Lexical Stress Detection in Spoken English

Aditi Singhal, Madhavi Srinivasan, Vivek Kumar

INTRODUCTION

We intend to design a neural network architecture to automate the classification of lexical stress that would be integrated into computer aided pronunciation learning (CAPL) tools for English secondary language learning.

PROBLEM STATEMENT

Blue Canoe Learning, our sponsor, helps non-native speakers improve English pronunciation using machine learning and Artificial Intelligence (AI) . The current models in the company provide feedback on consonant substitution and quality of vowels spoken but fail to correctly detect errors related to lexical stress in continuous speech. We have implemented a lexical stress classifier to detect two levels of stress at phoneme-level.

In English pronunciation, stress is important. Misplacing the syllabic stress can alter a word's part of speech or even alter the meaning of the word. Stress is seen as a property of the syllable or vowel center of the syllable. Stress can be of two kinds, lexical and rhythmic. Lexical stress focuses on the individual syllables in individual words, whereas rhythmic stress focuses on the prominence of syllables in stretches longer than an individual word, such as a sentence. In this project, we will be focusing on lexical stress. The English language consists of 14 vowel phonemes, and each phoneme can have three levels of stress (no stress, primary stress and secondary stress).

We intend to create a deep learning system that takes spoken sentences sliced at a phoneme-level as input, and classifies each phoneme into a binary-class output, where each class is an indicator of the stress level. We have recordings of non-native as well as native english speakers, with labeling at phoneme-level to train and test our model. Additionally, we have deployed our model using AWS Lambda and Fargate clusters to integrate with the existing learning app.

LITERATURE SURVEY

In [1], the authors have focused on rhythmic stress to recognize stress in whole sentences. Firstly, the system uses a forced alignment speech recognition using Hidden Markov Models (HMM) recognizer. The recognizer maps specific segments of the utterance to each vowel in the target using the phonemic transcription of the utterance. Then the prosodic and vowel quality features of these segments are extracted and normalized. Normalization is important to reduce variation due to different speakers, recording situations and utterance contexts.

A number of prosodic features like pitch, amplitude and duration have been used. The absolute value of the duration of a vowel segment is influenced by many factors other than stress, such as the intrinsic durational properties of the vowel, the speech rate of the speaker, and local fluctuations in speech rate within the utterance. Therefore in place of absolute duration of the vowel segment, a normalised duration is used, which indicates how much longer or shorter this vowel segment is than that vowel would “normally” be spoken by an “average” speaker. Amplitude changes as the vowel is pronounced and hence there can be many features capturing the change in amplitude, initial amplitude, max amplitude, etc. The most commonly used one is the RMS (root mean square) of the amplitude values across the vowel.

In [2], the authors have used a similar approach of using a HMM recognizer to do forced phoneme alignment and identify the time boundaries of the pronounced phoneme. Certain spectral and non-spectral features were extracted from the speech signal of different phonemes. The feature vector used consisted of acoustic features from targeted syllable as well as the preceding and succeeding syllables. The final features set was then fed into two different classifiers, Convolutional Neural Networks (CNN) and Deep Neural Networks (DNN) to classify the level of stress in each phoneme as primary stress, secondary stress or no stress.

Our approach is majorly influenced by the second research paper mentioned above.

DATA REVIEW

- For training, we have about 20 GB data containing about 460 hours of speech of 1100 distinct native English speakers from LibriSpeech[3]
- In addition to this, we also have OSCAAR[4] (Online Speech/Corpora Archive and Analysis Resource) data which contained about 4907 audio files, with speaker variation of 10 females and 2 males
- For testing, we have~500k recordings which are on ~5 seconds each from Blue Canoe Users, that are non-native English Speakers
- We have speech data from users belonging to different nationalities, from Spanish, British, Chinese speakers, as well as speakers from different states of the US. This enables us to work with different accents and understand the linguistic context in each of these accents.
- Initially, the modelling process was started with 3 classes Primary, Secondary and No Stress. Since very few words have secondary stressed vowels,there was a class imbalance problem. Additionally secondary stress is very similar to primary stress, hence, it was decided to model this as a binary classification problem with classes primary stress and no stress.

