Deep Learning Approach to Accent Classification

Leon Mak An Sheng, Mok Wei Xiong Edmund
`{ leonmak, edmundmk }@stanford.edu`

1 Introduction

Despite the progress of automatic speech recognition (ASR) systems that have led to assistants like Alexa and Siri, accent is still an issue in developing robust ASR systems that deliver high performance across diverse user groups [1]. Statistical analysis has identified gender and accent to be most important factors of speaker variability affecting the fluency of ASR systems [2]. Our motivation stems from the fact that both team members are Singaporeans, who are known to have a unique, strong and distinctive accent very unlike the American and British accents. ASR systems like Google Now and Siri are usually trained on and perform best for these accents [3], and our experience has shown that speaking in our native accent with these ASR systems typically end up with not much success. We then usually resort to a forced accent in order to get the ASR to recognize the speech correctly, which is unnatural and proof that ASR systems can still be improved. Since accent is such a crucial aspect in ASR, we were inspired to build an accent classification machine learning model which could be used as a preliminary step in the ASR pipeline, allowing it to adopt a more suitable speech recognition model adapted to the identified accent for better performance. Other possible applications of accent classification include immigration screening [3]. In this project, our goal is to develop a deep learning model that is able to identify and classify a speaker by his or her predicted native language. The input to our algorithm is an utterance of a word by a speaker.

2 Related Work

Previous work has been done on foreign accent classification using traditional machine learning techniques. Chen, Lee, and Neidert [4] have used SVM, Naïve Bayes and logistic regression to obtain 57.12% test accuracy with SVM for Mandarin and German non-native speakers, using the CSLU database. Wang et al. [5] identified that models trained on male data do not generalize well on female data. They used a layered classification, first classifying by gender and then by accent, specifically on word-level utterances. Ge, Tan and Ganapathiraju [6] used Perceptual Linear Predictive features instead of MFCCs, and also focused on vowel extractions for their dataset after observing that most accents appeared in the pronunciation of vowels rather than consonants. We felt that this approach was quite clever but difficult to perform on a large dataset. Upadhyay [7] developed a new dataset of 5 speakers from China, India, France, Germany, Turkey and Spain from online videos, and was different from most of the existing research as he had used deep learning, specifically deep belief networks, to perform classification. Ma, Fan and Zhou [8] identified that applying a Gaussian Mixture Model approach, together with Hidden Markov Models to be the best approach in accent classification.

3 Dataset and Features

3.1 Dataset

We chose to use the Wildcat Corpus of Native and Foreign-Accented English[9] since it was available for free and contained a scripted reading scenario in which participants clearly enunciated a scripted list of words one at a time. This was useful in our preprocessing step where we segmented out individual word utterances as separate audio clips from the original speech recording, producing many word-level utterances for us to perform further feature extraction from. As the dataset from Wildcat Corpus consists of predominantly Chinese, English and Korean native language speakers, we decided to use these three native languages as our accent classification task classes.

3.2 Preprocessing

We used a peak detection library [10] to preprocess the audio recordings by segmenting out the word utterances in each speech recording and extracting the audio signals within the intervals when the speaker was pronouncing a word.

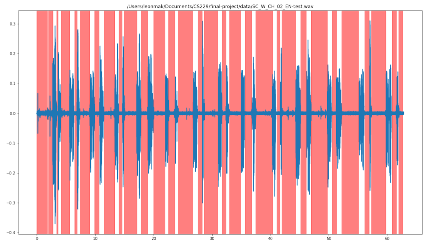


Figure 1: Regions below the threshold energy density (in red)

Time windows of unit length 0.18 seconds were extracted when its energy density was more than 4.8% of the average energy density of the entire audio signal in the file. We experimented with several different values of the time window length and energy density threshold and found that the above values gave the best isolation of word utterances from the silences in the speech recording on manual inspection of the extracted word utterances. Energy density is calculated as the squared sum of its amplitude over the time window.

With many word level utterances for each native language class, we then used the Librosa [11] library to extract MFCCs from each of the extracted audio segments. We chose to extract MFCCs because it accounts for human perception sensitivity with respect to frequencies, and thus is appropriate for speech/speaker recognition [3]. For each utterance, we fixed its length to 1 second by either padding or trimming the utterance, and extracted 50 MFCC bands from the utterance.

