Text-Independent Speaker Recognition Using Deep Learning

Pranjal Naman[a], Satyarth Vats[a], Raunaq Bhalla[a] and Monarch Batra[a]
[a]*Depatment of Instrumentation and Control Engineering, Netaji Subhas Institute of Technology, Dwarka, New Delhi*

Abstract. Speaker recognition is the process of recognizing the speaker by using speaker-specific information. A speaker recognition system can be classified into text-dependent speaker recognition and text-independent speaker recognition systems. In a text-dependent system, the recognition phrases are fixed (known beforehand). The user can be prompted to read a randomly selected sequence of numbers. While in a text-independent speaker recognition system, there are no constraints on the words which the speakers are allowed to use. What is spoken in training and what is uttered in actual use may have completely different content. The entire domain of speaker recognition can be further categorized into speaker identification and speaker verification. Speaker verification evaluates whether the voice belongs to some person while speaker identification tries to find out the person it belongs to. In this paper, Mel Frequency Cepstral Coefficients (MFCC) were extracted from the audio files. These features were then fed a Convolutional Neural Network (CNN). This CNN was then optimized in order to increase model accuracy. Over the span of 6 runs of varying parameters, a maximum accuracy of approx. 97% was achieved.

Keywords: Speaker Recognition, Text-independent, MFCC, CNN, Deep Learning

1. Introduction

Speech, apart from the linguistic information it carries, also incorporates numerous other useful features such as emotion, gender and identity of the individual. And therefore, decoding these abstract features into numerical features helps us in diverse ways. Speaker recognition systems find applications ubiquitously — be it for voice authentication in biometric systems or for compiling customer feedback from their responses.

Automatic speech recognition is quite essentially the intersection of two widely different fields of Digital Signal Processing and Machine Learning. What we aim to establish from this article is an insight into how an automatic speaker recognition system might be implemented using deep learning. This task can be defined as a typical supervised learning problem. Consequently, it may be divided into three parts — extracting the features from voice samples, building a multiclass classification model using the extracted features and their respective labels, and ultimately testing this model on our test samples.

Another aspect to speech recognition systems is text-independence. While text-dependent models rely on certain fixed utterances from the speaker for classification, a text-independent model does not rely on fixed utterances and can classify a user, based on voice samples that the model has previously not experienced. In recent times, a lot of successful approaches including Gaussian Mixture Models, Hidden Markov Models and Deep Learning methods like Convolutional Neural Networks (CNNs) have been proposed for biometric recognition and verification tasks. The methods mentioned above make use of features like MFCCs and Spectrograms extracted from the audio features. Inspired by the success of features like MFCCs for speaker recognition, the model proposed

in this paper also makes use of MFCC features for training.

The rest of the paper follows this structure. Section 2 contains information about the dataset. Section 3 elucidates the concept of Mel Frequency Cepstral Coefficients (MFCC). Section 4 and 5throw light on the features that are extracted and how these are arranged to form the training and testing dataframes. Section 6, in general talks about the use of CNN as a method for Deep Learning. In section 7, the actual CNN architecture has been explained. along with the methodology used. The paper concludes with results and conclusions in sections 9 and 10 respectively. The references have been cited in the last section, section 11.

2. Dataset and Setup

The dataset employed in this research paper is publicly available as Voxforge Speech Corpus. The dataset contains speech samples from more than 1300 speakers. However, the dataset contains numerous samples from anonymous sources and including these in the project would not be productive. Each speaker has about 50 or more unique utterances each of about 3 seconds in length. After cleaning and ignoring the anonymous voice samples, a total of 1227 speakers' data was used for the purposes of this project. A total of about 50 thousand voice samples from 1227 speakers was used for training and the model was tested on about 13 thousand unseen samples.
The setup used for the purposes of this research paper is rather modest. A system with an i5 processor and 16 gigabytes of RAM was used. *Sublime Text* was used as the text editor and a metric tracking tool provided by Weights & Biases was used to record the training and testing processes.

