Extended Essay: Computer Science

Investigating the Impact of Various Feature Extraction Algorithms on Performance in Automatic Speech Recognition Systems

How Effective Are Mel-Frequency Cepstrum Coefficients in Speech Recognition Neural Networks Compared to Other Speech Feature Extraction Algorithms?

3942 Words

Contents

List of Abbreviations

ASR	**A**utomatic **S**peech **R**ecognition
HMM	**H**idden **M**arkov **M**odel
NLP	**N**atural **L**anguage **P**rocessing
MFCC	**M**el **F**requency **C**epstral **C**oefficient
DWT	**D**iscrete **W**avelet **T**ransform
DDC	**D**irected **D**ialogue **C**onversations
NLC	**N**atural **L**anguage **C**onversations
RNN	**R**ecurrent **N**eural **N**etwork
LSTM	**L**ong **S**hort **T**erm **M**emory
STFT	**S**hort-**T**ime **F**ourier **T**ransform
CTC	**C**onnectionist **T**emporal **C**lassification
WER	**W**ord **E**rror **R**ate

Chapter 1

Introduction

Automatic speech recognition (ASR) is the ability of a machine to recognize language from human speech and convert it into written text [1]. Due to the endless number of possible and practical applications of ASR technology in our developing world, major corporations still aspire to develop an accurate yet optimized speech recognition system for their products and services. For instance, Baidu spent a total of $4.7 billion dollars on research and development [2], of which a large portion headed into developing their voice assistant, DuerOS. Hence, the implementation of ASR can improve the ergonomics of any task that requires human interaction, such as personal assistants, smart healthcare and household appliances, and media transcription to support people with disabilities, hearing impairments, or language barriers. However, such a task is difficult due to the sheer complexity of any spoken language, as well as the varying and unpredictable nature of different people and their speaking habits. Even big-name companies such as Alphabet, Microsoft, and Apple, who pride themselves in their sophisticated multi-million dollar speech recognition systems, have been countlessly ridiculed for misinterpretations in their voice recognition products even today.

The field of ASR is rather new compared to other fields in computer science as significant developments in ASR only began in the 50s, with the innovation of Bell's Audrey and IBM's Shoebox [3], [4]. These systems fell into a category of ASR systems that recognized individual words to an isolated vocabulary using manually set parameters for phoneme recognition [4]. By the end of the decade, ASR technology could distinguish words with a maximum of four vowels and nine consonants and by the 70s, systems could recognize up to 1000 words, which is around the vocabulary of a three-year-old toddler [3]. It was not until the late 90s that researchers transitioned outwards of phoneme detection to Hidden Markov Models (HMM) and focused on natural language processing [4]. Deep learning concepts and technologies emerged and were investigated to be a reliable alternative to HMMs, to the extent to which it is prominently used today.

However, a plausible deep learning system that can recognize continuous speech within a low margin of error is very strenuous to build. A well-designed system can consist of multiple complex layers that can be very complicated to integrate into one another. Nowadays, large tech companies boast an accuracy of almost 95%, which is still fairly low in the context of sophisticated deep-learning models [3]. One of these layers is commonly a feature extraction layer that can optimize an input waveform and isolates specific qualities of the inputted speech data [5], [6]. Modern speech recognition models widely use algorithms such as Mel-Frequency Cepstrum Coefficients (MFCCs) as a preprocessing layer. Yet, MFCCs often have shortcomings when processing audio signals, such as being excessively sensitive to noise and imperfections in waveforms [5].

This report aims to investigate the extent to which Mel-Frequency Cepstrum Coefficients outperform other feature extraction methods in automatic speech recognition systems. The performances of short-term Fourier transform spectrograms, MFCCs, Mel spectrograms, and discrete wavelet transforms (DWTs) will be assessed against each other by being processed through identical ASR neural network architectures. From this investigation, the effectiveness and limitations of different speech feature extraction methods will be better understood and advise forthcoming decisions in ASR neural networks.

Chapter 2

Background

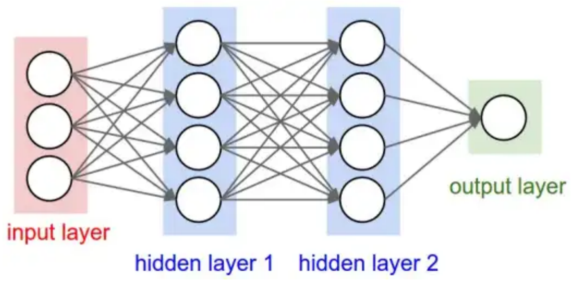
2.1 Automatic Speech Recognition

Automatic speech recognition models can be split into two general categories of complexity. First of all, there is single-word classification to an isolated vocabulary where directed dialogue conversations (DDC) are analyzed; its overall capabilities can be seen in early ASR efforts such as IBM's Shoebox. The other variant is continuous speech recognition, where natural language conversations (NLCs) are processed and transcribed to a vocabulary of around 20,000 to 100,000 words. Many technologies today utilize this system, including video subtitles, voice input systems, and customer services at call centres. While DDC classification models can be fairly simple to understand and produced through feed-forward networks, continuous NLC networks require the use of recurrent neural networks.

2.2 Recurrent Neural Networks for ASR

Recurrent neural networks (RNNs) are a variant of deep learning neural networks that use time-sequential data without a fixed input shape. These systems are inspired by the biological composition of neurons in the brain; modern society often adapts its greatest innovations from nature. Each neuron, out of around 100 million, is interconnected with one another and passes a signal via its axon to receiving axon terminals belonging to other neurons [7]. Similarly, in an artificial linear neural network, a combination of weights and biases stacked in layers are constantly optimized through gradient descent to produce a matrix of probabilities that highlight the best possible output the network can generate in its current state. In each cell, a dot product of the input matrix and the cell's weight matrix is done and summed with the cell's bias matrix to create an output matrix of values, which is eventually fed into a proceeding layer of nodes (Figure 2.1) [7].

At first, these matrices carry out operations using senseless, randomly generated weights and biases that result in a completely random output. However, as the model calculates its loss from an evaluation of the net deviation of the predicted output from the ground truth, a gradient is formulated that adjusts each layer's parameters closer to what would create an output with a lower loss [7]. The higher loss an output receives,



$$y = w \cdot x + b$$

w: weights, b: biases

FIGURE 2.1: The nodal, layer-by-layer architecture of a feed-forward neural network (Source: Mukul Rathi)

the greater magnitude of the gradient is passed to backpropagation and the more each node in the network is changed, with the vice versa applying as well.

In a typical feed-forward neural network, the input and output shapes of the model are fixed and have to be consistent in order to be trained properly. Since the length of the NLC audio input can vary in this investigation, recurrent neural networks will be used to process continuous speech data that last an indeterminate length of time. RNNs accomplish this task by accepting input for each time step into a recurrent cell and feeding the cell's generated output to the succeeding cell through a hidden layer that stores a certain number of hidden units, doing so until it reaches the end of the input batch [7], [8]. This results in the interconnected nature of a recurrent neural network that enables it to find relationships in continuous data [9], namely in stock market prediction, and influence future outputs. The framework of a standard RNN cell is laid out below (Figure 2.2), where xt is the inputted time step data, ot is the cell output after the tanh function is carried out, ht-1 is the previous cell's hidden output, and ht is the current cell's hidden output to be sent to the next cell.

The distribution of input data and output data can vary from network to network and is ultimately based on the shape of the input data and the desired shape of the output data, unlike feedforward networks [9]. For example, a music generation RNN could take in a dataset of Vivaldi's works as an input and output an infinite length of completely original Vivaldi-styled music: this model would be classified as a one-to-many RNN structure. Table 2.1 describes four main types of RNN structures that satisfy most criteria of networks that use continuous data.

Due to the continuous nature of transcribing NLCs, there are an indeterminate number of inputs and outputs present. Thus, a many-to-many RNN structure will be used as the framework for this investigation's experimentation. Additionally, many high-performance NLC RNNs include another layer of recurrent cells that examines the input data in reverse as well, called bidirectional RNNs, in order to better distinguish the meaningful attributes and relationships in language.

