End-to-End Speaker Verification

A Project Report

Submitted by:

Abhigyan Ganguly (17-1-4-026)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

at

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

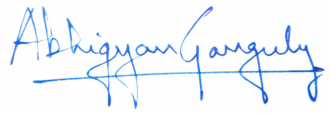
NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR,

SILCHAR, ASSAM (INDIA)-788010

MAY, 2021

DECLARATION

I hereby declare that the project entitled "End-to-End Speaker Verification" submitted for the B. Tech. (ECE) degree is my original work and the project has not formed the basis for the award of any other degree, diploma, fellowship or any other similar titles.



Place : NIT Silchar **Signature of the Student**

Date : 16/05/21

CERTIFICATE

This is to certify that the project titled “End-to-End Speaker Verification” is the bona fide work carried out by **Abhigyan Ganguly**, a student of B Tech (ECE) of National Institute of Technology Silchar (An Institute of National Importance under MHRD, Govt. of India), Silchar, Assam, India during the academic year 2020-21, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (Electronics and communication Engineering) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

Place :NIT Silchar **Name and Signature of the Supervisor**

Date :

ABSTRACT

Speaker recognition in general and Speaker verification in particular have been an important application area in voice biometrics and speech-based user security applications. It draws ideas from the fields of audio and signal processing, probability theory, machine learning and others. Although traditional speaker verification models were largely based on probability theory; current systems have been driven by state of the art machine learning algorithms. Due to the abundance of training data in the modern world, deep learning based architectures have proved to be at the forefront, producing lowest error rates and high accuracy as compared to their predecessors. However, a recent development in the domain of deep learning, is the End-to-End architecture paradigm which aims to simplify the model architecture and provide state-of-the art results whilst also reducing the domain specific knowledge which machine learning algorithms sometimes pose as a challenge. In that spirit, I believe that End-to-End architectures must be explored in the domain of speaker verification as well, as it has shown great results in similar sequence modelling tasks. In this project I have firstly proposed a two-level voice biometric system consisting of a Password detection module and an End-to-End Text-Dependent Speaker Verification (TDSV) module which is a modified version of a baseline TDSV model. The architecture proposed in this project leverages the two-way (previous and future) context capturing capability of a Bidirectional LSTM and also propagates it to a 3 layer architecture, since the training dataset is relatively small compared to the baseline dataset. Secondly, the domain of Text-independent speaker verification is explored with End-to-End learning and an attention module over the baseline model architecture is utilized in order to better capture the essential context for each cell of the LSTM. Both approaches have shown promising results which along with the proposed architectures and implementations, have been discussed in greater depth in the later sections.

ACKNOWLEDGEMENT

It gives me immense pleasure in bringing out this report for my 8th semester-final year B.Tech project entitled “End-to-End Speaker Verification”. I would like to express my sincere gratitude to my Supervisor and Associate Professor of Electronics and Communication Engineering of National Institute of Technology, Silchar, Dr. R.H. Laskar sir for extending his kind help, guidance and sustained inspiration for the work on the project. I would also like to express my sincere gratitude to Dr. Mohammad Azharuddin Laskar sir for his valuable suggestions and guiding me in every step of the project.

I deem it to be a solemn duty on my part to express a deep gratitude to the faculties and other staff members of Electronics and Communication Engineering Department, NIT Silchar, for providing me with the lab facilities and all the necessary help. I would also like to express my deepest appreciation to my peers for motivating and helping me in this project.

With sincere thanks and gratitude,

Abhigyan Ganguly

Table of Contents

1. INTRODUCTION

Speaker Verification is a 1-1 check for the specific enrolled voice and the new voice that comes under the umbrella of Speaker Recognition which is a 1-N task. They can be compared to Binary classification and Multi-class classification respectively. Speaker Verification is the process of verifying, based on a speaker's known utterances(voice sample), whether an (test) utterance belongs to the speaker or to an imposter. There are primarily two types of Speaker Verification : Text dependent and Text Independent. When the spoken utterances are constrained to a single word or phrase across all users, this variant is called text-dependent speaker verification. This compensates for phonetic variability, which poses a significant challenge in speaker verification. Text Independent Speaker verification has no such constraints and thus due to the increased phonetic variability, there is a requirement of large training data and training time, since the model needs to generalise well to all speaker utterances.

End-to-End Machine Learning refers to a training paradigm whereby a possibly complex learning system is represented by a single model. The idea is to model lengthy processing pipelines into one black-box type architecture. This reduces the domain specific knowledge requirement and simplifies the model training and architecture, at the expense of an increased training data requirement and training time.

1.1 Problem Definition

Hence the task at hand is to implement an Efficient Speaker Verification system with capability of both TD as well as TI recognition whilst incorporating the state of the art End-to-End ML paradigm along with the proposed novel ideas. The overall outcome goal is to have a product of voice biometric security and hence an Initial Password Word Detection system is also developed along with the SV module.

Speaker verification is a key technology in the field of voice biometrics and personal security in the 21st century. A robust, efficient and cost effective real-time implementation is thus crucial to

ensure that this feature can be installed in scenarios with varied background conditions and phonetic variability

1.2 Project Overview :

This project is based in the domain of Speaker Verification using the End-to-End Machine Learning paradigm. The project is broadly divided in two directions, the first half deals with developing a two-level voice biometric system consisting of a Password detection module and an End-to-End Text Dependent Speaker Verification Module. The second half deals with addressing the Text Independent Speaker verification and probes into the use of an Attention Module to improve the performance of the same. Both The speaker verification modules are inspired from the same baseline End-to-End model and certain changes and additions in the architecture are proposed with relevant motivations in order to improve the overall efficiency of the system.

2. LITERATURE SURVEY

S.No	Title	Journal/ Conference Name	Year of Publication	Findings
1.	Speaker recognition by machines and humans: A tutorial review	IEEE Signal Processing Magazine	2015	This paper gave a tutorial overview and introduction to the various aspects and fundamentals of Speaker recognition. It also introduced traditional speaker recognition methods.
2.	Locally-Connected and Convolutional Neural Networks for Small Footprint Speaker Recognition	INTERSPEECH 2015	2015	Here the authors presented a Deep Learning based approach to speaker recognition and CNN implementations to reduce model size.
3.	End-to-end text-dependent speaker verification	IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)	2016	This paper presents an 2End-to-End approach to this task; thus significantly simplifying the architecture and removing the enrollment stage. I would like to implement a variant based on this system as a baseline.
4.	Generalized end-to-end loss for speaker verification.	IEEE ICASSP	2018	The authors of this paper present a new,more efficient Loss function for speaker verification tasks, which reduced both EER and training time.
5.	Deep Learning for Audio Signal Processing	Journal of selected topics of signal processing	2019	In this paper the authors discussed the preprocessing aspects of speech based tasks, and their role in the corresponding sequence model.

2.1 Existing Systems :

2.1.1 Traditional Speaker Verification Overview:

There are 3 main phases in Speaker verification:

a. Training : In the training stage, a suitable internal speaker representation is obtained from the utterance, allowing for a simple scoring function. This representation depends on the type of the model (such as a Gaussian subspace model or deep neural network), the representation level (e.g., frame or utterance), and the model training loss (e.g., maximum likelihood or softmax). State-of-the art representations are a summary of frame-level information, such as i-vectors and d-vectors.

b. Enrollment : In the enrollment stage, a speaker provides a few utterances which are used to estimate a speaker model. A common choice is to average the i-vectors or d-vectors of these utterances.

c. Evaluation/Testing : During the evaluation stage, the verification task is performed and the system is evaluated. For verification, the value of a scoring function of the utterance X and the test speaker *spk*, *S(X, spk)*, is compared against a predefined threshold. The speaker utterance is accepted if the score exceeds the threshold, i.e., the utterance X comes from speaker spk, and is rejected otherwise.
A common scoring function used is the Cosine Similarity given as,

$$S(X, spk) = [f(X)^T m_{spk}]/[||f(X)||\ ||m_{spk}||]$$

Where, …. (1)

m_{spk} is the speaker model, and X is the given evaluation utterance.

