Assessing English accent similarity using various audio-to-embedding schemes and dimensionality reduction methods

Katya Katsy kkatsy@ucsd.edu
Mason Kellogg m1kellogg@ucsd.edu
Morgan Lafferty mlaffert@ucsd.edu
Sean Lindstrom slindstrom@ucsd.edu

1 Introduction

Our project investigates distinct English accents and how well different embedding schemes and clustering algorithms are able to capture the subtle similarities and differences between them. We test multiple methods of generating vector embeddings from audio files, including features derived from Mel Frequency Cepstral Coefficients (MFCC), from a bag-of-words approach for representing phoneme frequency within the audio files, and from FacebookAI's Wav2Vec2.0 model, which converts audio directly into embeddings. We then evaluate and compare the effectiveness of these methods by performing dimensionality reduction on the embeddings using both linear (PCA) and nonlinear (t-SNE, UMAP) techniques and visualizing the results. Our findings show that native and non-native speakers of English are most easily separated in the latent spaces while further separation of non-native English speakers is challenging. This suggests that more powerful embedding schemes may aid in accent invariant speech recognition systems, especially for non-native English speakers. We hope our experiments motivate future work in this area and provide avenues for more specialized research on speech recognition inclusion.

2 Related work

Ahamad et al. (2020) define four key requirements (variety of speakers, words vs sentences, uniformity of content, and semantic requirement) for creating a well-curated database of speech samples in non-native accents for training and testing robust automatic speech recognition (ASR) systems. Using these principles, they introduce AccentDB, a database that contains samples for 9 distinct metropolitan Indian-English accents. The authors used Mel-Frequency Cepstral Coefficient (MFCC) extraction to represent audio files as vectors. They perform dimensionality reduction and plotting (PCA, t-SNE, UMAP) to evaluate separability of accents in latent space (Figure 5). Generally, UMAP showed that the accents are similar (are valid for comparison) and t-SNE showed that they are separable (can be treated as independent classes). This paper is useful for our study because it compares different methods for dimensionality reduction. Of note is the high (40 per audio frame) dimensionality of the MFCC embeddings.

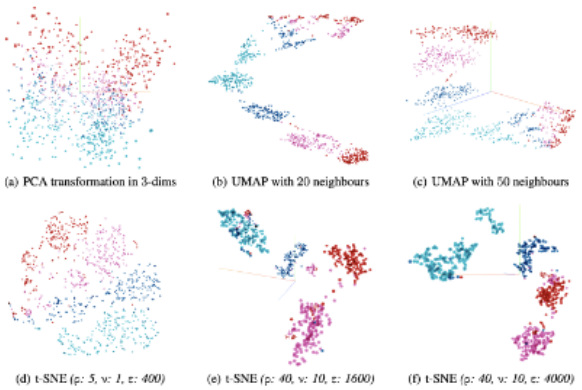

Figure 1: Projection of MFCC embeddings for 4 non-native Indian accents (Ahamad and Anand (2020)).

Nisioi (2016) discusses the connection between accent recognition and text classification. They emphasize the importance of accent classification to improve the accuracy of speech recognition. Nisioi states "the detection of different native accents can prove useful to reduce the error rate of speech recognizers" (Nisioi, 2016). In their study, they compare text and speech classifiers on a large data set of English second language speakers in various countries in Asia. To extract features from audio speech files, they use moving low pass filters to obtain 1582 features and 38 low level descriptors based on openSMILE. Some takeaways from the study are that multi-labeled classification

techniques are challenging and that multi-class approaches are more reliable and effective. Furthermore, their study shows the effectiveness and efficiency of speech classification and achieves 80-90% accuracy on speech classification of speakers' native languages with as little as 2 seconds of audio data.

