MEL SPECTROGRAM FEATURES FOR THE KALDI VOXCELEB V2 RECIPE

Uma Bahl (ub2151)

Columbia University, New York

ABSTRACT

The Mel Frequency Capstal Coefficient Computation, or MFCCs, are generally the most widely used features for speaker recognition systems. However, MFCCs undergo compression and decorrelation, which may remove important information from the signal.

Mel Spectrogram features have been gaining popularity, as the less compressed visual representation is conducive to detailed and accurate information about a signal. This paper explores the performance of the VoxCeleb v2 speaker recognition recipe with both MFCC and Mel Spectrogram features. We find that the MFCC based model outperforms the Mel Spectrogram model by 5.1% EER.

1. INTRODUCTION

In the field of Spoken Language Processing, two diverging approaches have emerged: the approach of using audio features such as MFCCs, and the approach of using vision techniques such as Mel Spectrograms. This paper aims to explore using Mel Spectrogram features with the Kaldi VoxCeleb v2 recipe.

1.1. The Kaldi VoxCeleb V2 Recipe

We utilize the Kaldi VoxCeleb recipe for speaker recognition. In the first version of the recipe, representations called i-vectors are used as features. These i-vectors are learned from the universal background model (UBM), and a probabilistic linear discriminant analysis (PLDA) classifier is used to make the speaker recognition decisions [9].

The second version of the recipe, Deep Neural Networks (DNN) are used to generate embeddings called xvectors. The embeddings are initially 512 dimensions, extracted from one of the final layers of the model. The embeddings are reduced to 120 dimensions, and a PLDA classifier is used.

Both versions of the recipe use MFCC features. The features are of the following specification: frame length 25ms, frequency 20 - 7600Hz, 30 mel-frequency bins, and 30 cepstra. The v2 recipe also utilizes data augmentation by adding noise and reverberation to increase the amount of data.

1.2. The Mel Spectrogram

To the human listener, the linear frequency scale does not sound linear. The Mel Scale is logarithmic representation that transforms the signal's frequency such that sounds of equal distance in frequency sound of equal difference to the human listener. Because it is logarithmic, it reflects how lower frequencies are easier to differentiate than higher frequencies. The Mel scale was introduced by Stevens in 1937 [1]. Mel is a unit of pitch, equal to one thousandth of the the pitch of a simple tone with frequency of 1000 Hz with an amplitude of 40 dB above the auditory threshold [1].

The Mel Spectrogram is a representation of the spectral content of a speech signal [1]. It is a three dimensional representation, where the horizontal axis is time, the vertical axis is frequency. The third dimension is conveyed through the darkness of the marking, where darker markings signify high energy and white signifies no energy.

A Mel Spectrogram captures a lot of information about the signal. For one, whether the signal is narrowband or wideband, based on whether the lines in the spectrogram are horizontal or vertical [1]. Additionally, it captures formant representations. For example, children typically have a shorter vocal tract length, and therefore the formant locations move up in frequency, something that can be captured in a Mel Spectrogram. This representation is helpful for tasks such as speaker recognition, a task the VoxCeleb Kaldi recipe aims to

do.

1.3. Mel Spectrogram vs MFCC Features

There are several ways to compute MFCCs [1]. One approach uses spectral estimation, Mel-scale warping, cepstral computation, and Discrete Fourier Transforms (DFTs). One of the advantages of the cepstrum is its invariance to linear spectral distortions. Overall, MFCCs convey information about the physical aspects of the speech signal.

MFCCs have mean subtraction capability, which can prevent them from getting affected by noise [1]. Cepstral mean normalization (CMN), or cepstral mean subtraction, is valuable in speaker recognition as it removes the DC component of a segment. It is achieved by estimating the mean of the cepstral features and subtracting it from all the features. This approach has been shown to improve results for speaker recognition. The Kaldi VoxCeleb recipe applies Cepstral Mean and Variance Normalization (CMVN), an approach that normalizes the variance as well.

Filter bank coefficients can be highly correlated, which is not always ideal in certain machine learning models (Gaussian Mixture Models, Hidden Markov Models) [13]. A compressed and decorrelated representation can be obtained by applying the Discrete Cosine Transform (DCT), resulting in MFCCs. DCT is a linear translation, which may discard information in speech signals that are non-linear. A Mel Spectrogram will retain more of this information, which may make it better for certain tasks.

