International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Efficient Recognition and Classification of Stuttered Speech Signal using Deep Learning Technique

Mrs. Mohandoss Rajalakshmi[1*], Mr. Ramasubbu Rengaraj[2], Mr. Giri Rajan Babu Venkatakrishnan[3], Mrs. Jeya R.[4]

Submitted: 26/12/2023 **Revised**: 06/02/2024 **Accepted**: 14/02/2024

Abstract: Speech recognition systems in modern-day devices are a popular feature that has facilitated human-machine interaction. Users need not learn complex programming languages to communicate with their devices and can give commands using their voice to perform multiple tasks. However, its usage is limited if it encounters stutter in the voice input of a person with this disfluency. This work is based on building a system that not only classifies the speech as stuttered or normal but also rectifies the discourse by removing stuttered portions from the signal. It first takes the speaker's audio as input and performs segmentation on the speech signal to divide it into segments of 300ms. MFCC feature extraction from these segments is done. These features are fed into the model for classification of the audio segment, which is then corrected to give stutter-free audio, along with its text conversion.

Keywords: *Automated Speech Recognition System (ASSR), Deep Neural Network (DNN), MFCC Feature Extraction, Stuttered Speech Recognition, Speech-to-Text API.*

1. Introduction

1.1. Background and Objective

Stuttering — also known as infancy stammering or fluency deficiency — is a condition of speech that causes frequent and inherent obstructions in normal fluency and speech flow. Those who stutter know what they want to say but face trouble in speaking it out. They may repeat or prolong a word, a syllable, a consonant, or a vowel sound. Or perhaps they pause during speech because they may have reached a troubling tone word. Young children stutter when learning to communicate in their initial stages, which makes it a common part of their speech development. Young kids may stutter when their speech and language skills are not sufficiently established to keep up with what they wish to say. Most children outgrow the stuttering phase. However, stuttering may persist as a chronic problem until a person reaches adulthood, due to which people may suffer from lower self-esteem and lose confidence while trying to interact.

More than 1% of the world (about 70 million people) suffers from the disability of stuttering.

The key thing to understand about speech is that human-generated sounds are mediated by the vocal tract form like tongue, teeth, and so on.

This shape defines the sound that comes out. For few, this may cause consonants to become unclear, and vowels to blend together, causing them to stutter. In the traditional stuttering identification procedure, the speech input is taken and analyzed for signs such as repeating a syllable, a term, prolonged pauses, fragmented expressions, incomplete sentences, and interjections. (Stuttering involves disfluencies like dysarthria, apraxia, stuttering, cluttering, lisping, whispering, mumbling, and many more.) The frequencies of each of these disfluencies are then counted. The main drawbacks of making such an assessment are that it may be time-consuming, subjective, or inconsistent. In addition, different judgments on the same material may lead to conflicts in agreement.

This system aims to propose a solution for the people with stuttering disability to combat it by taking in their speech input, processing it to produce a clear audio output devoid of pauses and convert the audio into text.

2. Literature Survey

A thorough study of the existing classification techniques used in previous recognition systems was conducted. It was found that most of the existing work based on this subject is just the classification of the speech audio into 'stuttered' or 'normal' speech. The points to be noted from the competent approaches among all these works is first, the pre-emphasis of audio and then extraction of audio

[1*]*Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, SRM Nagar, Kattankalathur, Chennai, Tamil Nadu, India.*
[2]*Associate Professor, Department of Electrical and Electronics Engineering, Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam, Chennai, Tamil Nadu, India.*
[3]*Associate Professor, Department of Electrical and Electronics Engineering, Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam, Tamil Nadu, India.*
[4]*Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, SRM Nagar, Kattankalathur, Chennai, Tamil Nadu, India.*
** Corresponding Author Email: rajalakm2@srmist.edu.in*

features (MFCC feature extraction was found to be the most efficient), followed by training the classifier models (based on ANNs, HMMs, SVMs, etc.) and MATLAB being used as the language of programming.

2.1. Background Study

A research conducted by AndrzejCzyzewski, AndrzejKaczmarek&BozenaKostek [1] reviews automatic recognition of stuttered audio in normal, and altered feedback speech with frequency. It exhibits different procedures for analyzing stuttered audio and describes ways to define those arguments that represent a stuttering event. It was inferred from the paper that the process of calculating stammering events can be carried out more equitably through the automated observation of stop-gaps, syllable recurrence, and prolongation of vowels. The alternative methodology may be focused on the theoretical measurement of speech rhythmicity, which could focus on a subjective assessment of the process. Meanwhile, automatic detection of intervocalic interruptions, stop-gaps, voice start time, and vowel duration may rely on the speaker, and the precedent derived from a single speaker may be inaccurate if universal. All of these restrictions make ANN Classifier a bit redundant as it has many dynamic variables and this data mismatch dramatically reduces the accuracy of its ANN model. This configuration resulted in the accuracy of their model being 71 percent.