Chitturi and Hansen (2008) discuss the influence of speech recognition in the technology sector. In their study, they attempt to distinguish US, UK and Australian English using a combination of audio and written data. They use MFCCs for audio classification and n-gram models and LSA classifiers on the written data. Individually, classification based on text documents outperformed classification based on audio speech data. When combined, the model's classification accuracy increased for all dialects. Their results show the strength of MFCCs in audio-speech recognition and present the unique idea of combining speech recognition with text classification. They suggest that we can use speech recognition in conjunction with larger language models to potentially increase performance of transcribing.

Valid accent classification could lead to lower error in speech recognition and more personalized responses. Shih (2017) attempts to classify four of the most common English accents: British, French, Spanish, and Mandarin. Interestingly, Shih used small dataset of 430 speech examples. In the study, MFCCs are used to extract features from the audio data and only the 13th lowest order coefficients are used. They state that "MFCCs are commonly used features in speech recognition systems because they approximate the important features that the human auditory system detects in audio signals" (Shih, 2017). A softmax regression model as well as a long short-term memory neural network are used for classification. They achieved a relatively low testing accuracy of about 52% for the LSTM. We found this result to be promising, given the small dataset size. An interesting idea presented for future work is splitting the audio into individual words and obtaining embeddings per word.

Li et al. (2021) notes that speech recognition models perform worse when tested on unseen accent speech. They discuss two common methods for building accent-robust ASR models: domain-adversarial training (DAT) and multi-task learning (MTL). They introduce accent embeddings as another method for accent-robust models. Using a database of 20 non-US-English accent speech, the authors compared the effectiveness of these three methods. An influential result is that their MTL model trained with wav2vec2.0 embeddings is very successful in learning accent invariant features of speech and improving performance on unseen accents. They also found that wav2vec embeddings have "more advantages for building accent-robust ASR when no accent labels are available for training supervised embeddings" (Li et al., 2021). This work is useful for our study, because we are also relying on Facebook AI's wav2vec for generating vector embedding of audio files. The paper also includes some hyperparameter recommendations for t-SNE dimensionality reduction on wav2vec2.0 embeddings. It is an interesting result that their model generalized better to unseen accents when no accent labels were available during training. This helped inform our choice of training method (phoneme labels). This paper also supports the goal of our study: to add to the body of work on inclusive speech recognition.

3 Dataset

We used the "Speech Accent Archive" dataset collected by researchers at George Mason University and made available for download on Kaggle. This dataset contains 2,140 .mp3 audio samples, each from a distinct speaker reading the following passage in English:

> *Please call Stella. Ask her to bring these things with her from the store: Six spoons of fresh snow peas, five thick slabs of blue cheese, and maybe a snack for her brother Bob. We also need a small plastic snake and a big toy frog for the kids. She can scoop these things into three red bags, and we will go meet her Wednesday at the train station."*

The speakers in the dataset are from diverse backgrounds, representing 214 different native languages across 177 countries. There is an about even split between males (1123) and females (1049) with a median speaker age of 28 (IQR 22-41).

For our project, we reduced this dataset to include all speakers from the 6 most represented native languages: English, Spanish, Arabic, Mandarin, French, and Korean. We do so because (1)

clustering results would be very difficult to interpret had we included all 214 different native language classes and (2) some languages have only a few samples in our dataset which could result in erroneous and noisy clusters. After excluding all other samples, we have 579 English, 162 Spanish, 102 Arabic, 65 Mandarin, 63 French, and 62 Korean speakers.

The global scale of our dataset and the fact that all recordings are done in English lends itself well to our objective of assessing accent similarity around the world. However, this remains a difficult task as the differences in accents are likely much more subtle than the differences between each speaker's native language. In the following sections we describe our efforts to evaluate whether vector embedding schemes are able to capture these differences in a way that can be distinguished by dimensionality reduction techniques.

3.1 Data Processing

3.1.1 Pre-processing

We first converted the audio clips from .mp3 files to .wav files in order to generate the embeddings.