However, conventional recurrent neural networks have two major problems that need

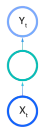
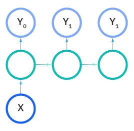
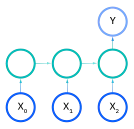
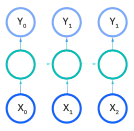
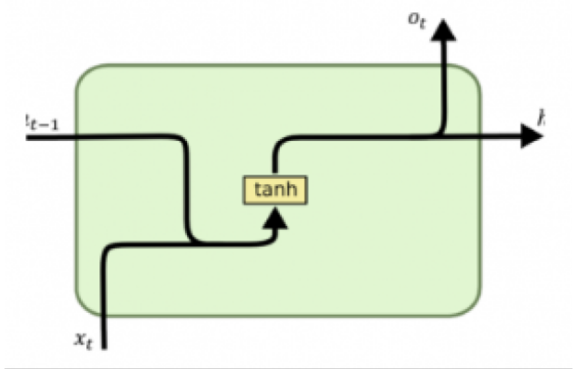
Architecture	**Characteristics**	**Application**	**Diagram**
One-to-One	Single input Single output	Image classification	
One-to-Many	Single input Sequence of outputs	Image captioning	
Many-to-One	Sequence of inputs Single output	Sentiment analysis	
Many-to-Many	Sequence of inputs Sequence of outputs	Language translation	

TABLE 2.1: Four types of recurrent neural network structures (Source: IBM [9])



$$h_t = \tanh(W(\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}))$$

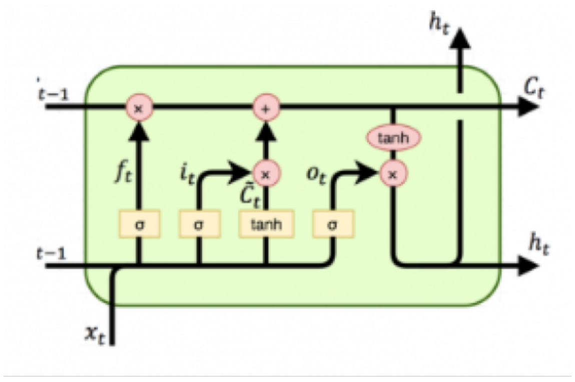
FIGURE 2.2: The internal composition of a recurrent network cell (Source: Lopez [10])

to be considered: the vanishing and exploding gradient problem, and a long-term dependency on speech waveforms. The vanishing and exploding gradient problem appear during the backpropagation stage of the network, where a gradient propagates through each cell and updates the weights and biases of each node for the network to learn [11]. The problem arises when the number of layers starts to increase and the gradients start to travel and multiply through more and more layers; when a gradient with a slight deviation from 1 is returned to the first node during backpropagation, it begins to increase or decrease exponentially as it progresses through each cell. Correspondingly, as it reaches the end of backpropagation, the gradient is either too minuscule to the point where the end weight is scarcely updated at all, or too large where the weight cannot be adequately optimized [11], [12].

Going on, audio waveforms can take up a significant portion of input data, dependent on sample frequency, as a one-second audio clip of someone dictating a word can take up to 16,000 time steps of data at a sample rate of 16 kHz. The extreme short-term memory of a simple hyperbolic tangent recurrent network cannot remember what sound signal was inputted into the cell 5,000 iterations earlier, to the extent that it is significant towards producing the output. This is detrimental towards the ultimate performance of the neural network, as a large segment of a model's decision on a phoneme, character, or word depends on contextual data beforehand [7]. For example, a model could have difficulties discriminating between the letter "k" and "q" with no context. However, the prerequisite knowledge that it was preceded by a "loo" will likely increase the probability that the following letter is "k". These two setbacks of recurrent neural networks introduced a revolutionary recurrent cell concept first established in 1997 and prominently used today, termed the Long Short-Term Memory cell [12].

2.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) cells counter the two problems above by incorporating an extra hidden memory state that propagates throughout recurrences of the network. Adding a memory state that can remember important specifics from many recurrences beforehand influences the current output [7], [12]. Contrary to the perception-based framework of traditional neural networks, an LSTM cell is much more convoluted and contains substantially more weights and biases in the form of specialized gates: the forget gate, input gate, and output gate (Figure 2.3), each with a specialized purpose [7], [13]. The combination of these properties provides the fundamental framework that conceives the LSTM network's ability to recognize context-sensitive information and is the reason why these networks are the standard for building reliable speech recognition systems.



$$\begin{pmatrix} i_{gate} \\ f_{gate} \\ o_{gate} \\ g_{gate} \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \cdot c_{t-1} + i \cdot g$$

$$h_t = o \cdot \tanh(c_t)$$

FIGURE 2.3: The internal composition of an LSTM cell (Source: Lopez [10])

A simple speech recognition system is usually composed of two different components: a feature extraction layer and the neural network itself. The purpose of the feature extraction layer is to process the raw input data into a more usable form that captures a certain aspect of sound [5]. In the case of ASR, input data is typically in the form of waveform data embedded in mp3 or wav files. This one-dimensional waveform data has several limitations due to having less useful features and larger variability in the speech signal than a two-dimensional format of audio signals [5], such as spectrograms. Spectrograms are a visual representation of the amplitudinal spectrum of frequencies of an audio signal with respect to time[14]. These amplitude maps of frequency can be extracted through the discrete short-time Fourier transform (STFT), which splits an audio sample into smaller segments of waveform data and processes the segment through a Fourier transform to represent the waveform as a set of frequencies for a frame of time [15]. Concatenating these frames together creates a two-dimensional time series of frequencies that are called spectrograms. The discrete STFT function is defined as:

$$X\left[n, w\right[= \sum_{m=-\infty}^{\infty} w[n-m] \cdot x[m] \cdot e^{-iwm}$$
$$n : frameindex, w : frequency$$

Spectrograms are frequently used in convolutional neural networks due to their multi-dimensional property and are the reliable standard in terms of extracting audio features from waveform data since they can accurately represent their original audio signals [16]. In a spectrogram, the x-axis refers to time and the y-axis refers to the frequency that's played at a certain point in time, with a lighter colour indicating that the amplitude is larger with regard to magnitude (Figure 2.4) [16].

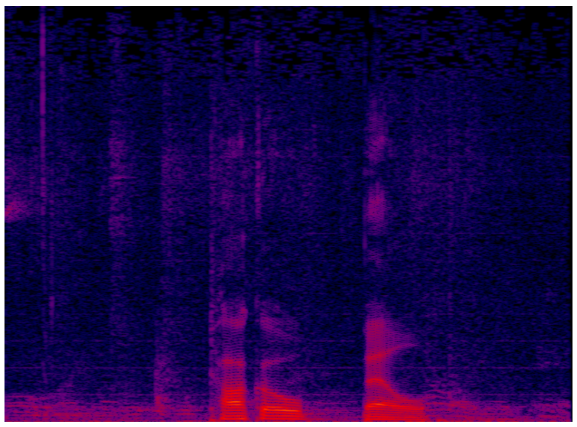
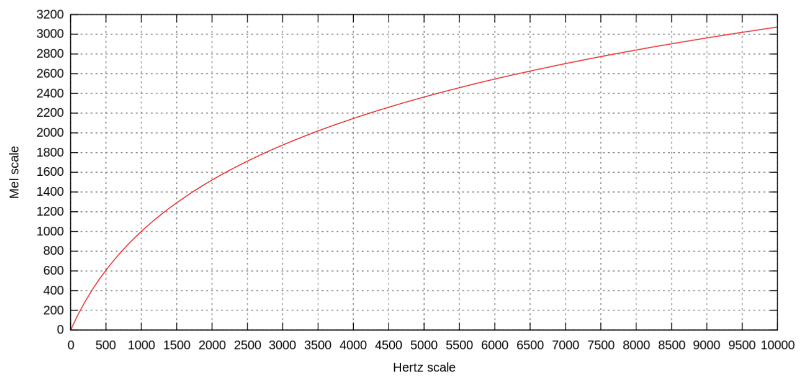


