Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 5- Issue 1, Paper 33 January 2022

Speaker Identification from Voice

Yashu Kedia
Department Of Computing Technologies
SRM IST Kattankulathur
Kattankulathur-603203

Nagadevi S.
DepartmentOfComputingTechnologies
SRM IST Kattankulathur
Kattankulathur-603203

Abstract - The speech signal conveys information about the identity of the speaker. The area of speaker identification is concerned with extracting the identity of the person speaking the utterance. As understanding how to recognise complex, unstructured and high-dimensional voice/speech/audio data is one of the greatest challenges of our time. In this project, we proposed an idea for identifying a speaker by machine on the basis of his/her voice with the help of deep learning approach. Some of the paradigms or algorithms that will be used are MLP, CNN,RNN,LSTM and GRU.

Keywords

Convolution Neural Network (CNN). Recurrent Neural Network (RNN). Long Short-Term Memory (LSTM). Multilayer Perceptron(MLP). Gated Recurrent Unit(GRU)

1. INTRODUCTION

The objective of the machine is to identify the speaker from voice. This can only be achieved if there is a certain level of similarity in the repository of the machine for that voice and that of the speaker's voice. The next aspect of identifying a speaker is to train the machine by testing an unidentified utterance against the training data and accordingly arriving upon the judgement. The target speaker is identified as the speaker of the test utterance. In recent times there has been a lot of interest generated in the field of alternative speech parameterizations based on using features that are formant. Essential are formant frequencies for the development of speech spectrum. However this process of discovering the formants from the speech signal can be very difficult and therefore might not be possible every time. Which is why instead of assuming the resonant frequencies, what can be used is formant like features. This entire ordeal is divided into two parts:
To identify a speaker from a sample with regard to repository of samples already collected.
To verify the claimed identity of a voice signal or sample generated.

1.1 Motivation

Speaker identification makes it possible to use the speaker's voice to verify their identity. And control access to services such as voice dialing, banking by telephone, database access services, voice mail, security control for confidential information areas, and remote access to computers etc

1.2Problem Statement

Understanding how to recognize complex, unstructured, and high-dimensional voice/speech/audio data is one of the greatest challenges of our time.
Traditional (GMMs) approach suffers from an inherent assumption of linearity in speech signal dynamics. Such approaches are prone to overfitting and have problems with generalization.

Volume 5- Issue 1, Paper 33 January 2022

1.3Objective

The goal is to use the machine to determine a speaker's identification based on his or her voice. The user makes no claim to his or her identity. In this paper, we offered an idea for identifying a speaker by machine based on his or her voice using–

•**Deep Learning approach**
oMLP(Multilayer Perceptron)
oCNN(Convolutional Neural Network)
oRNN(Recurrent Neural Network)
oLSTM(Long Short Term Memory)

1.4Automatic Speech Recognition(ASR)
The listener receives multiple levels of information from the spoken signal. Speech, at its most basic level, delivers a message through words. Speech, on the other hand, original evidence about the language being said and also the speaker's emotions, gender, and general identity. Automated speaker identification systems try to extract, categorise and recognise the information included in a voice signal that conveys the speaker identity, while speech recognition systems strive to recognise the words said in a speech signal as they are uttered.
Two more fundamental tasks are encompassed within the scope of this:

1.4.1 Speaker Identification

The task of detecting who is speaking from a group of known voices or speakers is known as speaker identification. Because the unknown person makes no claim to identify, the algorithm must classify them as 1: N. The task is sometimes referred to as closed-set identification since it is thought that the unknown voice must come from a defined group of recognised speakers.

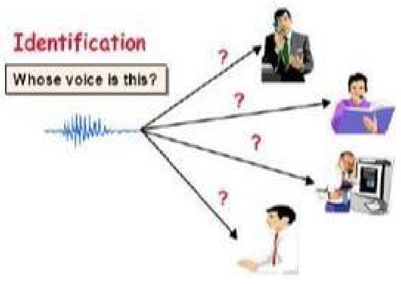


Figure 1.1 Speaker Identification (Figures courtesy of Douglas Reynolds, MIT Lincoln Labs)

1.4.2 Speaker Verification

Speaker verification (also known as speaker authentication or detection) is the task of determining whether a person is who he/she claims to be (a yes/no decision). Since it is generally assumed that imposters (those falsely claiming to be a valid user) are not known to the system, this is referred to as an open-set task. By adding a "none-of-theabove" option to closed-set identification task one can merge the two tasks for what is called open-set identification.

Figure 1.2 Speaker Verification (Figures courtesy of Douglas Reynolds, MIT Lincoln Labs)

1.5Deep Learning Technique

1.5.1Multilayer Perceptron

Feed forward artificial neural network is what defines a Multi-Layer Perceptron(MLP). The three layers that comprise an MLP are: 1) An Input Layer 2) A Hidden Layer 3) An Output Layer. A non linear activation function is what is used by every layer barring the input layer for the purpose of training. The difference between an MLP and a single layer perceptron is the fact that the former has multiple layers which in turn enablesittodistinguishbetweenthosedatawhicharen’tlinearlyseparable.