OVERVIEW OF FEATURES

- Mel Frequency Cepstral Coefficients(MFCCs) - help in extracting the components of the audio signal that are useful for identifying the linguistic content and discarding all the other information like background noise, emotion etc.

- Delta MFCCs and Delta-Delta MFCCs – first and second derivative of MFCCs, to capture trajectories of MFCC coefficients over time
- Formants (F0 - F4) – to capture concentration of acoustic energy around a particular frequency in the speech wave. Different formants occur at roughly 1000Hz intervals. Each formant corresponds to a resonance in the vocal tract

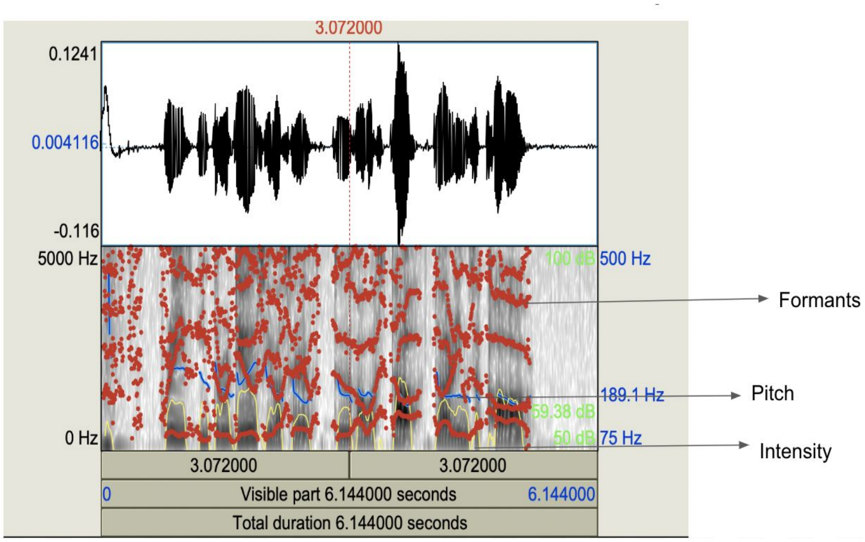


Figure 1. Acoustic Features of an audio sample

EXPLORATORY DATA ANALYSIS

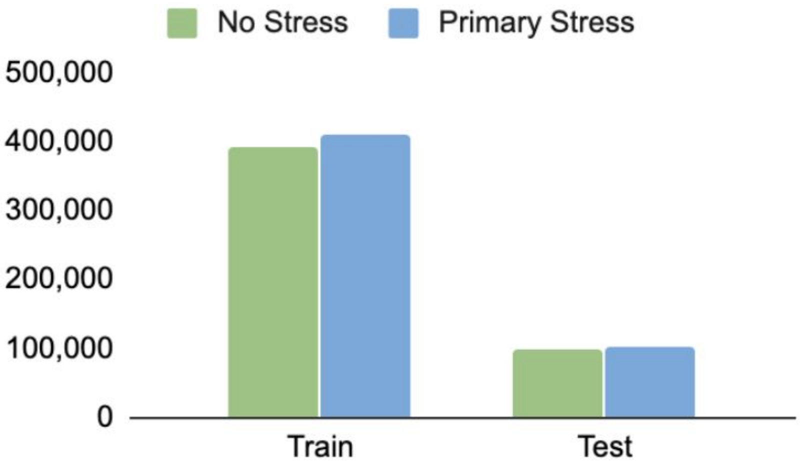


Figure 2. Data Distribution

The plot above shows the distribution of data samples across different stress levels and train and test sets. It can be observed that the classes are mostly balanced.