Another study by Anusuyaa and Katti, [2] presents Automatic Speech Recognition (ASR) and addresses the advantages of utilizing an acoustic-phonetic procedure that speculates the nearness of phonetic units (phonemes) in audio input and why these phonetic units are usually represented by a couple of acoustic properties that can be shown in the location signal as time passes by. While the acoustic segments of phonetic units contrast at a further extent, with the two speakers and nearby disturbance (implied to as the impact of coarticulation), the acoustic-phonetic strategy expects that the rules for variety are clear and can be utilized. The subsequent stage is the division and marking process, where the voice signal is divided into stable acoustic areas, joined by at least one phonetic name added to each sectioned locale bringing about a phoneme called Speech Lattice Characterisation. The last advance in this methodology means to decide a legitimate word (or word string) from the phonetic imprint arrangement delivered by the stretching to name. In this procedure, phonetic limitations on the errand (i.e., jargon, linguistic structure, and other semantic principles) are conjured to get to the dictionary for word deciphering dependent on the phoneme cross-section. The issues with ASR are that it is exceptionally inefficient, not even 30 percent of the words could be distinguished accurately.

Manu Chopra with his team presented a research where they used 3 different implementations (namely Neural Networks, Naive Bayes, and SVM) over their MATLAB models. Neural Networks coupled with standard mean feature extractor helped them achieve overall accuracies of 85.4% for men and 78% for women. A baseline classifier was first made based on previously used neural networks. The idea was to keep the classifier as simple as possible by making a two-layered neural network and to analyze the effects of different audio features on the success of the classifier (as well as to verify the effectiveness of MFCC features on stuttering/non-stuttering classification). The goal of the baseline classifier was to test the simplest possible solution that could create a two-layer neural network in TensorFlow that captured only MFCC features of the audio files passed. The final epoch value was set to 5000 and the learning rate to 0.01. There were about 53 audio files (half stuttered and half non- stuttered) in their database. This configuration yielded the best accuracy of 66.0%. This paper shows the effectiveness of neural networks on serving as classifiers for stuttered and non-stuttered speech.

A paper published by Afroz and Koolagudi, [3] recognizes pauses in the input speech by examining the duration, frequency, position, and distribution of pauses in the sound. In this work, qualities like audio length, its recurrence, distribution, and particular places of pauses in the audio are estimated and measured. The stammered audio input corpus by UCLASS is taken and analyzed for this examination. The following steps were used for analysis:

(1) Perceptual Analysis:

This analysis was used to understand how pauses affected the normal flow of speech. For this, the pauses' duration, their frequencies, and respective locations were studied. This method helped the authors understand whether the person was stammering and also gave information about the severity level of the disorder.

(2) Acoustic Analysis:

This was used to understand the features of pauses in stuttering. Acoustic analysis required some voicing measures such as fundamental frequency, model and capacity, decibel level (the sound pressure level is a physical measurement which is approximately associated with loudness perception), amplitude, et cetera. These were used for further examination of the speech and to distinguish between normal and stuttered voices.

A method known as Automatic Visually Blind Segmentation (it is a novel algorithm which is not used much, hence this procedure does not require any prior knowledge regarding the speech to be segmented) was implemented for the division of the speech signal into

spoken and halted areas, utilizing an effective cap that depended on energy and ZCR (Zero Crossing Rate). Four formant frequencies were used to distinguish delays found inside the voiced areas. It was seen that the length of an ordinary delay extended between 150 ms and 250 ms.The short pauses had a time duration of 50 ms-150 ms. While in case of stammering the brief delays ran from 10 ms to 50 ms, the long postponements stretched out from 250 ms to 1-2 seconds. Qualities like term energy, zero-crossing rate, pitch, and formant frequencies have been used for the unmistakable verification of postponements and stammers in the sound sign. Since blind segmentation has been used, there was no convincing motivation to upgrade the model for every language/speech style or accent as this procedure does not require any prior information for the division. The language algorithm did not need to be redesigned or modified. After recognizing delays, all voiced and paused areas were picked up. By fixing a limiting value of more than 150 ms as long and under 150 ms as short, every delayed area was characterized into long and short classifications. Regardless, this work revolved basically around perceived delays in speech. The edge taken can fluctuate according to the originator and cause various faults. According to this investigation, the classification of pauses was done physically in the previous works. On a typical for 108 records, the manual division realized 14 to 21% more divisions than customized blind segmentation, the cause for this distinction was that in the manual strategy what was found in a waveform as quietness or gap sought out to count (the smaller fragment of voice) as one portion. This incited irregular division. While in the offered procedure for this work, about 98% of the voice segments could be obtained precisely, and this perceived exactness was affirmed by the creators by performing the manual and perceptual examination. This arrangement has a wide demand in the clinical field especially for discovering speech issues.

