 12].

## 3. Implementation and Experiments

Our team used Google Colaboratory to collaborate throughout the development of the project. Our final published code can be found on github.

### *3.1. Dataset*

Our work focuses on a global, open-source voice database by Mozilla, known as Common Voice. It includes a plethora of human voice recordings in a plethora of languages for different scripts. Since our project is concerned with the English language, our data has 16 English accent classifications which include: US, Australian, Canadian, Irish, UK, Filipino, Hong Kong, Indian, Malaysia, New Zealand, Scottish, Singaporean, Welsh, South Atlantic, Southern African, and West Indies and Bermuda.

Due to the limited capacity and computation power of our machines, we used a large enough kaggle dataset, derived from

Mozilla's Common Voice, to train and test our models. Another reason we decided to use the kaggle dataset is because Mozilla recently changed their data collection policy on accentedness. Instead of allowing the contributing speakers to identify their country of origin, Common Voice collects the geographical location of the speaker as the region of origin, which is severely inaccurate for the purpose of our application. Similar to any data collection effort, Common Voice has some limitations and external factors that play a role in the speakers' collected speech recordings, including speaker spontaneity, fluency and the effect of globalization on non-native accents.

### *3.2. Audio Processing and Features Extraction*

The kaggle dataset required extensive data cleaning. We initially deleted all recordings with a blank accent label using a python script, and then converted the mp3 audio recordings to wav files using python's pydub library.

In the features extraction phase of our project, we analyzed our audio recordings through understanding specific linguistic features and commonalities. This is to ensure that we utilize a sensible feature relevant to accentedness, which we can later input into our different neural networks models to achieve high classification accuracy. We plot each audio recording, computed their Fourier transforms and plotted their different spectrograms as shown in Figure 1.

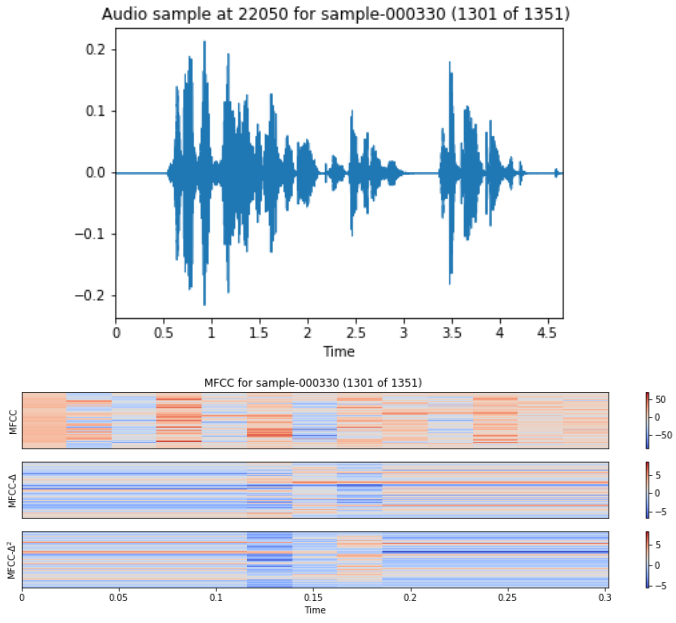


Figure 1: A sample plot of an audio recording followed by its corresponding MFCC and MFCC-deltas spectrograms using python.

In our final implementation, we specifically focus on the Mel Frequency Cepstral Coefficient (MFCC) and its deltas as our main feature, due to its wide and established use in different speech recognition research. The Mel scale relates the perceived frequency of a tone to the measured frequency, which helps to mimic or showcase how humans identify small changes in speech. This is done by the MFCC, since it can accurately depict the envelope of the time power spectrum of the speech signal generated by humans depending on their vocal tract. In other words, the MFCC features represent phonemes, which are distinct units of sound generated by the vocal tract, through which different human accentedness can be showcased and analysed.

All Common Voice recordings are 1-2 sentences long. Accordingly, we attempted to chunk the recordings into word utterances in order to overcome issues with different word durations for different speakers, as well as the frequency effects of native and non-native accents. However, this attempt did not prove to be reliable for a couple of reasons. One of them is the recordings do not have the same script or sentences, therefore chunking the audio to word utterances would not have as much of an effect on the input. Secondly, the speech speed, i.e. the silence between the words in a spoken sentence, can be considered a feature related to accentedness that we do not want to eliminate. This is especially that different languages have different rhythms and frequencies that we believe affect the accentedness of the speaker as well. We alternatively decided to preprocess a set duration from all the audio files as a control. This is to ensure we have a more holistic approach to the problem, especially since humans generally recognize accents from a stream of words or a spoken sentence and not only one word.