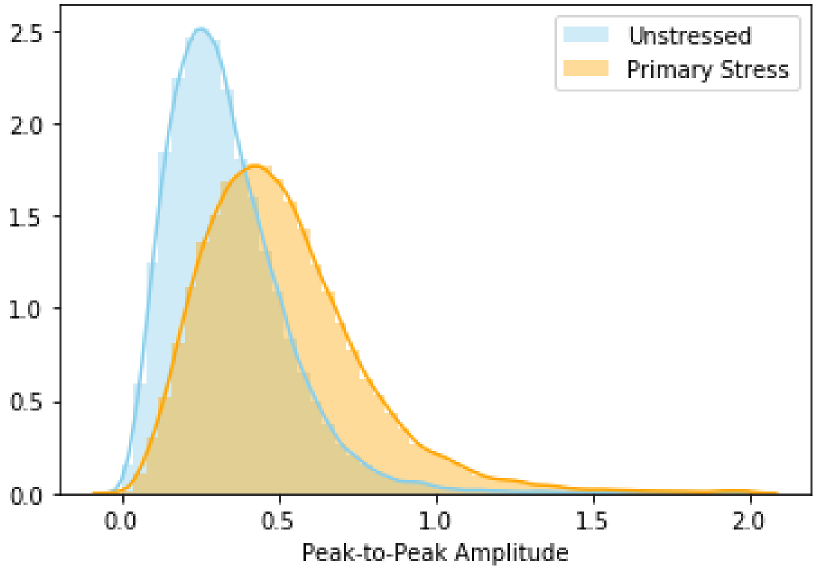


Figure 3. Data Distribution of peak-to-peak amplitude across different stress levels

The plot above shows the distribution of peak-to-peak amplitude values across different stress levels. Primary stress data shows a relative high mean in peak-to-peak amplitude value compared to Unstressed data.

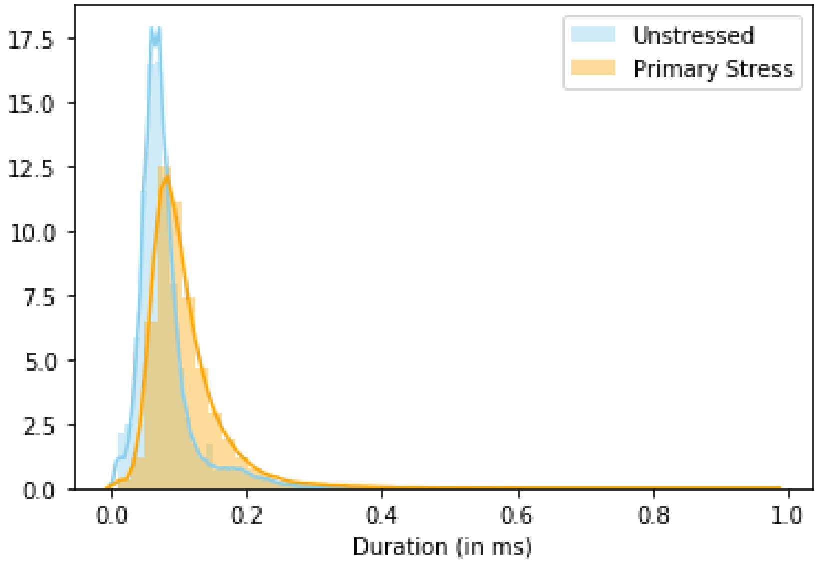


Figure 4. Data Distribution of duration across different stress levels

The plot above shows the distribution of duration values across different stress levels. It can be observed that the maximum value of duration for unstressed data is higher than primary stress data.