In research presented by A Suryaa and Varghese in 2017 [4], SVM (support vector machines) was used for model training with MFCC feature extraction. Three methods for recognition of stuttered speech were proposed:

(1) Supervised model for stuttered speech recognition

Using MFCC for feature extraction from the audio signals and SVM for classification had two stages: training and testing. SVM analyses the data for classification and regression analysis with the help of associated learning algorithms. An accuracy of 76% was acquired in the identification and classification of the words. The precision of this procedure could be boosted by using more training data.

(2) Stuttered speech recognition by stuttering pruning

The speech was converted into amplitude/time audio signals and then maximum amplitude was given to the neural network to compute the threshold value based on which the classification was done. Speech correction methods implemented using neural networks acquired a low accuracy of about 62% which could be further enhanced by incorporating more training data as well as by integrating some other attributes like frequency, energy and others, of the audio input for training.

(3) Automated text-to-speech based stuttered speech recognition

ANN (Artificial Neural Network) based model was trained with intelligent guesses to predict the vowel and consonant terms in the speech. It analyzed the inputted speech with its training experience and produced equivalent texts. The outputted text was then inputted to a dictionary. The word with the highest matching was picked. In this way, all stuttered parts were eliminated. This method achieved an accuracy of 80%.

In another work by Lim Sin Chee and his team [5], the performance of MFCC features was analyzed for the identification of lengthenings and recurrence in speech signals containing stutters. The classifiers used were k-NN(kth-nearest neighbor) and LDA(Linear Discriminant Analysis), k-NN being a lazy learner showed lower performance levels and gave an inaccurate output, whereas LDA which is popularly used for pattern recognition gave a comparatively higher accuracy for the detection of prolongations and repetitions in audio samples containing stuttered speech.

Summary

After carefully reviewing the existing literature, the authors planned to use DNN classifier, coupled with MFCC feature extraction technique for the classification of segments as “stutter” or “normal” in a given speech audio. A DNN or Deep Neural Network is a multilayered Artificial Neural Network (ANN) between the input and output layers; every mathematical manipulation as such is called a layer, and since there are several layers in a complex DNN, hence the term "deep" networks. The authors decided to use Python for this work as it is fairly more compact and readable than MATLAB and offers more choices in graphics packages and toolsets along with better in-built packages/ libraries.

3. Proposed Methodology

In this system, MFCC (also known as Mel Frequency Cepstral Coefficient) has been used for extracting features from the input sound. A Deep Neural Network-based classifier is built for classification of speech. The workflow of the complete process is as follows:

3.1.Input Speech

The audio files from the UCLASS dataset were used. UCLASS stands for University College London Archive of Stuttered Speech. This specific dataset was chosen because it contains recordings of monologues, conversations, and readings of different speakers, between the ages of 7 years to 20 years. The audio files were in .wav format. This is the preferred format as these files are uncompressed and render lossless audio vis-a-vis other formats. Figure 1 shows that the System Architecture Block Diagram.