3. Related Works

The problem of speaker recognition, in recent times, has attracted various modelling techniques, including but not limited to, deterministic techniques like Dynamic Time Warping (DTW), Vector Quantization and statistical modelling methods like Gaussian Mixture Model (GMM) and Hidden Markov Models (HMMs). In fact, one of the very first works on speaker identification by Reynolds D. in 1995 incorporated the use of GMM-UBM (Universal Background Model). Of all these methods, the most popular one is the Gaussian Mixture Models [3, 4, 8, 9]. The distribution of feature vectors extracted from a person's speech is modelled by a Gaussian mixture density [9]. The GMM based approach is essentially an unsupervised, clustering algorithm that first generates GMM models for different speakers and then uses pattern matching of a test sample with all of these models to predict the identity of the speaker.
This paper presents speaker recognition as a typical supervised learning problem and uses the power of convolutional neural networks to generate a well performing speaker recognition models.

4. Mel Frequency Cepstral Coefficients (MFCCs)

Fourier analysis can provide meaningful interpretation to only stationary signals. Since, sound is a non-stationary signal, we work with frames. In frames a signal is assumed to be stationary and hence we can apply the spectral analysis to it. Framing is dividing the speech signal into frames of 20-30 ms with an optional overlap of 1/3-1/2 of frame size. Frame size is usually taken as power of two, to make it feasible for FFT. Hamming window removes the discontinuities from the framed speech signal. A Discrete Fourier Transform of the signal is calculated using the FFT algorithm. The DFT is calculated using the formula

$$X_i(k) = \sum_{n=0}^{N-1} x_i(n) e^{-\frac{j\pi 2nk}{N}}$$

where *i* is the frame number and *N* is the number of points to calculate the DFT. FFT converts the signal from time domain into frequency domain and also finds the magnitude of frequency response of each frame. Mel-frequency warping is converting the actual frequency of each tone of speech signal measured in Hz into a subjective pitch which is measured on the 'Mel' scale.
Mel spectrum is computed by passing the Fourier transformed signal through a set of bandpass triangular filters called the Mel filter-bank. The approximation of Mel from physical frequency is given by

$$Mel(f) = 2595\, log_{10}\,(1 + \frac{f}{700})$$

In Cepstrum, the log mel spectrum is converted back to time. The result obtained is mel frequency cepstrum coefficients. This set of coefficients obtained for each speech frame of about 30ms with overlap is called an acoustic vector. These acoustic vectors are useful in representing and recognizing the voice characteristic of speaker.

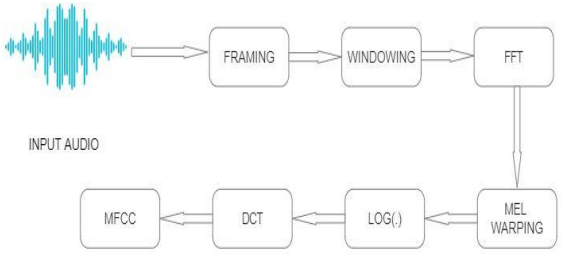

Figure 1: Extraction of MFCCs

5. Feature Extraction

The speech signal varies slowly with time; therefore, it is called quasi stationary. Its characteristics remains quite stationary, when examined for a short period of time (between 5 and 100ms). However, for longer period of time (order of 0.2s or more), the signal characteristics varies to reflect the different speech sounds. Therefore, the speech signal is required to be framed in samples short enough to be considered stationary.

As discussed earlier, the features to be extracted from the raw audio input samples are MFCC features. Each audio input is about 3 seconds long. However, it is not necessary that all audio inputs are exactly of the same length. So, framing the audio input might not result in the same number of frames for each sample. For this very reason, the frames are padded using *numpy* to 196 i.e. regardless of the number of frames in each audio input, the feature is padded up to 196 frames.