Based on this scoring function, a Similarity Matrix is generated for a batch of N speakers, each with M utterances (batch size = N*M). M individual utterances of each of the N speakers are enrolled to obtain the respective speaker model and then this speaker model is compared with M utterances each of the remaining N-1 speakers as well as the true speaker's utterances, and this gives one similarity matrix for that particular speaker.. The process is similarly repeated for the rest N-1 speakers, to obtain a total of N similarity matrices for one batch.
If the value of the cosine similarity is greater than a certain threshold then the model classifies the utterance to be that of the speaker otherwise the model classifies the utterance as that of an imposter and rejects it.

Fig 1. Traditional Speaker Verification Approach

Some of the Models/Techniques used in speaker verification in the past are:

a. Gaussian Mixture Model-Universal Background Model
b. Support vector machines.
c. I-vector approach etc.

2.1.2 Deep Learning Based Approach (D-vector):

This approach utilises Deep Neural Network based architectures in speaker verification. 'D-vectors' are derived from a deep neural network (DNN), as a speaker representation of an utterance. A DNN consists of the series of layers of different(or same) nonlinear activation functions in order to transform the speaker utterance into a vector where a decision can be easily made.

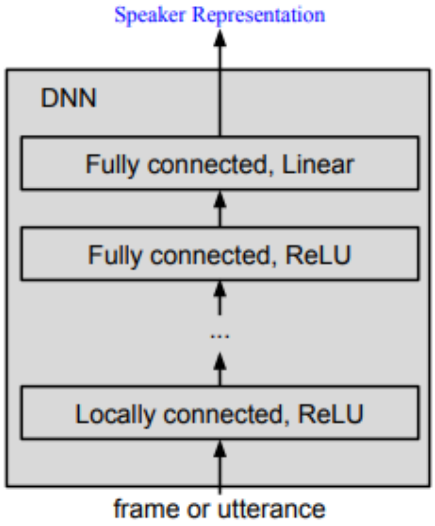


Fig 2: DNN-based(d-vector) approach for Speaker Verification

a. All layers use ReLU activated except the last, which is linear, after which a softmax function is applied.
b. During the training stage, the parameters of the DNN are optimized using the softmax loss:

$$l_{softmax} = -\log \frac{\exp(w_{spk}^T y + b_{spk})}{\sum_{s\bar{p}k} \exp(w_{s\bar{p}k}^T y + {}_{s\bar{p}k})} \quad \ldots (2)$$

where, w_{spk} and b_{spk} denote the weights and biases

the activation vector of the last hidden layer is denoted by *y* and *spk* denotes the correct speaker and *spk'* denotes the competing training speakers (imposters).

c. During Enrollment, the speaker model(Utterance d-vectors) of a speaker is obtained by averaging the activation vectors of the last hidden layer for all frames of an utterance, and then averaging over the enrollment utterances.
d. During Evaluation, a scoring function is used between the speaker model d-vector and the d-vector of a test utterance.

Constraints with this approach :

i. Limited context in speech representation in the d-vectors.
ii. The softmax loss doesn't follow the typical speaker verification protocol and is more apt for a multi-class classification problem than a binary classification counterpart.
iii. Also the softmax loss does not scale well with increased data as is the case with End to End learning, since the computational complexity increases linearly with the increase in number of speakers in training data.

2.1.3 End-to-End Baseline Approach

In the current scenario of Big data and ever increasing need for accuracy and credible real-time performance, End-to-End machine learning systems are becoming highly lucrative as they greatly simplify the architecture and reduce the number of disconnected blocks in the model.

To summarize :

i. Reduced requirement of domain specific knowledge to solve the problem.
ii. End-to-End systems are easier to maintain and deploy as a singular pipeline.
iii. Highly accurate after sufficient training and data inputs.
iv. Gradient estimation(Backpropagation) and Parameter updates of the entire network are jointly optimized, which greatly simplifies training.
v. Can handle minimal footprint Speaker Verification efficiently.

Hence, I would like to implement my End-to-End Speaker verification model based of of the given baseline model below :

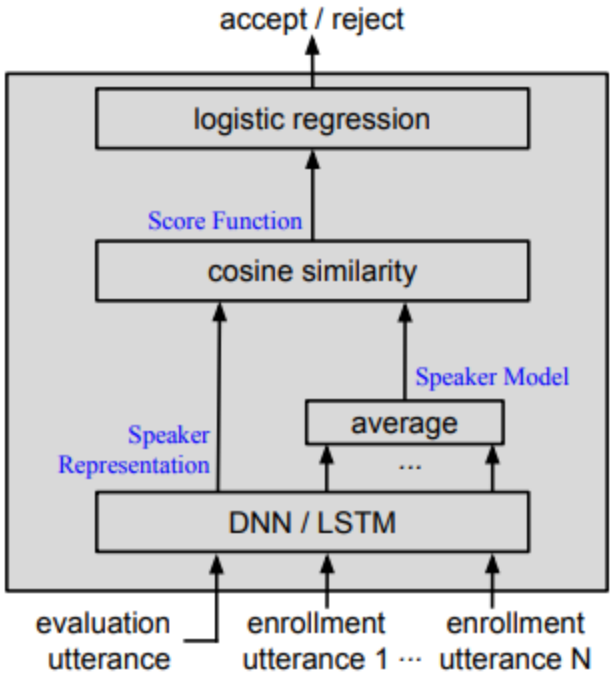


Fig 3: End-to-End Speaker Verification model[2]

The salient features of this model are :

i. Integrating the steps of the speaker verification protocol into a single network
ii. The input of this network consists of an ”evaluation” utterance and a small set of ”enrollment” utterances (N+1 in total).
iii. The output is a single node indicating accept or reject
iv. The cosine similarity(scoring function) between the speaker representation and the speaker model, *S(X, spk)* is computed, and fed into a logistic regression layer, to obtain the final output
v. If the output is above a certain threshold then the speaker utterance is accepted, else rejected.
vi. The architecture is optimized using the end-to-end loss.
vii. After the training step, all network weights are kept fixed, except for the bias of the one-dimensional logistic regression which is manually tuned on the enrollment data.
viii. There is no enrollment stage as a result.

ix. At test time, we feed an evaluation utterance and the enrollment utterances of a speaker to be tested to the network, which directly outputs the decision.

x. The following network is used to obtain the speaker representation :

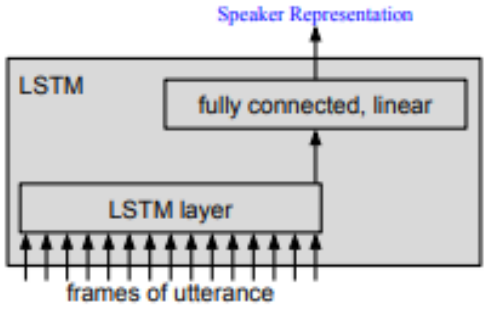


Fig 4: Structure of the DNN/LSTM layer shown in Fig 3**.**

2.2 Proposed System and Ideas :

One of the key drawbacks of any End-to-End Machine Learning system is that they require tremendous amounts of training data to perform at decent accuracy. Most of the end-to-end systems implemented so far have been trained on very large datasets and are deployed in applications where there is a large variety in the training data distribution.
This poses a problem for implementation of such a system or learning paradigm to smaller-scale target areas where availability of a large training data is not feasible. These typically include small offices, schools or restaurants where a speaker verification system is needed to be deployed for the employees.

Hence I would like to propose a

a) Two-level voice biometric system consisting of :

i) A Password word detection system : This will act as the first layer of security and the system will only accept those speakers who utter the right password. This would help reject any Correct-non target and Imposter-non target speakers at the very onset.

ii) An E2E three layered Bi-LSTM based TDSV module.
This is a modification of the original E2E TDSV model[1], and the idea is to be able to better capture the context of the utterance by using a Bi-LSTM.

b) A TISV model with attention module.
The behaviour of the E2E baseline model is explored under Text independent scenario, since TISV provides more flexibility to the user, and then an attention module based modification is proposed and tested. The ability of the attention module to capture small-footprint based context, is the motivating factor here for variable length utterances.

The proposed ideas are discussed below in depth :

2.2.1 Two-level voice biometric system :

I. Password Detection System :

As previously mentioned, the password-check system that will only allow speakers who utter the correct password-word to pass. In the obtained dataset, the password used was : “Activate”, and all other words/phrases are treated as negative utterances.

Some specifications of the model are :

a. The dataset consists of 10 second audio clips containing both positive and negative words.
b. In order to increase training data, data augmentation is performed by superimposing random background on the audio clips.
c. The output time steps predict a value between 0 and 1 (probability) and if this value is greater than a certain threshold (t), then the model makes a “bell” sound (during testing) to indicate that the password word is correct and the speaker has been accepted.

