Accent Recognition with the Co-Training Algorithm

Siddi Avinash Chenmilla, Spencer Riggins
schenmil@gmu.edu, sriggins2@gmu.edu

Introduction

In recent years, voice recognition technology has become prevalent in daily life. Products like the Amazon Echo and Google Home have been utilized to simplify many tasks from ordering groceries, to controlling smart home products. However, one major setback of these technologies is their ability to understand a wide variety of spoken accents.
Since the Fall of 1998, the linguistics department at George Mason University has been compiling an archive of voice clips, with the intent of distinguishing the differences between the accents of people from different regions[1]. The clips last approximately 30 seconds each, and contain speakers in a quiet environment, speaking the sentence, "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." This sentence allows speakers to sound out a large number of phonemes, which can be used to compare varying accents. Each speech sample is labeled with the speaker's age, country of origin, languages spoken, and other information about the speaker's familiarity with the English language.
The purpose of this project is to explore and examine different features, and algorithmic techniques for the purpose of voice accent classification. We are trying to see if a good accent classifier can be built when starting with only a small amount of labeled data. We will use to co-training algorithm to accomplish this. We will determine if performing the co-training algorithm with a small amount of data can approach the accuracy rate given by training classifiers on a large amount of labeled data.

1 Background

A number of studies have been conducted in the field of accent recognition, with the intent of increasing the robustness of voice recognition software. Most of these studies have been predicated on the extraction of Mel Frequency Cepstrum Coefficients (MFCC), an extremely popular feature in audio classification, which is derived from the Fast Fourier Transform on a window (chunk) of the sound wave data. Essentially, this operation creates a spectral image from a sound wave. Many of these studies have also used neural networks, among other classifiers, for learning on this extracted audio data. Leon Mak An Sheng and Mok Wei Xiong Edmund in their paper Deep Learning Approach to Accent Classification used MFCC and trained on CNN classifier[2]. The MFCC, however, is not the only feature that can be extracted from audio wave data, and little work has been done to investigate using other features to train accent classifiers. In addition to MFCC, we will also employ Chromagram, or Chroma, features in an attempt to bolster the accuracy of our accent classifiers.

2 Approach

Since a great deal of study regarding this topic has gone into training classifiers on MFCC, it is significant to explore how well the Chroma feature is suited to the problem of accent

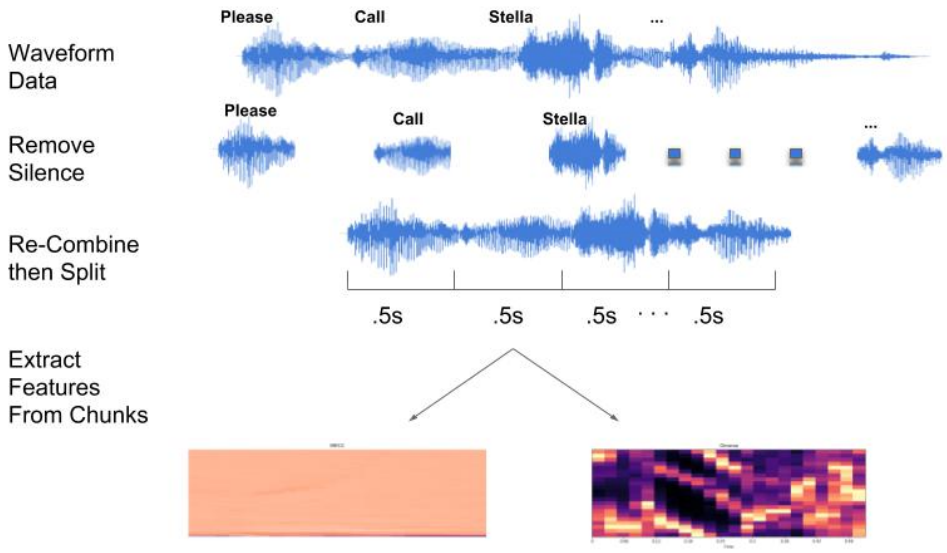

Figure 1: Feature Extraction Process

recognition. We attempt to determine whether MFCC, Chroma, or a combination of these two features is best for training accent classifiers based on neural networks, particularly when only a small amount of labeled data, and a large amount of unlabeled data is available to train the classifiers.[3] We will also examine the results achieved by co-training in an attempt to optimize the algorithm for greater classification accuracy.