The next step is to calculate MFCC features for each of these frames. MFCC features are calculated using the Python library *librosa*. Now we know that for each frame, twenty MFCC features are calculated for each frame resulting in a *20 x 196* feature vector for each audio sample. The *Conv2D* layer doesn't accept inputs of this form, so the dimensions are expanded using *numpy* to *20 x 196 x 1* for each audio input.

All in all, the input audio sample is converted into a 3-dimensional numerical feature vector. There are 52728 audio samples for training and 13182 samples for testing. So, the training and testing input vectors are of shape (52728, 20, 196, 1) and (13182, 20, 196, 1) respectively.

6. Convolutional Neural Networks

A Convolutional Neural Network is a deep learning algorithm that proceeds with the application of simple filters applied repeatedly to get a feature map, which is an implicit extraction of significant features from an image.

Therefore, while in other methods, features are extracted beforehand, a CNN has the benefit of learning these features internally without having to be manually extracted and fed to the network.

A typical CNN Architecture would consist of:

- An input layer, allows for the CNN to take as input a raw image which would require flattening of the image in a feedforward neural network.
- The Convolution layers, A kernel is applied to the input image and a convolution operation takes place between these two matrices. This is done by shifting the kernel over the image in predetermined steps called strides. The resultant image could then be reduced or increased in dimensionality, by using another operation called padding. Conventionally, the convolutional layers are designed in a way that the initial ones are responsible for extraction of low-level features and the complexity of the features extracted increases with every layer. This would give the network a complete understanding of the input image.
- The Pooling layers are responsible for reducing dimensionality of the results from convolution operation, to decrease the computational power required to process the data. It also acts to reduce noise, to make dominant features much more visible.
- The classification layer or the Fully Connected Layer, it learns the nonlinear combinations of features extracted from the combination of convolution and pooling layer.
- The output obtained from this network can be flattened and fed to a feed forward neural network, with backpropagation to get a more accurate classification model.

Layer	Filters	Kernel	Stride	Padding	Activation	Output Shape
Input	-	-	-	-	-	(20, 196, 1)
Conv1	16	(3, 3)	(2, 2)	'same'	relu	(10, 98, 16)
BatchNorm1	-	-	-	-	-	(10, 98, 16)
Conv2	32	(3, 3)	(2, 2)	'same'	relu	(5, 49, 32)
BatchNorm2	-	-	-	-	-	(5, 49, 32)
Conv3	64	(3, 3)	(2, 2)	'same'	relu	(3, 25, 64)
BatchNorm3	-	-	-	-	-	(3, 25, 64)
Conv4	64	(3, 3)	(2, 2)	'same'	relu	(2, 13, 64)
BatchNorm4	-	-	-	-	-	(2, 13, 64)
Flatten	-	-	-	-	-	1664
Dropout	-	-	-	-	-	1664
Dense	-	-	-	-	relu	2454
Dropout	-	-	-	-	-	2454
Output	-	-	-	-	softmax	1227

Table 1: CNN Architecture Employed

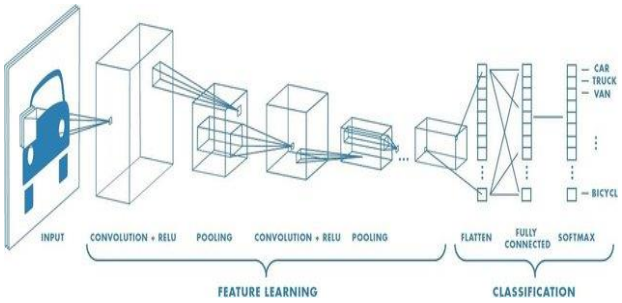

Figure 2: A Typical CNN Architecture

7. Model Architecture

In line with the general architecture of a convolutional neural network discussed in the previous section, the CNN model that we employ consists of stacked convolutional and dense layers to produce the final result. However, drawing conclusions from [8] we decided to replace max-pooling layers with strided convolutional layers. This was done because an increase in model performance, if not very significant, was observed on replacing max-pooling layers with strided convolutional layers [11]. Each convolutional block also contains a batch normalization layers to normalize and scale all inputs to the next layer [10]. There are four convolutional blocks followed by a *Flatten* layer and a *Dense* layer. The optimizer used is Adam with learning rate set at 0.0002 and momentum set at 0.5. The whole architecture can be found in Table 1.