FIGURE 2.4: A spectrogram of a person saying the phrase, "Taco Bell"

2.4 Feature Extraction Algorithms

However, spectrograms are not very well suited nor optimized for visualizing human speech. For instance, a human voice at a conversational level will normally not exceed a frequency range between 60 Hz and 500 Hz and it's not necessary to include non-phonetically vital properties of speech signals in the input data. Therefore, Mel-frequency spectrograms were intended to artificially replicate the human hearing system on how we perceive different frequencies [21, towardsdatascience]. Mel spectrograms are computed through a precise logarithm function, dubbed the Mel-frequency filter bank, which retains the phonetically vital properties of speech signals and plots the result on the Mel scale (Figure 2.5) [17].



$$mel(f) = 2595x \log_1 0(1 + f/700)$$

FIGURE 2.5: The logarithmic relationship between the Hertz and Mel scale (Source: Vedala [18])

On the other hand, Mel-Frequency Cepstrum Coefficients are a set of coefficients that represent the changes in a spectrum of Mel frequencies over time [5]. The process involved in developing MFCCs includes taking the logarithm of a Mel spectrogram and applying the inverse discrete cosine transform to create a cepstrum of the original spectrum [19]. The result is a small set of coefficients, typically around 10-20, that ultimately represent the main speech features of the original waveform data. The number of coefficients outputted can be tuned as a hyperparameter to represent different types of audio data optimally. The small space complexity of MFCCs results in fast training and prediction times.

Another speech feature extraction function that will be evaluated is discrete wavelet transforms (DWTs). DWTs decompose a signal using a defined set of wavelets at a particular set of scales to represent the waveform and can extract local spectral and temporal information simultaneously [20], [21]. It uses an algorithm similar to the STFT in addition to convolutions in convolutional neural networks. A specific wavelet, or window, "slides" through the input and multiplies itself with the input waveform [21], [22], creating a convolutional output. The wavelet executes multiple passes through the waveform, changing its wavelength each time, and outputs how much of the wavelet is within a signal, similar to how kernels behave in convolutional neural networks [22] (Figure 2.6).

The DWT of a signal is determined/specified by the function [22]:

$$T_{m,n} = \int_{-\infty}^{\infty} x(t) \cdot \psi(t - n/m) dt$$

$$\psi : wavelet, m : scaling factor, n : translation parameter$$

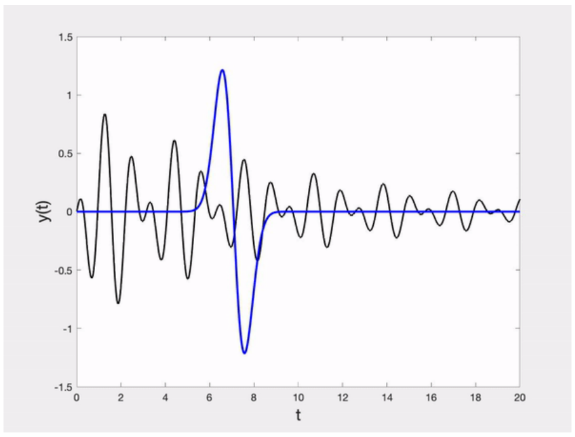


FIGURE 2.6: A wavelet traversing through a waveform (Source: Towards Data Science [21])

Chapter 3

Experiment Materials & Methodology

In this paper, heavy importance is placed on primary experimental observations and data. Thus, an LSTM neural network is implemented to perform the investigation, with the use of Google's Tensorflow machine learning API. Using this API allowed for ease of visualizing performance data and greater freedom in the network's architecture. The network architecture will remain a control variable throughout all trials. The independent variables are the speech feature extraction algorithms, which include STFT spectrograms, Mel spectrograms, MFCCs, and DWTs. The dependent variables are the loss calculated by connectionist temporal classification (CTC) loss, word error rate (WER), and the time taken by each network when processing the test dataset. WER is calculated by the number of substitutions, insertions, and deletions divided by the number of characters and it is a more accurate metric to assess linguistic proximity than loss [23].

3.1 Datasets Used

The LJ Speech Dataset is used to train the neural network that conducted the experiment [24]. The dataset contains 13,100 NLC audio clips of around 4-8 seconds that are spoken by a single speaker reading from a selection of non-fiction literature [24], amounting to a total of approximately 24 hours of data. 10% of this data is used for validation while 90% is fed into training. All in all, there is an average of 17 words that are spoken in each 6-second clip and there are around 13,800 distinct words said among all the audio clips [24].

3.2 Pre-Processing

The dataset provided the sound files in wav format, which made it very simple to extract its waveform data. Tensorflow's audio analysis library in combination with PyWavelets [25] performs each speech feature extraction algorithm on every sample in the dataset to obtain the respective input data (Figure 3.1).

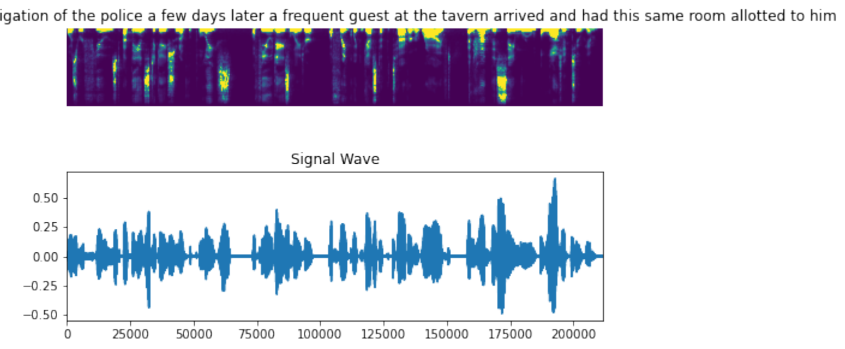


FIGURE 3.1: Sample of training data paired with its transcription, visualized as a Mel Spectrogram using Tensorflow's signal library and its respective signal wave

3.3 Network Architecture

This report uses an end-to-end recurrent character-based neural network that consists of two convolutional layers, five layers of bidirectional LSTM cells, a dense layer, and a decoding layer to process and display the final output. The LSTM layers take in a time step of data (whose shape is dependent on the feature extraction algorithm), utilize 128 hidden units in each cell and output a one-hot matrix of shape (31, 1) at each time step. The output is a probability matrix of 31 characters that the model predicts the audio signal refers to at that point in time, which includes all of the Latin alphabet, three punctuation marks, as well as a space and a blank character. A character classification system is used over phoneme-based or word-based systems since with the current LSTM network architecture, it is capable of distinguishing between similar-sounding characters in the English dataset used. A working result of this model would output a sequence of slurred letters that, once decoded with CTC loss, would output a word; the model could output "b_uu_t_ _n_oo", which suggests that the original audio input is of a person uttering "but no" [26], [27]. After this procedure, these characters can be fed into an English language model to be further refined into comprehensible written text as the final output. Refer to Appendix C for model details.

The LSTM model is trained on 389 batches of training data with each batch consisting of 32 audio clips. The model is trained for 5 epochs for each set of independent variables, with a learning rate of 0.0001 to avoid overfitting, and evaluated by three validation datasets. Each of the three trials holds 21 batches of data that determine the accuracy of each model.

Regarding the feature extraction algorithms, the MFCC trials will have 21 coefficients and the DWT trials will use a biorthogonal 6.8 wavelet (Figure 3.1).

3.4 Experimental Process

It is essential to clarify some notable semantics of this investigation. The general performance of a model within this investigation is assessed on a synthesis of its accuracy

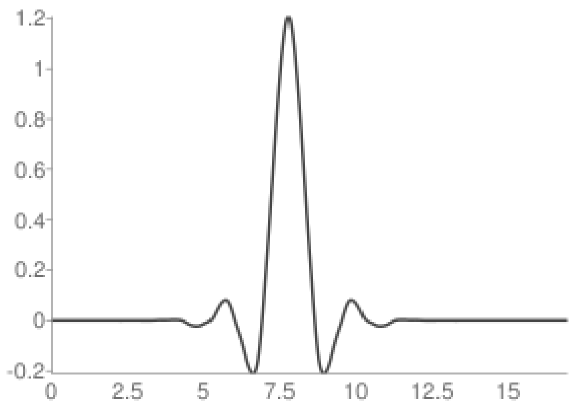


FIGURE 3.2: A biorthogonal 6.8 wavelet, visualized (Source: Waselewski [28])

and its efficiency as well as how it compares with other models, though with the greatest emphasis on accuracy. The accuracy of a model is calculated appertaining to both its loss from CTC and its word error rate, and the efficiency of a model is gauged on a comparison of its estimated time complexity and a measured difference of time. For each speech feature extraction algorithm, there will be three trials held, amounting to a total of 4 models and 12 trials.