3 Experimental Design

3.1 Pre-Processing and Feature Extraction

We pulled our data from GMU speech archive. The speech accent archive uniformly presents a large set of speech samples from a variety of language backgrounds. As mentioned, the Speech Accent Archive has compiled over 2,700 audio clips of individuals from varying countries of origin, speaking the same phrase in English. We took around 150 MP3 files for each class(India, North America and China). We then converted them into .WAV format using the Python library PyDub, which provides functionality for manipulating audio data. We used PyDub to remove the silences from the WAV files and then split the resulting data into 0.5 second long "chunks". We found that this gave the closest approximation to splitting the WAV files by word, which was not possible, since we needed our chunks to be of uniform length. After completing the previously described chunk-creation process we were left with, 7,521 samples for India, 6,356 samples for North America and 9,630 samples for China. Then we used another audio manipulation library called Librosa to extract MFCC and Chroma features for every chunk. After the feature extraction process, we are left with 23,507 MFCC images, and 23,507 Chroma images, one for each 0.5 second WAV chunk. Both the MFCC and Chroma features were created with dimensionality of 22 x 22.

3.2 Training on CNN

From the features we extracted in the pre-processing step, we converted the MFCC and Chroma images into two-dimensional Numpy arrays using the Numpy Python library, then passed them as input for the training the CNN. We used Keras, which is an open source neural network library in Python that uses TensorFlow as a backend for building our CNN's. K-fold(5-fold) cross validation was performed for three different models of CNN on both the features. We did this to find out the best classifier for each feature to use later in the co-training algorithm. We divided the data into 5 folds by random sampling, each fold with around 4700 samples i.e 20 percent of the data. Therefore 80 percent of the data is used for training and rest 20 percent for testing. The first model of the CNN has two convolution layers with a kernel of size of 3 x 3 and relu activation function with stride of 1. The first convolution layer has 32 output filters and the second convolution layer has 64 output filters. It is then followed by max pooling layer of pool size 2 x 2 and a dropout with 0.25 rate. After that we added a dense layer with relu activation function and a dropout with 0.5 rate. It has 128 output units. The last layer is the dense layer with softmax activation function. The number of output units is equal to the number of classes. The second model of CNN follows the same as the first models but without any dropout regularization. The third model is the same as the first one, except with an increased kernel of size 5 x 5.
We configured each model with the categorical cross entropy function as the loss function and Adadelta as the optimizer with accuracy as the metric. Then we fit the model with batch size of 100 and ran it for 10 epochs.

3.3 Co-Training Algorithm

We chose the two most successful classifiers from k-folds, one for MFCC features and one for Chroma features, to be used as the classifiers in the co-training algorithm. The co-training algorithm is a semi-supervised learning algorithm that utilizes two complementary views of the data, MFCC and Chroma in our case, to train classifiers when only a small amount of labeled data, and a large amount of unlabeled data is presented.
In order to simulate this, we initially sampled out only 1,000 samples as our training set. That is, our labeled data initially contains 1,000 MFCC features, and 1,000 WAV features, which correspond to 1,000 of the 0.5 second audio chunks, each labeled with their corresponding class label. Similarly before training, we also prepare 500 samples as a test set. Finally, we sample 1,000 samples of unlabeled data to be evaluated by each classifier at the end of each co-training iteration. The rest of the samples remained as the large pool of unlabeled data. The co-training algorithm is an iterative process, which seeks to train each classifier on the pool of labeled data, and and then predict on the small pool of 1,000 unlabeled samples.
During each iteration, each classifier is allowed to select a given number of it's most confident predictions of each class to be taken from the sampled pool of unlabeled data, and added to the sampled labeled data with their predicted label as training data for the next iteration. In our implementation, we allow each classifier to choose it's 10 most confident samples for each class i.e. each classifier is allowed to label 10 samples of each class, for a total of 40 samples (40 MFCC features, and 40 Chroma features corresponding to 40 0.5 second audio chunks) taken from the unlabeled data and placed into the labeled data per iteration when training on two classes (60 samples when training on 3 classes). Notice that if the MFCC classifier chooses to add a sample to the labeled data, then both the MFCC and Chroma feature corresponding to that sample are added to the labeled data, and likewise for the Chroma classifier. At the end of each iteration, the small pool of unlabeled samples is replenished by the large unlabeled data pool, so that the unlabeled data pool contains 1,000 samples at the beginning of each iteration.