DATA PREPROCESSING

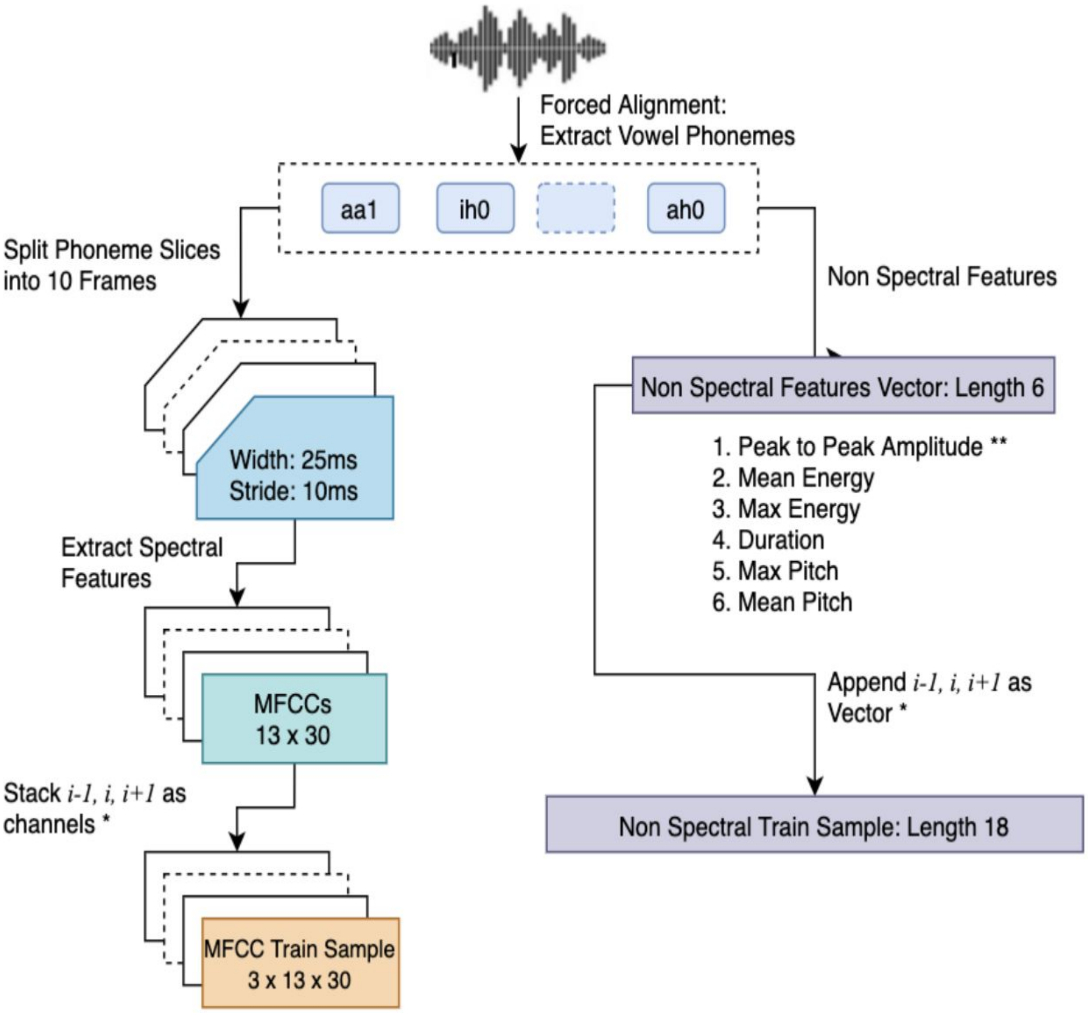


Figure 5. Data Preprocessing Pipeline

The Data Preprocessing pipeline consists of two parts, one which creates the spectral features, namely, MFCCs and another which extracts temporal features. Each individual step is described in more detail in the next section.

WORK DONE

We model the lexical stress detection use case as a binary classification problem where the inputs are features derived from each phoneme and the output is one of the two stress levels - primary stress or no stress. We are following the approach in [2] for feature extraction and model training.

Forced Alignment

We have used gentle[6] which is built on Kaldi [7], to perform forced phoneme alignment. In forced alignment, the acoustic model is given an exact transcription of what is being spoken in the speech data. The system then aligns the transcribed data with the speech data, identifying which time segments in the speech data correspond to particular phonemes in the transcription data.