Chapter 4

Experimental Results

4.1 Tabular Presentation of Data

The tables in Appendix A and Appendix B show the raw experimental results of all the trials, with each table presenting the performance of each feature extraction algorithm in pre-processing speech for ASR neural networks.

4.2 Graphical Presentation of Data

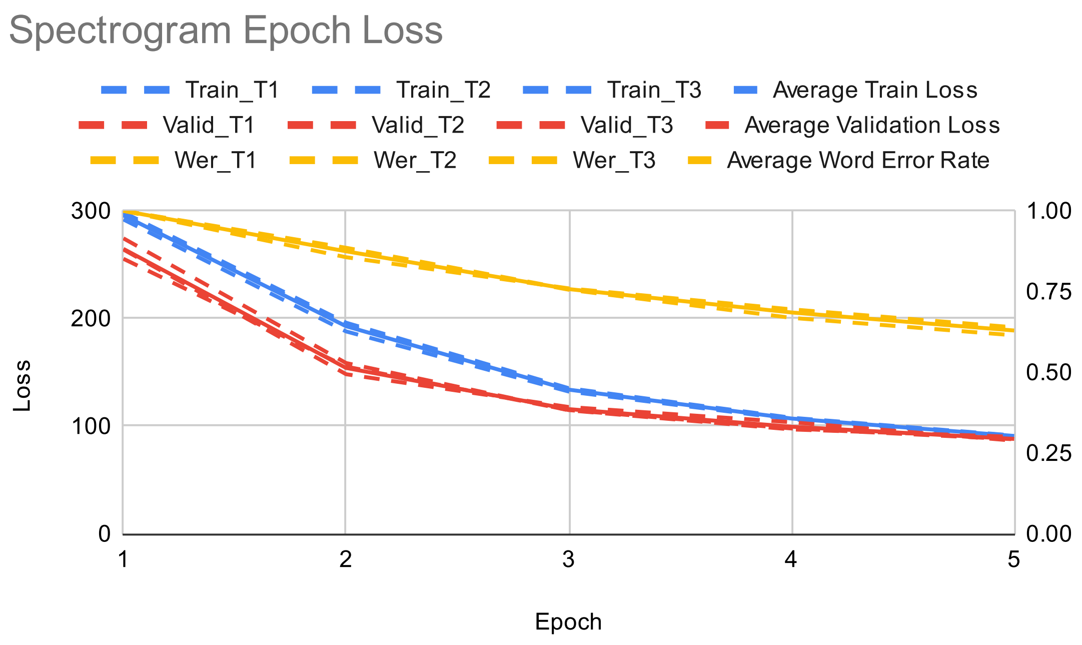


FIGURE 4.1: Spectrogram

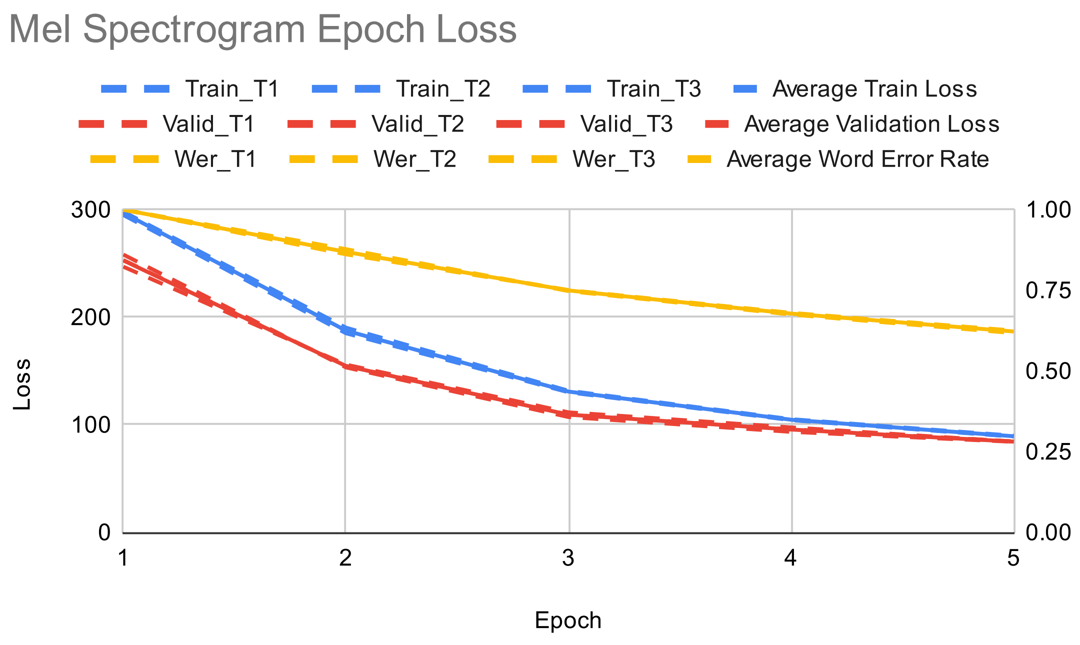


FIGURE 4.2: Mel Spectrogram

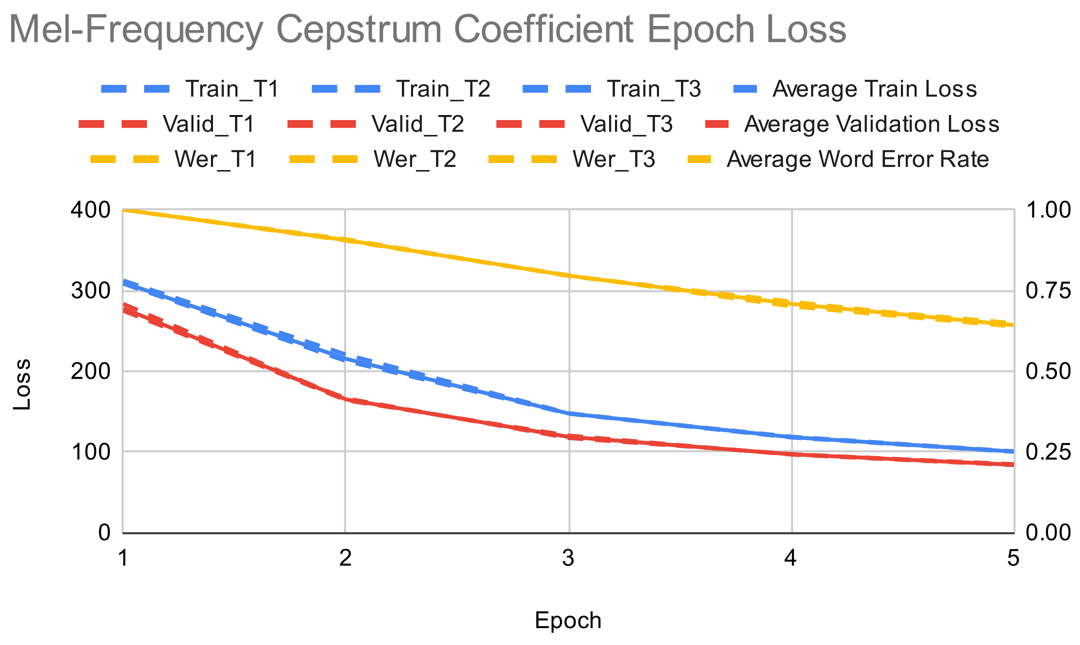


FIGURE 4.3: MFCC

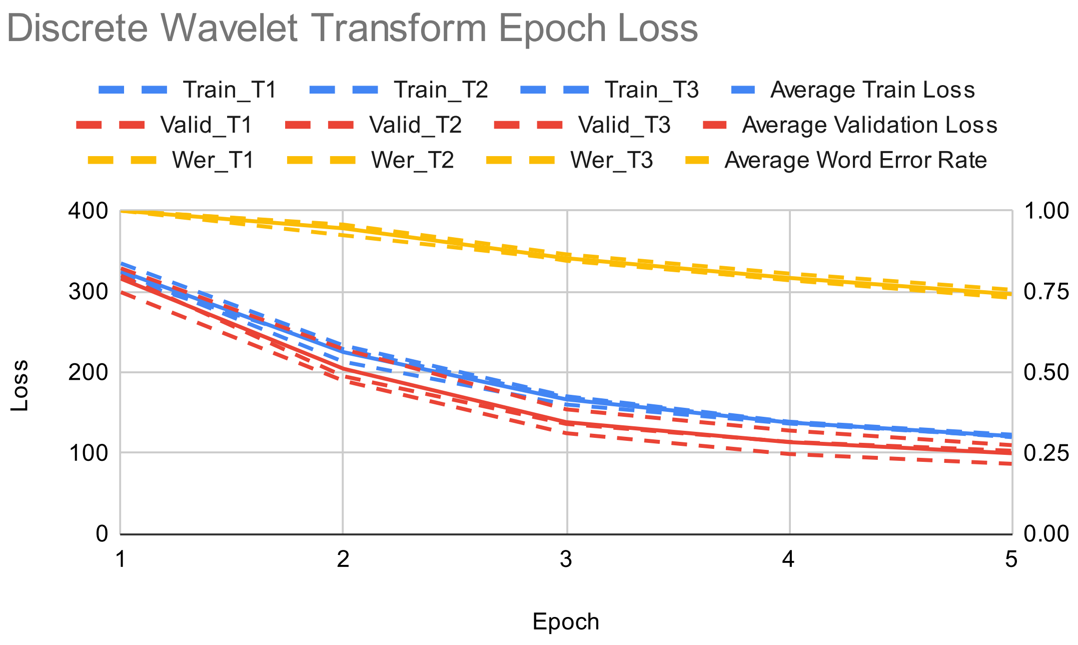


FIGURE 4.4: DWT

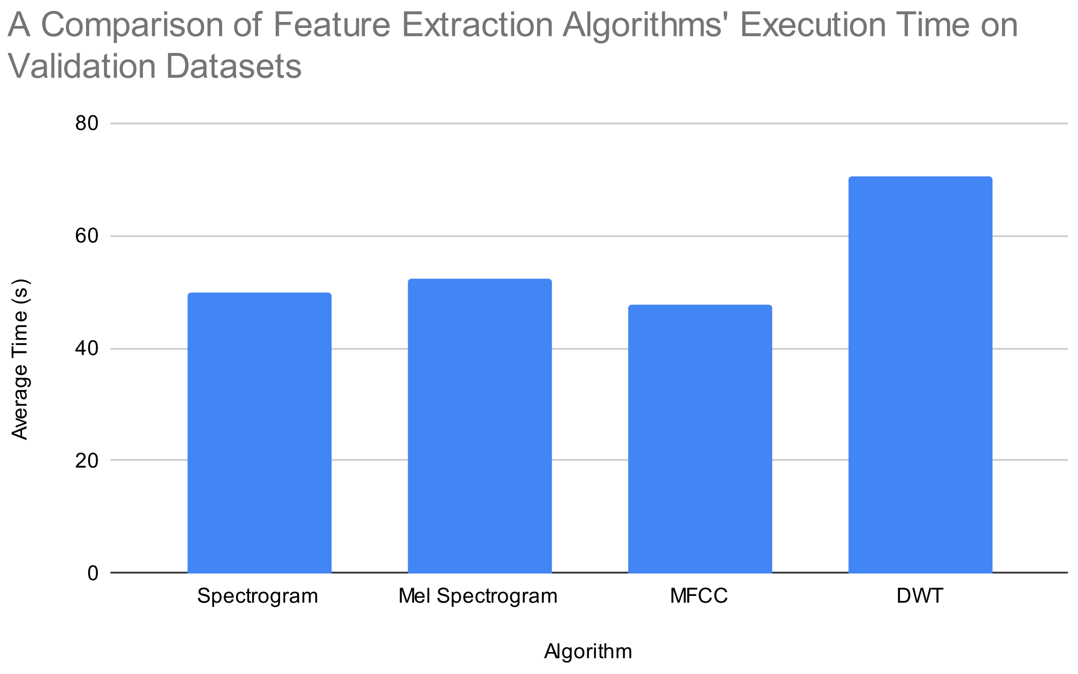


FIGURE 4.5: Execution Times

Chapter 5

Analysis & Conclusions

5.1 Experimental Analysis and Trends

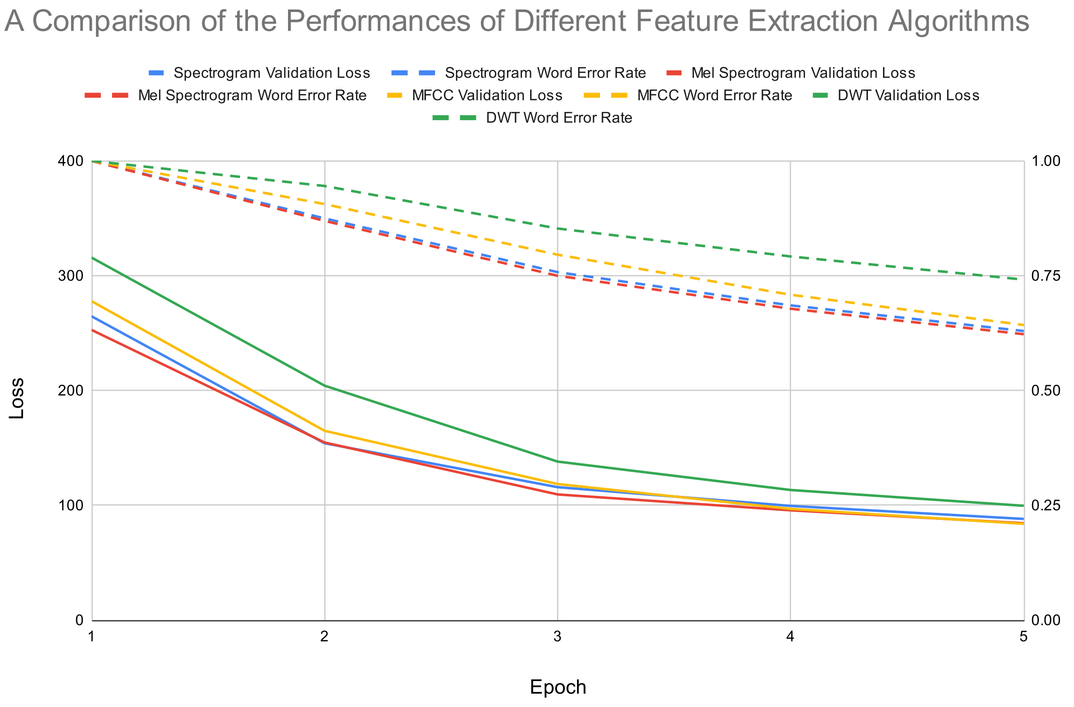


FIGURE 5.1: A comparison of extraction algorithms' accuracies against each other

In the visualizations of the experiment results above, much can be inferred about how differing feature extraction algorithms perform against each other. As observed in (Figure 5.1), spectrograms, Mel spectrograms, and MFCCs performed similarly, with similar results in loss and WER, while DWT's performance is noticeably worse than any other algorithm in terms of both accuracy and time spent. While other speech extraction algorithms had an ending WER of roughly 63%, DWT's average WER in epoch 5 is 74%. DWT's poor performance could be attributed to two possible causes: either the algorithm is inefficient in extracting audio features or there is error present in implementing the algorithm. DWT outputs coefficients whose values indicate the overall resemblance of a wavelet to any window of a specific waveform. The number of coefficients can vary but are generally very low compared to the number of data present in spectrograms, resulting in an inevitably lossy output within this investigation. Additionally, it is the sole algorithm that did not use Tensorflow's signal library and instead PyWavelets, which may have caused the significant disparity in execution time; this factor is further elaborated below.