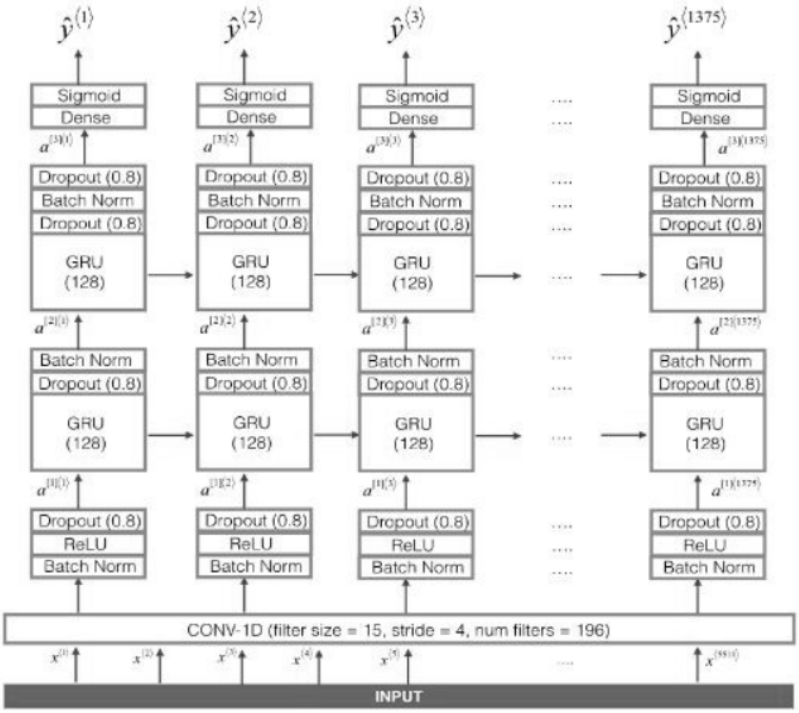


Fig 5 : Model architecture

A. **Data Augmentation :**

Data augmentation helps to generate synthetic data from existing dataset such that generalisation capability of the model can be improved.

The test set consisted of a number of audio samples from few users, which created an unnecessary bias. Hence augmentation on the test set is also proposed to improve generalisation. The dataset for password detection is augmented using four augmentation methods namely Noise Injection, Time shifting, Changing pitch and changing speed of utterance. The CSTR VCTK dataset is however only noise augmented (for making the model robust to noise) since it has relatively sufficient and varied training data.

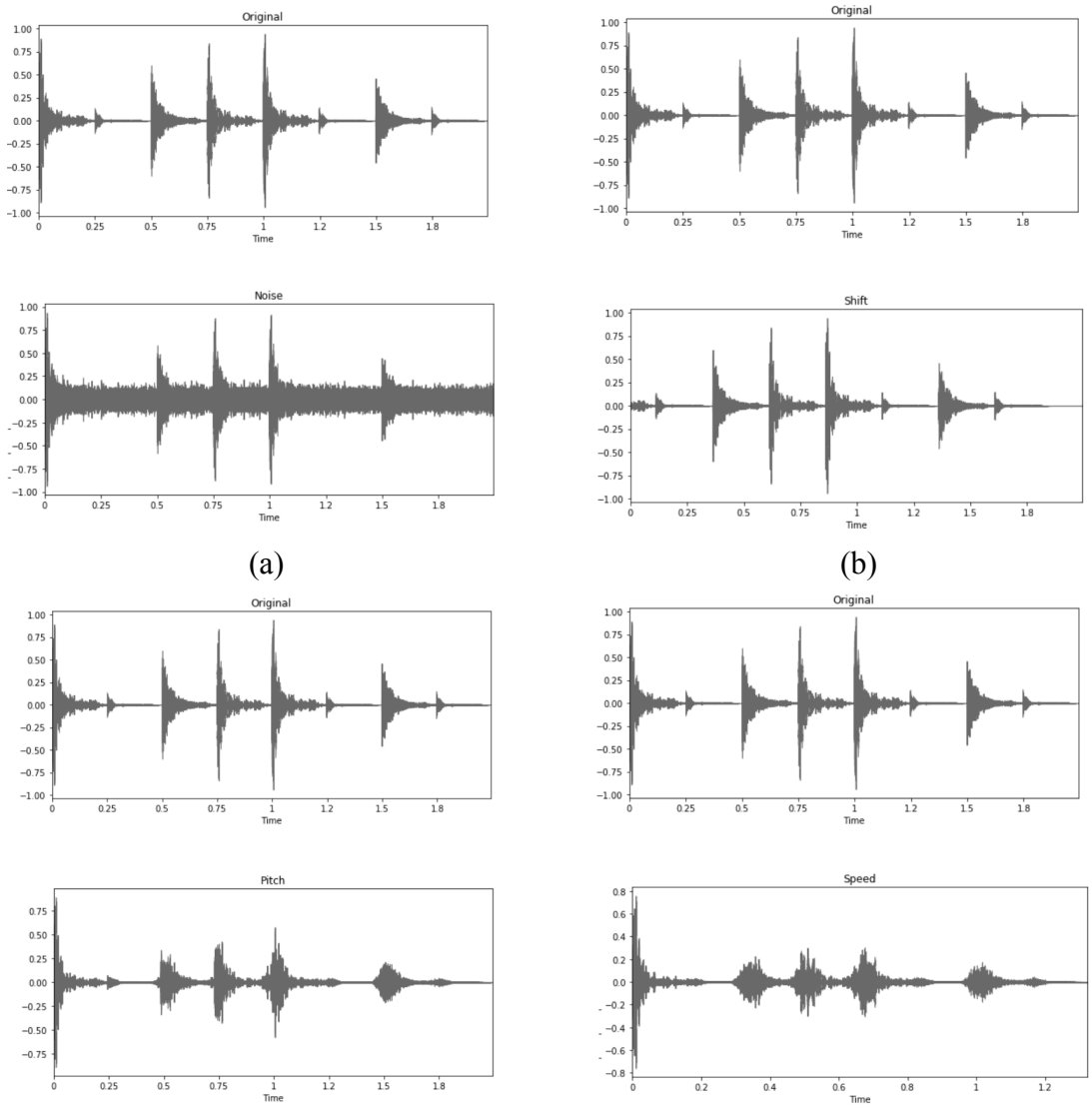


Fig 6: The four audio-data augmentation techniques :

(a) Noise Injection (b) Time Shifting (c) Changing Pitch (d) Changing Speed

II. End-to-End Text dependent Speaker Verification Module :

As Text Dependent verification requires a certain password/passphrase, the dataset is preprocessed to extract the text dependent utterances, i.e the common utterances among the speakers. This can be avoided if an exclusively Text dependent dataset is available. Since the above process gives a limited training data after filtering, data augmentation is performed using noise. Then The Baseline model is implemented on the database and its performance evaluated. Thereafter the following modifications are proposed to the model and the results are compared to check if there is an improvement in its performance.

a. Bidirectional-LSTM in 3 Layered Approach

It is known that Text Dependent Speaker verification is predominantly reliant on the context of the speaker utterance. In other words this means that unlike the Text Independent scenario, here the model judges not only who is speaking but what is being said as well (the passphrase/password). Although it is common practice to use LSTMs in order to gauge the context, I would like to propose using a Bidirectional LSTM or Bi-LSTM for short which can capture both the past as well as future context in training, since every cell takes the cell outputs from the previous as well as the future cells. Bi-LSTMs have shown good improvement in sequential based tasks such as Text Generation.

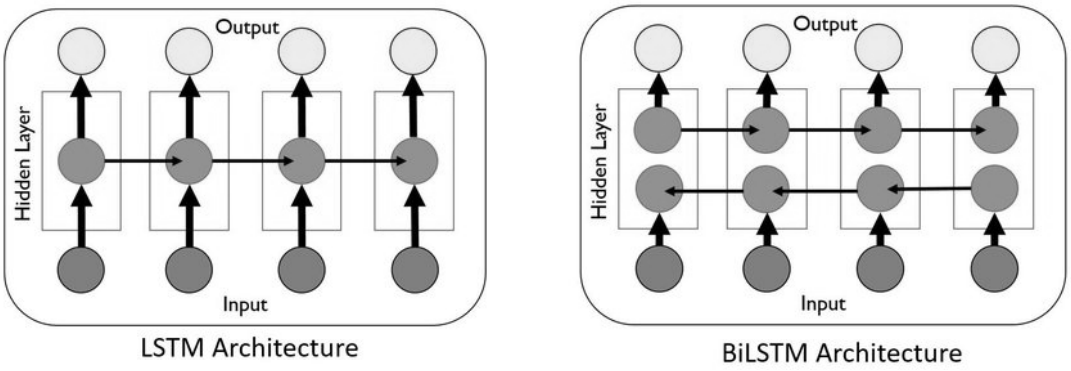


Fig 7 : Comparison between the two model architectures.

