Accent Recognition for Speech Recognition

170070007 170070046

Problem Statement

Our task was to identify the accent of a given speaker. Automatic identification of foreign accents is valuable for many speech systems, such as speech recognition, speaker identification, voice conversion, etc.

Methodology

There are two parts to this problem 1) Feature extraction and 2) Classification via extracted features.

Since differences in accent are due to both prosodic and articulation characteristics, a combination of long-term and short-term feature extraction training was used.

The feature extraction pipeline for our approach is as follows :-

Speech -> Silence Removal -> Feature Extraction -> PCA -> Improved Features.

(PCA was not used by us because it lead to a reduction in accuracy. Also Silence removal consists of removing all the blank spaces from the wav file)

We extract two types of features i) Long term features ii) Short term features. The long term features were then fed to a Deep NN with the parameters as suggested in the paper (further details in the section implementation details) and the output was a 5 node softmax layer which would represent the probabilities of the input being in a specific class. The short term features are fed into a RNN and its probabilities for the input being in one of the classes was calculated. The two probabilities were merged by assigning weights to them corresponding to their prediction accuracies.

Existing Approaches

There are many existing approaches related to this problem but the approaches we liked were 1) Accent identification by combining deep neural networks and recurrent neural networks trained on long and short term features by Jiao et al. (implemented in this project)

2) Improved Accent Classification Combining Phonetic Vowels with Acoustic Features by Zhenhao Ge (this is described below)

The author has tried to combine phonetic knowledge in the accent recognition problem instead of just using the acoustic features. Also his feature extraction pipeline is a little bit different than the one shown above. As for the classifier he uses a GMM-UBM model which is described as follows. He used a simple fact that most identifiable accents are presented from the pronunciation of vowels rather than consonants and thus computed multiple vowel-specific GMMs with features of the vowel components.

Implementation Details

The RNN and DNN models were written using Keras library and all testing and training was done on google collab.

VAD and segmentation

First the empty spaces and the stops from the speech are removed using VAD(voice activity detection). Then the dataset samples were cut down into 4 second segments(they were originally of size 36 seconds) hence around 8-9 segments (as some data is removed after passing through VAD) from a single data sample were created.

Long term feature extraction

This was done by a module called openSmile in which we had to write a config file to extract the required set of parameters. Also these features were introduced in Interspeech 2013 competition and they had given how the features were to be extracted. The features were of length 6373 per 4s segment.

Small term feature extraction

This was done by a python library called python_speech_features which gives mfcc features directly. The 4s segment was sampled by taking a 25ms window which was shifted by 10ms so that it would cover the entire 4s. This would give 399 samples and 39 mfcc features per sample. But 39 features were taking too much time when fed to a RNN, hence we decided to remove the delta and the double delta features.

Training of long term features using DNN

The long term features were then passed through a DNN whose structure is as follows.

The input layer contains 6373 features. Three hidden layers with 256 nodes for each followed. Rectifier linear units (“ReLU”) were used at the output of each layer and we use the dropout method to prevent over fitting, each input unit to the next layer can be dropped with 0.3 probability. The output layer contained 5 nodes corresponding to the 5 accents with softmax activation functions. Adam optimizer was used with learning rate 1e-5 and batch size was fixed to 128.