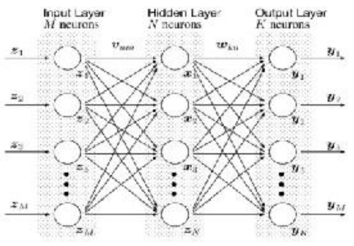


Figure 1.3 Typical Multilayer Perceptron Architecture (Figures courtesy MDPI)

1.5.2Convolution Neural Network (CNNs)

Deep learning models have yielded ground breaking achievements on a range of pattern identification tasks, including computer vision and speech recognition, in the past few years. An important component in achieving these outcomes has been a kind of neural network known as a convolution neural network.
When it comes to deep learning, convolution neural networks (also known as ConvNets) are a kind of deep neural network that is most widely used in the analysis of visual images and the recognition of speech. CNNs use a kind of multi-layer perceptrons that is meant to need the least amount of preprocessing. Their shared weights design and translation in variance features have earned them the nickname “shift invariant” or “space invariant” artificial neural networks (SIANN).
Natural processes were used as inspiration for the design of convolution networks, since the pattern of connecting between neurons mirrors the architecture of the visual cortex of an animal brain. A narrow portion of the visual field known as the receptive field is occupied by individual cortical neurons that react to visual inputs. Different neurons’ receptive fields partly overlap, allowing them to span the whole visual field. A revolutionary development in the area of computer vision, convolution neural networks have become one of the most widely used and influential tools. They have outperformed typical computer vision methods and delivered findings that are state of the art in their field. These neural networks have shown to be effective in a wide range of real-world case studies and applications, including the ones below:
Classification of Images, Detection of Objects, segmentation, Recognition of Faces;

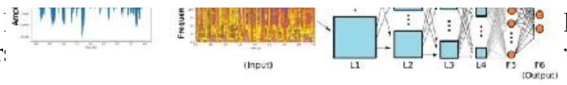
CNN based vision systems leveraged by self-driving cars;
Voice Recognition;
Etcetera.
A CNN has three layers: an input layer, a  lution, layers of RELU .e layers of activation function, layer: malising are the many components of a CNN's hidden layer.

Figure 1.4 Architecture of the CNN (adapted from [8])

1.5.3 RNN

One of the most powerful and robust type of neural network out there are Recurrent Neural Network (RNN) as they are the only ones with internal memory.
Due to their capacity to memorize, RNNs are very efficient in making predictions as they are able to remember and hence expect what is about to come.
Due to this reason precisely, they are the preferred choice of algorithm for data which is sequential like series of time, text, speech, data on finance, video, audio, climate and many more as they can develop an understanding of the deeper kind along with its context when compared to other algorithms.
.
"Whenever there is a sequence of data and that temporal dynamics that connects the data is more important than the spatial content of each individual frame."

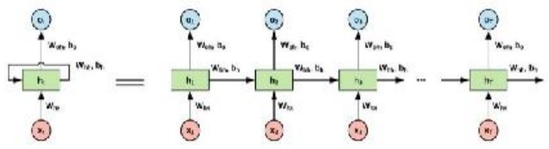


Figure 1.5 Sample RNN structure (Left) and its unfolded representation (Right)

1.5.4 LSTM

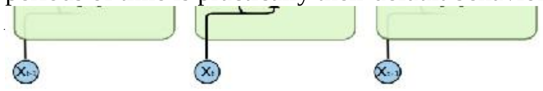
Long Short Term Memory networks usually just called "LSTMs" - are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter&Schmidhuber (1997) and were refined and popularized by many people in the following work. They work tremendously well on a large variety of problems and are now widely used.
LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn.
 hain of repeating modules of a neural network. In standard ery simple structure, such as a single tanh layer.

Figure 1.6 The repeating module in a standard RNN contains a single layer

LSTMs also have this chain-like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

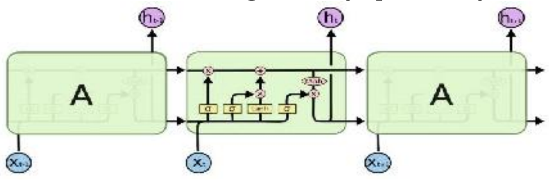


Figure 1.7 The repeating module in an LSTM contains four interacting layers

Yashu Kedi, Nagadevi S.

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)
Volume 5- Issue 1, Paper 33 January 2022

1.6Summary

MLP (Multilayer Perceptron), Convolution Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory were all introduced in this chapter (LSTM).
The rest of the report is structured as follows:
The existing solutions are presented in the second chapter. It introduces many available options as well as their downsides.

The recommended solution for Speaker Identification is presented in Chapter three. The working model's results are presented in Chapter 4 along with implementation details.

The project work is eventually completed in Chapter 5.

2. LITERATURE SURVEY

While selecting few Speaker Identification approaches under consideration, more focus was kept on recently proposed techniques as those overcome most of the limitations of previously proposed approaches in this domain.