2. FORMULATION FROM THE STATE OF THE ART

Prior work includes using Log-Mel Spectrograms for various applications. Mehta et al. [3] used 2D Log-Mel Spectrograms and CNNS with transfer learning to classify music into genres. Meng et al. [4] used 3D Log-Mel Spectrograms for speech emotion recognition, achieving results that improved on the previous state-of-the-art methods. Meghanani et al. compared MFCCs and Log-Mel spectrogram based features for Alzheimer's Dementia recognition, and concluded that Log-Mel spectrograms are effective features [5].

Additionally, other researchers have attempted to improve the performance of the Kaldi VoxCeleb recipe with spectrogram based approaches. Zeinali et al [12] considered Convolutional Neural Network (CNN) based embeddings. They combined the embeddings with the energy-based VAD from the Kaldi SRE16 recipe. They also attempted to improve the TDNN topology. They attempted adding more epochs, adding more layers, adding more neurons, and using PLP and FBANK features. The FBANK features combined with the other methods resulted in a slight improvement in their result. The FBank features are 40 dimensional, 16kHz, with frequency limits 20-7600Hz, and 40 filter-bank channels. This result shows that utilizing more of the filter bank or using more expressive features may improve the performance of the recipe.

3. APPROACH AND FORMULATION

3.1. The Approach

The approach was to alter the Kaldi VoxCeleb v2 recipe to use Mel Spectrogram features instead of MFCCs. The approach also involved slightly simplifying the training process, as the original recipe requires a large amount of resources for training. The altered recipe is defined in a new run.sh script and consists of the following steps.

The initial stage is to prepare the data. It creates the spk2utt and utt2spk files for the training and test data. It creates the data/voxceleb1_test and data/voxceleb1_train directories for the data. Then, it combines the training data into the data/train directory. Since a simplified version of the model was trained, the training data is composed of the voxceleb1_train data, rather than the voxceleb1_train, voxceleb2_train, and voxceleb2_test data.

The first stage is to create the features for both the training and test data. In the original recipe, MFCC features are computed for the data. This stage is altered to compute spectrogram features instead. The spectrogram features are computed using the Kaldi compute-spectrogram-feats functionality. The energy-based VAD is computed as well.

The second stage of the original recipe consists of augmenting the training data with reverberation, noise, music, and babble. The third stage consists of taking a random subset of the augmentations then computing the features for the augmented data. To maintain a simpler model, the augmentation steps were omitted in the altered recipe.

The remaining stages were largely unchanged. The

fourth stage prepares the features to generate examples for xvector training. The fifth stage is to filter out the features and speakers. First, it removes features that are too short, so that all the utterances that remain are at least five seconds. Next, it removes speakers with fewer than eight utterances. Stages six through eight are for training the model. Stage nine extracts the x-vectors for centering, LDA, and PLDA training. It also extracts x-vectors for the evaluation. Stage 10 centers the evaluation xvectrors, decreases the dimensionality, and trains the PLDA model. Stage 11 makes predictions on the test set. Stage 12 computes and displays the EER and minDCF of the predictions.

3.2. The Model

The Time-Delay Neural Network (TDNN) is a beneficial architecture for Speaker Recognition, as it works well with the dynamics of speech as a time-varying signal [1]. They were first introduced in 1989 by Waibel et al., with the frame-based analysis of speech in mind. The framing process involves isolating a window of the signal before extracting features.

The TDNN model was inspired from the Recurrent Neural Network (RNN) model, as an RNN operates on state transitions through time. The TDNN also shares similarities with a Feed Forward Neural Network (FFNN). However, the TDNN delays each input in (typically) increments of one frame of audio.

The model consists of nine layers [1]. It takes in an input segment with T frames and the input dimension.

The first five layers operate on the speech frame, and all use the ReLU activation function. The first layer uses the temporal context of two frames on both sides, centered at the current frame, therefore giving it five frames of total context t [9]. The second layer uses the spliced output at frames $t-2, t$, and $t+2$, and therefore sees nine frames of context. The third layer uses the spliced output at $t-3, t, t+3$, and therefore sees 15 frames of context. The last two frames continue with these 15 frames of context.