3.1.2 Post-processing

After generating our vector embeddings using MFCCs and Wav2Vec2.0, more processing was required prior to clustering as the resultant feature vector embeddings were of different lengths. This can be attributed to the varying speaking speeds and cadences across our dataset when each person was reading the given passage. To manage such differences, we either zero-padded every vector to be the same length as the maximum feature vector length present or trimmed each embedding to the average length for MFCC and Wave2Vec2.0, respectively. We chose not to pad our Wav2Vec2.0 like we did MFCC due to extremely large embedding sizes and computational constraints. No post-processing was required for our phoneme-based embeddings as a Bag-of-Words approach was utilized so all the vectors were the same size.

4 Baselines

First, we developed a simple baseline on which to evaluate more advanced subsequent models. We generated 40-dimension MFCC embeddings for all speakers. For our baseline, we used a 3-feature representation for each speaker: length of embedding vector, mean value of embedding vector, and standard deviation of embedding vector. Based on the definition of MFCC above (and discussed in the literature review), we are confident that the length of the embedding is a measure of speaking speed for each speaker. We believe that the mean and standard deviations of the coefficient values reflect a speaker's average speaking magnitude and speech magnitude fluctuation, respectively.

In our more advanced models discussed in the next section, we clustered by first performing dimensionality-reduction techniques which determine latent features that capture the most variance. However, for our baseline we leave these summary statistic features as-is, in the hopes to retain some interpretability when plotting. Figure 3 is a chart that averages these three features within-class; note the differences between classes. Also note that the second through fourth row for each language represents its centroid in this table. We plotted these three features for speakers from the classes of interest defined earlier. Figure 2a emphasises that we did not see a significant spread between mean MFCC value and sd of MFCCs. Figure 2b emphasises the spread in length of MFCC embedding vectors. The interpretability of this baseline model was of particular interest to us. We see visually that the majority of the variance is explained by the length of MFCC embedding vector feature, which indicates that the speed that a speaker reads the prompt is a decent (as a baseline) method of clustering speech. Specifically, we see that most native English speakers read the prompt in less time than other speakers. We also see less within-class variation in this metric for native English speakers. The opposite is true for non-native English speakers.

To set quantitative baselines for how distinct the language clusters are, we measured the distance of each non-native English speaker's cluster to the native English language speakers centroid, shown in the last column of Figure 3.

As an aside, we also evaluated whether 'silence time' in the .wav files was a good separator of language classes. To measure this metric, we calculated the number of entries in the MFCC embedding matrix which were less than 5 in magnitude for each speaker. The resulting plot is shown in Figure 2c. It appears that 'silence time' is a proxy for the speaking time feature, given that there appears to be a linear relationship. This makes intu-

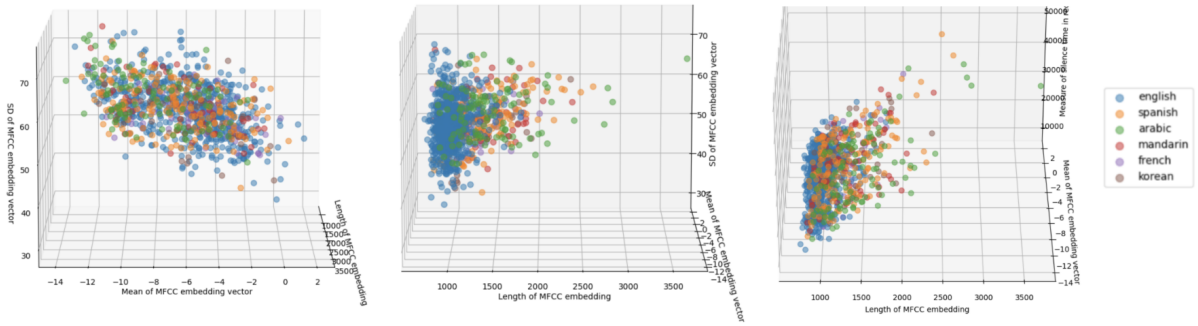

Figure 2: From left to right: 2a. Baseline features colored by native language of speaker emphasizing similarity in mean and standard deviation of MFCC embedding between classes. 2b. Baseline features colored by native language of speaker emphasizing variation in length of MFCC embeddings between classes. 2c. Baseline features colored by native language of speaker emphasizing linear relationship between length of MFCC embedding and 'silence' time (number of entries in MFCC embedding less than 5 in absolute magnitude) in recording.