Feature Extraction

After phoneme alignment on an audio file, we slice out the phoneme of interest (p_i) as well as the preceding (p_{i-1}) and succeeding phoneme (p_{i+1}). For each phoneme we extract seven temporal features given in Table 1. As the speech signal is distributed over multiple frequencies, we also use features on the Mel Scale. Each phoneme is split into ***n*** non overlapping frames of 10ms and the first 27 Mel Frequency Cepstral Coefficients (MFCC) are extracted. To adjust for variable length of phoneme the maximum number of frames is set to a constant ***N***. If ***n*** > ***N***, the middle ***N*** frames are used and if ***n*** < ***N***, zero padding is used. The MFCCs are arranged as a three dimensional matrix of shape N x 27 x 3 where the three channels are the features from the three phonemes. We also have the temporal features which are arranged sequentially into a vector of length 6 x 3 = 18.

Feature	Description
f_1	Peak-to-peak amplitude over syllable nucleus
f_2	Mean energy over syllable nucleus
f_3	Maximum energy over syllable nucleus
f_4	Nucleus duration
f_5	Maximum pitch over syllable nucleus
f_6	Mean pitch over syllable nucleus

Table 1. Non-Spectral acoustic features

Model Training

The architecture, shown in Figure 5, consists of two different neural networks whose outputs are combined before passing onto the softmax layer. The first part is a CNN which takes the MFCCs as input and outputs a vector. The second part is a deep neural network which takes in the vector of non-spectral acoustic features and outputs a vector. The two vectors are concatenated and passed onto a fully connected (FC) layer followed by a softmax layer. Two different networks are required because features f1 to f6 are non-spectral and cannot be treated similarly. This architecture is majorly inspired from [2].