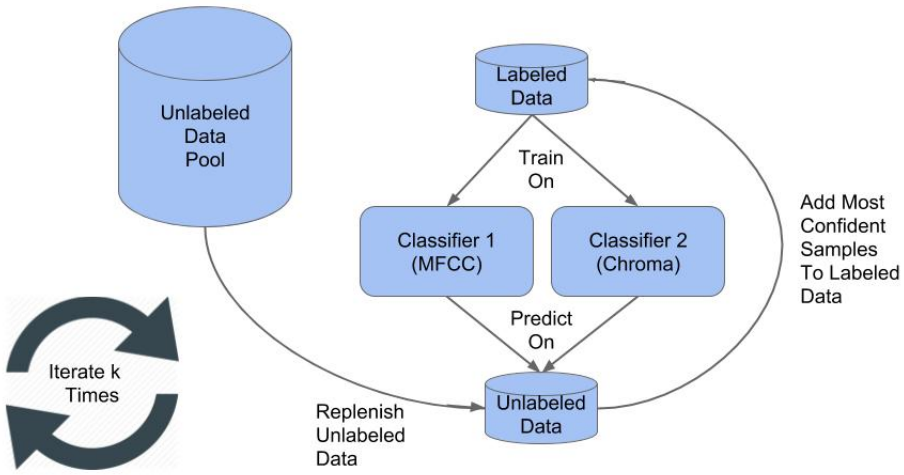

Figure 2: Co-Training Algorithm

At the end of each iteration, we test each classifier on our test set of 500 samples to find the accuracies of each classifier. We allow this process to iterate 100 times. We implemented the co-training algorithm in the manner described above two times, once for 2 classes, India and North America, and once for 3 classes, India, North America, and China.
In addition to the co-training implementation, we also trained each model on all but a test set of 500 samples (23,007 samples for 3 classes, and 13,377 samples for 2 classes) in order to show how accurate each classifier can be when trained on an ample amount of data. These accuracies will be compared to the accuracies found from the co-training algorithm to determine if co-training allows the classifiers to approach their optimum possible accuracies.

4 Experimental Results

After training the three models with 5-fold cross validation on the two features we got the following results. We chose the model 1, CNN with dropout regularization, for MFCC features and model 3, CNN with increased kernel size, for Chroma features for the co-training algorithm, as they have the highest average accuracy when compared to other models.[Figure 3]
We then ran the co-training algorithm for 2 class case[Figure 4] and 3 class[Figure 5] case and attained the following results. The blue line corresponds to the accuracy attained by the MFCC classifier at the end of each co-training iteration, and the orange line corresponds to the accuracy attained by the Chroma classifier at the end of each co-training iteration. The green dashed horizontal lines correspond to the optimal accuracies of the MFCC classifier described in the previous section. Likewise, the red dashed horizontal lines correspond to the optimal accuracies of the Chroma classifiers.
What follows is an analysis on the number of times each classifier chose to add mislabeled samples to the labeled training data during co-training. In the following bar graphs[Figure 6], the total number of mislabeling that each classifier makes are separated by the true class of the sample that was being labeled. The final analysis performed deals with the instances where the two classifiers disagree on the labeling of a sample[Figure 7]. For instance, the MFCC classifier chooses a sample to be added to the labeled training set with a label of North America, but