Native language	Average MFCC embedding length	Average MFCC coefficient mean	Average MFCC coefficient standard deviation from mean	Euclidean distance from English centroid
English	989	-5.99	53.2	0
Arabic	1463	-8.28	57.6	474
Mandarin	1351	-6.41	55.2	362
Spanish	1288	-6.72	54.7	299
Korean	1331	-6.51	53.3	342
French	1175	-5.79	52.3	186

Figure 3: Table describing baseline feature centroids. Last column is Euclidean distance of each centroid from naive english speaking class centroid.

itive sense, because if the speaker is not speaking, their total time to record the passage will likely be higher. Although this isn't surprising, it does show that the baseline is performing as we would expect, which is encouraging.

5 Your approach

5.1 Conceptual Approach

We attempt to distinguish varying accents by creating vector embeddings from audio clips of people reading the same passage in English and performing dimensionality reduction on these to cluster each by the native language of the speaker. To do so, we experiment with different embedding schemes including MFCCs, Wav2Vec2.0 and Phonemes followed by both linear (PCA) and non-linear clustering methods (t-SNE, UMAP) and compare results to a simple baseline.

5.1.1 Embedding Schemes

Phonemes: Phonemes were derived from the audio files using Allosaurus - a pretrained universal phone recognizer that converts waveforms into strings of IPA phonemes. The Allosaurus python library is based on the work of (Xinjian Li, 2020).

The resulting string of IPA phonemes of an audio file is then encoded using a bag-of-words approach: we first obtain a list of all the IPA phonemes that appear within our audio files, and then we proceed to count the number of times each phoneme appears within a recording and use these counts as a feature vector.

We counted 57 unique IPA phonemes, so we created feature vector of length 58 - the unique phonemes plus a symbol representing periods of silence in the recordings, which is generated automatically by Allosaurus.

MFCCs: Mel Frequency Cepstral Coefficients (MFCCs) is a widely-used form of data compression in automatic speech recognition that allows audio to be converted to feature embeddings. The mel scale is a logarithmic transformation of a signal's frequency that converts hertz into the *mel* unit. *Mels* are defined in such a way that sounds of equal distance on the Mel Scale are perceived to be of equal distance to humans. The mel-frequency spectrum is a representation of the short-term power spectrum of a sound based on the mel scale. MFCCs are the values that represent the mel-frequency spectrum: each MFCC value represents how similar the mel-frequency spectrum is to cosine shapes created from different frequencies.

For our project, we generate 40-dimension MFCC embeddings for each audio clip.

Wav2Vec2.0: Wav2Vec 2.0 is a transformer-based neural network model created by Facebook that automatically learns discrete speech units and projects recorded speech into a learned embedding space (Alexei Baevski, 2020). We used the pre-trained Wav2Vec2.0 from PyTorch, specifically, the base model pre-trained on 960 hours of unlabeled audio from LibriSpeech dataset.

The result is 12 matrices of feature outputs per one waveform audio, corresponding to the output of each transformer. In order to get a single feature array, we average these matrices, and then flatten the resulting feature array into a feature vector.