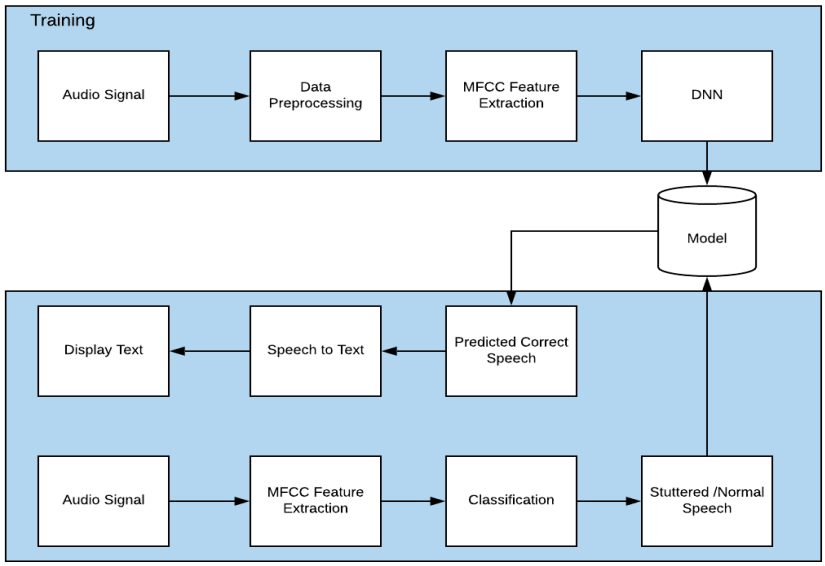

Fig 1: System Architecture Block Diagram

3.2.Data Preprocessing

Dataset preprocessing, extraction of features and feature engineering are steps we take to extract information from the underlying data, information that should be useful for predicting the category of a sample or the value of some target variable in a machine learning context. In audio analysis, this process is largely based on finding components of an audio signal which can help us to distinguish it from other signals. The database had audio files in 2 releases, release 1 and release 2. Release 1 had 16 files that had time-aligned transcriptions and release 2 had 4 files. The model in this system was trained using the files present in release 1. The audio files were in .wav format and the corresponding time- aligned transcriptions are in CHILDES CHAT format. The orthographic transcriptions were in plain text. Since the orthographic transcriptions did not have the text for corrected speech but had the transcription for the stuttered speech, they had to be transcribed manually. Time-Aligned transcriptions were used to split every data file into ‘stuttered’ segments and ‘normal’ segments for the purpose of model training. The transcriptions had the start time and end time (in milliseconds) for each segment. So 12,633 segments were registered after splitting the audio files.

Table 1: Audio Data Statistics

	ALL	STUTTER	NORMAL
COUNT	12633	2643	9990
MAX (ms)	17044	17044	14499
MIN (ms)	0	1	0

MEAN (ms)	315.0925	762.5323	196.7158
MEDIAN (ms)	192	486	168
MODE (ms)	109	201	93

From Table 1 it can be observed that the average duration of all the segments was approximately 315ms. The stuttered segments were about 2.5 times longer than the average segment and the longest stutter found from the dataset was around 17 seconds. Training the model on such skewed data would not be helpful because reproducing a stuttered segment that was as long as 17 seconds is highly unlikely. So, instead of using the segments of variable duration, the segments of the file were further segmented down to less than or equal to 300 ms which is close to the average length of the segments as inferred from the table. This segmentation created 17,545 segments that were used for training the model.

3.3. Feature Extraction Using MFCC

The purpose of MFCC feature extraction is to identify the components of an audio signal that are useful for its identification and discard the rest. It is popularly known that a sound wave is constantly changing with time. On a smaller scale, audio signals do not change very rapidly, so the audio signals were divided into small frames of less than or equal to 300 ms duration, which is close to the average length of the segments, as mentioned above, to closely examine them and extract accurate features. This process led to the dataset finally consisting of 17,545 segments, the features of each of which were extracted using MFCC.

The load() method in the Librosa library in Python was used which loads an audio file as a floating-point time series. These values are nothing but the amplitudes of the waveform of the audio samples. Through this method, audio is automatically resampled to either the rate specified, or to the default sampling rate which is 22050 Hz. The sampling rate (sr) of the audio is defined as the number of samples per second of the audio. Now that the amplitude values were available, Mel spectrogram for each audio segment could be plotted. The floating-point time-series was inputted into the melspectrogram() method of Librosa, along with its sampling rate. This method first computes its magnitude spectrogram and then maps it onto the mel scale by mel_f.dot(S**power). A default power=2 operates on a power spectrum. Since most sounds humans hear are concentrated in very small frequency and amplitude ranges, the amplitude spectrogram does not provide much scope for feature extraction. Therefore, the amplitude_to_db() method was used to convert the amplitude spectrogram obtained to a dB-scaled (Decibels) spectrogram. A spectrogram is like a heat map where the varying colors show the transitions in the intensities of sound.