2.1 Related work

Speaker Identification and Clustering using Convolutional Neural Networks, by: Yanick Lukic, Carlo Vogt, Oliver Durr, and ThiloStadelmann [1]
A new study shows that machine learning (ml, namely convolution neural networks (CNNs), have resulted in huge advancement in computer vision and related fields during the last several years. In part, this progress may be credited to the transition from manufacturing features and subsequently discrete semi to advanced deep learning and recognition systems from essentially unprocessed input and running them end-to-end. Speaker clustering is still a common use for customised handling sequences, such like MFCC features and GMM based models. Using basic spectrogram as input, we study the most optimal CNN architecture for speech detection and grouping in this article. The process of converting a speaker identification network into a speech clustering network is also discussed. When applied to a very well TIMIT dataset, we show that our technique can achieve results that are equivalent to the best-in-class-all without the usage of handmade features.
Using the TIMIT dataset, they achieved a 97.0 percent accuracy rate.

Speaker identification using Neural Networks on an FPGA, by: F. Trujillo-Romero and S. O. Caballero-Morales [2]
This paper contains the formulation and construction of the speech recognition system, which was created by the authors. The vectors supplied for such recognizer's neural network were encoded using Linear Prediction Coding (LPC). A DigilentNexys 2 boards with only an XILINX Spartan-3E Field Programmable Gate Array was used to build this four-layer neural network (FPGA). Ten people were trained and tested on the new system before it was made available to the public. A total of 60 samples were collected from each user, half of which were utilised for education and the other half for evaluation. This approach was able to attain an identification rate of 98 percent.
Speaker recognition from raw waveform with SincNET, by: MircoRavanelli and YoshuaBengio[3]
This original study SincNet architecture aids the very first convolution layer in discovering more relevant filters. Sinc-Net is abnd pass filtering approach based on parameterized sinc functions. Traditional CNNs learn each filter's higher and lower cut off frequencies from data, but our technique just remembers the high and low cut-off frequencies. This technique creates a small and effective bespoke filter bank that is well suited for application. When trained on raw waveforms, the suggested architecture outperforms a normal CNN in both voice recognition and speaker verification. SincNet obtained an 85 percent chance of getting on the TIMIT dataset.

Speaker identification via support vector classifiers, by: M. Schmidt and H. Gish [4]

Researchers came up with an innovative approach to identify speakers.
Using Vapnik's support vectors, the technique is fascinating for a number of reasons. With the support vector approach, instead of needing to approximate speaker concentrations, which is a cumbersome intermediary step, the boundaries between speakers' sounds are modelled explicitly in some feature space. As a result of their limited discriminating strength, vector support discriminant classifiers reduce test mistakes. Classifiers with much more attributes than instructional points maybe generated as a consequence of this method. The Vapnik's theory, on the other hand, determines which class of distinguishing functions should be utilised based on the quantity of training data by predicting the predicted number of test mistakes. In comparison to other discriminant functions, gradient boosting classifiers are quicker to calculate. The Switchboard corpus outperforms the BBN modified Gaussian Bayes judgement system by 88 percent, according to early data.

Deep Speaker: an End-to-End Neural Speaker Embedding System, by: Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, Zhenyao Zhu [5]
Cosine similarity is used to quantify the similarity of speakers in this study, which proposes a neural embedding system called Deep Speaker. The embeddings of Deep Speaker maybe utilised for a variety of applications, including speaker identification, verification and clustering. Mean pool embeddings for speaker embeddings are created using ResCNN or GRU architecture, and triplet loss is used on cosine similarity. According to three different dataset comparisons, Deep speaker looks to beat a DNN based i-vector benchmark. The verification equal mistake rate is reduced by about half (roughly) and the identification accuracy is increased by around 60 percent using Deep Speaker, for example (relatively). They also found that utilising a Mandarin-trained model improves the accuracy of the English speaker identification.

Speaker Recognition With Recurrent Neural Networks, by: Shahla Parveen, Abdul Qadeer, and PhilGreen[6]
In this paper, they use recurrent neural nets to solve an open set text dependent speaker detection problem. They employ a feed-forward net architecture based on Robinson et alwork's. Between the input and the state nodes and the output, we introduce a completely connected hidden layer. They demonstrate that this concealed layer improves the efficiency of learning complex classification tasks. Backpropagation across time is used in the training. Each speaker has one output unit, with training targets that match to speaker identification. We get a real acceptance rate of 100% with a fake acceptance rate of 4% for 12 speakers (a mix of male and female). These statistics are 94% and 7 percent for 16 speakers respectively.

2.2 Summary

This chapter has a list of prior articles and research, some of which are still in the research phase. With the developments in deep learning, we are seeing a trend of Deep Learning in Speaker Indentification, when earlier individuals relied on machine learning for voice and these new deep learning approaches perform significantly better than the previous methods.

3. PROPOSED WORK

The proposed approach uses four models for the implementation of Speaker Identification from voice. The identification is done using Deep Learning models which are MLP, CNN, RNN, and LSTM at the end results are compared.