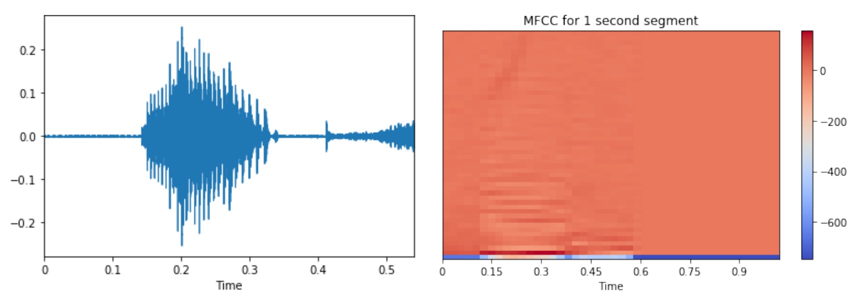


Figure 2: Plot of audio segment for word 'legs' (left) and its MFCC after padded to 1 second (right)

We then normalized the MFCC samples by subtracting the mean and dividing by the standard deviation. The result of the preprocessing and feature extraction step is input data is an m x 50 x n tensor, where m is the total number of utterances, and n is the number of frames sampled at 22050 Hz. Our final dataset consisted of 23910 examples, split into a training set of 19128 examples (80%), a dev set of size 2391 examples (10%), and a test set with 2391 examples (10%). Furthermore, we also applied data augmentation to the training set by adding random Gaussian noise (drawn from standard Gaussian) to each example, doubling the size of our training set to 38256 examples. The idea behind this form of data augmentation is that different individuals naturally speak with different vocal frequencies (which are reflected in the small differences in MFCCs) even if they share the same accent, so the Gaussian noise serves to provide this natural variation in producing more training examples.

4 Methods

We first implemented some traditional machine learning methods, specifically ensemble learning methods like Random Forests and Gradient Boosting methods, using Sci-kit Learn library. We wanted to use these models as baseline performance for our neural networks and thus we mostly used the default values provided by the library.

We tried 2 deep neural network architectures: the Multi-layer Perceptron (MLP), Convolutional Neural Networks (CNN). All neural networks were implemented in Python using the Keras [12] neural network library.

The first neural network architecture we tried to implement was the MLP, which consists of multiple stacked fully connected layers of neurons. The MLP has the simplest architecture out of the three networks implemented, and was used to establish a baseline performance for the subsequent networks. The last layer of the MLP is a softmax layer, performing softmax regression over the three classes.

During training, a prediction is made for each example in the batch by forward propagation and the loss, computed with categorical cross-entropy loss function, is back-propagated to find the error with respect to each weight in the network, so that they can be adjusted to descend the loss function and decrease the loss value.

$$L_i = -\sum_j t_{i,j} \log(p_{i,j}) \tag{1}$$

$$softmax(y)_i = \frac{\exp(y_i)}{\sum_j \exp(y_j)} \tag{2}$$

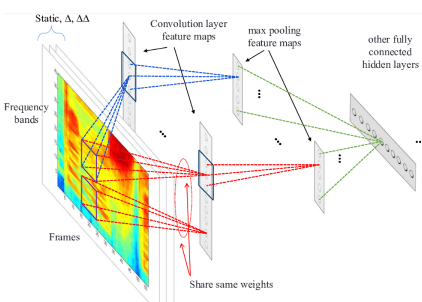
Figure 3: The categorical cross-entropy loss function (**Equation 2**) and softmax function (**Equation 3**)

Next we used a CNN which consisted of 2 convolutional layers with 3x3 filters and Rectified Linear Unit Layers (ReLU) which apply the activation function $x := max(0, x)$, and max pooling layers with 2x2 filter. Batch normalization [13] was also used to speed up training time.

Convolutional layers preserve the spatial relationship between pixels by learning local patterns, using subsamples of input data, as opposed to densely connected layers which learn global patterns, and learning image features.

The Max-Pooling layers retain important information about the image while reducing the dimensionality of the input and thus the computations in the network.

The final layers are densely connected with the last layer having a softmax layer to output the confidence of each class prediction. The Adam algorithm was used for optimization with a learning rate of 0.001.



Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 48, 42, 32)	320
batch_normalization_24 (Batc	(None, 48, 42, 32)	128
max_pooling2d_4 (MaxPooling2	(None, 24, 21, 32)	0
conv2d_7 (Conv2D)	(None, 22, 19, 64)	18496
batch_normalization_25 (Batc	(None, 22, 19, 64)	256
max_pooling2d_5 (MaxPooling2	(None, 11, 9, 64)	0
dropout_30 (Dropout)	(None, 11, 9, 64)	0
flatten_8 (Flatten)	(None, 6336)	0
dense_37 (Dense)	(None, 256)	1622272
dropout_31 (Dropout)	(None, 256)	0
dense_38 (Dense)	(None, 3)	771

Total params: 1,642,243
Trainable params: 1,642,051
Non-trainable params: 192

Figure 4: General architecture of CNN [14] (left) and summary of CNN used (right)

$$J_{L2} = J + \frac{\lambda}{2}||W||^2 \tag{3}$$

Figure 5: L2 regularization on neural networks

A number of measures were found to be useful in reducing overfitting. Dropout layers, which drop hidden and visible units (with their connections), were placed between layers. L2 regularization was also applied to reduce overfitting. Early stopping was also used to stop training once training any more would increase generalization error.

5 Results

5.1 Model Analysis

Traditional machine learning methods such as Gradient Boosting, Random Forest, were also used to construct a baseline. After doing 10 fold cross validation, the following results were obtained.

Model	Test Accuracy (%)
Gradient Boosting	69.1
Random Forest	69.1
MLP	80.0
CNN	88.0

5.2 Network Analysis

Our neural networks were had lesser number of layers compared to pre-trained models such as VGG for CNN as the shape produced by MFCCs were less complex than images of real life-objects. Increasing the number of filters in each layer did not lead to measurable change in the accuracy, but lead to longer training times.

Model	Data Augmentation	Train / Dev / Test Accuracy (%)
MLP	No	83.9 / 79.0 / 78.4
MLP	Yes	83.5 / 79.42 / 80.0
CNN	No	85.7 / 87.6 / 87.8
CNN	Yes	85.12 / 88.37 / 88.0

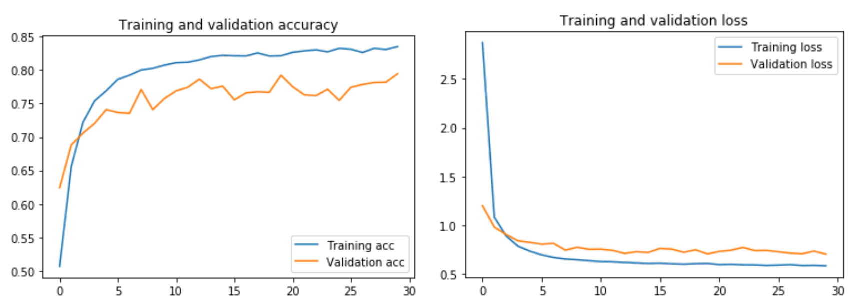


Figure 6: Training and validation accuracy and loss for MLP

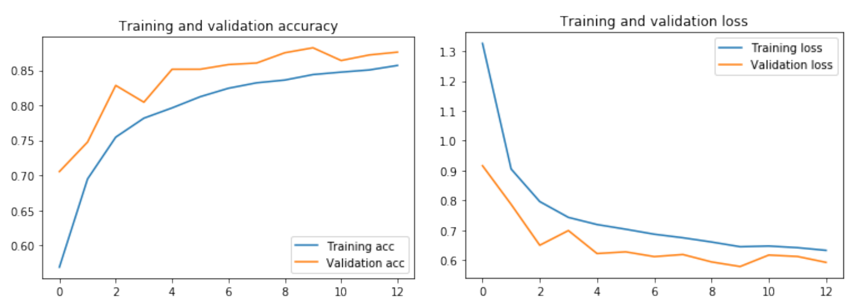
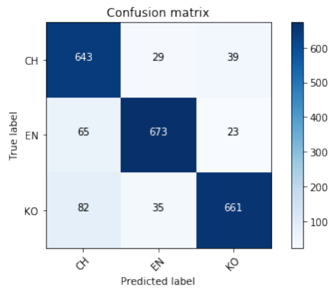


Figure 7: Training and validation accuracy and loss for CNN



Native Language	Precision	Recall	F1 Score
Chinese	0.904	0.814	0.857
English	0.884	0.913	0.898
Korean	0.850	0.914	0.881

Figure 8: CNN Confusion Matrix (left) and Precision-Recall table (right)

5.3 Error analysis

When examining examples that were misclassified, we found that some examples had noticeable background noise, or were from external sources such as dropping the microphone. These segments were incorrectly extracted as they had been loud enough to be detected by the extraction script. Thus if the data were better pre-processed to detect these anomalies, better results could have been obtained.

