11-785 PROJECT MILESTONE REPORT: SPEAKER IDENTIFICATION USING DEEP NEURAL NETWORKS

Fan Hu
fanh@andrew.cmu.edu

Linwu Zhong
linwuz@cs.cmu.edu

Siyuan Wang
siyuanw@cs.cmu.edu

Zinian Zhao
zinianz@andrew.cmu.edu

ABSTRACT

The project aims to develop a neural network model to identify if two speech recordings belong to the same speaker. Different feature extraction and representation methods as well as various neural network architectures have been being compared and investigated to implement an optimal model, which hopefully can achieve an equal error rate of less than 5%.

1 INTRODUCTION

In this project, we would like to work on NIST dataset and build a deep neural network (DNN) model to identify if two speech recordings belong to the same speaker. The project scope includes finding optimal feature representations from the acoustic data, exploring the best network and architecture for the model, and also identifying the most suitable loss function etc.

2 FEATURE EXTRACTION

In this project, the feature extraction consists of 8 main steps (1) Voice Activity Detection (2) Pre-emphasis (3) Framing (4) Windowing (5) Fourier-Transform and Power Spectrum (6) Mel spectrograms and Filter banks (7) Mean Normalization (8) Feature Length Standardization

2.1 VOICE ACTIVITY DETECTION

In the original speech recording files, there are some silence parts in the audios. This silence data contains no useful information, thus should be removed to make the signal data cleaner. We used a python library which interfaces to the WebRTC Voice Activity Detector (VAD) to detect and remove the unvoiced signals from the original speech data.

2.2 PRE-EMPHASIS

In general, the high frequencies components in the original speech signal have smaller magnitude than the low frequencies components. In order to balance the frequency spectrum, we applied a pre-emphasis filter on the speech signal to enhance the high frequencies components. The pre-emphasis filter is formulated as emphasized_signal(t) = signal(t) - k $\cdot$ signal(t-1), where the filter coefficient k is set as 0.97 in this project.

2.3 FRAMING

The frequencies in the speech signal are not always stationary over a long period of time. However, we could consider the signal to be stationary over a short period of time. In order to keep the frequency contours of the speech signal, we divided the signal into short time frames with an overlap of a small frame size. After that, we could apply Fourier transform over each short time frame and then combine the short frames to get the frequency contours. The short frame size we set in this project is 25 ms, and the overlap size is set as 15 ms.

2.4 WINDOWING

After dividing the signal into short time frames, we applied Hamming window function to each frame to smoothen the signal.

2.5 FOURIER-TRANSFORM AND POWER SPECTRUM

After the signal pre-processing steps, we applied N-point Fast Fourier Transform (FFT) to convert the signal data of each frame from time domain to frequency domain. We then calculated the power spectrum based on the FFT result using the formula: $P = (1/N) \cdot FFT(x) \cdot FFT(x)$, where N is set as 512 in this project.

2.6 FILTER BANKS

After applying FFT and calculating the power spectrum, we applied filters on a Mel-scale to extract frequency bands. The reason to use Mel-scale is due to that human ear perception of the frequency contents for speech signals is not in a linear scale. The applied filters are triangular filters with amplitude of 1 at the center frquency and decrease linearly to 0 till they reach the center frequencies of the adjacent filters. Amd we used 40 triangular filters in total. In order to filter out some noises, when converting from Hertz to Mel-scale, the high frequency bound we set is 3600 Hz, and the low frequency bound we set is 20 Hz.

2.7 MEAN NORMALIZATION

We also did mean normalization on the mel-scaled filter banks to improve the Signal-to-Noise (SNR).

2.8 FEATURE LENGTH STANDARDIZATION

After we removed unvoiced audio parts, some of the speech files only had less than 30 seconds' signal data. We discarded such speech files as these audios may be too short to extract any representative features. For the remaining speech files, in order to make it easier for the batch processing in our model, we fixed the time dimension in the features as 30000 by appending duplicates using the original extracted features.