The sixth layer is a statistics pooling layer. It combines all the output frames from the previous layer for the input segment and computes the mean and standard deviation. We set the maximum chunk size, or maximum number of frames the stats layer is able to pool over, to 100 seconds. If an input is longer than 100 seconds, multiple xvectors are computed and averaged.

Since this layer aggregates the frames, the remaining layers operate on the full segment. The seventh layer is a segment-level layer using the ReLU activation function. This layer is where the xvector is extracted from.

The eighth layer is another segment-level layer using the ReLU activation function. The xvectors could be extracted from this layer as well, but the best performing xvectors come from the previous layer. The last layer is a softmax output layer which is used to classify the speakers.

3.3. Training Process

Two models were trained. The first, the baseline model with MFCCs features. The second, the experimental model with the Mel Spectrogram features.

Both models were trained in a Google Cloud Platform Virtual Machine (VM) instance. The VM environment used was an Ubuntu 18.04 n1-highmem-8 (8 virtual CPUs, 52 GB memory), with Intell Haswell and 1.2 TB disk space.

The models were trained using one Nvidia Tesla K80 GPU. The Kaldi training script was adjusted for one GPU by adjusting the num_initial_jobs and num_final_jobs parameters to use one GPU.

4. EXPERIMENTS AND RESULTS

4.1. Training Data

The experiments are run using the Kaldi VoxCeleb v2 recipe. This recipe focuses on speaker verification and uses four datasets [2].

The main data is sourced from the VoxCeleb dataset. The dataset consists of utterances sourced from interview videos on Youtube. It consists of two datasets: VoxCeleb1 and VoxCeleb2 [6].

The VoxCeleb1 Dataset consists of 100,000 utterances for 1,251 celebrities [7]. It contains utterances from 55% male and 45% female speakers, with 36 different nationalities. The VoxCeleb2 Dataset contains 1,092,009 utterances with 61% male and 39% female speakers, and 145 nationalities. [8].

[1] https://github.com/kaldi-asr/kaldi/blob/master/egs/voxceleb/v1/local/nnet3/xvector/tuning/run_xvector_1a.sh

[2] https://github.com/kaldi-asr/kaldi/tree/master/egs/voxceleb/v2

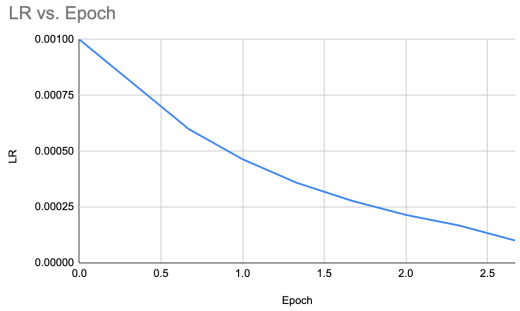
The VoxCeleb datasets are augmented with two datasets: The MUSAN and RIR datasets. Overlaying these datasets on to the VoxCeleb dataset increases the amount and diversity of the training data [9].

The MUSAN, or Music, Speech, and Noise Corpus contains three main sections [10]. For augmenting the VoxCeleb dataset, one of the following types of noise is randomly added. The first is Speech, consisting of 60 hours of book readings and government hearings in twelve different languages. Three to seven speakers are added to the training data to create "babble" in the background. The second portion is Music, which is over 42 hours of Western art music and popular genres. One music file is added to create background music. The third portion is Noise, about 6 hours of technical noises and ambient sounds. Noise is added at one second intervals throughout the utterance.

The RIR, or room impulse response, dataset consists of real RIRs and simulated RIRs [11]. Artificial reverberation is added to the VoxCeleb by augmenting with these files.