In the baseline implementation of the model, a single lstm layer was utilized to generate the speaker model from the utterances. This kept a relatively small number of training parameters, and the reason can be interpreted to be due to a wealth in training data. In my implementation I

used a multi-layered approach consisting of 3 lstm layers in order to improve the model performance as training data is limited when compared with the 'Ok Google' dataset of the baseline model.

b. Generalized-E2E Loss :

The generalized E2E loss as described in [7] is used as the loss function in our implementation as opposed to a Tuple Loss (TE2E Loss) or contrast (distance based) loss, since most tuples are taken randomly from the dataset. As a result there tends to be a bias in the enrollment utterances that are selected eventually leading to inefficient training.

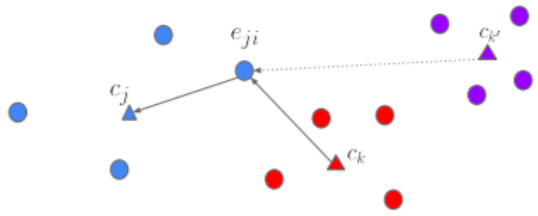


Fig 8: Speaker embedding e_{ji} (obtained from lstm) and the enrolled speaker embeddings

During the training, we want the embedding of each utterance to be similar to the centroid of all that speaker's embeddings, while at the same time, far from other speakers' centroids. Closest fake speaker to target speaker c_j in Fig is c_k.

The similarity matrix(Sji,j) is generated with respect to the c_k and e_{ji} using cosine similarity

Then optimized using the Generalized E2E loss function (softmax loss) :

$$L(\mathbf{e}_{ji}) = \mathbf{S}_{ji,j} - \log \sum_{k=1}^{N} \exp(\mathbf{S}_{ji,k}).$$

…. (3)

2.2.2 E2E Text-Independent Speaker Verification :

Text Independent doesn't require a fixed password/phrase, hence the entire labelled audio dataset can be utilized for training purposes. Having a Text-independent verification system increases the flexibility from the user perspective and makes it more convenient and futuristic, which makes it a motivating course to explore. However with an increase in training data, there comes a pitfall in the fact that the model is now tasked with

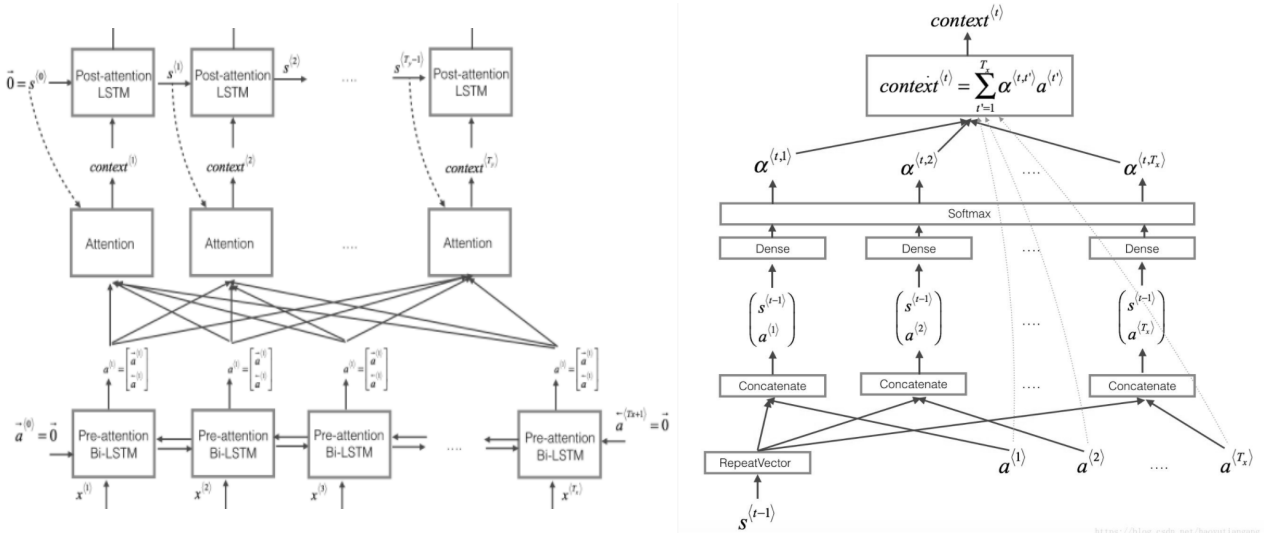
recognising a speaker from an imposter over any possible speaker utterance. This increased variability input makes it difficult for the model to give the same performance metric as compared to its TDSV counterpart under the same training conditions and architecture. The Attention Module is hence proposed in the motivation that it will help capture unique speaker phonetics from the utterance to a certain extent, as it has shown good results in previous context based sequential tasks : such as machine translation, speech recognition[11]

The Baseline model is first investigated on the dataset and then the proposed Attention Module is added to the architecture (as described below), its performance evaluated and the results compared.

a. **Text Independent with Attention :**

   To make the same E2E architecture feasible for smaller training sets and application areas with small phonetic diversity among the speakers, I would like to introduce an attention layer between the lstm layers of the model. The attention layer has been an important feature in speech recognition, especially small-footprint speech recognition, where each lstm/rnn cell learns to pay attention to only the required number of previous level cells.

   I believe the same can play an important role in minimal-footprint Speaker verification tasks and TISV in general.



(a) (b)

Fig 9: (a) The Attention layer between the two LSTMs

(b) Computations in a single attention cell

3. IMPLEMENTATION AND DESIGN

3.1 Datasets :

1) CSTR VCTK Corpus :

This corpus includes speech data uttered by 110 English speakers with various accents. Each speaker reads out about 400 sentences. The First utterance of each speaker ("Please call Stella") is used for Text-dependent training. A background noisy dataset is also provided which can be used for data augmentation. This dataset has been used in this project for training and testing purposes.

2) RSR 2015 :

Database for Text-Dependent Speaker Verification using Multiple Pass-Phrases.71 hours of recorded speech, with 300 participants (143 female and 157 male speakers) from 17 to 42 years old. With better computational capability, the proposed models are intended to be trained and tested on this dataset.

3.2 Two-level voice-biometric model :

3.2.1 Password Detection System

1) Input Features : CNN- Spectrogram

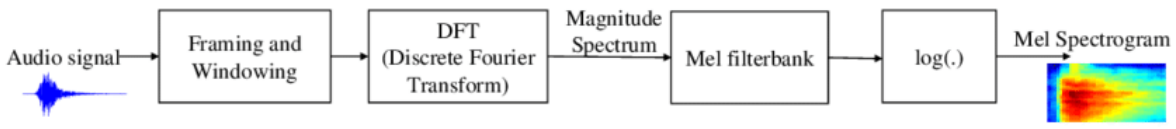


Fig 10: Mel Spectrogram fed as input to the Model (Fig)

Librosa (library for Recognition and Organization of Speech and Audio) and numpy are the two libraries used for augmenting the dataset.

2) Hyperparameters :

After training the model on various hit and trial combinations, these hyperparameters values were settled on :

a) Learning Algorithm : Adam Optimization algorithm
b) Learning rate = 0.0001,
c) β_1=0.9,
d) β_2=0.999
e) Learning rate decay =0.01
f) Loss function : binary cross-entropy (log loss function)
g) Epochs : Initially 200 on non augmented dataset (pre-training), then 50 Epochs further on the augmented dataset.