After plotting the audio wav files and taking their Fourier transforms using a sampling rate of 22.5 KHz (audible human sound), we extract the top 13 MFCC coefficients. Then using python scipy library we read the wav file and compute the sampling rate of each recording. Afterwards using the python speaking features library, python_speech_features, we get the mean of each of the top 13 MFCCs for each recording and use these values as the input to our neural networks.

### *3.3. Deep Learning Approaches and Models*

We examine three different neural network models for our application, namely, feedforward neural networks, convolutional neural networks (CNN) and recurrent neural networks (RNN). We compare the different architectures and performances of each with respect to the number of hidden layers and regularization methods in the Results and Discussion section.

For the feedforward neural network, we implemented a 2-layer with adam optimizer and categorical cross entropy loss, once using softmax activation function and once using ReLU. As expected, feedforward neural networks resulted in very low accuracies, below 50%, and therefore we did not continue with them.

From the literature review, we have confirmed that CNN model showed better performance in accent detection and classification than other above-mentioned models [2, 5, 13].

CNN is known to perform well on image classification tasks and, in most of the projects, including ours, the MFCCs were extracted from the utterances to form an image-like input that is fed into the CNN [2]. As such, we have effectively reduced the accent classification task from an audio to an image. We implemented 1-dimensional CNN model with various number of layers. Basic skeleton includes two Convolution 1D layers following batch normalization after each to keep weights in [0,1] range and maxpooling after all but the first Convolution, one Flatten layer (to add a dense vanilla layer later), one Dropout layer (to reduce overfitting), and the last single unit output Dense layer. A sample architecture is shown in Figure2. After several trials of adding more Convolution 1D layers to this skeleton, the best performance was obtained from using three Convolution 1D layers. Our model was built on top of the Keras deep learning Python package and largely based on a text processing example and various other open-source accent classification projects found online. The model was trained on 80% of our dev data, using full (1080,13)-dimensional MFCCs.

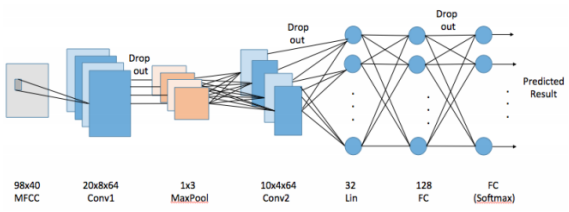


Figure 2: A sample model architecture of using CNN for accent classification. [14]

The third model we developed was an RNN model, specifically a long short-term memory (LSTM) RNN model. LSTM is a specific version of RNN that employs memory cells to preserve data throughout the sequence of unknown duration [5]. Our LSTM RNN model was built on top of Keras package. We made many adjustments and tuning to the parameter values and the number of layers. We decided on having three LSTM layers, followed by two Dense layers and one Dropout layer. We referred to Guo et al.'s LSTM-RNN model for binary accent classification, shown in Figure 3. Again, the model was trained on 80% of our data, using (n,1080,13)-dimensional MFCCs as its inputs. The data were either passed as 'stateful' or not, meaning that the last state for each sample at index $i$ in a batch will be used as initial state for the sample of index $i$ in the following batch–in this case, the 3-dimensional batch size must be passed explicitly into the model's first layer.

We choose adam optimizer since it is an efficient stochastic gradient descent method. To prevent overfitting we added dropout layers to all of the CNN and RNN models and also assessed the effect of different regularizers. As shown in Table 1, using L1 and L1-L2 regularizations gave the best performance for CNN 1D model, while single L1 and single L2 regularizations performed the best for RNN-LSTM.

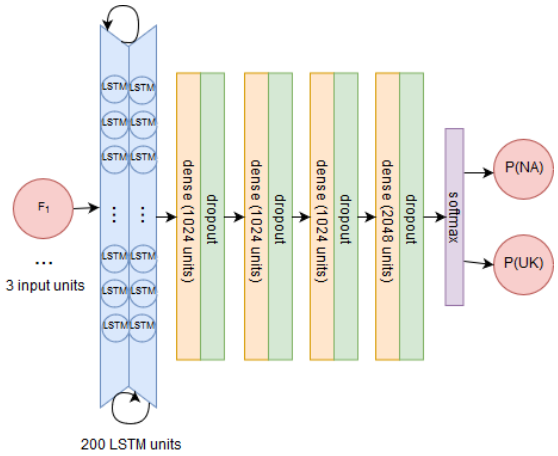


Figure 3: A sample model architecture of using RNN for accent classification. [5]

| | CNN 1D | RNN-LSTM |
|---|---|---|
| without regularization | 0.563 | 0.504 |
| L1 regularization | 0.570 | 0.537 |
| L2 regularization | 0.563 | 0.537 |
| L1-L2 regularization | 0.570 | 0.533 |