Concerning the performances of spectrograms, Mel spectrograms, and MFCCs, it is seen in (Figure 5.1) and (Figure 4.5) that their accuracies and times are very similar, but have small discrepancies that are important to distinguish. Mel spectrograms and MFCCs generally achieve better validation loss than raw spectrograms by a value of 4 while spectrograms and Mel spectrograms have slightly lower word error rates than MFCCs, by nearly 2%. Since the performance of Mel spectrograms remained consistently efficient in both metrics of accuracy, Mel spectrograms overall performed the best among all feature extraction algorithms explored in this study, though to a little extent.

Furthermore, comparing the time complexity of each algorithm reveals that MFCCs are the most efficient. This could be a result of the less trainable parameters needed in the neural network to process the compressed output of MFCCs. To elaborate, a time step of data produced with MFCC that is passed to an LSTM cell would consume around 21n spaces of data, with n being the number of time steps. In contrast, a spectrogram would normally take up nearly 193n spaces of data per time step[1], depending on the frequency range desired while computing the STFT, which would require levels of magnitude higher levels of computing required to process an audio signal.

5.2 Further Research Opportunities and Aspirations

The experimental process could be improved, as it led to a few sources of error: the first of which lies in the consistency of the speed of the computer the tests were run on. In Table B.1 and B.4 found in Appendix B, the large standard deviation of the spectrogram and DWT trials shows that there is a large uncertainty in measuring execution time can exist, possibly due to the varied use of the training CPU that occurred during the trials. This error can be disregarded if an uncertainty of ±1 second is added to the running times of each trial as a result of systematic irregularity in hardware.

Another source of error mentioned above is the inconsistency in the selection of audio processing libraries used to carry out the feature extraction algorithms. Consequently from the limited number of options available on any single audio analysis library that doesn't include the entirety of my independent variables, such as Librosa, multiple libraries had to be used, which altogether reduces the uniformity within my methodology. This error can be eliminated by implementing the algorithms manually without using third-party libraries.

[1]For a frame step of 160, frame length of 256, and fast Fourier transform length of 384

To expand the scope of future investigations regarding the research topic, there are several improvements that could be implemented. To attain more accurate, representative data and better overall performance of each algorithm in the neural network, more epochs of training data could have been trained if hardware limitations were not present. Additionally, a downside of the current dataset is that all of the NLC speech is spoken by one speaker, which is unrealistic in practical scenarios. Obtaining a dataset with a variety of speakers would better reflect the performances of these algorithms through more realistic analyses.

There are several research opportunities that could be explored regarding the research question. The speech feature extraction algorithms examined in this paper are a portion of the most conventionally used computer scientists use today for speech recognition. Even so, there are some modern and advanced yet ambiguous speech feature extraction algorithms out there that may perform even better than the ones investigated in this report but don't have the recognition to gain traction in the field of ASR. Investigating the extent to which these are viable as a replacement for MFCCs could yield interesting results.

In the future, it would be worthwhile to examine the extent to which MFCCs can perform well in noise-heavy environments with the addition of white noise and the use of noise reduction/cancellation systems; the experimental results currently presented do not draw out clear conclusions on whether MFCCs should be used if a system expects a high magnitude of background noise.

5.3 Conclusions

This paper presented several conclusions regarding the research question at hand through an analysis of the performances of different speech feature extraction algorithms. The results show that although MFCCs are considerably optimized and contain denser key vocal information of a waveform, Mel spectrograms overall possess the highest degree of accuracy while maintaining a reasonable processing time with large inputs. In a more practical sense, the use of spectrograms, Mel spectrograms, and MFCCs are generally all viable within a speech feature extraction layer in most non-industry ASR neural networks.

Bibliography

[1] R. Wang, *Automatic speech recognition 101: How asr systems work*, 2021. [Online]. Available: https://www.dialpad.com/blog/automatic-speech-recognition/.

[2] L. L. Thomala, *Baidu: Ramp;d spending 2021*, Apr. 2022. [Online]. Available: https://www.statista.com/statistics/1079978/china-baidu-research-and-development-costs/.

[3] S. Authors, *A short history of speech recognition*, 2022. [Online]. Available: https://sonix.ai/history-of-speech-recognition.

[4] S. L. Authors, *Speech recognition software: History, present, and future*, Jun. 2021. [Online]. Available: https://summalinguae.com/language-technology/speech-recognition-software-history-future/.

[5] P. V. Janse, S. B. Magre, P. K. Kurzekar, and R. R. Deshmukh, "A comparative study between mfcc and dwt feature extraction technique," *International Journal of Engineering Research amp; Technology*, vol. 3, no. 1, Jan. 2014. DOI: 10.17577/IJERTV3IS11110.

[6] S. Ajibola Alim and N. Khair Alang Rashid, "Some commonly used speech feature extraction algorithms," *From Natural to Artificial Intelligence - Algorithms and Applications*, Dec. 2018. DOI: 10.5772/intechopen.80419.

[7] P. Koehn, *Neural Machine Translation*. Cambridge University Press, 2020.

[8] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 38, Mar. 2013. DOI: 10.1109/icassp.2013.6638947.

[9] I. C. Education, *What are recurrent neural networks?* Sep. 2020. [Online]. Available: https://www.ibm.com/cloud/learn/recurrent-neural-networks.

[10] d. lopez dProgrammer, *RNN vs LSTM vs GRU*. Apr. 2019. [Online]. Available: http://dprogrammer.org/wp-content/uploads/2019/04/RNN-vs-LSTM-vs-GRU-1024x308.png.

[11] S. Mehrotra, *Let's understand the problems with recurrent neural networks*, Jul. 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/07/lets-understand-the-problems-with-recurrent-neural-networks/..

[12] J. Brownlee, *A gentle introduction to long short-term memory networks by the experts*, Jul. 2021. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/.

[13] V. Lendave, *Lstm vs gru in recurrent neural network: A comparative study*, Aug. 2021. [Online]. Available: https://analyticsindiamag.com/lstm-vs-gru-in-recurrent-neural-network-a-comparative-study/.

[14] S. K. Mahanta and A. Padmanabhan, *Audio feature extraction*, May 2021. [Online]. Available: https://devopedia.org/audio-feature-extraction.

[15] J. O. Smith. W3K Publishing, 2007. [Online]. Available: https://ccrma.stanford.edu/~jos/sasp/.

[16] G. Green, *Seeing sound: What is a spectrogram?* Sep. 2018. [Online]. Available: https://blogs.bl.uk/sound-and-vision/2018/09/seeing-sound-what-is-a-spectrogram.html.

[17] L. Li, "Performance analysis of objective speech quality measures in mel domain," *Journal of Software Engineering*, vol. 9, no. 2, pp. 350–361, Jan. 2015. DOI: 10.3923/jse.2015.350.361.

[18] K. Vedala, *Mel scale*, Oct. 2022. [Online]. Available: https://en.wikipedia.org/wiki/Mel_scale#/media/File:Mel-Hz_plot.svg.

[19] T. Singh, *Mfcc's made easy*, Jun. 2019. [Online]. Available: https://medium.com/@tanveer9812/mfccs-made-easy-7ef383006040.

[20] M. C. Nechyba, *Introduction to the discrete wavelet transform (dwt)*, Feb. 2004. [Online]. Available: https://mil.ufl.edu/nechyba/www/eel6562/course_materials/t5.wavelets/intro_dwt.pdf.

[21] S. Talebi, *The wavelet transform*, Dec. 2022. [Online]. Available: https://towardsdatascience.com/the-wavelet-transform-e9cfa85d7b34.