5.1.2 Clustering Methods

PCA: Principal component analysis, or PCA, is a popular dimensionality reduction technique that linearly transforms data into a new coordinate system in which the variation in the data can be described with fewer dimensions. We believed that PCA would be a good method to identify accents that are strongly dissimilar even though it might not be able to capture the more subtle differences between accents due to its linearity.

t-SNE: The t-distributed stochastic neighbor embedding (t-SNE) is a nonlinear dimensionality reduction technique for embedding high dimensional data in a lower dimensional latent space. The main advantage of t-SNE is its ability to preserve local structure, where points that are close to one another in high-dimensional space tend to remain close in the low-dimensional space.

UMAP: Uniform Manifold Approximation and Projection (UMAP) works similarly to t-SNE in that it uses a graph layout to approximate data into a lower dimensional space. However, UMAP tends to be more efficient than t-SNE and also can be more effective by doing a better job at maintaining some of the global structure of the higher dimension space (McInnes et al., 2020).

5.2 Implementation

We successfully implemented all embeddings schemes and dimensionality reduction techniques mentioned above. All our code can be found on our shared Github. Code is grouped into folders by embedding scheme which are labeled accordingly (i.e. MFCC, Phonemes, Wave2Vec2, Baseline).

We chose to reduce the dimensions of the feature vectors to 3 so that the results could be easily visualized in 3D for analysis.

5.3 Compute

The majority of processing was done using Google Colab's cpu with the exception of generating Word2Vec2.0 embeddings which proved to be computationally expensive and required use of Colab's available gpu. We also found subscribing to Google Colab Pro to be beneficial.

5.4 Runtime

5.4.1 Generating Embeddings

Creating the MFCC embeddings was relatively quick and took just over 2 minutes in total. In comparison, generating the phoneme embeddings took approximately 3-4 hours. Wav2Vec2.0 embeddings were costliest to produce: generating embeddings took a total of 7-8 hours and loading those embeddings back into the colab environment took 3 minutes.

5.4.2 Running Clustering Algorithms

For MFCCs, the most computationally expensive component was running hyperparameter tuning for the t-SNE and UMAP algorithms as generating one t-SNE plot took between 40 sec and 2 min depending on the number of iterations specified while each UMAP plot required about 1 min.

For the Phoneme embeddings, running each clustering algorithm only took about 1min on average but a couple minutes to run through many different hyperparameters to find the best results.

For Wav2Vec2.0, we used the High-RAM Google Colab Pro environment for clustering. In this setting, each clustering method took about 5 min to run since the embeddings were so large, around 1 million features parameters per audio file.

5.5 Results

5.5.1 3D Clustering

As seen in Figure 4, distinct clusters for each accent group were difficult to achieve. While most pairs of embedding schemes and dimensionality reduction techniques (with the exception of Wave2Vec2.0 + t-SNE) were able to separate those whose native language was English from those whose was not, further distinction between non-native English speakers was limited. We see handfuls of each non-English accent class cluster to-