The next thing to do was to extract the MFCCs (mel-frequency cepstral coefficients) which are a classic input feature for recognition related work like voice activity detection, phoneme recognition, et cetera in an audio sample. In practice, the first 13 MFCC coefficients are used to describe the instantaneous, spectral envelope shape of the speech signal. The acoustical speech signal depicts a sequence of transitions between phonemes. It is the information about such transitions (obtained from the spectrogram) that is particularly required for identifying the characteristic properties of different types of speech segments. In order to achieve this, the first difference of signal features is determined, known as the *delta* of a feature. Specifically, for a feature f_k at time-instant k, the corresponding delta is defined as Eq.(1)

$$\Delta_k = f_k - f_{k-1} \quad (1)$$

The second difference, known as the delta-delta, is correspondingly Eq.(2)

$$\Delta\Delta_k = \Delta_k - \Delta_{k-1} \quad (2)$$

Hence, the 13 features calculated in the recognition engine in this work were then typically appended by their Δ and ΔΔ -features to triple the number of features (meaning a total of 39 MFCC features for every segment) with a very small computational overhead. This method of computing deltas is successful not only because they are simple to calculate, but because they also provide a clear benefit over the instantaneous features of the spectrum. [6] The MFCC vs Time, Δ MFCC vs Time, and the Δ Δ MFCC vs Time spectrograms can be plotted for every segment.

Finally, the root-mean-square (RMS) value was computed for each frame from the audio samples time-series. Hence, a total of 40 features were acquired pertaining to each segment. Features thus extracted for all the segments were then appended into a vertical stack which was fed into the classifier model for training and further classification of new audio segments as stuttered or normal. [7-12]

3.4. Neural Networks

In Machine Learning, Deep Neural networks (DNN) are Artificial Neural Networks(ANN) with multiple layers between input and output. One common feature of the Neural Network is to deal with unordered and unstructured data. DNNs are formulated of multiple layers. The computations in each layer are hidden, hence the layers are termed as “hidden layers". DNN is a supervised learning algorithm where the machine recurrently performs the same task on every element of the sequence where the output of each task is dependent on every previous calculation. The layers’ equation for our model can be represented as : Eq.(3)

$$y' = \text{softmax}((w3.\text{ReLU}(w2.(\text{ReLU}(w1.x + b1) + b2) + b3)) \quad (3)$$

To begin, the input layer/primary layer of the network receives the features of the data (audio segment) as input. The ultimate layer/output layer receives the classified output from the internal/hidden layers. The processing layers are the internal/hidden layers of the network. These layers perform specific mathematical tasks wherein the data received as input is processed according to the above equation. Here, w1-w3 are the weights and b1-b3 are the biases. The weight’s parameter decides the amount of influence of the input on the output and the bias is helpful to the network for the case when the input is zero, as, with the bias parameter, the neurons will still fire. They together compute the output data which is then passed on to the next layer as input. This method learns in a sequence called feature hierarchy, where the features at the top of the hierarchy are determined using features at the bottom of the hierarchy. Therefore, after performing the linear tasks on each of the data in every node of the layer, the activation function was applied to the calculated output before sending it to the next layer.

The activation function used here is the Rectified Linear Unit(ReLU). The main purpose to apply an activation function is that it provides a non-linear element to linear neural network equations, so for the hidden layers, ReLU was implemented on the data. While for the output/ultimate layer Softmax activation function was used. The Softmax activation function is conventionally used to predict the probabilistic scores during the classification of the output. One of the major features of Softmax is that the sum of values of the ultimate layer after the probabilistic distribution is always equal to 1.There are 3 hidden layers in the proposed neural network and 10 neurons in each layer.

The learning rate is a hyperparameter of the machine learning model. The use of this parameter determines how much to change the model each time the weights are updated. A very small value of the learning rate parameter leads to a large training period, and a large value leads to an unstable training process hence the value was set to 0.001. Along with learning rates, one requires more training epochs which in this scenario were 1200. Epoch is aparameter that determines the number of times the algorithm will work on the entire training dataset file. A cost function was used to calculate and minimize the inaccuracy in the machine learning model by predicting the relations between x(input data) and y’(output data). It is generally represented by the difference of predicted value and actual value. In simple terms, the Cost function penalizes the neural network when an error occurs while predicting values. The main objective of the cost function is to boost the accuracy of prediction and curb the error.

3.5.Audio Correction

The model was trained on audio segments of duration <= 300ms, thus the audio to be corrected also had to be segmented with a duration of 300 ms.This made it difficult to detect stutter boundaries. Preparation of the input audio was thus done by overlapping all the segments by 200 ms, instead of naively segmenting the audio in a contiguous manner. This type of overlapping helped differentiate between stuttered and non-stuttered parts with a granularity of 100 ms. After segmentation, these segments were sent to the classifier for classification.