8. Hyperparameter Tuning

In a machine learning model such as a neural network, optimization can be achieved either through changing the bias/weights attached to each node, or by varying the parameters that control the learning process. The former is achieved by the model itself in the training phase, the latter requires domain knowledge to find the accuracy optima. For a neural network these parameters consist of number of epochs and batch size. The activation function type could also be varied. The end goal of the process being to reduce variance and bias. For most intents and purposes, hyperparameter tuning can be done through trial and error by a domain expert, however, for much larger data processing tasks, algorithms to find the optimal hyperparameters can be used. These algorithms would act as loops on the model itself.

A hit and trial method has been tried for hyperparameter tuning and the hyperparameters have been randomly updated and the results for different runs have been compiled and mentioned in Table 2.

9. Results

The training accuracies and categorical cross entropy losses have been recorded and plotted as can be seen in Figure 3 and Figure 4. Model can be seen converging for all six successful runs and the losses and accuracies for the same have been recorded and tabulated in Table 2. The maximum training accuracy that has been achieved is 97.80%.

Runs	Batch Size	Epochs	Number of Classes	Accuracy	Loss
VoxForge_SuccessfulRun_1	32	8	1227	91.39%	0.8388
VoxForge_SuccessfulRun_2	32	8	1227	91.98%	0.8557
VoxForge_SuccessfulRun_3	64	8	1227	94.03%	0.7735
VoxForge_SuccessfulRun_4	128	8	1227	95.35%	0.7954
VoxForge_SuccessfulRun_5	64	16	1227	96.25%	0.5505
VoxForge_SuccessfulRun_6	64	32	1227	97.80%	0.4355

Table 2: Comparison of Accuracies and Losses for Different Runs

However, the actual performance of the model can only be tracked by evaluating the model on unseen samples and seeing how well it generalizes. The model is evaluated using the formula:

$$\%\ of\ correct\ recognitions = \frac{\#\ of\ correctly\ identified\ samples}{total\ \#\ of\ samples}$$

The research paper cited as no. [12], using Deep Belief Networks to model the statistical features of sound waves could achieve an accuracy of 95%.

Another paper with the same objective cited as no. [7] in references, used spectrogram images of audio clips are used as an input and applied CNN on it to get an average accuracy of 95.83%.

The model presented in this paper is tested on over 13000 previously unseen samples from the 1227 speakers and an accuracy of 96.95% was achieved.