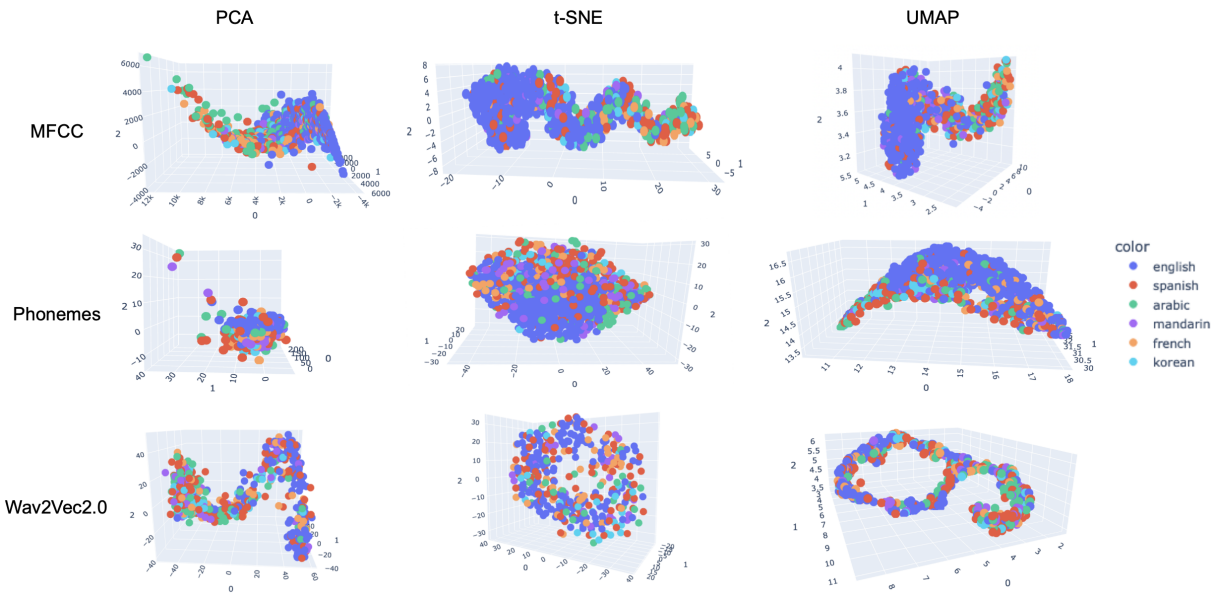

Figure 4: 3D cluster plots for each pair of embedding scheme and dimensionality reduction technique. Each audio clip is represented as a point that is colored according the native language of the speaker.

gether, however, no recognizable and dense clusters are represented.

The MFCC embedding technique appears to have the best clustering from a visual perspective. It seems to have the most separation between English and the other languages and slightly more distinct clusters. On the other hand, the Wave2Vec2.0 audio embeddings seem to produce the worst clusters as for t-SNE and UMAP, even the native English speaker audio samples do not separate well. In regards to the different clustering techniques, PCA seems to do the best job of separating the English and non-English native accents. This could suggest that the task of clustering and separating accents does not require non-linear techniques.

5.5.2 Distances

One method we use to analyze the clustering results is to compare the distances between the centroid of each language cluster to the English centroid. We choose English as our reference point because each audio clip is recorded in English and we take this to be the "standard" accent. We expect these distances to reflect the accepted lexical distance relationships between different languages.

The distances of the five non-English native language cluster centroids to the English centroid for each embedding and clustering type can be seen in Figure 5. The table does not display actual distance numbers as they are not strictly relevant

Ranking of Distances to English Centroid						
Embedding	Clustering	1	2	3	4	5
MFCC	PCA	French	Spanish	Korean	Mandarin	Arabic
	tSNE	French	Spanish	Korean	Mandarin	Arabic
	UMAP	French	Spanish	Korean	Mandarin	Arabic
Phonemes	PCA	French	Mandarin	Spanish	Korean	Arabic
	tSNE	French	Mandarin	Spanish	Korean	Arabic
	UMAP	French	Mandarin	Spanish	Korean	Arabic
Wav2Vec2	PCA	French	Spanish	Korean	Mandarin	Arabic
	tSNE	Spanish	French	Arabic	Mandarin	Korean
	UMAP	French	Spanish	Korean	Mandarin	Arabic

Figure 5: Ranking of closest (1) to furthest (5) distance of specified language centroid to the English centroid.

and are scaled differently; instead, we present a ranking of which languages centroids are closest to the English cluster centroid. The MFCC audio embedding technique resulted in distances that most reflect what we expected. The French cluster centroid was the closest, followed by Spanish, Korean, Mandarin, and finally Arabic. Each clustering technique resulted in the same relative distances using MFCCs as well which could be an indicator that MFCCs are a good embedding technique. Looking at Phonemes, Mandarin was surprisingly the second closest language cluster to English. This could indicate that Mandarin and English actually share more similar sounds and phonemes compared to the other languages. Similarly to MFCC embeddings, phonemes also had the same relative cluster distances across all clus-