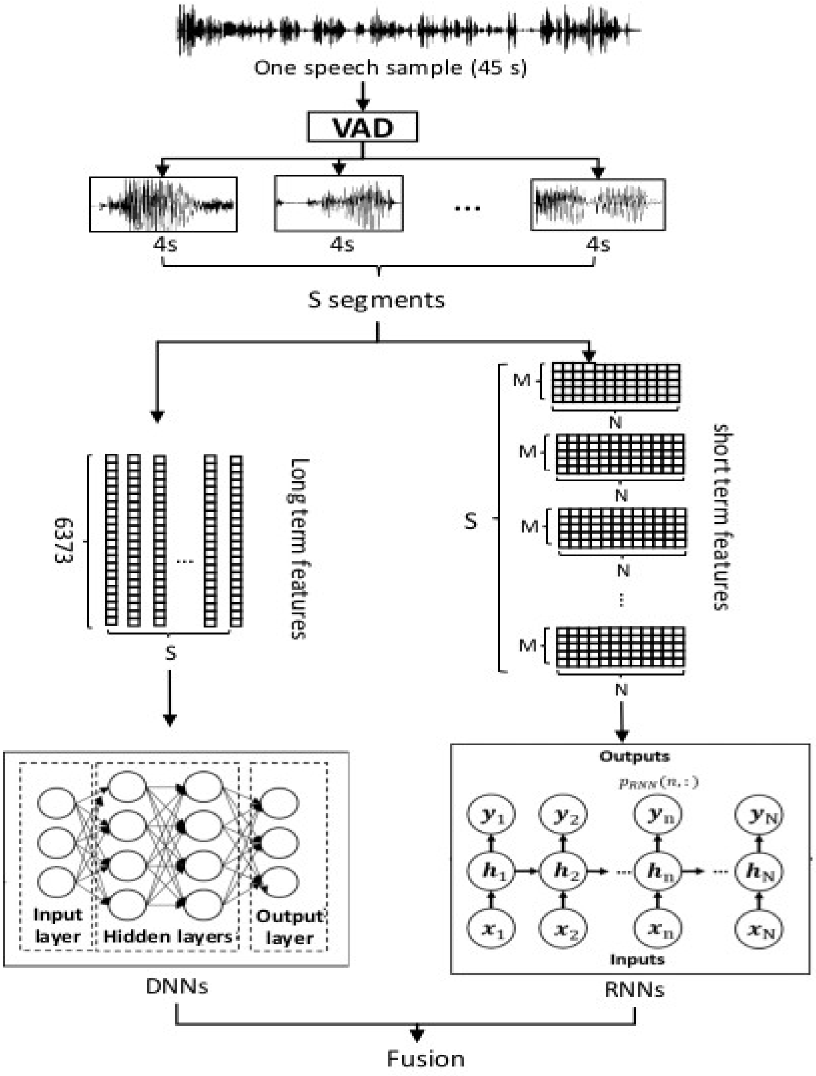

Figure 1: The proposed system of combining long and short term features using DNNs and RNNs.

Short term feature training using RNN

The RNN was trained on the short-term features extracted from the speech. Categorical labels were assigned to each frame of the segment. The results for each sample were calculated by averaging the predictions on all frames in all segments. The structure of RNN is as follows: The input data is sequentially fed into the RNN frame-by-frame. Each frame is of dimension 39. Two hidden layers with 512 long short term memory (LSTM) nodes were used. The activation function for the gates was a ‘logistic sigmoid’ and for updating the cell state, we used a ‘tanh’. The accent label was assigned to every 25ms speech frame - the LSTM layers allowed the model to learn long-term dependencies by taking the output of the previous hidden nodes as part of the inputs to the current node s. The hyper parameters for epochs was chosen to be 1(it itself takes around 6 hours of training time), batch-size=1, optimizer was Adam with lr=1e-5 and validation split=0.2.

Final accuracies by combining the results of DNN and RNN

The final output will be calculated by fusing the results from the DNN and the RNN. This is done by first calculating the accuracies they get individually on the validation set.

$$P(j) = \frac{1}{S}\left[w_{\mathrm{DNN}}\sum_{i=1}^{S} P_{\mathrm{DNN}}(i,j) + w_{\mathrm{RNN}}\sum_{i=1}^{S} P_{\mathrm{RNN}}(i,j)\right],$$

Where the weights w__DNN and w__RNN are calculated as follows:

The accuracies are those which are calculated on the **validation**[1] sets of the respective structures.

$$w_{DNN} = \frac{Acc_{DNN}}{Acc_{DNN}+Acc_{RNN}}$$
$$w_{RNN} = 1 - w_{DNN}$$

Experiments and Discussion

The dataset consisted of 63 wav files each of Arabic, English, Mandarin, French and Spanish languages. Each wav file was of nearly 36 seconds and all of them consisted of everyone saying the same sentence. After VAD and segmentation the total number of wav files came out to 2466.

Model Peculiarity	Train Accuracy	Validation Accuracy
DNN (epochs = 50)	53.12%	52.50%
DNN (epochs = 100)	70.56%	53.3%

[1] It was changed to what ma’am had said in the presentation (i.e instead of using training accuracies use validation accuracies)

RNN (epochs = 1 and input feature space trimmed)	65%	55%
DNN with RNN	NA	Test accuracy 54.9%[2]

The paper which was used as reference had 11 classes and had achieved accuracies of 50.2% on all the classes.

Confusion Matrix

The confusion matrix was calculated on the validation set which consisted of 400 samples.

PRE/TRUE	Arabic	English	French	Mandarin	Spanish
Arabic	69	5	4	10	6
English	2	51	3	7	4
French	9	4	40	13	9
Mandarin	5	7	1	63	3
Spanish	9	7	3	2	64

As can be seen here French is mis-classified the highest number of times.

Summary

In this project we tried to classify accent by separately training on the long term and short term features and then fusing the results together. This project could be extended to perform better by including the phonetic vowel dependencies as was mentioned in the related work (second paper)

References

1) Accent identification by combining deep neural networks and recurrent neural networks trained on long and short term features (link here)

2) Improved Accent Classification Combining Phonetic Vowels with Acoustic Features (link here)

3) Keras documentation (link here)

[2] Same changes were made as 1