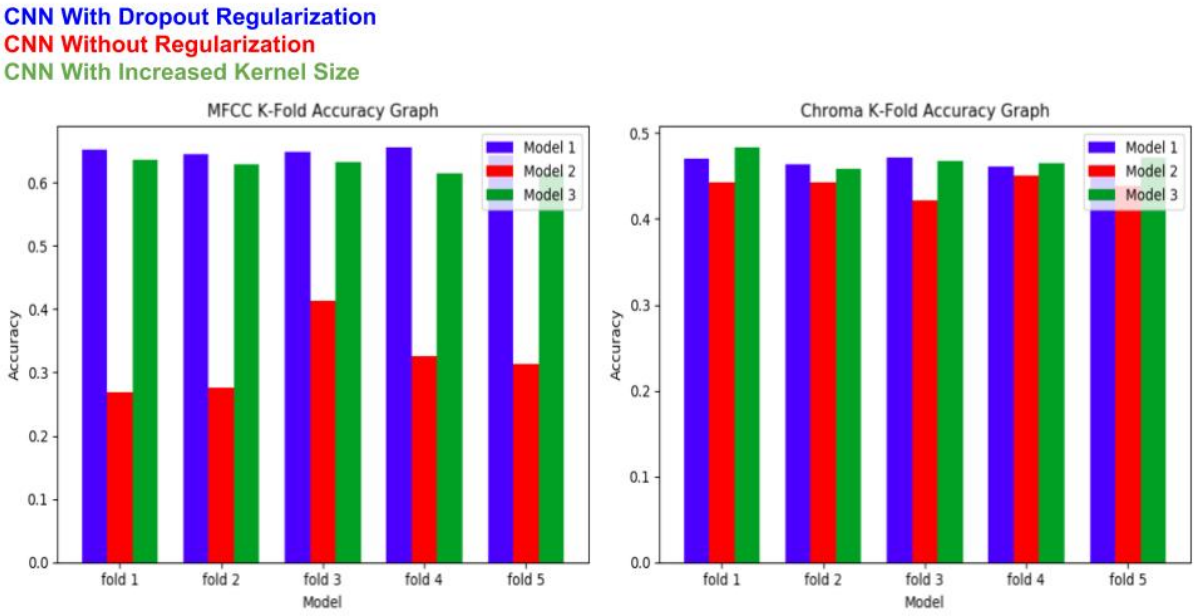

Figure 3: K-fold Results

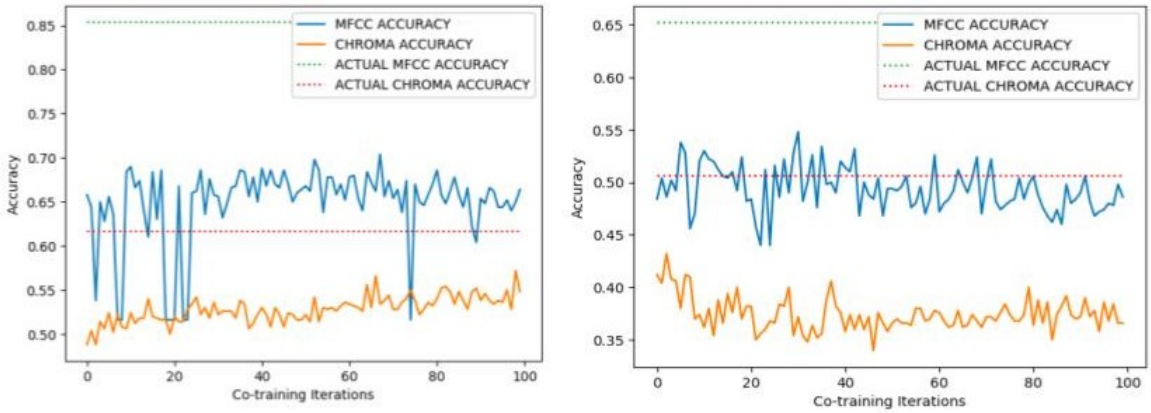

Figure 4: Class Co-Training Results

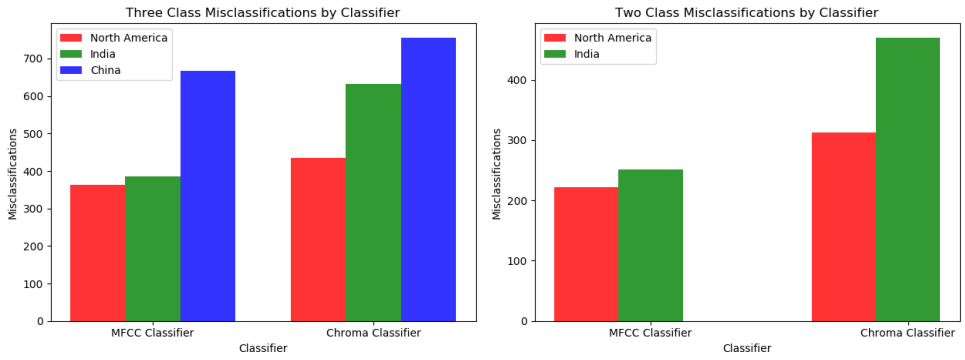

Figure 5: Misclassifications by Classifier

	MFCC	Chroma	Neither
N_America	12.0	1.0	0.0
India	9.0	8.0	0.0

Figure 6: Two Class Disagreement Matrix

	MFCC	Chroma	Neither
N_America	9.0	10.0	1.0
India	8.0	6.0	10.0
China	26.0	7.0	11.0

Figure 7: Three Class Disagreement Matrix

the Chroma classifier chooses to add that same sample to the labeled training set with a label of China. Interestingly, this problem does not seem to be addressed in the current literature on co-training, and thus, a strategy for handling these cases does not exist. In our implementation, when two classifiers disagree on the labeling of a sample, the sample is added to the labeled training set twice, once with each predicted label. This is done in an effort to offset the disagreement of the two classifiers, but we are tracking these cases in an effort to hypothesize a more optimum strategy. The following matrices are constructed by counting the instances of disagreement between the two classifiers, separated by the true class of the sample, as well as the classifier that was proposing the correct label i.e. in the three class example[Figure 8], there were nine instances where the the two classifiers proposed different labeling for a sample whose true label was North America, and the MFCC classifier proposed the correct label. Note that the 'Neither' column corresponds to instances where neither classifier proposed the correct label.