Table 1: Initial validation accuracies of each of the CNN 1D and RNN LSTM models with different regularization methods (30 epochs)

## 4. Results and Discussion

We determined that CNN 1D with 3 hidden layers and L1-L2 regularization has the best performance of our models and ran it for 100 epochs. As displayed in Figures 4 and 5, we can see the classification accuracy compared to the training accuracy as well as the validation and training cross entropy losses for our two main models, RNN LSTM and CNN respectively.

While there is a significant gap between training and test accuracy, the test accuracy our model achieves is significantly higher than those reported by the literature [9]. The larger the dataset we used, the higher the accuracy our model outputs, due to more diversity and the better quality of the data. Therefore, we can attribute the increased accuracy we obtained due to the more diverse speech we used for our input data. The recordings were of people speaking different English script. Therefore instead of chunking the script to words and aligning them as done in the literature, our method allowed the neural network to capture more accent features rather than learn speech recognition. This is especially that many words are monosyllabic and relatively hard to recognize the speaker's accent from. By thinking about how humans perceive and recognize different accents, we realized it is not only about the pronunciation, but it also includes the tempo and the rhythm of human speech that need to be accounted for. So it is necessary to input a long enough speech, i.e. 1 or 2 sentences to extract the necessary accentedness features.

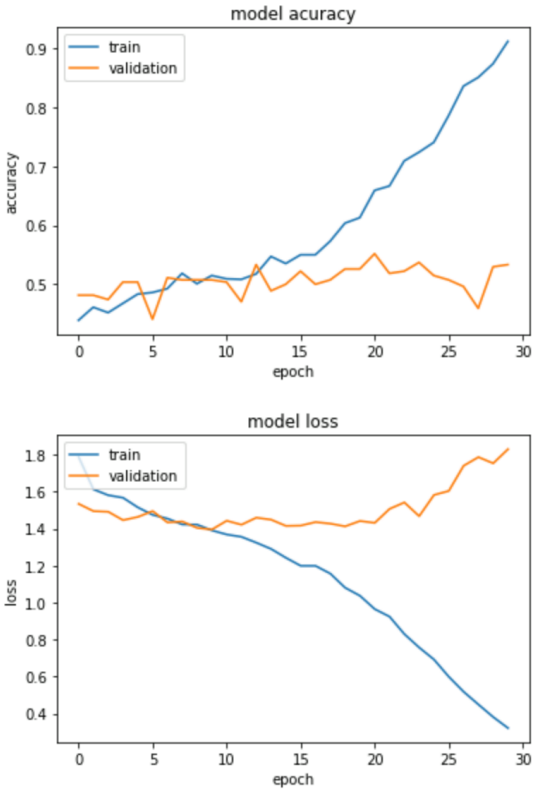


Figure 4: The classification accuracy and cross-entropy loss of RNN LSTM for 30 epochs. Maximum Accuracy is around 50%

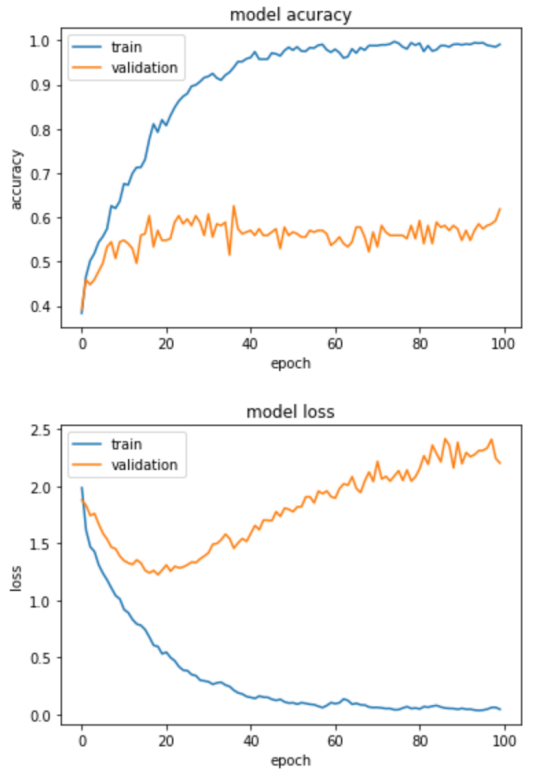


Figure 5: The classification accuracy and cross-entropy loss of CNN 1D for 100 epochs. Maximum accuracy is around 61%