3 MODEL: TRANSFER LEARNING

The overall architecture of the model consists of two steps: first pre-train a multi-way classification model to learn utterance level feature embedding and then fine-tune the model by using triplet loss to further maximize the cosine distance between two speakers.

3.1 PRETRAINING: MULTI-WAY CLASSIFICATION

Deeper networks tend to have larger capacity over shallower networks but the training cost is usually much more expensive. To tackle this problem, He et al. (2016) proposed ResNet model based on conventional CNN to simplify the training process. The core of a Resnet architecture incorporates a stacked residual blocks within which output from previous layer is added on top of the input in following layers, which is shown in Figure 1.

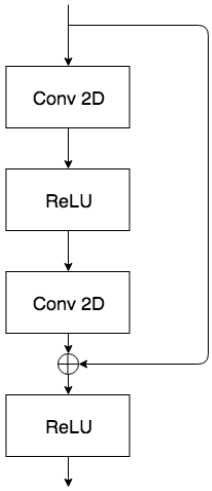

Figure 1: Detailed View of a ResNet Block

layer name	kernel/structure	stride
conv16	5×5	2×2
res16	3×3	1×1
conv32	5×5	2×2
res32	3×3	1×1
conv64	5×5	2×2
res64	3×3	1×1
conv128	5×5	2×2
res128	3×3	1×1
conv256	5×5	2×2
res256	3×3	1×1
conv512	5×5	2×2
res512	3×3	1×1
mean pooling	along time dim	-
linear	512×classes	-

layer name	kernel/structure	stride
conv32	5×5	4×4
res32	3×3	1×1
conv128	5×5	4×4
res128	3×3	1×1
conv512	5×5	4×4
res512	3×3	1×1
mean pooling	along time dim	-
linear	512×classes	-

Table 1: Model A (large) and Model B (small)

Li et al. (2017) has demonstrated the capability of ResNet to extract features of voice data. In this project, we employed a similar ResNet as the pre-trained model because of its ease to train and accurate feature extraction ability.

The details of our proposed pre-trained model are shown in Table 1. One basic residual block contains two convolutional layers with kernel size of 3X3 and stride of 1. In Model A, six residual blocks are concatenated using single convolutional layers with kernel size of 5X5 and stride of 2. Output channels are increased by two times as the model goes through each basic residual block in hope to learn significant features from the training data. Subsequently, an average pooling layer is applied whose result is projected linearly into a softmax layer to identify the label. In addition, batch normalization was also adopted between every consecutive convolutional layer and activation function layer. We used cross entropy loss as the loss function and Adam as the optimizer with 0.0001 learning rate.

Besides exploring different complexities of the pre-trained model, a 1-D ResNet model was also built to learn utterance level embeddings. The number of input channels of its input 1-D convolutional layer depends solely on frame-level representation, whereas the whole utterance becomes a single input channel for the input 2-D convolutional layer of the 2-D ResNet model.

3.2 TRIPLET NETWORK FOR SPEAKER IDENTIFICATION

The goal of pretraining is to learn an initial embedding of each speaker utterance. To achieve speaker identification, we would like the distance between embeddings from different speakers as far as possible. However, it is highly likely that the distributions of two different speakers' utterances overlap with each other and the pre-trained model does not suffice to accurately identify the difference between the data points in the overlapping region. In light of this, hence, we further train the pre-trained model using triplet loss to separate the distributions of data points from different speakers. We define cosine similarity as the distance between two embeddings. In other words, the scaler from cosine similarity also indicates the probability of two embeddings representing the same person. This would allow us to use triplet loss in this project. The triplet loss takes in three samples: Anchor, Positive and Negative. Each time, an anchor sample or a specific speaker utterance is selected followed by a positive sample which comes from the same speaker. A negative example needs to be an utterance from a different speaker. Formally, we want the cosine similarity between anchor and positive examples greater than that between anchor and negative to a certain extent called margin. The margin further separates two speakers' embeddings. It is vital that the training samples must be selected in a way that does not satisfy the above-mentioned condition for the purpose of learning. We call such training triplets as "hard" triplets which is defined the same as Li et al. (2017).