[22] N. S. Nehe and R. S. Holambe, "Dwt and lpc based feature extraction methods for isolated word recognition," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2012, no. 1, Jan. 2012. DOI: 10.1186/1687-4722-2012-7.

[23] Y.-Y. Wang, A. Acero, and C. Chelba, "Is word error rate a good indicator for spoken language understanding accuracy," *2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No.03EX721)*, pp. 577–582, Nov. 2003. DOI: 10.1109/asru.2003.1318504.

[24] K. Ito and L. Johnson, 2017. [Online]. Available: https://keithito.com/LJ-Speech-Dataset/.

[25] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. OLeary, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, Apr. 2019. DOI: 10.21105/joss.01237.

[26] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," *ICML 2006 - Proceedings of the 23rd International Conference on Machine Learning*, vol. 2006, pp. 369–376, Jan. 2006. DOI: 10.1145/1143844.1143891.

[27] H. Scheidl, *An intuitive explanation of connectionist temporal classification*, Jun. 2018. [Online]. Available: `https://towardsdatascience.com/intuitively-understanding-connectionist-temporal-classification-3797e43a86c`.

[28] F. Wasilewski, *Biorthogonal 6.8 wavelet (bior6.8)*, 2008. [Online]. Available: `https://wavelets.pybytes.com/wavelet/bior6.8/`.

[29] H. Ali, N. Ahmad, X. Zhou, K. Iqbal, and S. M. Ali, "Dwt features performance analysis for automatic speech recognition of urdu," *SpringerPlus*, vol. 3, no. 1, Apr. 2014. DOI: `10.1186/2193-1801-3-204`.

[30] K. Daqrouq, A.-R. Al-Qawasmi, K. Y. Al Azzawi, and T. Abu Hilal, "Discrete wavelet transform amp; linear prediction coding based method for speech recognition via neural network," *Discrete Wavelet Transforms - Biomedical Applications*, Sep. 2011. DOI: `10.5772/20978`.

[31] A. Hannun, "Sequence modeling with ctc," *Distill*, vol. 2, no. 11, Nov. 2017. DOI: `10.23915/distill.00008`.

[32] A. Lindgren and G. Lind, "Language classification using neural networks," *Dissertation*, 2019. DOI: `urn:nbn:se:uu:diva-385537`.

[33] K. S. Rao and M. K. E., "Speech recognition using articulatory and excitation source features," *SpringerBriefs in Speech Technology*, 2017. DOI: `10.1007/978-3-319-49220-9`.

[34] Y. Wang, X. Deng, S. Pu, and Z. Huang, "Residual convolutional ctc networks for automatic speech recognition," *CoRR*, vol. abs/1702.07793, Feb. 2017. DOI: `10.48550/arXiv.1702.07793`.

[35] J. Zhang, "English speech recognition system model based on computer-aided function and neural network algorithm," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–11, Jan. 2022. DOI: `10.1155/2022/7846877`.

[36] A. Biswal, *Recurrent neural network (rnn) tutorial: Types and examples [updated]: Simplilearn*, Nov. 2022. [Online]. Available: `https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn`.

[37] Mlearnere, *Learning from audio: The mel scale, mel spectrograms, and mel frequency cepstral coefficients*, Apr. 2021. [Online]. Available: `https://towardsdatascience.com/learning-from-audio-the-mel-scale-mel-spectrograms-and-mel-frequency-cepstral-coefficients-f5752b6324a8`.

[38] B. Ramakrishnan, *Lecture 16: Connectionist temporal classification, sequence prediction*, May 2021. [Online]. Available: `https://www.youtube.com/watch?v=RowViowx1Bg`.

[39] B. Ramakrishnan, *Lecture 17: Connectionist temporal classification (ctc), sequence to sequence prediction*, Apr. 2021. [Online]. Available: `https://www.youtube.com/watch?v=5Rj0J9AuGw0`.

[40] A. Taspinar, *A guide for using the wavelet transform in machine learning*, Aug. 2021. [Online]. Available: `https://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/`.

Appendix A

Appendix A - Epoch Loss

TABLE A.1: Spectrogram Epoch Loss

	Epoch	**1**	**2**	**3**	**4**	**5**
Training Loss	Train_T1	296.7976685	194.3861389	134.335434	107.0815659	91.11164093
	Train_T2	298.5567017	196.0717926	135.2273865	107.8201065	91.3082962
	Train_T3	291.9888611	187.9899597	132.1474152	106.3984909	90.31215668
	Av. Loss	295.7810771	192.8159637	133.9034119	107.1000544	90.91069794
Validation Loss	Valid_T1	274.4621887	158.631134	114.7272415	96.96182251	90.11564636
	Valid_T2	255.4617767	155.3216095	114.933876	97.87284088	87.8312149
	Valid_T3	264.06427	148.2462921	117.7789459	103.6573257	86.51172638
	Av. Loss	264.6627452	154.0663452	115.8133545	99.49732971	88.15286255
Word Error Rate	Wer_T1	1	0.8822	0.7558	0.6687	0.6146
	Wer_T2	1	0.8856	0.7573	0.6923	0.6345
	Wer_T3	1	0.8564	0.7599	0.695	0.6394
	Av. WER	1	0.8747333333	0.7576666667	0.6853333333	0.6295

TABLE A.2: Mel Spectrogram Epoch Loss

	Epoch	**1**	**2**	**3**	**4**	**5**
Training Loss	Train_T1	298.1916504	190.3350525	131.6872101	105.2731323	89.91330719
	Train_T2	295.0480347	185.1468048	130.2666931	104.3753891	89.07975006
	Train_T3	297.911764	187.3469113	130.8143732	104.652378	89.20348672
	Av. Loss	297.050483	187.6095895	130.9227588	104.7669665	89.39884799
Validation Loss	Valid_T1	258.2694092	153.4085236	107.2570343	93.50734711	84.25429535
	Valid_T2	247.0012207	155.9919586	111.6265259	97.62954712	84.40084839
	Valid_T3	253.1349877	154.7935403	109.5709144	95.81934759	84.61218773
	Av. Loss	252.8018725	154.7313408	109.4848249	95.65208061	84.42244382
Word Error Rate	Wer_T1	1	0.8617	0.7508	0.6785	0.6248
	Wer_T2	0.9999	0.8685	0.7508	0.6804	0.6254
	Wer_T3	1	0.8777	0.7478	0.6752	0.6176
	Av. WER	0.9999666667	0.8693	0.7498	0.6780333333	0.6226

TABLE A.3: MFCC Epoch Loss

	Epoch	**1**	**2**	**3**	**4**	**5**
Training Loss	Train_T1	312.3415527	220.6151123	147.9094543	118.0336456	100.1157913
	Train_T2	308.6204834	212.7577515	147.4524078	118.8214722	100.6679077
	Train_T3	308.2146301	212.3941498	147.0379944	117.6222458	99.95289612
	Av. Loss	309.7255554	215.2556712	147.4666189	118.1591212	100.2455317
Validation Loss	Valid_T1	283.334259	164.7569122	118.3945618	96.96144104	84.69897461
	Valid_T2	274.635437	166.2772675	117.0184784	97.25863647	83.58209991
	Valid_T3	275.5388794	163.729599	120.2116318	96.44332886	83.41139984
	Av. Loss	277.8361918	164.9212596	118.5415573	96.88780212	83.89749146
Word Error Rate	Wer_T1	1	0.909	0.7945	0.7097	0.6487
	Wer_T2	1	0.9042	0.7952	0.7152	0.641
	Wer_T3	1	0.9046	0.7973	0.7005	0.638
	Av. WER	1	0.9059333333	0.7956666667	0.7084666667	0.6425666667