3.1 Dataset Description

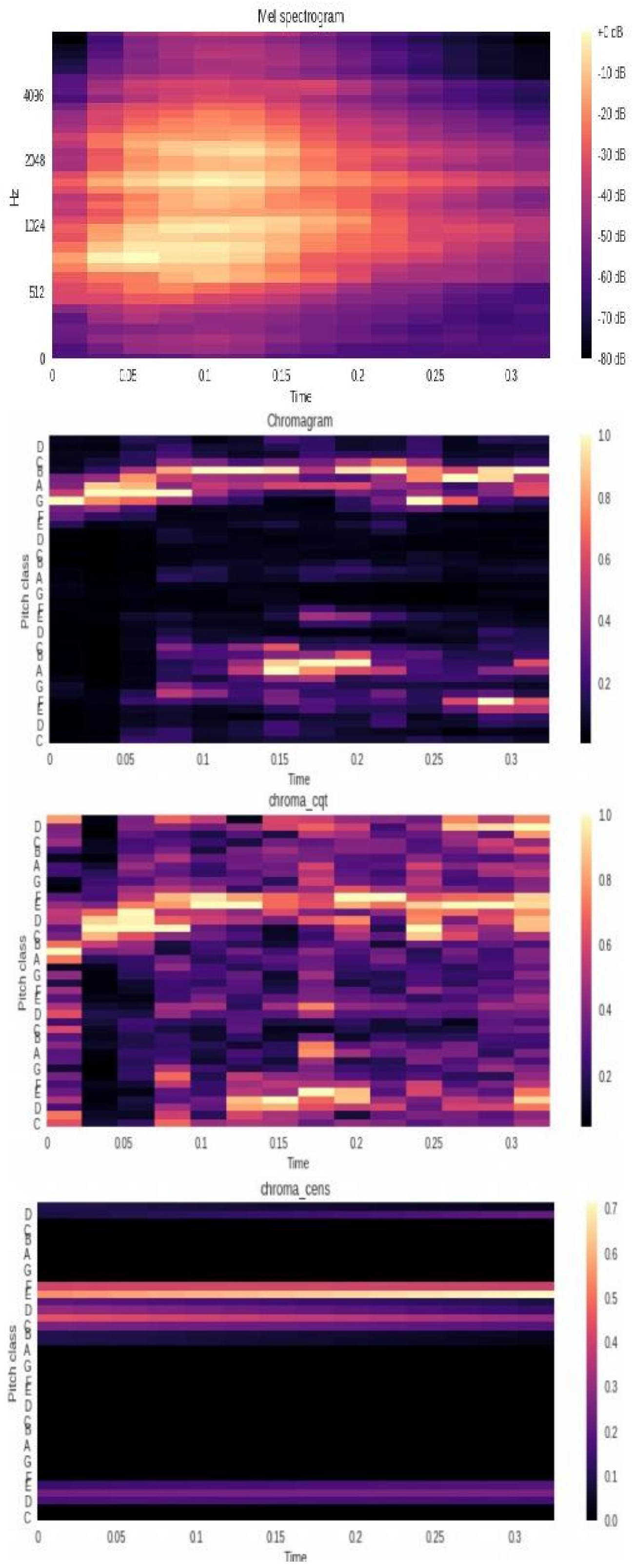
In this project, we aim towards identity of a speaker by machine on the basis of his/her voice where no identity is claimed by user. The dataset consists of various voice recording with each voices record in different pitch, speed and with some noise. With such a divergent dataset, we are able to train our system to good levels and thus obtain good results.
We analyze 1,330 voice recordings from 14 classes and each class contains about 90 to 100 voice and the extension of voice is .wav. Each class label is set with a speaker name.
Feature extraction is done by mfcc (Mel-frequency cepstral coefficients), melspectogram (mel-scaled spectrogram), chroma_stft (Short-Time Fourier Transform), chroma_cqt (Constant-Q transform), and chroma_cens (Chroma Energy Normalized). The neural network is trained by applying these features as input

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 5- Issue 1, Paper 33 January 2022

parameters. From each voice, extracting 200 features by mfcc, melspectogram, chroma_stft, chroma_cqt, and chroma_cens which means 40 from each.
If we plot the the feature of one voice as image:
Figure 3.1 Sample of Features as in the form of image

Volume 5- Issue 1, Paper 33 January 2022



Volume 5- Issue 1, Paper 33 January 2022

3.2 Work Flow Diagram

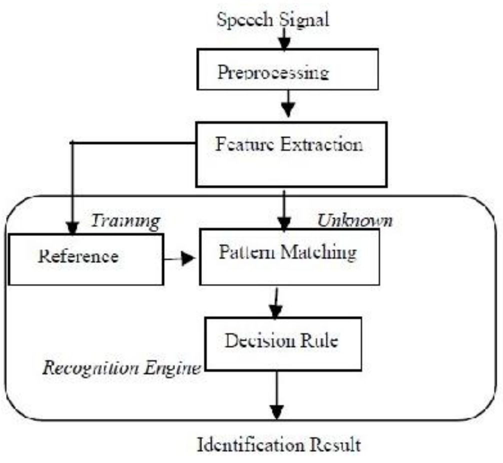


Figure 3.2 Work flow diagram

3.3Implementation using Deep Learning

In this implementation, MLP, CNN, RNN, and LSTM has been used which consist of an input and an output layer, as well as multiple hidden layers.
In MLP the total number of layer is four, one input layer two hidden layers and one output layer. In this, we are using relu activation function and in output, we are using a softmax activation function.
In CNN the total number of layer is four, two layer is for convolution, one for dense layer and one output layer which is fully connected. In this, we are using relu activation function and in output, we are using a softmax activation function.
In RNN the total number of layer is four, Two layer is for RNN, one for dense layer and one output layer which is fully connected. In this, we are using tanh activation function and in output, we are using a softmax activation function.
In LSTM the total number of layer is four, Two layer is for LSTM, one for dense layer and one output layer which is fully connected. In this, we are using relu activation function and in output, we are using a softmax activation function. Every model is originally written withCUDAto run withGPUsupport. More details have been discuss in the next section of this chapter.