6 Discussion

The two ensemble models that we planned to use as baseline performance for our neural network implementations, Gradient Boosting and Random Forests, performed respectably well at 69% test accuracy. Even though they did not perform as well as the neural networks, they were easier to implement and this taught us to respect traditional machine learning techniques despite deep learning methods gaining popularity recently.

Among our neural networks, CNN performed better than MLP, as we expected. This is likely because CNN is known to perform well on image classification tasks and in our context, we had extracted the MFCCs from the utterances to form an image-like input that is fed into the CNN. As such, we have effectively reduced the accent classification task from an audio one to an image one, thus using the CNN gave better performance.

Our initial data was based on file level (full speech sentence) sampling at a fixed length, but we were unable to obtain reasonable performance (best test accuracy 40% with 5 classes on a different dataset, but similar preprocessing). This difference could be due to the fact that at the word level rhythmical characteristics except intonation is captured and can be used to distinguish english accent[15].

We also observed that data augmentation did help to boost the performance of our deep learning models, as can been seen in the results table.

7 Conclusion and Future Work

The results from our project show the capabilities of deep neural network architectures to classify both native and non-native english speakers. Using MFCC extracted from recordings, our CNN model was able to perform the classification the best among the algorithms we tested. It also turned out that audio-preprocessing and initialization of the CNN and MLP were major factors in affecting performance, and data augmentation, L2 regularization and dropouts were helpful in reducing overfitting.

More classes of non-native speakers could be included to see if our model is able to handle a wider variation of non-native speakers and to discern more subtle variations across those classes. Other statistical audio features like MFCC n-order derivatives (deltas) and mel-spectrograms could be used, or prosodic features such as range and sub-band energies could also be used. Given that our training classes had samples from both male and female examples, we could get better accuracy if we had trained models separately on them. In an end-to-end system, a model could be used to classify male and female samples before classifying for native language. More complex neural network architectures can be created by combining several types of neural network architectures, for example LSTM and DNN taking a final weighted probability[16].

References

[1] Russell, M., Najafian, M., Modelling Accents for Automatic Speech Recognition. 23rd European Signal Processing Conference (EUSIPCO), pages 1568. *IEEE*, 2015

[2] Huang, C., Chen, T., and Chang, E., Accent Issues in Large Vocabulary Continuous Speech Recognition In *International Journal of Speech Technology 7* , pages 141–153, 2004

[3] Upadhyay, Rishabh, Accent Classification Using Deep Belief Network, University of Mumbai, page 1, 2017.

[4] Neidert, J., Chen, P., Lee, J., Foreign accent classification, *http://cs229.stanford.edu/proj2011/ChenLeeNeidert-ForeignAccentClassification.pdf*

[5] Wang, X., Guo, P., Lan, T., Fu, G., Study of Word-Level Accent Classification and Gender Factors *http://students.cse.tamu.edu/xingwang/courses/csce666_accent_native_indian.pdf*

[6] Ge, Z., Tan, Y., Ganapathiraju, A., Accent Classification with Phonetic Vowel Representation

[7] Upadhyay, Rishabh. Accent Classification Using Deep Belief Network, University of Mumbai, pages 6-7, 2017.

[8] Ma, B., Yang, F., Zhou, W., Accent Identification and Speech Recognition for Non-Native Spoken English *https://web.stanford.edu/class/cs221/2017/restricted/p-final/boweima/final.pdf*

[9] Wildcat Corpus of Native and Foreign-Accented English *http://groups.linguistics.northwestern.edu/speech_comm_group/wildcat/content.html*

[10] Peak extraction script, adapted from *https://github.com/libphy/which_animal*

[11] Librosa, a python package for music and audio analysis, *https://librosa.github.io/librosa/*

[12] Chollet, F., Keras. *https://github.com/fchollet/keras.*

[13] Ioffe, S., Szegedy, C., Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

[14] Abdel-Hamid, O., Deng, L., Yu, D., Exploring convolutional neural network structures and optimization techniques for speech recognition, *Interspeech*, 2013

[15] J. C. Wells, Accents of English. *Cambridge, U.K.: Cambridge University Press, 1982, vol. I, II, III*

[16] Jiao, Y., Tu, M., Berisha, V., Liss, J., Accent Identification by Combining Deep Neural Networks and Recurrent Neural Networks Trained on Long and Short Term Features, *Interspeech*