Variance Explained				
Embedding	Clustering	Comp 1	Comp 2	Comp 3
MFCC	PCA	0.264	0.069	0.040
	t-SNE	0.184	0.001	0.016
	UMAP	0.185	0.097	0.012
Phonemes	PCA	0.567	0.075	0.037
	t-SNE	0.35	0.025	0.039
	UMAP	0.344	0.04	0.056
Wav2Vec2	PCA	0.044	0.016	0.014
	t-SNE	0.028	0.025	0.010
	UMAP	0.034	0.023	0.013

Table 1: Variance explained by the first 3 components of our clustering algorithms.

tering techniques. Wav2Vec2.0 however, had different centroid distances when using t-SNE compared to PCA and UMAP. Wav2Vec2.0 using PCA and UMAP resulted in the same distances as the MFCC embeddings, which is expected as these techniques result in more dense embeddings based on the actual audio. The clustering using t-SNE on the Wav2Vec2.0 embeddings, however, result in unexpected results with Spanish being closer to English than French and Arabic being closer than both Mandarin and Korean.

These centroid distance comparisons suggest that the MFCC embeddings work well as they result in consistent expected results whereas the Wav2Vec2.0 and Phoneme distances do not always consist with the expectations. In terms of clustering techniques, PCA and UMAP are the only two that are always consistent with each other and the expectations.

6 Error analysis

6.1 Explained Variance

For all the embedding methods, we observed PCA produced the highest total explained variance, with t-SNE and UMAP capturing significantly less variance. A potential explanation for this is that the goal of PCA is to capture maximal variance, while t-SNE and UMAP have no such optimization scheme, and, instead, focus on capturing local and global relationships when identifying components.

Additionally, higher explained variance does not equate to better clustering performance. For example, the highest explained variance was for phoneme embeddings and PCA clustering, a total of 68%, however, this approach led to the misclassification of Mandarin being more similar than Spanish, which was not made by any of the other approaches.

An explanation for this may be that some embedding schemes are easier to cluster: phoneme embedding are easier to cluster since the feature vectors are smaller and simpler, so it is easier to break down differences into highest-variance components, but it does not guarantee that the resulting clusters will capture as much information as those created from more complicated embeddings.

6.2 Outliers

We performed a manual error analysis of audio file inputs that lie far from their native language centroids.

This analysis is straightforward for our baseline model because we developed intuitive baseline features, as discussed in Section 5. Note that in Figure 5, the points that lie separated from their native language clusters do so because of an extreme value in length of MFCC embeddings. For example, we see a distinct green cluster of native Arabic speakers in the range of 1000-1500 MFCC embedding length. Points outside this range (especially the recording at MFCC embedding length of 3700) are far from the centroid and thus more likely to be clustered with the native Spanish or Mandarin speakers classes. Discussed in Section 5, we note that the MFCC embedding length feature of our baseline is analogous to amount of time for the speaker to recite the prompt. It is also linearly related to the amount of time the speaker is paused (not speaking). Thus, our baseline fails on inputs where the speaker in a native language class takes significantly longer to recite the prompt than other speakers in their class, indicating that special care should be given to ensure that the speaker is not interrupted during their reciting. We also suggest allowing the speaker to read the prompt before reciting, so that their recording may be as smooth as possible, without losing accent-specific pauses. Any recordings where there is a time interruption should be re-taped.