TABLE A.4: DWT Epoch Loss

	Epoch	**1**	**2**	**3**	**4**	**5**
Training Loss	Train_T1	317.0186768	228.4565277	168.7938538	137.921936	119.5729599
	Train_T2	334.7597046	233.082016	170.1693726	138.6927338	122.6289673
	Train_T3	323.619873	212.8789063	159.9281158	136.3984909	120.3121567
	Av. Loss	325.1327515	224.8058167	166.2971141	137.6710536	120.838028
Validation Loss	Valid_T1	319.665802	194.9621735	135.9151306	113.6958008	102.68367
	Valid_T2	328.6252441	228.4376373	153.9455414	127.7307587	109.6193008
	Valid_T3	299.0671387	189.2872314	124.4108429	98.65732574	86.51172638
	Av. Loss	315.7860616	204.2290141	138.090505	113.3612951	99.60489909
Word Error Rate	Wer_T1	1	0.9568	0.8648	0.8053	0.7552
	Wer_T2	1	0.9554	0.8449	0.7857	0.7391
	Wer_T3	1	0.9238	0.8486	0.7846	0.7301
	Av. WER	1	0.9453333333	0.8527666667	0.7918666667	0.7414666667

Appendix B

Appendix B - Execution Time

TABLE B.1: Spectrogram Execution Time

Epoch	**1**	**2**	**3**	**4**	**5**	**Average**
Trial 1	52.4	51.5	51.4	51.5	51.5	
Trial 2	53	48.9	55.8	55.3	55.3	
Trial 3	46.5	44.2	43.1	40.6	45.2	
Average Execution Time (s)	50.63333333	48.2	50.1	49.13333333	50.66666667	49.74666667

TABLE B.2: Mel Spectrogram Execution Time

Epoch	**1**	**2**	**3**	**4**	**5**	**Average**
Trial 1	52.7	52.3	51.8	51.8	53.8	
Trial 2	55.1	53.2	54.2	51.8	52.9	
Trial 3	52.8	51	50.2	50	50.7	
Average Execution Time (s)	53.53333333	52.16666667	52.06666667	51.2	52.46666667	52.28666667

TABLE B.3: MFCC Execution Time

Epoch	**1**	**2**	**3**	**4**	**5**	**Average**
Trial 1	49.5	48.6	48.6	48	48.2	
Trial 2	48.4	46.7	46.9	46.9	47	
Trial 3	50.3	47.1	47.1	46.9	47.1	
Average Execution Time (s)	49.4	47.46666667	47.53333333	47.26666667	47.43333333	47.82

TABLE B.4: DWT Execution Time

Epoch	**1**	**2**	**3**	**4**	**5**	**Average**
Trial 1	85.2	83.4	77.6	69	57.2	
Trial 2	78.5	64.9	66.1	71.8	73.9	
Trial 3	88.5	52.7	53.3	53.5	83.5	
Average Execution Time (s)	84.06666667	67	65.66666667	64.76666667	71.53333333	70.60666667

Appendix C

Appendix C - Model Graph

TABLE C.1: Sample Network Architecture for Spectrogram

Layer (Type)	**Output Shape**	**Number of Parameters**
input (InputLayer)	"(None, None, 193)"	0
expand_dim (Reshape)	"(None, None, 193, 1)"	0
conv_1 (Conv2D)	"(None, None, 97, 32)"	14432
conv_1_bn (BatchNormalization)	"(None, None, 97, 32)"	128
conv_1_relu (ReLU)	"(None, None, 97, 32)"	0
conv_2 (Conv2D)	"(None, None, 49, 32) "	236544
conv_2_bn (BatchNormalization)	"(None, None, 49, 32) "	128
conv_2_relu (ReLU)	"(None, None, 49, 32) "	0
reshape_1 (Reshape)	"(None, None, 1568)"	0
bidirectionalLSTM_1 (Bidirectional)	"(None, None, 1024)"	8523776
dropout (Dropout)	"(None, None, 1024)"	0
bidirectionalLSTM_2 (Bidirectional)	"(None, None, 1024)"	6295552
dropout_1 (Dropout)	"(None, None, 1024)"	0
bidirectionalLSTM_3 (Bidirectional)	"(None, None, 1024)"	6295552
dropout_2 (Dropout)	"(None, None, 1024)"	0
bidirectionalLSTM_4 (Bidirectional)	"(None, None, 1024)"	6295552
dropout_3 (Dropout)	"(None, None, 1024)"	0
bidirectionalLSTM_5 (Bidirectional)	"(None, None, 1024)"	6295552
dense_1 (Dense)	"(None, None, 1024)"	1049600
dense_1_relu (ReLU)	"(None, None, 1024)"	0
dropout_4 (Dropout)	"(None, None, 1024)"	0
dense (Dense)	"(None, None, 32) "	32800
Total parameters		35039616

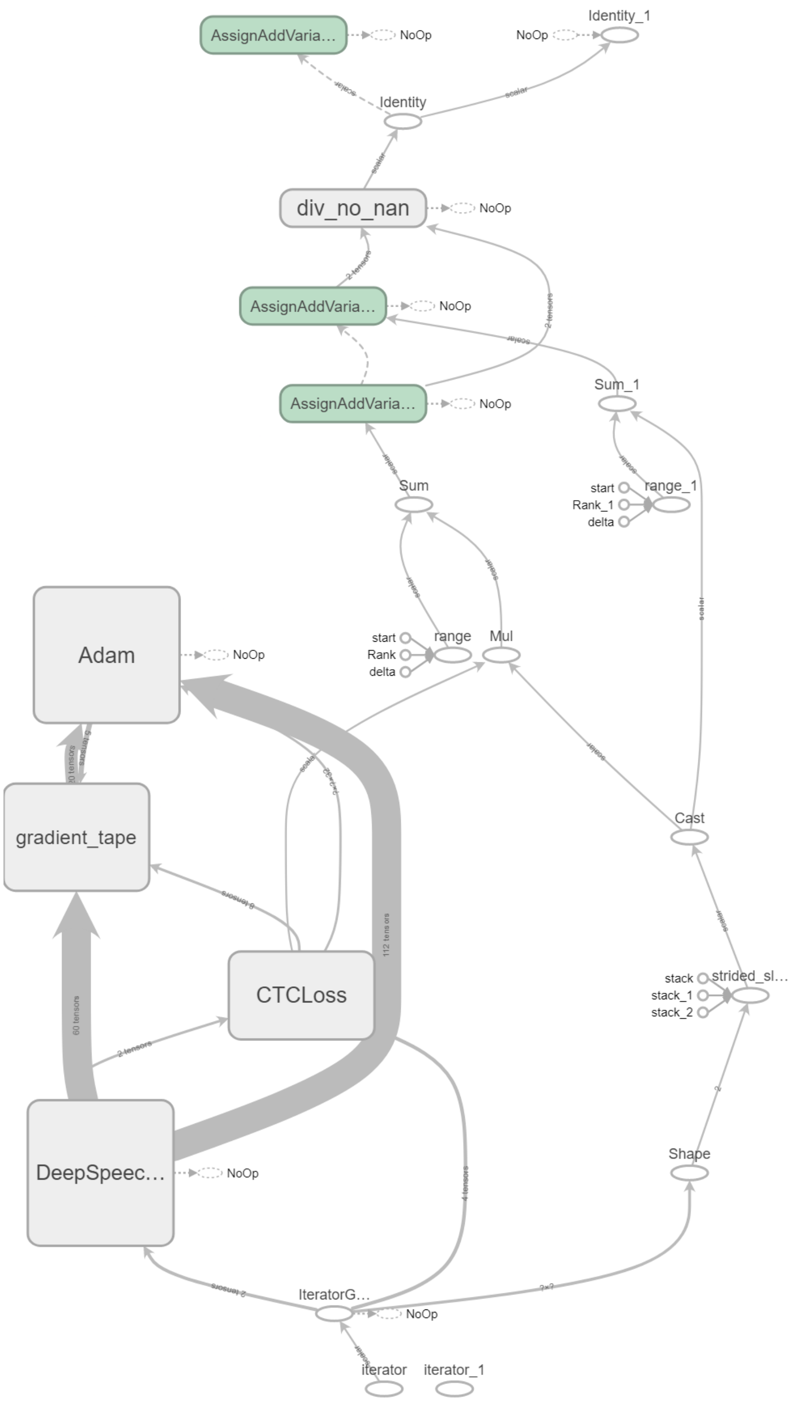


FIGURE C.1: A graph of the general Tensorflow model used in this investigation - deep speech is the name of the neural network outlined above