5 Analysis of the Results

In accordance with the cross-validation results, the CNN with dropout was chosen for the MFCC classification, and the CNN with dropout and increased kernel size was chosen for Chroma classification in the co-training algorithm.

The first thing to notice about the results of the 2 class co-training is that although neither classifier's accuracy approaches their optimal accuracy, both the accuracy of the MFCC classifier and the Chroma classifier seem to be on a general uptrend. This is most noticeable in the Chroma classifier's accuracy, which starts at around 48% at the first round of co-training, and eventually ends at around 54% by the 100Th iteration. This would imply that more samples are being labeled correctly than incorrectly, and thus the co-training process is working. The MFCC classifier's accuracy, however, did not yield quite as promising results. The overall trend, although slightly upward, still remains relatively flat. Also, on a few iterations the accuracy drops all the way down to around 50%. The reason for these dips remain unexplained. Perhaps they are a result of the way that the Keras library was initializing the weights of the classifier at the beginning of co-training. For the purpose of our analysis, though, they can be treated as outliers. Regardless, as is to be expected, neither classifier approaches their optimal accuracy. The 3 class co-training results are more easily explained, but less exciting. The accuracies of both classifiers decrease as the number of co-training iterations decrease. This is also somewhat expected, because with more class labels, the chance that either classifier adds a mislabeled

sample to the labeled training data increases.

During the three class implementation of co-training it can be seen that the Chroma classifier mislabels a greater number of total samples, it also shows that although the Chinese samples make up only about 41% of the total number of samples, the Chinese samples make up about 45% of the misclassifications, making Chinese accents the most difficult for our classifiers to identify. Notice also, that the MFCC classifier does a much better job of classifying Indian accents than the Chroma classifier does.
The results from the three class co-training are mirrored in the two class co-training results. Once again, the MFCC classifier does a much better job of distinguishing between North American and Indian accents than the Chroma classifier does.

The disagreement matrix for the three class follows quite naturally from the misclassification data, but it does show some new interesting results. China accounted for 50% of the disagreements, but only make up 41% of the labeled data, once again making China the most difficult class to predict on. However, in cases of disagreement where the true label of the sample is China, the MFCC classifier was correct 59% of the time, which implies that the MFCC classifier is much better at classifying Chinese samples than the Chroma classifier, at least in cases where the confidence of the prediction is high.
The two class disagreement matrix further cements the notion that the MFCC classifier generally performs better than the Chroma classifier. However, in both the two and three class case, in instances where the true class of the sample is India, the number of times where the MFCC classifier is correct and the number of times the Chroma classifier is correct are quite close.
From the disagreement matrix results, a good strategy may be to give preference to the MFCC's classifier's prediction of the China class, and give equal preference to each classifier on the other two predictions.

6 Conclusion and Future Work

It is well-known that the MFCC is a suitable feature for accent classification from previous work. We have shown though, that the Chromagram can be used to train better-than-chance classifiers, and is therefore a potential candidate to be a complementary feature for future accent classification systems. We have demonstrated intriguing exploratory results for utilizing MFCC and Chroma features to implement the co-training algorithm on small amounts of labeled accent data, especially in the case of binary classification, where the accuracies of our classifiers trended upward with the number of co-training iterations. We have also analyzed the misclassification rate, and cases of disagreement between the classifiers used in co-training to understand which of the used accents were hardest to distinguish from one another, as well as which classifiers performed best at predicting each class. Future work on this topic could include implementing our proposed strategy of handling labeling disagreements, as well as determining if similar analysis of classifier disagreements could be generalized to other co-training implementations. Work could also be done to determine the optimal tunings of the co-training algorithm to achieve desirable results in the accent classification domain i.e. number of samples labeled per iteration, size of the starting labeled and unlabeled data sets, etc. One could also simply run the co-training algorithm for a greater number of iterations to more fully understand the effect of the co-training algorithm on our classifiers, as well as attempt to find other more suitable features for co-training in the accent classification domain than MFCC and Chroma.

References

[1] George Mason University:*the speech accent archive*(http://accent.gmu.edu/)

[2] Leon Mak An Sheng, Mok Wei Xiong Edmund: *Deep Learning Approach to Accent Classification* (http://cs229.stanford.edu/proj2017/final-reports/5244230.pdf)

[3] Avrim Blum, Tom Mitchell: *Combining Labeled and Unlabeled Data with Co-Training*(https://www.cs.cmu.edu/ avrim/Papers/cotrain.pdf)