3.4.1 Convolutional Layer

A convolution is applied by the convolution layer to the input which is then passed on to the next layer. This is directly in assimilation with the response of a neuron to a visual stimulus. Only for its own receptive field does each convolutional neuron process data. Even though to classify data can we use fully connected feed forward neural networks, it becomes impractical to apply this architecture in case of audio and image. In the event of a shallow architecture, the number of neurons required to process an image would be enormous, as each pixel would be an important variable. Consider the following example: a totally linked layer for a little picture with a size of 100x100 will have 10000 neurons that are important to each neuron in the subsequent second layer. The convolution procedure solves this problem by reducing the number of free parameters while using fewer parameters in the deeper network. If, for example, a tiling area with the same shared weights is 5x5 in size, independent of picture size just 25 learnable parameters would be required. When using back-propagation to train multi-layer neural networks, this is how the problem of disappearing gradients is overcome.

3.4.2 Activation Function

To make the output layer non-linear we use back-propagation. The output of the convolution will be passed via the activation function in the in the case of a Convolutional Neural Network. It's possible that this is the ReLU activation function.
The ReLU function is defined as follows:

$$f(x) = \max \quad (0, x)$$

3.4.3 Pooling Layer

In most cases, a pooling layer is added after a convolution layer. Pooling layers are used to decrease the amount of parameters and computations, which is accomplished by lowering the dimensionality of the data in an ongoing way. The act of overfitting is thus controlled and the time of training is thus reduced.
The most frequent pooling technique is max pooling, which takes the highest maximum value in each window. From beforehand itself do we need to specify the technique of pooling. Average pooling is another example, which established clusters of neurons from the previous layer and then takes their average value.

3.4.4 Dropout Layer

A dropout layer maybe added to the model to alleviate the overfitting issue. Using a dropout layer, it is possible to switch off portion of the neurons at random times in order to reduce a tiny degree of dependence on the training data. The hyperparameters that decide what proportion of neurons we wish to turn off may be adjusted to our preferences. As a consequence, since, as previously mentioned, the same state of activity for each and every neuron will be different always at similar instances of time, the model will never be able to memorise all of the data.

3.4.5 Fully connected Layer

This accomplished by linking every single node for one stratum to just about every neuron in another layer, which is accomplished by the totally connected layer. In principle, it is similar to multilayer perceptron neural network (MLPN).

3.4.6 Optimizers

It is during the training process when we try to minimise the loss function by changing or tweaking the parameters (weights) of the model.
In reaction to the outcome of the loss function the optimiser links the parameters and the loss function together to form the optimal solution. In layman's words, what it does is fiddle with the parameters of the model in order to turn it into the most accurate description or form possible. The loss function's functionality directs the optimiser in the appropriate path in terms of his or her work.

3.4.7 Gradient Descent

Gradient Descent is the most important technique and the foundation of how we train and optimize Intelligent Systems. It is also the most difficult to master. After finding the minima, controlling the variance and updating the model's parameters, Gradient Descent leads us to Convergence, which is the last step.

Volume 5- Issue 1, Paper 33 January 2022

3.4.8 Stochastic Gradient Descent

Instead of calculating the gradients for all of your training examples on every pass of gradient descent, it's sometimes more efficient to only use a subset of the training examples each time. Stochastic gradient descent is an implementation that either uses batches of examples at a time or random examples on each pass.

3.4.9 Momentum

A very popular technique that is used along with SGD is called Momentum. Instead of using only the gradient of the current step to guide the search, momentum also accumulates the gradient of the past steps to determine the direction to go.

3.4.10 Adam

Adaptive moment estimation is the full form of Adam which is another way of estimating the current gradients with help of past gradients. Due to the addition of fractions of previous gradients to the current one, Adam utilizes the concept of momentum. It is an optimiser which has a reputation that is pretty widespread and hence is used to train neural nets.
Heuristics of both Momentum and RMSProp is combined by algorithms of Adam or Adaptive Moment Optimization.

4. IMPLEMENTATION AND RESULT

4.1 Implementation Details

4.1.1System Requirement

System used for the experimental purpose has following configuration:

- Processor: Intel(R) Core(TM) i3-6006U CPU @ 2.00 GHz
- RAM: 8.00 GB
- GPU: 2.00 GB
- System Type: 64-bit
- Operating System: Windows 10

4.1.2Software Requirement

- IDE : Anaconda (Jupyter, Spyder), Google Colaboratory
- Programming Language: Python
- Packages :Keras, Matplotlib, Pandas, NumPy, Scikit-learn, LibROSA

4.1.3 Experiment

The experiment was carried out on a computer running the Windows 10 operating system and Google Colaboratory. Python is used to write the source code. For the project, around 1,330 voice recordings from 14 classrooms were analysed, with each class including approximately 90 to 100 voices.
mfcc (Mel-frequency cepstral coefficients), melspectogram (mel-scaled spectrogram), chroma stft (Short-Time Fourier Transform), chroma cqt (Constant-Q transform), and chroma cens (Constant-Q transform) are used to extract features (Chroma Energy Normalized). These properties are used as input parameters to train the neural network. mfcc, melspectogram, chroma stft, chroma cqt, and chroma cens retrieve 200 features from each voice, resulting in 40 from each.