3.2.2 End-to-End TDSV System with 3-layer Bi-directional LSTM:

1) Input Features : MFCC (Mel Frequency Cepstral Coefficients)

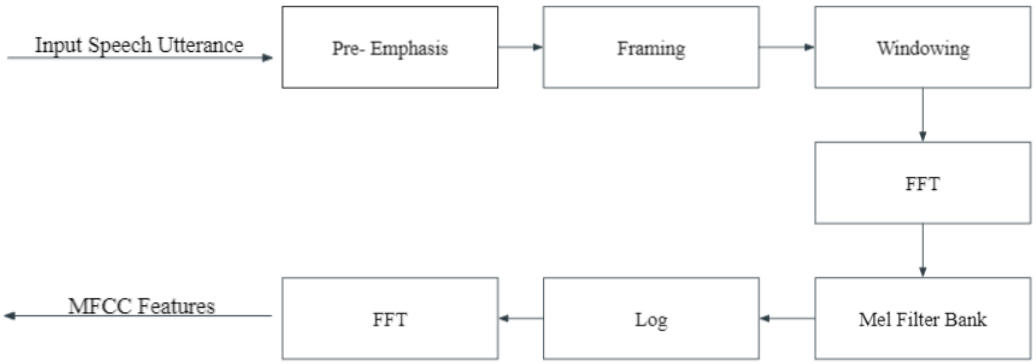


Fig 11: MFCC feature extraction process

Preprocessing parameter values :

a) Sampling rate : 8000
b) Fft kernel size : 512
c) window length (ms) : 0.025
d) hop size (ms) : 0.01

2) Model Architecture :

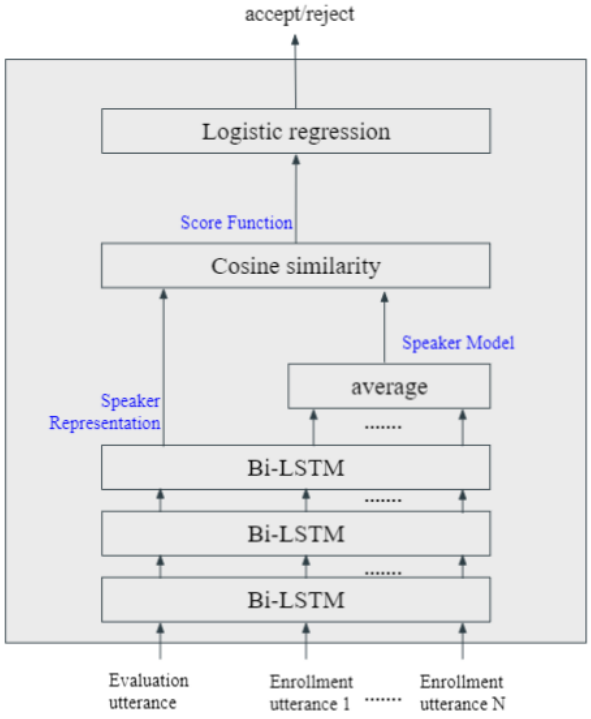


Fig 12: Proposed TDSV architecture

3) Hyperparameters :

After training the model on various hit and trial combinations, these hyperparameters values were settled on :

a) Learning Rate : 0.01
b) Learning Method : Batch Gradient descent/RmsProp/Adam
c) N, number of speakers per batch : 4 (default)
d) M, number of utterances per speaker : 5 (default)
e) β_1=0.5,
f) β_2=0.9
g) Epochs : 100000
h) Alpha-decay : 0.5
i) Mini batch size : 20 utterances, (5 utterances from 4 speakers)
j) Train/test split : 0.9/0.1
k) Loss function : Generalized end-to-end loss.

3.3 End-to-End TISV System with Attention Model:

1) Input Features : MFCC (Mel Frequency Cepstral Coefficients)

   In order to deal with the variable length of utterance in Text Independent scenario, we perform the sliding window operation to compute the mfcc features and feed it to the LSTM under the following condition :

   a) If utterance length > minimum length :

      Take the first 180 frames and last 180 frames, and feed it to the model.

   b) Else

      Feed the first 180 frames to the model.

   where minimum length = (Number of frames*hop_size + window size)*sampling rate

2) Model Architecture :

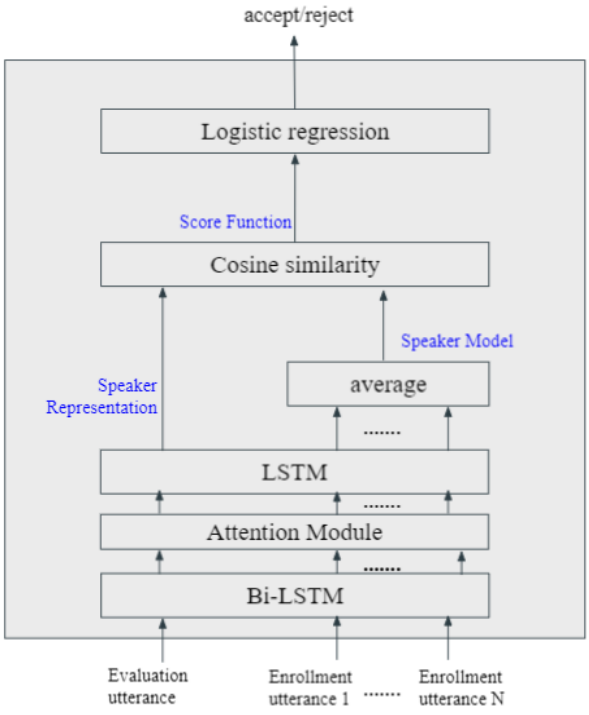


Fig 13: Proposed attention based TISV architecture

3) Attention Algorithm pseudo code

   a) Single cell attention :

      A function is defined for a single attention cell with the algorithm :

I. Taking the previous hidden state (s_prev) of the post attention LSTM and repeating it Tx times, where Tx is the number of time steps in the pre-attention Bi-LSTM.

s_prev = repeater(s_prev)

II. Concatenating ‘a’ (the hidden state output of all the pre-attention Bi-LSTM cells) with s_prev

concat = concatenator([a, s_prev])

III. Using a dense small fully-connected layer to propagate contact to compute the "intermediate energies" variable e.

e = dense1(concat)

IV. Using a second dense fully-connected layer to compute the "energies" variable energies.

energies = densor2(e)

V. Finally using a softmax activation function on the energies to compute the attention weights “alphas”..

alphas = softmax(energies)

VI. Taking the dot product of "alphas" with "a" to compute the context vector to pass on to the next post-attention LSTM-cell.

context = dot([alphas, a]).

b) Iterating over each of the Ty(number of post attention LSTM cells) cells and passing the context vector of the corresponding attention cell below, and computing the hidden state output and the activation output of that cell.

4) Hyperparameters : Same as TDSV

3.4 Software and Hardware Specifications :

1. Operating System support : Windows/Mac/Linux.
2. CPU Processor : AMD Ryzen 5/ Intel Core i7 or upwards preferred
3. CPU RAM : 8GB, Quad-Core.
4. GPU (recommended) : Radeon RX 560x (8GB) or equivalent and upwards.

5. *If Nvidia GPU is available, then CUDA support can be leveraged however it wasn't used in this project.
6. Programming Language Breakdown : Python (100%)
7. Tools and Utilities : Anaconda, Jupyter Notebook, Spyder, Google Colab.
8. Specific Libraries to install : Librosa, Tensorflow-Keras, Scikit learn.