We also computed the confusion matrix for the best CNN-1D model with the optimal test accuracy of 0.611, using python scikit. From there we deduced a further limitation in our dataset, where there is a severe discrepancy in the percentage (availability of audio files and labelled data) of each accent. In our case for example, we have a majority of "US" English accents but very limited number of "African" labelled accent. This adversely impacts the accuracy the model outputs due to the lack of training data for these minority accents and can lead to overfitting.

## 5. Conclusions and Future Work

We have shown that our CNN-1D model results in the best performance of our three neural network models. We obtained a final classification accuracy of around 61%, which significantly exceeds the maximum 40% classification accuracy for five accents as reported in the literature [9, 10, 12].

There are several potential measures we could take to further develop and improve our results in the future. One important aspect for future models would be to spend more time on hyperparameter tuning, since it requires an extensive and time-consuming grid search.

Another interesting direction would be to use an image processing method to input the different MFCC and MFCC delta plots of the recordings to neural networks models instead of the numerical values. This is a direction many of the literature utilized for different speech recognition applications and we were initially intending to take but were limited by the computational power and memory of our machines.

Furthermore, part of our project proposal included collecting audio recordings from the NYU Abu Dhabi community given the diversity of the student body. Unfortunately, due to an abrupt decision by NYUAD's administration for students to evict the Abu Dhabi campus starting mid-April we were unable to ask the community to contribute to our project during such uncertain times. An obvious next step would be to collect these audio files once school resumes back to in-person classes, to help assess our models better and even contribute with a dataset that has a more diverse classification of accents.

The boundaries between different accents are becoming more fluid nowadays due to the immigration and globalization of our times. Therefore, in future work, it would be important to consider developing neural network models that account for the globalized non-native accents. Where one speech can be classified by more than one label to accommodate for the complex reality of globalized accents.

## 6. References


[1] Wildcat Corpus of Native- and Foriegn-Accented English. (link)

[2] Sheng, Leon Mak An and Mok Wei Xiong Edmund. (2017). Deep Learning Approach To Accent Classification. (pdf)

[3] Baker, R. E., M. Baese-Berk, L. Bonnasse-Gahot, M. Kim, K. J. Van Engen, and A. R. Bradlow. (2011). Word durations in non-native English. Journal of Phonetics, 39, 1-17. (pdf)

[4] Hernandez, S.P., Bulitko, V., Carleton, S., Ensslin, A., Goorimoorthee, T. (2018). Deep Learning for Classification of Speech Accents in Video Games. AIIDE Workshops. (pdf)

[5] Guo, Y. Violet. (2019). Speaker Accent Classification with Deep Learning. (link)

[6] S. Yoo, I. Song, and Y. Bengio. (2019). A Highly Adaptive Acoustic Model for Accurate Multi-Dialect Speech Recognition. (pdf)

[7] Abdel-rahman, Mohamed. (2014 thesis). Deep Neural Network acoustic models for Automatic Speech Recognition. (pdf)

[8] Graves, Alex, Mohamed, Abdel-rahman, and Hinton, Geoffrey. (2013). Speech Recognition with Deep Recurrent Neural Networks. (pdf)

[9] Chu, Lai, and Le. (2017). Accent Classification of Non-Native English Speakers. (pdf)

[10] Matthew Seal, Matthew Murray, Ziyad Khaleq. (2011). Accent Recognition with Neural Network (pdf)

[11] Bird, J. J., Wanner, E., Ekárt, A., Faria, D. R. (2019). Accent classification in human speech biometrics for native and non-native english speakers. In Proceedings of the 12th ACM International Conference on PErvasive Technologies Related to Assistive Environments, PETRA 2019 (pp. 554-560). ACM. (link)

[12] Morgan Bryant, Amanda Chow, Sydney Li. (2014). Classification of Accents of English Speakers by Native Language. (pdf)

[13] Kevin Chionh, Maoyuan Song, Yue Yin. (2018). Application of Convolutional Neural Networks in Accent Identification. (pdf)

[14] Xuejiao Li, Zixuan Zhu. (2017). Speech Command Recognition with Convolutional Neural Network. (pdf)