The classifier marked ‘1’ for segments having stutters and ‘0’ for the “normal” ones. This allowed for the removal of the segments which were labelled as ‘1’ (that is, stutter) and combine the remaining NORMAL segments. One way of assembling the segments was to append contiguous chunks together but this would result in sharp interjections at the point of concatenation, which in turn would yield an overly artificial sounding voice. Hence, rather than appending the adjacent chunks, cubic interpolation was used to find out the missing audio samples that were lost during the removal of STUTTER segments. To fix this issue and generate an audio sample that is close to the original sample, the audio samples were fed into the cubic interpolation algorithm that reproduced more points between the end of the previous chunk and the beginning of the current chunk. Now after fetching all the possible points between 2 normal points on the audio sample, the cubic interpolation algorithm bent the interpolated points around the original audio sample to get a more polished sound. This method is more preferable than using linear interpolation because linear interpolation makes straight lines when the interpolated points are merged onto the audio sample and since an audio sample follows a wave-like structure, using a cubic interpolation algorithm maintains the quality of the audio input and allows to achieve a smoothened corrected speech.

This corrected speech was then written out and saved in .wav format, which allows maintaining the quality of the corrected speech. This corrected audio file is sent to Google's Speech-to-Text API. The API analyses the audio file, finds features, and transcribes the audio file and finally returns the transcribed text. Thus, a textual equivalent of the audio could be displayed on the UI.

4. Results

From the dataset that has been used to train the model, the time-aligned transcriptions were used to divide the audio file into segments and manually distinguish and label them as 'STUTTER' or 'NORMAL'. The total number of segments were then computed and distributed, as tabulated in Table 1. After calculating the mean of all the segments it was observed that the 'stuttered' segments were generally 2.5 times longer than the average segment. To overcome this disparity in segment length, the authors decided to keep a fixed segment length of 300 ms which was close to the average segment length.-

Next, the features of all the segments were extracted using MFCC feature extraction. Thirteen MFCC features were extracted from the spectrogram obtained for a segment. Figure 2 depicts a dB-scaled Mel Power Spectrogram for one of the audio segments. Similarly, spectrograms for any audio sample can be obtained. The next step was to compute the Δ and ΔΔ of each of the MFCC features. Figure 3 shows the MFCC vs Time, ΔMFCC vs Time, and ΔΔMFCC vs Time spectrograms for one of the segments. These 39 features along with the RMS values for all segments were stored in a vertical stack.