4. RESULTS

4.1 Two-level voice-biometric model

4.1.1 Password Detection System :

a. Training loss/accuracy after 50 epochs :

```
In [23]: model.fit(X, Y, batch_size = 5, epochs=50)

Epoch 48/50
1/1 [==============================] - 0s 6ms/step - loss: 0.4275 - accuracy: 0.9543
Epoch 49/50
1/1 [==============================] - 0s 3ms/step - loss: 0.4263 - accuracy: 0.9549
Epoch 50/50
1/1 [==============================] - 0s 2ms/step - loss: 0.4267 - accuracy: 0.9555

Out[23]: <tensorflow.python.keras.callbacks.History at 0x1bfc49ef3d0>
```

- Accuracy : 95.55 %
- Loss : 0.4267

b. Test set accuracy :

```
In [24]: loss, acc = model.evaluate(X_dev, Y_dev)
         print("Dev set accuracy = ", acc)

         1/1 [==============================] - 0s 3ms/step - loss: 0.3188 - accuracy: 0.9540
         Dev set accuracy =  0.9540363550186157
```

i. Loss : 0.3188

ii. Accuracy : 95.4%

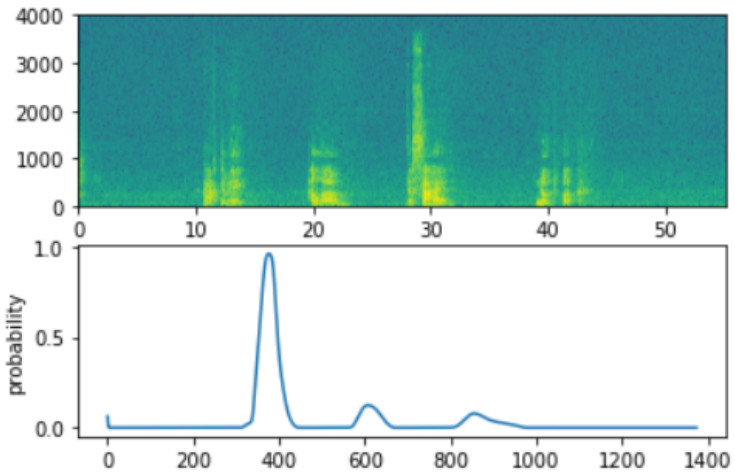


Fig 14: Model's output probability graph for a given input spectrogram.

From the above test set performance we infer :

iii. Model generalises better to noisy environments as well, since the dataset has been augmented with noise.

iv. Test set accuracy is improved by about 4% after performing data augmentation which shows that Overfitting of the model on the training set has also been reduced.

4.1.2 End-to-End TDSV System with Bi-directional LSTM:

a. Hyperparameters :
   - Learning Rate : 0.01
   - Learning Method : Batch Gradient descent
   - Epochs : 100000
   - Alpha-decay : 0.5
   - Mini batch size : 20 utterances, (5 utterances from 4 speakers)
   - Train/test split : 0.9/0.1
   - Loss function : Generalized end-to-end loss.

a. Dataset : CSTR VCTK dataset with added noise augmentation.

1. Baseline Model and Proposed Model Performance comparison:

The Hyperparameters are as mentioned except for the Epoch which is kept at 10000 iterations.

i. Baseline Performance

```
(iter : 9000) loss: 0.2504
model is saved!
(iter : 9100) loss: 0.2583
(iter : 9200) loss: 0.3584
(iter : 9300) loss: 0.2456
(iter : 9400) loss: 0.2267
(iter : 9500) loss: 0.1921
(iter : 9600) loss: 0.2410
(iter : 9700) loss: 0.2770
(iter : 9800) loss: 0.3261
(iter : 9900) loss: 0.1995
(iter : 10000) loss: 0.2888
model is saved!

E:\Projects\Final Year Project\Speaker Verificat
```

Fig 15: Loss over the last few training iterations.

```
C:\Windows\System32\cmd.exe
ckpt file is loaded ! ./tdsv_model_10k\Check_P
test file path :  ./test_tdsv
inference time for 40 utterences : 2.23s
[[[ 0.99 -0.13  0.78  0.23]
  [ 0.99 -0.13  0.79  0.26]
  [ 1.   -0.12  0.81  0.26]
  [ 0.99 -0.11  0.79  0.27]
  [ 0.99 -0.14  0.79  0.24]]

 [[-0.13  0.96 -0.04 -0.01]
  [-0.12  0.96 -0.05  0.03]
  [-0.14  0.96 -0.06  0.02]
  [-0.12  0.96 -0.03  0.04]
  [-0.13  0.95 -0.03  0.01]]

 [[ 0.82  0.06  0.98  0.21]
  [ 0.84  0.06  0.98  0.26]
  [ 0.85  0.04  0.98  0.26]
  [ 0.87 -0.01  0.98  0.21]
  [ 0.74 -0.04  0.96  0.23]]

 [[ 0.26  0.09  0.04  0.8 ]
  [ 0.31  0.07  0.09  0.83]
  [ 0.33  0.05  0.12  0.83]
  [ 0.37  0.03  0.21  0.88]
  [ 0.37  0.01  0.25  0.86]]]

EER : 0.05 (thres:0.83, FAR:0.05, FRR:0.05)

E:\Projects\Final_Year_Project\Speaker_Verific
```

Fig 16: Similarity Matrix for N=4,M=5 speaker batch.

ii. Baseline with 3-layered Bi-LSTM

```
(iter : 9100) loss: 0.2819
(iter : 9200) loss: 0.3610
(iter : 9300) loss: 0.2344
(iter : 9400) loss: 0.1844
(iter : 9500) loss: 0.4372
(iter : 9600) loss: 0.2168
(iter : 9700) loss: 0.3953
(iter : 9800) loss: 0.2552
(iter : 9900) loss: 0.3022
(iter : 10000) loss: 0.2656
model is saved!

E:\Projects\Final_Year_Project\Speak
```

Fig 17: Loss over the last few training iterations.

```
ckpt file is loaded ! ./tdsv_model_bi\Check_Point\mode
test file path :  ./test_tdsv_bi
inference time for 36 utterences : 0.28s
[[[ 9.85e-01  5.18e-02  7.11e-01]
  [ 9.87e-01  4.82e-02  6.56e-01]
  [ 9.87e-01  6.19e-02  6.72e-01]
  [ 9.91e-01  5.10e-02  6.80e-01]
  [ 9.89e-01  4.74e-02  6.81e-01]
  [ 9.88e-01  7.08e-02  6.81e-01]]

 [[ 3.23e-04  9.94e-01 -1.91e-01]
  [-1.58e-03  9.95e-01 -1.93e-01]
  [ 1.33e-02  9.95e-01 -1.79e-01]
  [ 1.04e-02  9.95e-01 -1.83e-01]
  [ 1.77e-03  9.94e-01 -1.91e-01]
  [ 9.81e-03  9.94e-01 -1.84e-01]]

 [[ 7.97e-01 -1.80e-01  9.55e-01]
  [ 7.78e-01 -1.60e-01  9.58e-01]
  [ 7.27e-01 -1.20e-01  8.81e-01]
  [ 7.62e-01 -5.87e-02  9.24e-01]
  [ 7.56e-01 -2.03e-01  9.59e-01]
  [ 7.54e-01 -1.30e-01  9.22e-01]]]

EER : 0.00 (thres:0.80, FAR:0.00, FRR:0.00)

E:\Projects\Final_Year_Project\Speaker_Verification-ma
```

Fig 18: Similarity Matrix for N=3,M=6 speaker batch.

iii. Performance Comparison after 10000 epochs :

Models	Loss	EER	Threshold
Baseline Model	0.2888	0.05	0.83
Proposed Model	0.2656	0.00	0.80