3.3 TRIPLETS SELECTION

To form triplets for training the model, we didn't follow the method used in Schroff et al. (2015) or Li et al. (2017) to select semi-hard examplers in the first few epochs. On the contrary, we extracted all hard triplets prior to using triplet loss for training. The selection process was based on the classification results from the pre-trained model. Wrongly classified data points as well as embedding of each training data point were recorded during pre-training, which would guide us to select hard triplets. For each wrongly classified training sample, we would select it as an anchor example and then iterate all other data from the same speaker. Each of the Anchor-Positive (AP) pair will then be matched with all other training data from the speaker the anchor data misclassified to. Subsequently, cosine similarities of anchor and positive (AP) as well as anchor and negative (AN) data are computed. Only the triplets that have greater AN similarity than AP similarity are selected as the final training data for triplet loss training.

4 EXPERIMENTS AND EVALUATION

4.1 DATASET OVERVIEW

The data used in this project are NIST Speaker Recognition Evaluation (SRE) datasets from 4 different years: 2004 (12.25%), 2005 (7.47%), 2006 (50.10%), 2008 (30.18%).

The total number of speech recording files are 33687. 60.23% of these files belong to female speakers, and the other 39.97% files belong to male speakers. We also calculated the number of audio files for each speaker, and Figure 2 shows the overall distribution.

4.2 DATA PREPARATION

In the original set of speech recording files, there are 3726 different speakers. In order to better train the model, we discarded files that belong to speakers with less than 10 audio files. In the remaining files, there are 1303 different speakers. We then took 90% of the audio files for each speaker to form our training dataset, and used the other 10% as our validation dataset during pre-training.

4.3 EVALUATION METRICS

For the pre-training, we use CrossEntropy loss and accuracy as evaluation metrics. For the final network trained with triplet loss, we are using Equal Error Rate (EER), which is the common measure in the field of speaker identification. Equal error rate (EER) is defined as the common value at the threshold where the false acceptance rate is equal to the false rejection rate.

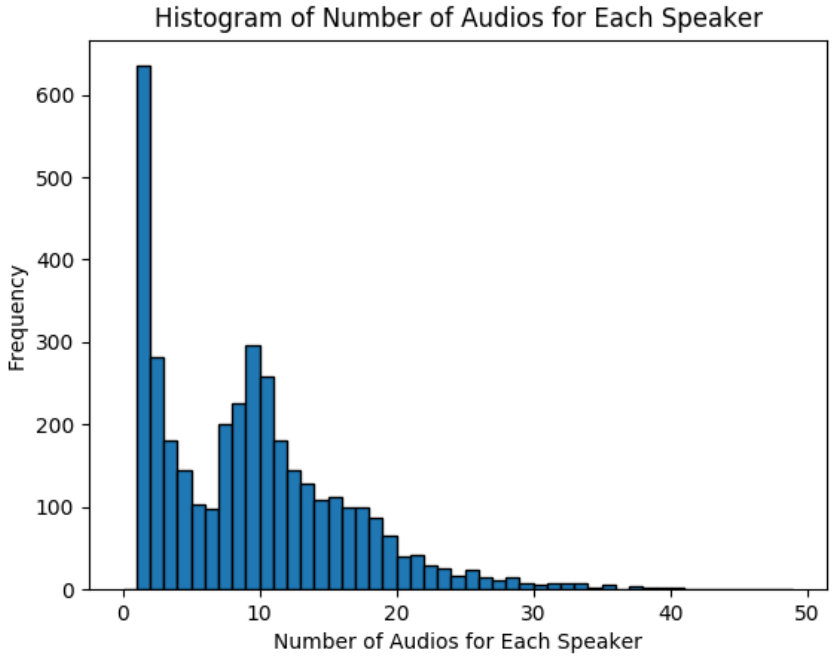

Figure 2: Histogram of Number of Audios for Each Speaker

4.4 RESULT

The most promising result that we got from the current pre-training multi-way class model could achieve classification accuracy of 77.41% on validation dataset. It is the small model (Model B in Table 1) that has three ResNet blocks.