We create a.csv file and save these features after extracting features from the voice. The data is divided into two categories: test and training. Train data contains 70 to 75 data points per class, while Test data contains 20 to 25 data points per class.
Deep Learning models CNN, RNN, and LSTM are used in the implementation, and the accuracy gained in both models is compared. On the CPU, deep learning takes around 12 hours to complete the training process with 10 epochs and 125 steps each epoch, whereas on the GPU, it takes about 1.5 hours with 25 epochs and 125 steps per epoch.

4.2 Results

4.2.1 Multi-layer Perceptron

MLP is a deep learning strategy that is used in the suggested methodology (Multilayer Perceptron). The input layer, which has 256 neurons, is where data is transferred. The overall number of hidden layers is four, with one being the input layer, two being hidden layers and one being the output. In this case, we are using the relu activation function, and in the output, we are using the softmax activation function (see below).

Adam optimizer with learning rate 0.001, and dropout 0.15
•**Model Accuracy**
Accuracy: Test = 98.35%, Train = 86.67%,
•**Model Loss**
Loss: Train = 0.8480, Test = 0.0321

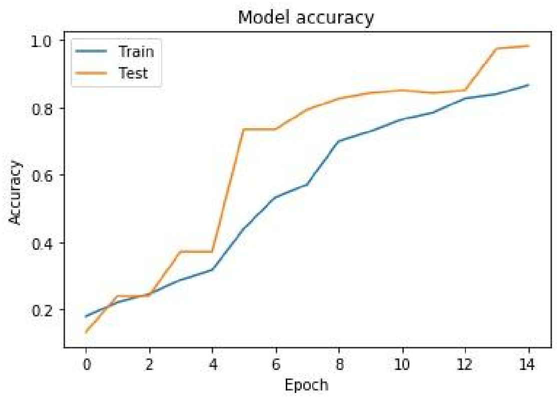


Figure 4.1 Model Accuracy of MLP

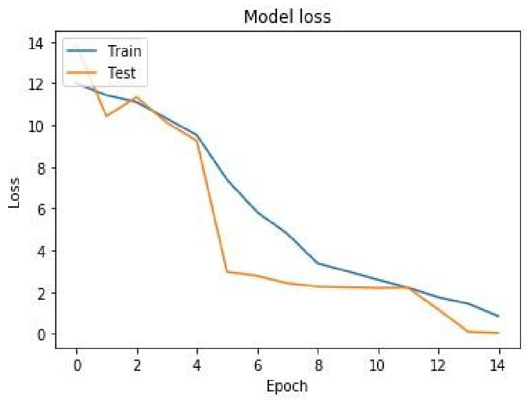


Figure 4.2 Model Loss of MLP

Yashu Kedi, Nagadevi S.

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 5- Issue 1, Paper 33 January 2022

4.2.2 Convolution Neural Network

The proposed methodology uses a deep learning approach which is CNN (Convolution Neural Network). Input data is passed to input deep learning module (CNN). Convolution operation in proposed work uses 2 layers. The total number of kernels are 64 with the size of 5*5 is used for the first convolution layer and last layer having numbers of the kernel are 128 of the same size which is 5*5. The last layer is fully connected followed by softmax layer in CNNs model.

Adam optimizer with learning rate 0.001, and dropout 0.15
•**Model Accuracy**
Accuracy: Test = 99.17%, Train = 99.38%
•**Model Loss**
Loss: Train = 0.0261, Test = 0.0248

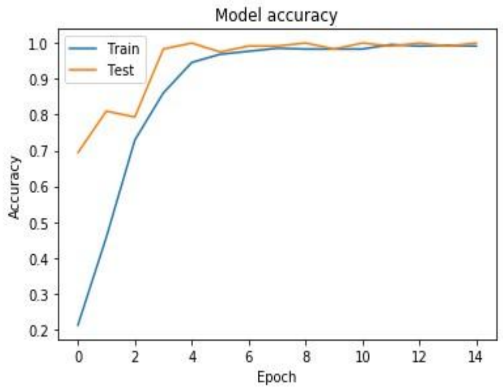


Figure 4.3 Model Accuracy of CNNs

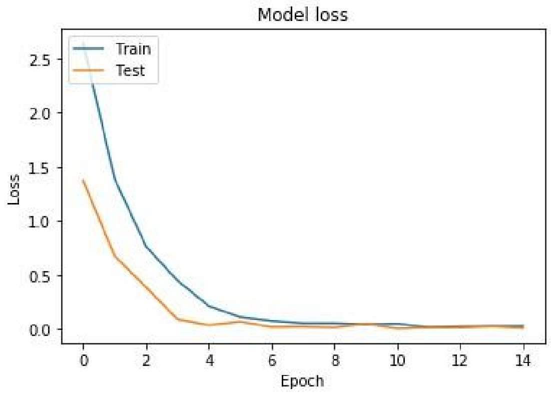


Figure 4.4 Model Loss of CNNs

4.2.3 RNN

The proposed methodology uses a deep learning approach which is RNN. Input data is from RNN layer with data size (40,5) The total number of layer is four, two for rnn, one for dense layer and last layer is for output layer. In this, we are using relu activation function and in output, we are using a softmax activation function.