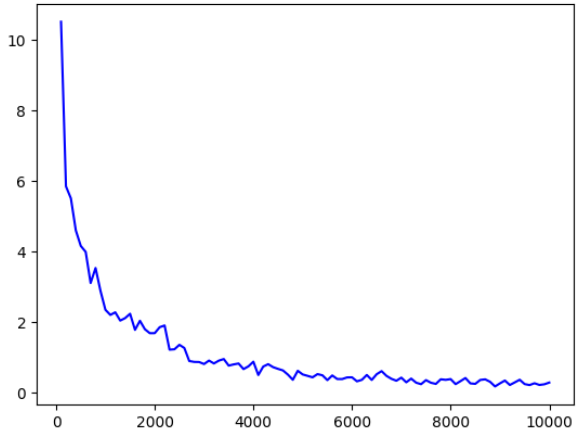


Fig 19: Training loss with iterations for proposed model

2. Loss comparison under different learning algorithms for Proposed Model :

a. Batch Gradient Descent : Final Loss : 0.2832

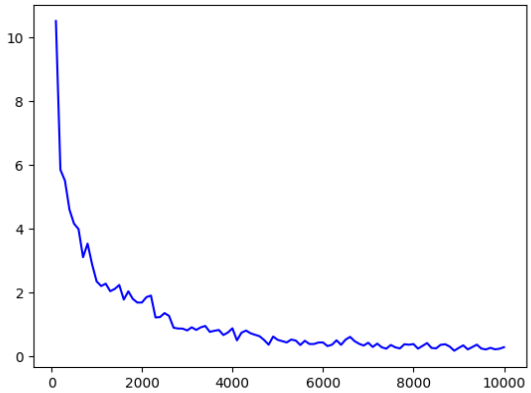


Fig 20: Loss v/s Iterations for Batch Gradient descent

b. Rms Prop : Loss = 4.3494

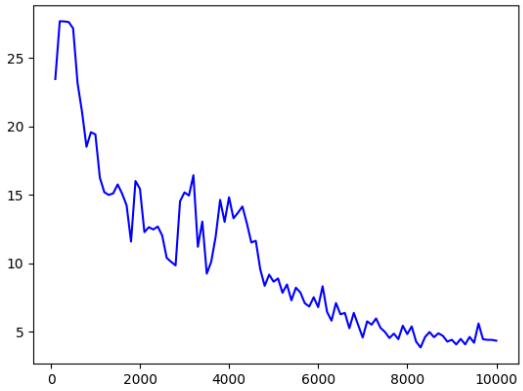


Fig 21: Loss v/s Iterations for Rms Prop

c. Adam : Final Loss = 1.1446

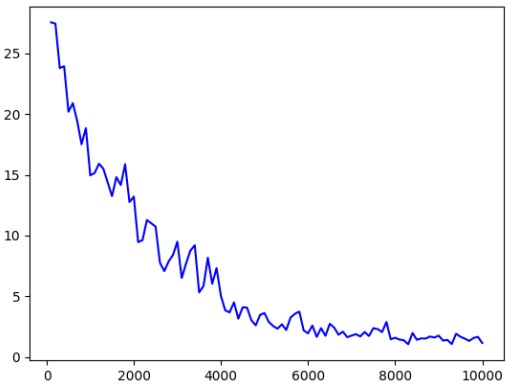


Fig 22: Loss v/s Iterations for Adam Optimizer

Loss function	Loss	Training/Convergence Time
Batch Gradient Descent	0.2832	Slowest
Rms Prop	4.3494	Moderate
Adam	1.1446	Fastest

3. Results on the Proposed TDSV Model after further training

   Epochs = 100000

```
(iter : 98800) loss: 0.0588
(iter : 98900) loss: 0.0612
(iter : 99000) loss: 0.0393
(iter : 99100) loss: 0.0449
(iter : 99200) loss: 0.0802
(iter : 99300) loss: 0.0575
(iter : 99400) loss: 0.0546
(iter : 99500) loss: 0.0486
(iter : 99600) loss: 0.0584
(iter : 99700) loss: 0.0285
(iter : 99800) loss: 0.0747
(iter : 99900) loss: 0.0515
(iter : 100000) loss: 0.0688
learning rate is decayed! current lr :  9.765625e-06
model is saved!

E:\Projects\Final_Year_Project\Speaker_Verification-m
```

Fig 23: Loss over the last few training iterations.

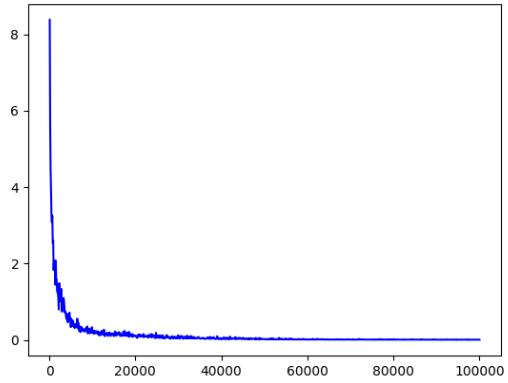


Fig 24: Loss v/s Iterations for 100000 epochs.

Loss after 100000 iterations : 0.0688

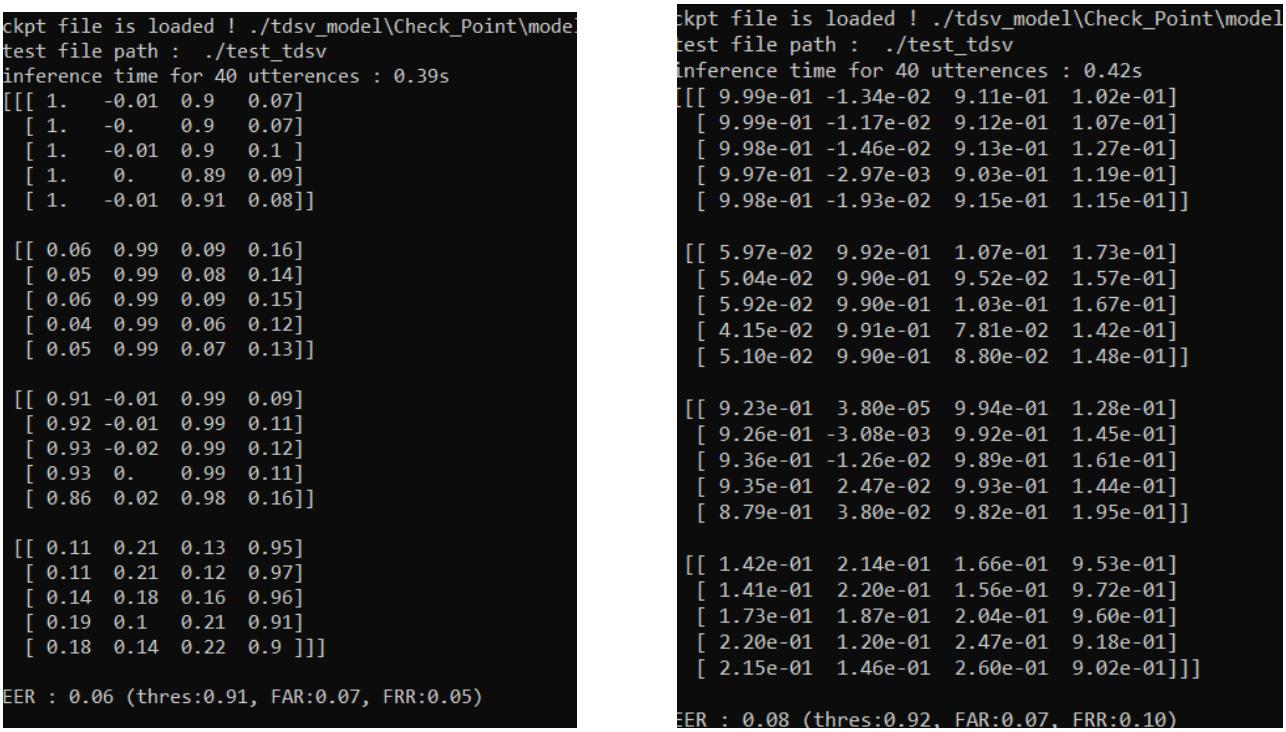


Fig 25: Similarity matrices for Model 6 and Model 7 respectively.

a. Test time comparison between Model 6 and Model 7

Models	EER	Threshold	FAR	FRR
Model 6	0.06	0.91	0.07	0.05
Model 7	0.08	0.92	0.07	0.10

Here, Model 6 : Trained model after 60000 iterations

Model 7 : Trained model after 70000 iterations

b. Comparison with Previous TDSV models (on the Ok Google Dataset) :

Models	Dataset	EER
I-vector	‘OK Google’	5.77
D-vector	‘OK Google’	2.90
E2E-DNN	‘OK Google’	1.87
E2E-LSTM	‘OK Google’	1.36
E2E-LSTM	CSTR VCTK	0.05
E2E-Tri-BiLSTM (Proposed Model)	CSTR VCTK	0.00

The results show that on the CSTR VCTK dataset the proposed model performs marginally better than the baseline model. However the performance of the proposed model on the ‘OK Google’ or a similar global-speaker level dataset would further support the claim.