We also performed a manual error analysis on our 3D cluster plots for our three embedding schemes. Note that in Figure 8, the nonlinear dimensionality reduction techniques (t-SNE, UMAP) tend to reduce the occurrence of extreme outliers. Therefore, we focus our outlier analysis to the PCA column. We are confident the linear features may better illuminate points dissimilar

to their native language classes. We focus our attention to the three points (Spanish, Arabic, Mandarin) separated from the majority of points in the PCA phonemes plot. Upon further inspection, we see that these recordings have a different phoneme distribution than their classes. Interestingly, we see that the age of speakers for these points are higher (above 40) than the average speaker in the dataset. Thus, our phoneme model fails on inputs where the speaker's age affects the phoneme distribution. If we inspect these points in the MFCC and Wav2Vec2.0 embeddings, we also see that they are far from their native language clusters. For MFCC, their embedding length is higher (corresponding to a longer time to recite the prompt). For Wav2Vec2.0, the recordings for these three speakers are relatively far from their centroid clusters, however the effect is less pronounced and the root cause is not well understood. This manual analysis informs our model choice in a couple ways. First, age of speaker may be more important than initially hypothesized. This may be affected by the age at which the person began to learn English ('age onset' in our dataset). Second, a combination of embeddings may be the best method for clustering accents because each method is individually sensitive to attributes of a speaker (such as age (phonemes) and speaking speed (MFCC)). A combined model may remedy these sensitivities.

7 Contributions of group members

- Katya: phoneme + Wav2Vec2.0 embeddings + post-processing, Wav2Vec2.0 clustering, embedding schemes, error analysis.
- Mason: baseline experiments, error analysis, introduction, related works.
- Morgan: data pre-processing, MFCC post-processing + embeddings + clustering, introduction, results, conclusion.
- Sean: phoneme clustering + analysis, related works, results, conclusion

8 Conclusion

Our goal of distinguishing and clustering accents proved to be an incredibly ambitious task. Originally, we had hoped our clustering visualizations would illuminate strong similarities/differences between different English accents and potentially even map to the geographical locations corresponding to the native languages of our speakers. However, despite extensive experimentation with both different audio-to-embedding schemes and dimensionality reduction techniques, we were surprised to achieve only minimal separation of accents, especially with dense feature embeddings like Wav2Vec2.0. That said, through such challenges we gained a new appreciation for the difficulty of building robust accent-invariant speech recognition systems capable of understanding the differences between accents which are much more subtle than those between languages. Additionally, we believe that our results emphasize the need for such systems as the consistent clustering of native English speakers away from non-native speakers highlights the strong differences between these populations and the importance of having universally accessible tools that are not biased towards certain speaking groups.

If we were to continue this project in the future, we would like to dedicate more time towards processing the audio files before and after generating embeddings to figure out what differentiates different accents the best; this could include generating embeddings per word or combining phonemes and a different audio embedding like MFCCs. Furthermore, we would like to acquire more samples for non-native English speakers as there is a large disparity in our dataset. Finally, we would like to explore classification of accents given the audio clips recorded in English.

9 Implementation Resources

The following is a list of resources we consulted when coding our implementation.

- Comparative Audio Analysis with Wavenet, MFCCs, UMAP, t-SNE, and PCA
- UMAP Variance Explained
- Speech Recognition Pipeline Tutorial
- Allosaurus Documentation

References

Ahamad, A., Anand, A., and Bhargava, P. (2020). Accentdb: A database of non-native english accents to assist neural speech recognition.

Alexei Baevski, Henry Zhou, A. M. M. A. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations.

Chitturi, R. and Hansen, J. H. L. (2008). Dialect classification for online podcasts fusing acoustic and language based structural and semantic information.

Li, J., Manohar, V., Chitkara, P., Tjandra, A., Picheny, M., Zhang, F., Zhang, X., and Saraf, Y. (2021). Accent-robust automatic speech recognition using supervised and unsupervised wav2vec embeddings.

McInnes, L., Healy, J., and Melville, J. (2020). Umap: Uniform manifold approximation and projection for dimension reduction.

Nisioi, S. (2016). Comparing speech and text classification on ICNALE. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3402–3406, Portorož, Slovenia. European Language Resources Association (ELRA).

Shih, C. (2017). Speech accent classification.

Xinjian Li, Siddharth Dalmia, J. L. (2020). Universal phone recognition with a multilingual allophone system.