Adam optimizer with learning rate 0.001, and dropout 0.15
•**Model Accuracy**
Accuracy: Test = 98.35%, Train = 96.04%
•**Model Loss**

Yashu Kedi, Nagadevi S.

Loss: Train = 0.1229, Test = 0.0358

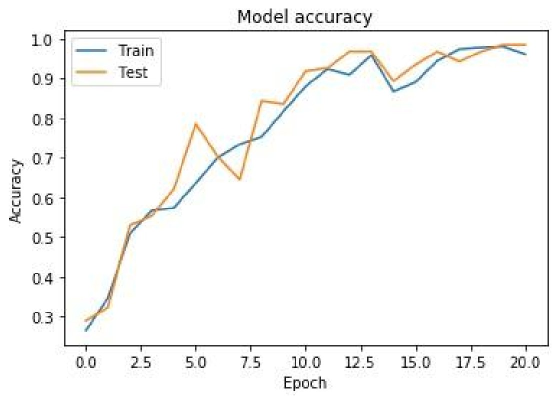


Figure 4.5 Model Accuracy of RNN

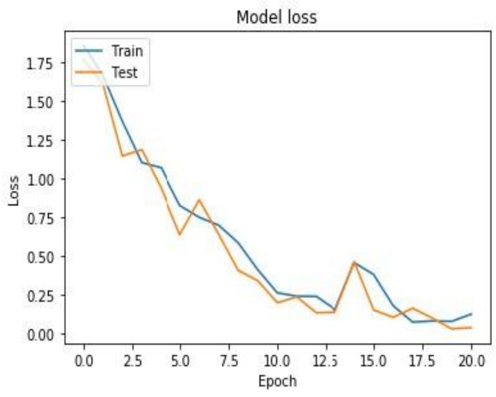


Figure 4.6 Model Loss of RNN

4.2.4 LSTM

The proposed methodology uses a deep learning approach which is LSTM. Input data is passed from LSTM layer with data size (40,5) The total number of layer is four, two for lstm, one for dense layer and last layer is for output layer. In this, we are using tanh activation function and in output, we are using a softmax activation function. **Adam optimizer with learning rate 0.001, and dropout 0.15**

•**Model Accuracy**

Accuracy: Test = 99.67%, Train = 99.58%

•**Model Loss**

Loss: Train = 0.0312, Test = 0.0091

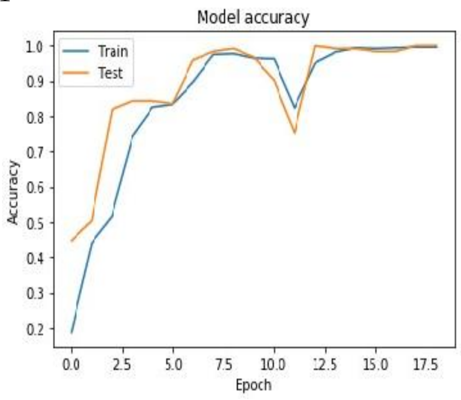


Figure 4.7 Model Accuracy of LSTM

Yashu Kedi, Nagadevi S.

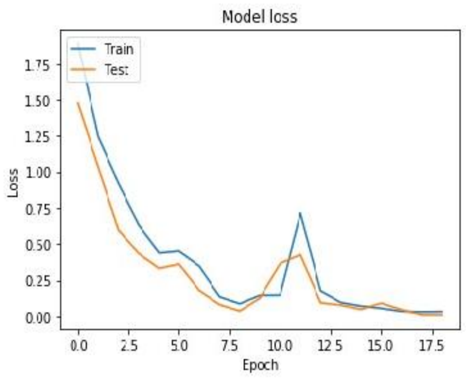


Figure 4.8 Model Loss of LSTM

4.2.5 GRU

The proposed methodology uses a deep learning approach which is GRU. Input data is passed from GRU layer with data size (40,5) The total number of layer is four, two for gru, one for dense layer and last layer is for output layer. In this, we are using relu activation function and in output, we are using a softmax activation function. **Adam optimizer with learning rate 0.001, and dropout 0.15**