4.2 End-to-End TISV System with Attention Model:

a. TISV on the Baseline Model :

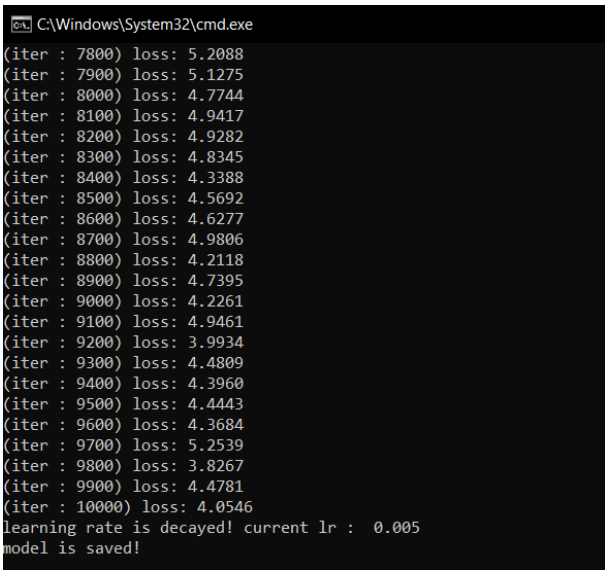


Fig 26: Loss over the last few training iterations.

```
ckpt file is loaded ! ./tisv_model\Check_Point\model.ckpt-0
test file path :  ./test_tisv
inference time for 40 utterences : 2.60s
[[[ 0.85  0.26  0.37  0.19]
  [ 0.91  0.4  -0.02  0.24]
  [ 0.85  0.37  0.04  0.2 ]
  [ 0.93  0.27  0.37  0.39]
  [ 0.79  0.18  0.57  0.08]]

 [[-0.14  0.6  -0.16  0.19]
  [ 0.63  0.73 -0.27  0.49]
  [ 0.46  0.79 -0.08  0.42]
  [ 0.07  0.37 -0.26  0.65]
  [-0.13  0.65 -0.14  0.01]]

 [[ 0.19 -0.24  0.98  0.14]
  [ 0.32 -0.39  0.92 -0.12]
  [ 0.27 -0.32  0.94 -0.04]
  [-0.   -0.25  0.9   0.14]
  [-0.01 -0.26  0.91  0.17]]

 [[ 0.25  0.44 -0.34  0.82]
  [ 0.39  0.06  0.11  0.88]
  [ 0.73  0.54 -0.17  0.74]
  [ 0.14  0.12  0.1   0.88]
  [ 0.15  0.25 -0.13  0.9 ]]]

EER : 0.05 (thres:0.57, FAR:0.05, FRR:0.05)
```

Fig 27: Similarity Matrix for N=4,M=5 speaker batch for Baseline-TISV

b. TISV with the proposed Attention Module :

```
model is saved!
(iter : 8100) loss: 4.0307
(iter : 8200) loss: 4.4742
(iter : 8300) loss: 4.8942
(iter : 8400) loss: 4.3729
(iter : 8500) loss: 3.6447
model is saved!
(iter : 8600) loss: 3.9906
(iter : 8700) loss: 4.0425
(iter : 8800) loss: 4.2146
(iter : 8900) loss: 3.9162
(iter : 9000) loss: 4.3025
model is saved!
(iter : 9100) loss: 3.7838
(iter : 9200) loss: 3.8408
(iter : 9300) loss: 3.7507
(iter : 9400) loss: 3.8965
(iter : 9500) loss: 4.1930
model is saved!
(iter : 9600) loss: 3.7915
(iter : 9700) loss: 3.9678
(iter : 9800) loss: 3.7160
(iter : 9900) loss: 3.9126
(iter : 10000) loss: 3.6649
model is saved!

E:\Projects\Final_Year_Project\Speaker_
```

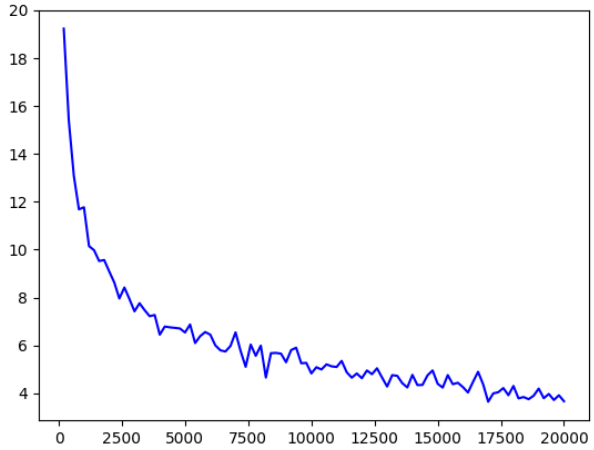


Fig 28: Loss over the last few training iterations and loss v/s iterations for 10000 epochs.

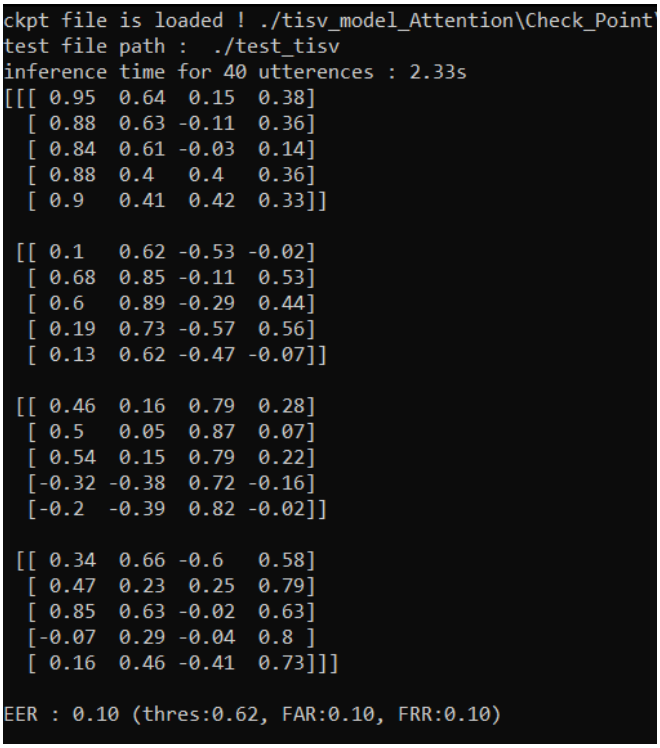


Fig 29: Similarity Matrix for N=4,M=5 speaker batch for Attention-TISV

c. Comparisons after 10000 epochs

Models	Loss	EER	Threshold
Baseline TISV	4.0546	0.05	0.57
TISV with Attention	3.6649	0.10	0.62

5. CONCLUSION AND FUTURE WORK

The two-level voice biometric system driven by End-to-End machine learning can serve as a security measure in various application scenarios in offices, homes, restaurants, schools etc, even under ambient noise . It would be expected to perform especially well in use cases involving small scale speaker variability, due to the relatively limited phonetic variations in the training dataset. The Text Independent model also showed promise when implemented with the attention model, although the results were not significantly improved. Still the user flexibility provided by TISV models is also to be taken into consideration, and hence the greater demand for a more efficient model.

In future, if greater computational capability is available, then the TISV model can be trained further for several iterations, till its performance in terms of Loss and EER can be brought down to the level of the TDSV model. Also the availability of a larger dataset such as the baseline model, having speaker variability at a global level, should be the next step forward in order to make the model more robust.

6. REFERENCES

[1] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in 2016 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[2] Yu-hsin Chen, Ignacio Lopez-Moreno, Tara N. Sainath, Mirkó Visontai, Raziel Alvarez, Carolina Parada, "Locally-Connected and Convolutional Neural Networks for Small Footprint Speaker Recognition". *INTERSPEECH 2015*. 1136-1140.

[3] Hansen, J. H., & Hasan, T. (2015). "Speaker recognition by machines and humans: A tutorial review". *IEEE Signal Processing Magazine*, 32, 74–99.

[4] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron J. Weiss, Kevin Wilson, "CNN Architectures for Large-Scale Audio Classification". IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*,* 2017.

[5] Arsha Nagrani, Joon Son Chung, Andrew Zisserman, "VoxCeleb: a large-scale speaker identification dataset". *Computer Speech & Language,* Volume 60, Elsevier*,* March 2020, 101027.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735– 1780, 1997.

[7] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. "Generalized end-to-end loss for speaker verification." IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*,* 2018.

[8] Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schlüter, Shuo-yiin Chang, Tara Sainath, "Deep Learning for Audio Signal Processing" *Journal of selected topics of signal processing*, vol. 13, no. 2, may 2019, pp. 206–219

[9] Li Wan, Quan Wang, Alan Papir, Ignacio Lopez Moreno. "Generalized End-to-End Loss for Speaker Verification". ICASSP 2018.

[10] Zacarias-Morales, Noel; Pancardo, Pablo; Hernández-Nolasco, José A.; Garcia-Constantino, Matias. "Attention-Inspired Artificial Neural Networks for Speech Processing: A Systematic Review" *Symmetry*, 2021.

[11] Changhao Shan, Junbo Zhang, Yujun Wang, Lei Xie. "Attention-based End-to-End Models for Small-Footprint Keyword Spotting",Interspeech 2018.