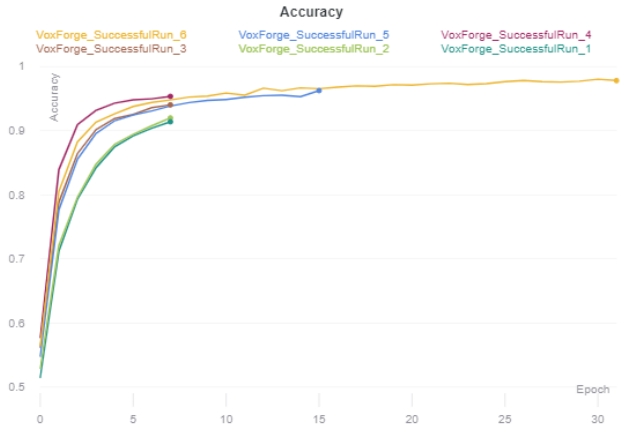

Figure 3: Accuracy vs Epochs

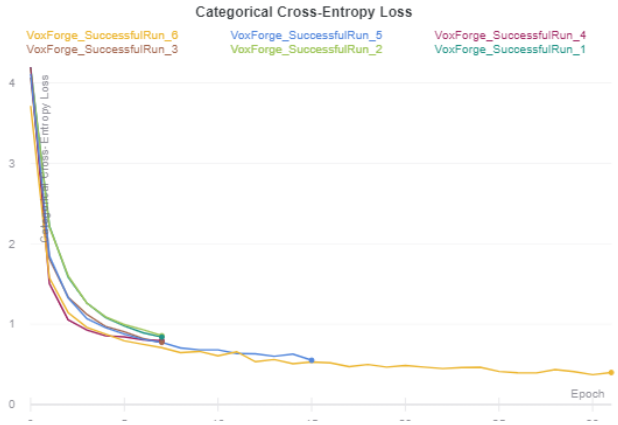

Figure 4: Loss vs Epochs

10. Conclusion and Discussion

A 96.95% accuracy still means that about 3 out of 100 results will be wrong which is one too many for it to be deployed in industrial usage. So, what can be done is a hyperparameter space search to receive optimal hyperparameters that, in turn, provide us with the best possible model. However, this approach does require a substantial amount of computing power and even then, we might have to wait for a long time (hours or even days) to get the optimal hyperparameter set. However, a 97% (approx.) test accuracy not only warrants research into the field of speaker recognition but also suggests the definite possibility that speaker recognition, in its improved version, can be used as a standalone biometric verification system.

11. References

[1] Furui, Sadaoki. (1996). An Overview of Speaker Recognition Technology. 43. 10.1007/978-1-4613-1367-0_2.

[2] M S, Sinith & Salim, Anoop & Sankar, K. & Nar yanan, K. & Soman, Vishnu. (2010). A novel method for Text-Independent speaker identification using MFCC and GMM. 292 - 296. 10.1109/ICALIP.2010.5684389.

[3] M S, Sinith & Salim, Anoop & Sankar, K. & Narayanan, K. & Soman, Vishnu. (2010). A novel method for Text-Independent speaker identification using MFCC and GMM. 292 - 296. 10.1109/ICALIP.2010.5684389.

[4] Mahboob, Tahira & Khanam, Memoona & Khiyal, Malik & Bibi, Ruqia. (2015). Speaker Identification Using GMM with MFCC. International Journal of Computer Science Issues. 12. 126-135.

[5] Santosh, K.Gaikwad & Bharti, W.Gawali & Yannawar, Pravin. (2010). A Review on Speech Recognition Technique. International Journal of Computer Applications. 10. 10.5120/1462-1976.

[6] Hasan, Md & Jamil, Mustafa & Rabbani, Golam & Rahman, Md. Saifur. (2004). Speaker Identification Using Mel Frequency Cepstral Coefficients. Proceedings of the 3rd International Conference on Electrical and Computer Engineering (ICECE 2004).

[7] S. Bunrit, T. Inkian, N. Kerdprasop & K. Kerdprasop (2019). Text-Independent Speaker Identification Using Deep Learning Model of Convolution Neural Network. International Journal of Machine Learning and Computing, Vol. 9, No. 2, April 2019.

[8] Douglas A. Reynolds & Richard C. Rose (1995). Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models. IEEE Transactions on Speech and Audio Processing 3(1). 72 – 83.

[9] Douglas A. Reynolds (1995). Speaker Identification and Verification Using Gaussian Mixture Speaker Models. Speech Communication, vol. 17, no. 1-2, pp. 91–108, 1995.

[10] Ioffe, Sergey & Szegedy, Christian. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

[11] Springenberg, Jost & Dosovitskiy, Alexey & Brox, Thomas & Riedmiller, Martin. (2014). Striving for Simplicity: The All Convolutional Net.

[12] A. Banerjee, A. Dubey, A. Menon, S. Nanda & G.C. Nandi.Speaker Recognition Using Deep Belief Networks to CCIS proceedings.