Figure 3 shows the train loss and validation loss changes by epoch. Figure 4 shows the classification accuracy changes by epoch on the validation dataset. From the plots, we could see that the train loss decreases from 7.13 to 3.45 and the validation loss decreases from 7.06 to 4.26 after 37 epochs, and the classification accuracy improves from 0.52% to 77.41% after 37 epochs.

The result shown in Figure 3 and Figure 4 are just mid-term result in our experiment. This model is still in training and has not converged yet. Due to the time limitation, we could only provide the current performance plots in the mid-term report. We will update the experiment results and plots in our final report. And we believe that we should be able to get a more promising result by the time we complete the project.

As for the triplet network, we have started training with triplets sampled from wrong classifications in train and validation dataset using the best pre-trained model we have at present. The starting EER on training triplets is about 37%. As of now, after two epochs, the training EER has decreased to 19% and is still decreasing. We think this is a promising temporary result, as these triplets were sampled from wrong classifications. Hence, the EER might be relatively lower in randomly sampled test trials, which we will get to in the final report.

4.5 DISCUSSION

We have conducted multiple experiments using different models and datasets to check which one could provide us with the best classification accuracy. From the experiment results, we have learnt that:

(1) Removing unvoiced audios from the speech files could significantly improve the classification accuracy.

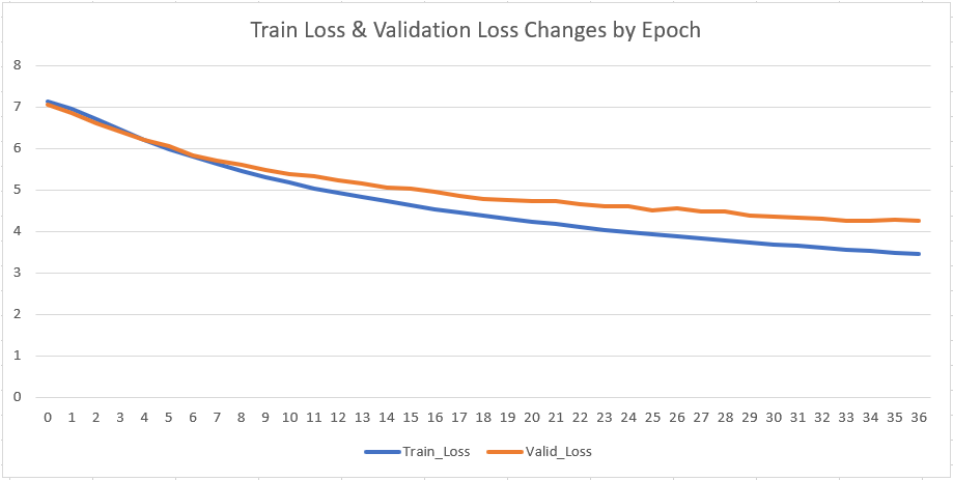

Figure 3: Train Loss and Validation Loss Changes by Epoch

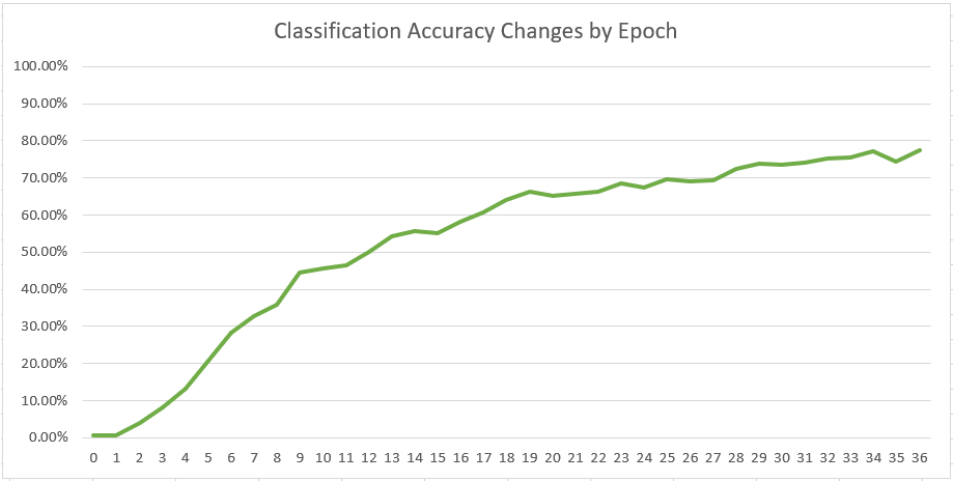

Figure 4: Classification Accuracy by Epoch

We conducted experiment to compare the effect of removing unvoiced audios from the speech files. In the experiment, we applied the same model on two datasets. The two datasets were generated from the same set of speech files. However, we removed unvoiced audios for Dataset A, and kept unvoiced audios for Dataset B. For Dataset A, both the validation loss and classification accuracy were almost unchanged after we trained the model for 100 epochs. But for Dataset B, the validation loss dropped 37.94% and the classification accuracy improved from 0.52% to 73.83% after just 30 epochs. From the result, we could see that removing unvoiced audios from the speech files could significantly improve the classification accuracy.

(2) Discarding speakers with less than 10 audio files could improve the classification accuracy.