4.2. Training of the Mel Spectrogram Model

Fig. 1. Learning Rate During Training of Mel Spectorgram Model

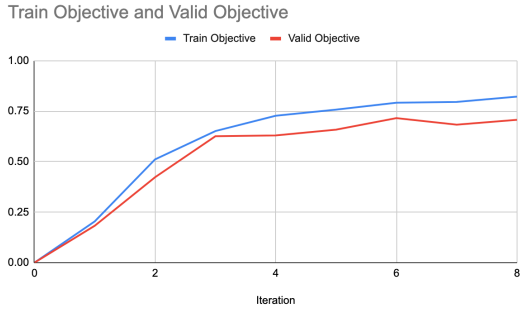


The Mel Spectrogram model trained in three epochs, with nine iterations. The model took approximately 22 hours to train, and there were 5.3 million parameters. The Learning Rate over the three epochs can be viewed in Figure 1. Information about the model can be viewed in Table 1. Lastly, the information from the accuracy output report can be viewed in Figure 2.

Information	Value
Feature Dimension	257
Frames per eg	1
Left Context	0
Number of Archives	3
Number of Frames	41,648,349
Right Context	200
Min Chunk Size	25
Max Chunk Size	10000
Total Training Time	21:26:22

Table 1. Information About the Mel Spectrogram Model

Fig. 2. Accuracy During Training of Mel Spectorgram Model



4.3. Training of the MFCC Model

The MFCC Model trained in three epochs, with 18 iterations. The Learning Rate over the three epochs can be viewed in Figure 3. Information about the model can be viewed in Table 2. The information from the accuracy output report can be viewed in Figure 4.

There were 5.1 million parameters, and the model took approximately 101 hours to train. This was a surprise, considering the Mel Spectrogram model had a larger input dimension and took approximately one fifth of the amount of time to train.

The hypothesis for this longer training time is that the Google Cloud Platform VM was experiencing a slowdown. Even simpler steps, like extracting the x-vectors, took a longer amount of time. Another hypothesis is that the frame length of the spectrogram features was larger than the 25ms frame length of the MFCC features. As seen when comparing Table 1 and Table 2, there was a substantial difference in the number of

Fig. 3. Learning Rate During Training of Mel Spectorgram Model

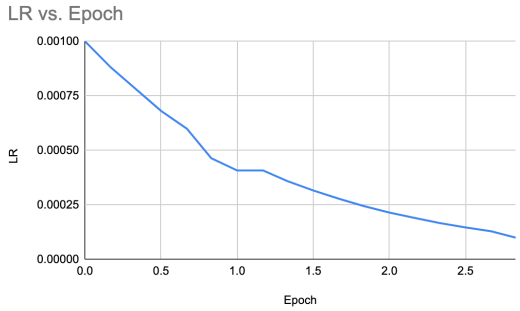
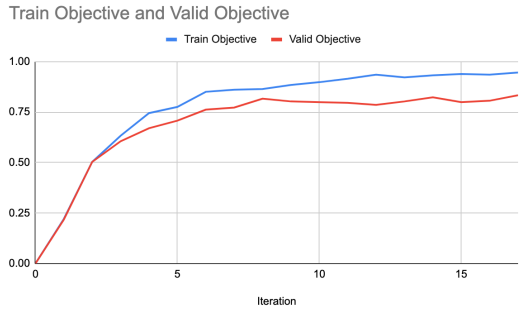


Fig. 4. Accuracy During Training of MFCC Model



frames.

Information	Value
Feature Dimension	30
Frames per eg	1
Left Context	0
Number of Archives	6
Number of Frames	113,289,836
Right Context	200
Min Chunk Size	25
Max Chunk Size	10000
Total Training Time	4 days, 5:19:18

Table 2. Information About the MFCC Model

4.4. Results

The first evaluation metric is the equal error rate (EER). The EER describes the performance of a speaker verification system, and is a convenient error percentage [1]. It can be a somewhat controversial error rate because the operating point for the rate is the point where there are equal false rejections and false acceptances. With the EER metric, a lower score is desired.

The second evaluation metric is the detection cost function (DCF). It is specified in terms of the cost of misses, cost of false alarms, prior probability a target speaker, and prior probability of a non-target speaker [1]. We measure it with p-targets of 0.01 and 0.001.

The results for EER and DCF can be viewed in Table 3. The MFCC based model outperformed the experimental Mel Spectrogram model by 5.1% EER, making it the clear winner. For reference, the improvement from the Kaldi Voxceleb v1 recipe to the v2 recipe was 2.201%.