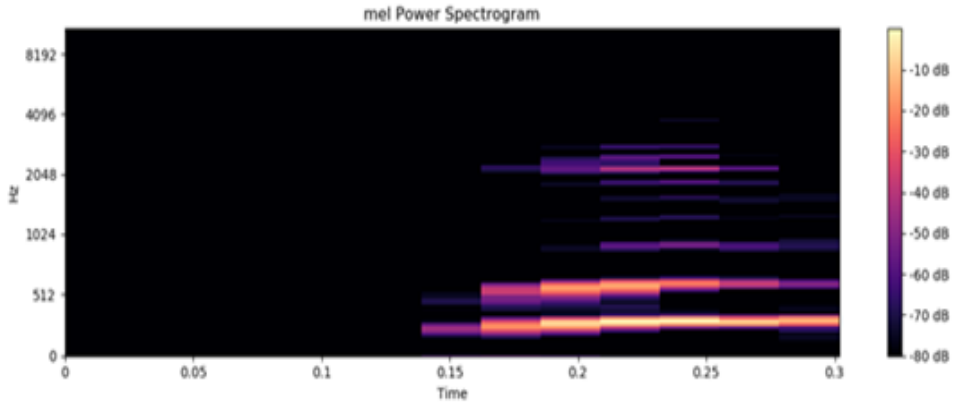

Fig 2: Mel Power Spectrogram of the input audio segment

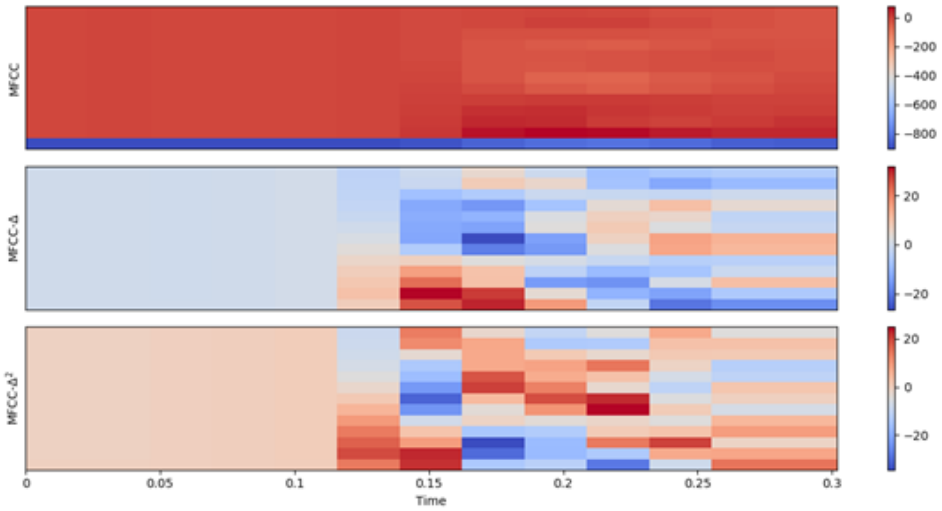

Fig 3: MFCC vs Time graphs for the input audio segment

The vertical stack containing the feature vectors of all the segments was then sent to the DNN (Deep Neural Network) classifier for model training. This is a supervised learning method of training wherein the model is fed data that is well labelled, telling it which segment is "stuttered" and which is "normal" in a particular audio file. It extracts features accordingly to train itself for further predicting the classification of other new unlabeled data that is added by the user. This concluded the training of the model which resulted in an accuracy of 85.9% as shown in Figure 4.

Fig 4: Model Accuracy

Now, when the user records new audio through the UI, the above steps are repeated, that is, the audio file first undergoes preprocessing. Next, features are extracted from the audio segments so obtained, which are then fed into the DNN classifier. The classifier, with the help of the already trained model, performs the prediction of classification of the audio segments. The DNN classifier consists of multiple layers between input and output. The processing layers are hidden and perform specific mathematical functions on the data received as input using weights and biases, according to Eq.(3) mentioned previously. The weights' parameter decides the influence of the input on the output and the bias is helpful to the network for the case when the input is zero. They together compute the output data and an activation function is applied to the calculated output before it is sent to the next layer. After the final computation of values acquired from the model, this process leaves all the segments marked as either '1' for STUTTER or '0' for NORMAL as shown in Figure 5 and concludes the classification.

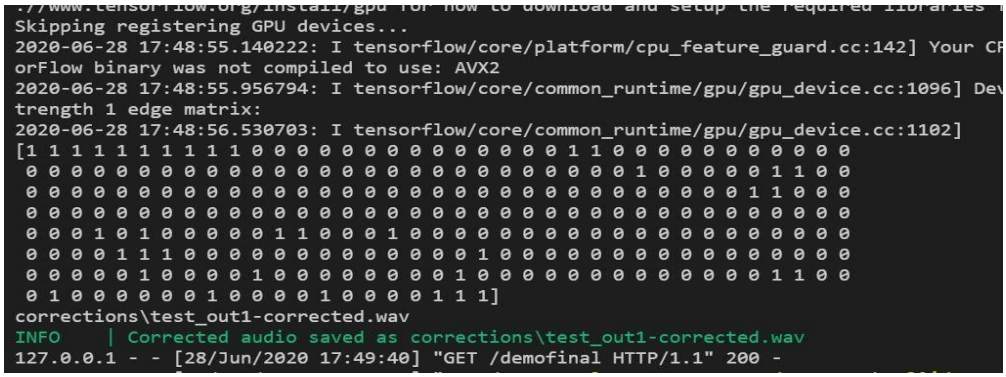

Fig 5: Classification of audio Segments

The classified data obtained is now sent to the Audio Correction module which takes all the segments which are classified as 'NORMAL' and interpolate them together to prevent any sharp interjections in audio that would have occurred because of those missing stutter points and gives a corrected audio file as the output.

Finally, the corrected audio is sent to Google's Speech-to-Text API to obtain its text interpretation which is displayed to the end-user on the UI. Figure 6 shows that the Speech to text conversion.