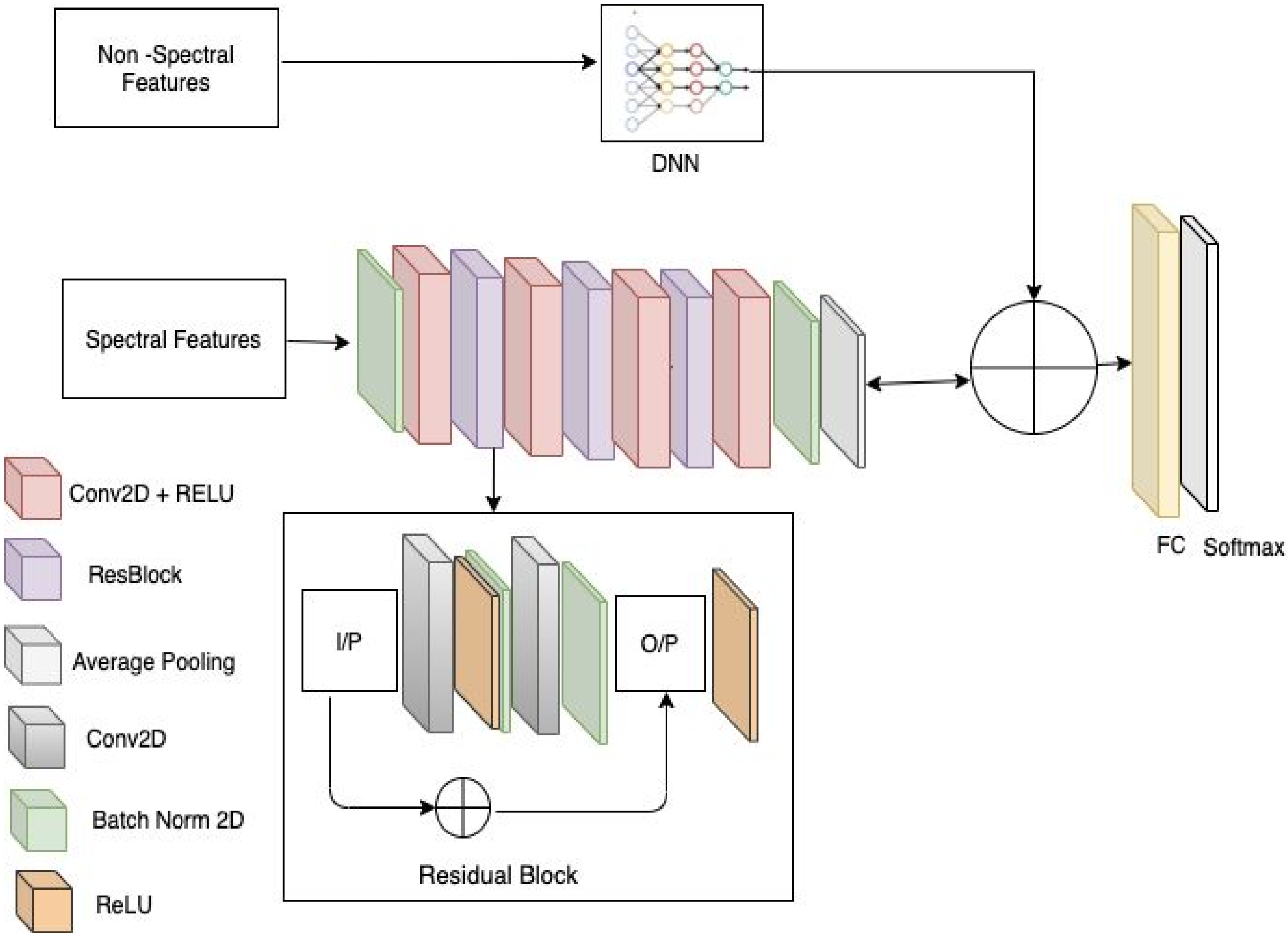


Figure 6. Model Training Network Architecture

Model Experiments

Stopword Removal

An initial model gave an accuracy of about 89%. On further investigation it was found that, most of the phonemes that were misclassified belonged to high-frequency common words like *the, at, to* etc. For the purpose of our application, it is more important to ensure that the model learns to classify the lexical stress level in other words than the high-frequency stopwords. Hence, learning from this feedback, an additional preprocessing step was added where around 80 stop words were removed from the data and the model was retrained. This increased the accuracy to about 96%.

Cyclical Learning Rate

Learning rate is one of the key parameters to tune while hyperparameter tuning. Cyclical Learning Rate[7] is a method by which the learning rate can be fluctuated between an upper bound and lower bound in a cyclical fashion. The key idea of why this might work is because this method ensures that the model jumps to another local minima in regular intervals. This method is also comparatively less computationally intensive than Adam Optimization technique. This did not enhance the performance of the model.

Data Augmentation

In order to introduce more speaker variation, additional audio transcripts were sourced from OSCAAR[3]. A total of 4907 audio files were added which added a speaker variation of 10 females and 2 males.

Phoneme Alignment

During the initial phase of the project, PocketSphinx was used for phoneme alignment. This caused a lot of failures and consequently resulted in data loss. This strategy was replaced by using Gentle[8] which caused more robust word alignments and hence preserved most of the data.

Architectural Changes

More layers were added post the concatenation of temporal and spectral features. This would ensure capture of relationships between these two different kinds of features.

Hyperparameter Tuning

- Reduced the learning rate during the training phase, when train loss increased (See Figure 9)
- After concatenating the features from the Deep Neural Network and Convolutional Neural Network, when the learning rate was kept small, the train accuracy was increasing and test accuracy decreasing, indicating overfitting. Hence, dropout regularization was introduced
- Batch Size was chosen to be 512 as proposed in the reference paper