The MFCC based model also outperformed the experimental Mel Spectrogram model by 0.2304 DCF10^{-2} and 0.2042 DCF10^{-3}. The improvement from the Kaldi Voxceleb v1 recipe to the v2 recipe in DCF10^{-2} was 0.1675, and in DCF10^{-3} was 0.1165. Again, this is a clear outperformance.

Evaluation Metric	Mel Spectrogram	MFCC
EER	12.79%	7.69%
DCF10^{-2}	0.8226	0.5922
DCF10^{-3}	0.9376	0.7334

Table 3. Evaluation Results

5. CONCLUSION AND FUTURE WORK

From the results, it is clear that the MFCC features outperformed the Mel Spectrogram features. The MFCC model achieved a 5.1% lower EER than the Mel Spectrogram model. It also achieved a 0.2304 and 0.2042 lower score for DCF10^{-2} and DCF10^{-3}, respectively.

This was a slightly surprising result, as I anticipated the more expressive Mel-Spectrogram features would lead to better results, especially in combination with the TDNN model. Perhaps this is because the MFCCs represent the signal in a compact and efficient manner, so they lead to better features for the model. Or perhaps this is because the MFCC features work well with noise due to the mean subtraction capability.

Another factor to consider is that this was an experimental model. The training process was simplified to

create a smaller model. If one wishes to further pursue this line of research, training the model on both the VoxCeleb1 and VoxCeleb2 datasets would be a reasonable next step. Additionally, testing with the augmented data could be another logical next step. Perhaps further tweaking on the frame length and other configurations of the spectrogram features would also yield favorable results.

Another future work possibility would be to experiment with augmenting the Mel Spectrogram features to the MFCC features, then training the model. A similar approach, Integrated Mel Linear Discriminant Analysis (IMELDA), uses LDA on the output of a Mel-scale filterbank [1]. IMELDA has been used in speech recognition tasks, and combined spectral features, cepstral coefficients, delta cepstral features, and delta-delta cepstral features.

Overall, this project was a wonderful learning experience. Working with MFCCs and Mel Spectrograms was a great way to learn about signal processing. Implementing the VoxCeleb recipe helped me gain confience using Kaldi, and I look forward to using it for future projects. Thank you for a great semester, Dr. Beigi!

6. REFERENCES

[1] Beigi, H. (2011). *Fundamentals of Speech Recognition*.

[2] Suh, J.W., Sadjadi, S.O., Liu, G., Hasan, T., Godin, K., & Hansen, J. (2011). *Exploring Hilbert envelope based acoustic features in i-vector speaker verification using HT-PLDA*.

[3] Mehta, J., Gandhi D., Thakur G., & Kanani P. (2021). *Music Genre Classification using Transfer Learning on log-based MEL Spectrogram*.

[4] Meng H., Yan T., Yuan F., & Wei H. (2019). *Speech Emotion Recognition From 3D Log-Mel Spectrograms With Deep Learning Network*.

[5] Meghanani, A., Anoop C.S., & Ramakrishnan, A.G. (2021).

[6] Nagrani A., Chung J.S., Xie W., & Zisserman A. (2019). *VoxCeleb: Large-scale speaker verification in the wild*.

[7] Nagrani A., Chung J.S., & Zisserman A. (2017) *VoxCeleb: a large-scale speaker identification dataset*.

[8] Chung J.S., Nagrani A., & Zisserman A. (2018) *VoxCeleb2: Deep Speaker Recognition*.

[9] Snyder, D., Garcia-Romero D., Sell G., Povey D., & Khundanpur S. (2018). *X-Vectors: Robust DNN Embeddings for Speaker Recognition*.

[10] Snyder, D., Chen, G., & Povey, D. (2015). *MUSAN: A Music, Speech, and Noise Corpus*.

[11] Ko, T., Peddinti, V., Povey, D., Seltzer, M.L., & Khudanpur, S. (2017). *A Study on Data Augmentation of Reverberant Speech for Robust Speech Recognition*.

[12] Zeinali, H., Wang S., Silnova A., Matejka, P., & Plchot, O. (2019). *BUT System Description to VoxCeleb Speaker Recognition Challenge 2019*

[13] Fayek, H. (2016). *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*. https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.