•**Model Accuracy**

Accuracy: Test = 97.52%, Train = 99.58%

•Model Loss

Loss: Train = 0.0105, Test = 0.1984

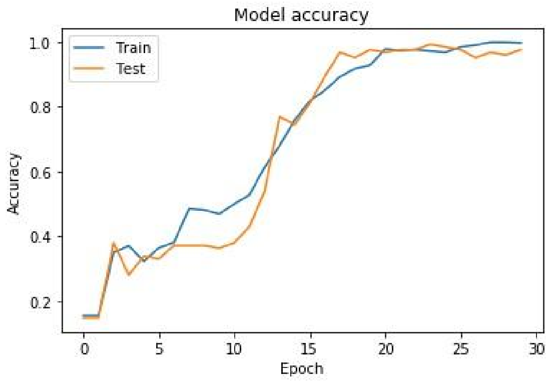


Figure 4.9 Model Accuracy of GRU

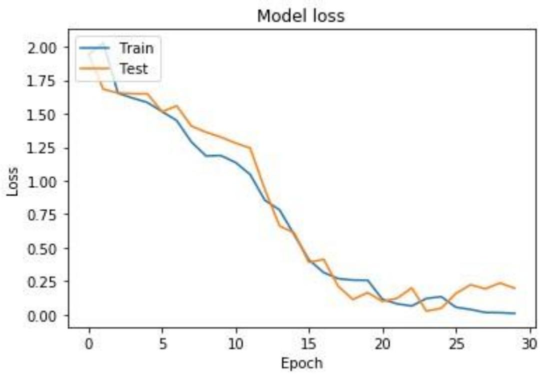


Figure 4.10 Model Loss of GRU

Yashu Kedi, Nagadevi S.

4.3 Graph of Accuracy and Loss of different models

4.3.1 Combined accuracy value graph of different Deep Learning techniques

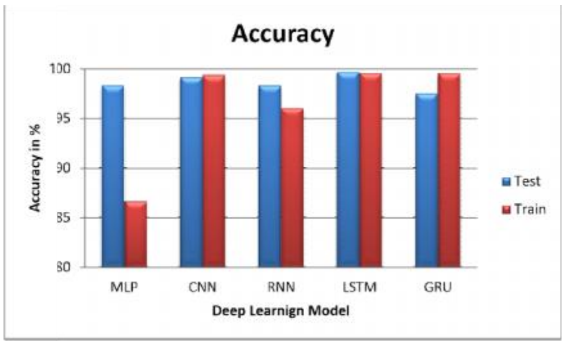


Figure 4.11 Accuracy graphs of different Deep Learning techniques

4.3.2 Combined loss graph of different Deep Learning techniques

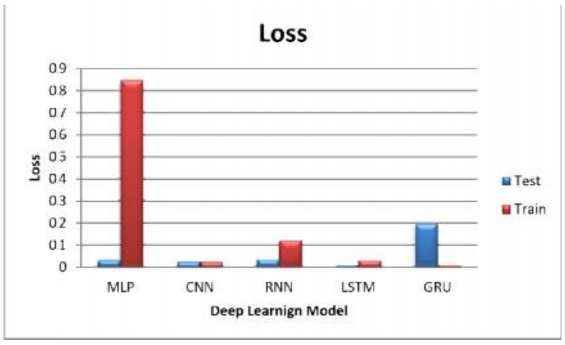


Figure 4.12 Loss graphs of different Deep Learning techniques

4.4 Summary

This chapter provided the implementation details of the proposed solution. In which different models are used for implementation and LSTM model gives better results among all of them.

5. CONCLUSION AND FUTURE WORK

In this project, we have presented an implementation of a speaker identification system on different Deep Learning models. This system used MLP, CNN, RNN, LSTM, and GRU in order to achieve recognition. The neural network was trained by using backpropagation as the learning algorithm.
This system was able to identify 14 different speakers in a satisfactory way. These speakers were the users from whom we took the samples to train the system. The speaker identification system was tested using different samples from those used to train it. The software works fine for identifying a speaker from a number of different speakers.
The achieved test accuracy from MLP, CNN, RNN, LSTM, and GRU was 98.35%, 99.17%, 98.35%, 99.67%, and 97.52% respectively.
The future work is like tagging the speaker from mix voice.

Volume 5- Issue 1, Paper 33 January 2022

REFERENCES

1) Yanick Lukic, Carlo Vogt, Oliver Durr, and ThiloStadelmann, "Speaker Identification and Clustering using Convolutional Neural Networks", 2016 IEEE International Workshop on Machine Learning for Signal Processing, SEPT. 13–16, 2016, Salerno,
2) Italy
3) F. Trujillo-Romero, S. O. Caballero-Morales, "Speaker identification using Neural Networks on an FPGA", 2012 Ninth Electronics, Robotics and Automotive Mechanics Conference
4) MircoRavanelli and YoshuaBengio, "Speaker recognition from raw waveform with SincNET," arXiv:1808.00158v2, 2018.
5) M. Schmidt and H. Gish, "Speaker identification via support vector classifiers", 1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings.
6) Xuewei Zhang, Xiao Liu, Ying Cao, and Ajay Kannan, Zhenyao Zhu, "Deep Speaker: an End-to-End Neural Speaker Embedding System", arXiv:1705.02304v1 [cs.CL] 5 May 2017
7) Shahla Parveen1,2, Abdul Qadeer2, and PhilGreen1, "Speaker Recognition With Recurrent Neural Networks", 6th International Conference on Spoken Language Processing (ICSLP 2000) Beijing, China October 16-20, 2000
8) R.V Pawar, P.P.Kajave, and S.N.Mali, "Speaker Identification using Neural Networks", World Academy of Science, Engineering and Technology, 12 2005
9) R M Makwana, Deep Face Recognition Using Deep Convolutional Neural Network,
10) AIeHive.com, http://www.ais.uni-bonn.de/deep learning/images/Convolutional NN.jpg