Application Integration

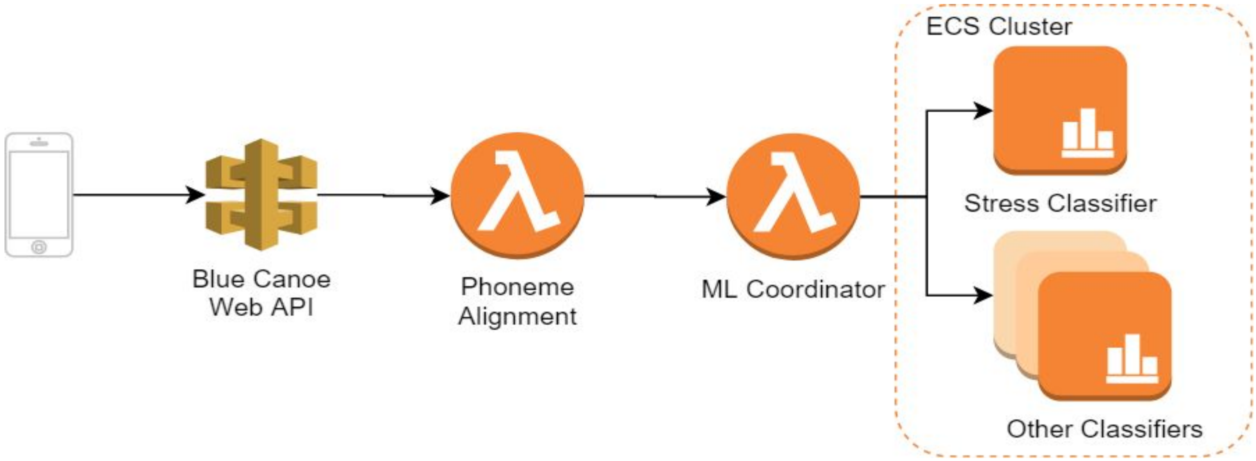


Figure 7. Application Integration Workflow

Blue Canoe is a Mobile App which asks users to speak predefined words or sentences and provide feedback on different aspects of pronunciation using Machine Learning classifiers. The stress classifier created, integrates with their existing infrastructure on AWS and extends their existing machine learning models.

RESULTS

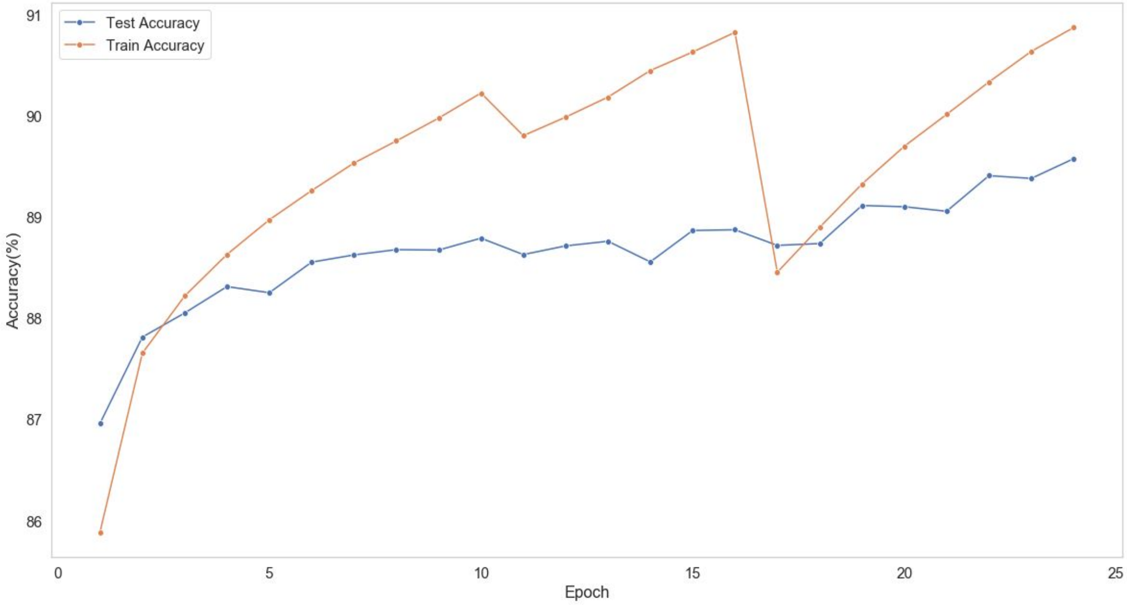
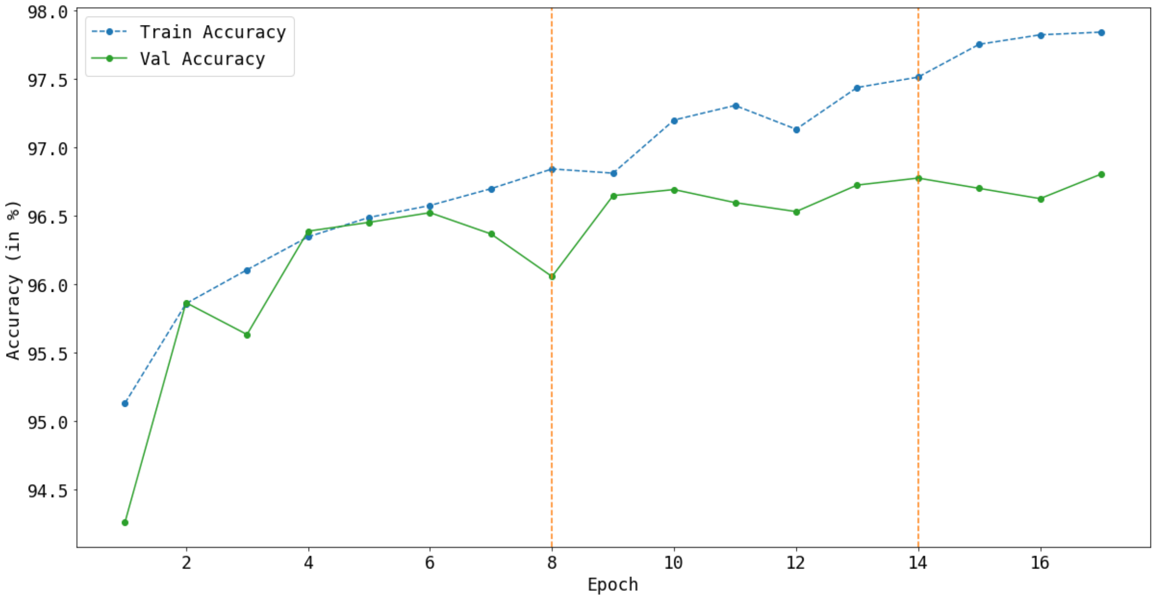