We also conducted experiment to compare the effect of discarding speakers with less than 10 audio files. In this experiment, we also applied the same model on two datasets. We applied the same signal processing methods and removed unvoiced audios on both datasets. However, we kept all the speech files for Dataset A, and discarded speakers with less than 10 audio files for Dataset B. For Dataset A, we did not see any changes on validation loss and classification accuracy after the model was trained for 10 epochs. But for Dataset B, the validation loss dropped 22.28% and the classification accuracy improved from 0.52% to 44.42% after 10 epochs. From this result, we could conclude that discarding speakers with less than 10 audio files could also help to improve the classification accuracy.

(3) The model does not have to be over-complicated to achieve a satisfying classification accuracy.

In another experiment, we compared the performance between Model A which has 6 ResNet blocks and Model B which has 3 ResNet blocks. Both models were running on the same dataset. For Model A, the validation loss was decreased by 13.16% and the classification accuracy reached 7.20% after 30 epochs. However, for Model B, the validation loss was decreased by 47.09% and the classification accuracy could reach 74.04% after 30 epochs. This experiment result told us that sometimes we did not have to use an over-complicated model to achieve a satisfying classification accuracy. In this experiment, although Model B is relatively simpler, it still has strong learning power. More importantly, since it has less parameters to tune, we could achieve a faster convergence speed compared to Model B.

5 PLAN

Our team's next move is to finish the pre-training process of the big ResNet model (Model A) and the small ResNet model (Model B). After that, we will continue training the two models using the triplet loss. Eventually, the equal error rates of the two models will be compared and discussed in the final report.

REFERENCES

Najim Dehak, Patrick J. Kenny, Reda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19:788–798, 2011.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Patrick Kenny. Bayesian speaker verification with heavy-tailed priors. *Odyssey*, 2010.

Hung-Shin Lee, Yu-Ding Lu, Chin-Cheng Hsu, Yu Tsao, Hsin-Min Wang, and Shyh-Kang leng. Discriminative autoencoders for speaker verification. *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5375–5379, 2017.

Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell McLaren. A novel scheme for speaker recognition using a phonetically-aware deep neural network. *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1695–1699, 2014.

Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. Deep speaker: an end-to-end neural speaker embedding system. 05 2017.

Timur Pekhovsky, Sergey Novoselov, Aleksei Sholohov, and Oleg Kudashev. On autoencoders in the i-vector space for speaker recognition. *Proc. Odyssey*, pp. 217–224, 2016.

Douglas A. Reynolds and Richard C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, 3:72–83, 1995.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. pp. 815–823, 06 2015.