Fig 6: Speech to text conversion

5. Conclusion

Pausing at the right moment is very essential for conversing. To the speaker, it gives a breather while speaking and to the listener, it helps to understand the speech of the speaker. However, pausing in the case of stuttering is very different as compared to normal pauses, as in this case, the pauses are disturbances to the overall flow of the speech of the speaker. These disturbances in speech fluency are referred to as stutters. Stuttering is a huge problem that affects more than 70 million people around the world. Advancements in technology today allow a normal person to effortlessly interact with the modern-day devices using just his voice as a medium. For those who stutter, the use of such technology becomes limited. This work is an attempt at building a system that takes in a person's stuttered voice as input, rectifies it by removing the stutter portions, and gives a fluent speech audio output, along with its textual interpretation.

Data Samples for this research were obtained from University College London Archive of Stuttered Speech (UCLASS) which consists of recordings of speakers who stutter and provide the authors with background details about the conditions in which the recordings were made. The first method implemented is data processing that follows the breaking up of audio input into multiple segments of a duration of 300 ms each. The segments obtained had a lot of error margin. To reduce this error margin, the authors decided to overlap the segments. This in turn helped in distinguishing between stuttered and non-stuttered parts with a granularity of 100 ms.This helped in extracting MFCC features from each segment which were used for training our model. The same process is carried out for newly recorded speech audios from the UI. Overlapped segments from these audios are then sent into the classifier along with their feature vector. The trained model is used for the classification of these segments. Through our system, segments are classified with an overall accuracy of 85.9% and it takes about 40 seconds to process an audio sample of 2 minutes. Next, all the segments which were marked as NORMAL are interpolated to preserve smoothness and quality. Finally, the corrected speech file so obtained is passed through Google's Speech-to-Text API and text output is displayed to the end-user.

5.1.Future Developments

In the near future, progress will likely be made with large vocabulary recognition (up to ten thousand words or more), speaker-independent continuous speech recognition (up to a thousand words or more), and a more precise understanding of naturally spoken language.

Robust Speech Recognition devices for stuttered speech will need Artificial Intelligence that can handle challenges such as accents and background noise more efficiently and smoothly. It is believed that virtual voice assistants will be integrated into every appliance that we use. From alarm systems to kitchen devices, from devices for entertainment to those used in hospitals and workplaces, voice assistants are likely to rule over our lives. We might even have voice-driven vehicles and location-based searches based using voice, all of which are meant for the users to be completely hands-free, devoid of any visual or physical contact with their devices. Such advancements will have to cater to the needs of their users having stuttered speech as well.

References

[1] A. Czyzewski, A. Kaczmarek, and B. Kostek, 2003. Intelligent processing of stuttered speech. *Journal of Intelligent Information Systems*, *21*, pp.143-171.

[2] M.A. Anusuya, and S.K. Katti, 2010. Speech recognition by machine, a review. *arXiv preprint arXiv:1001.2267*.

[3] F. Afroz, and S.G. Koolagudi, 2019. Recognition and Classification of Pauses in Stuttered Speech Using Acoustic Features. In *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)* (pp. 921-926). IEEE.

[4] A.A. Surya, and S.M. Varghese, 2016. Automatic speech recognition system for stuttering disabled persons. *International Journal of Control Theory and Applications*, *9*(43), pp.16-20.

[5] L.S. Chee, O.C. Ai, M. Hariharan, and S. Yaacob, 2009. MFCC based recognition of repetitions and prolongations in stuttered speech using k-NN and LDA. In *2009 IEEE student conference on research and development (SCOReD)* (pp. 146-149). IEEE.

[6] https://wiki.aalto.fi/display/ITSP/Deltas+and+Delta-deltas.

[7] P. Arbajian, A. Hajja, Z.W. Raś, and A.A. Wieczorkowska, 2018. Segment-removal based stuttered speech remediation. In *New Frontiers in Mining Complex Patterns: 6th International Workshop, NFMCP 2017, Held in Conjunction with ECML-PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Revised Selected Papers 6* (pp. 16-34). Springer International Publishing.

[8] D. Gartzman, 2020. Getting to know the mel spectrogram. *Towards Data Science*.

[9] K N, V. N., and S P, M. 2016. Detection and Analysis of Stuttered Speech. International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE), 5(4), pp. 952-955.

[10] S. Khara, S. Singh, and D. Vir, 2018. A comparative study of the techniques for feature extraction and classification in stuttering. In *2018 Second International Conference on Inventive*

Communication and Computational Technologies (ICICCT) (pp. 887-893). IEEE.

[11] https://librosa.org/librosa/generated/librosa.feature.melspectrogram.html.

[12] J. Loy, 2020. How to build your own Neural Network from scratch in Python. https://towardsdatascience.com/how-to-build-your-own-neural-network-from-scratch-in-python-68998a08e4f6