Figure 8. Train and Test Accuracy v/s epochs for an initial model



LR = 0.001 LR = 0.0005 LR = 0.0001

* LR - Learning Rate

Figure 9. Train and Test Accuracy v/s epochs

INSIGHTS

- The plot shown in Figure 8 shows the train and test accuracy over 25 epochs for an initial model, where stopwords were not removed. The maximum accuracy achieved on test set for this case was 89.58%

- After removing stopwords, the maximum accuracy achieved on the validation set was 96.80% and 97.84% on train set after 17 epochs with varying learning rates as shown in Figure 9

- The baseline accuracy without any training was 49.78% on validation set

LINK TO OUR CODE REPOSITORY

https://github.com/LexicalStressDetection/lexical-stress-detection

REFERENCES

[1] Huayang Xie, Peter Andreae, Mengjie Zhang, and Paul Warren, “Detecting stress in spoken English using decision trees and support vector machines,” in Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation. 2004, pp. 145–150, Australian Computer Society, Inc.

[2] M. Shahin, J. Epps, and B. Ahmed, “Automatic classification of lexical stress in English and Arabic languages using deep learning,” in Proceedings of Interspeech, 2016, pp. 175–179.

[3] V. Panayotov, G. Chen, D. Povey and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, QLD, 2015, pp. 5206-5210. Retrieved from http://www.openslr.org/12/

[4] Bradlow, A. R. (n.d.) ALLSSTAR: Archive of L1 and L2 Scripted and Spontaneous Transcripts And Recordings. Retrieved from https://oscaar3.ling.northwestern.edu/ALLSSTARcentral/#!/recordings

[6] https://kaldi-asr.org/doc/index.html

[7] Leslie N Smith. Cyclical learning rates for training neural networks. arXiv preprint arXiv:1506.01186v3, 2016.

[8] https://github.com/lowerquality/gentle