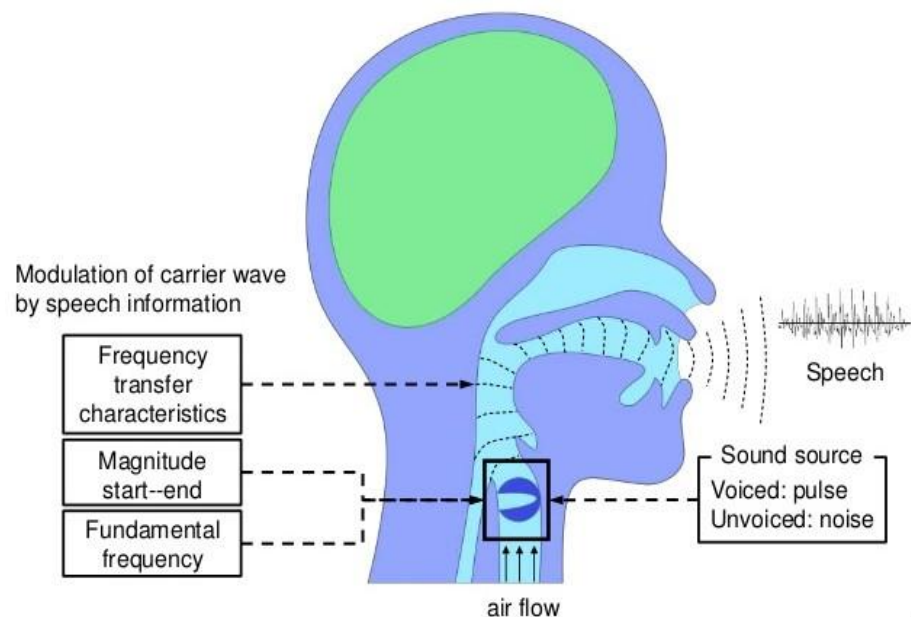


Text to Speech Synthesis Using Deep Neural Networks

Introduction

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read any text aloud. The input text may come from any of the sources like it may have been typed or scanned and read using OCR.

Introduction



Inspired by the human speech production system which is believed to have layered hierarchical structures in transforming the information from the linguistic level to the waveform level.

TTS Methodologies

Unit Selection :

In this model novel utterances are created by re-arranging fragments of pre recorded speech according to carefully crafted linguistic and acoustic cost functions. This is most used system in commercial use.

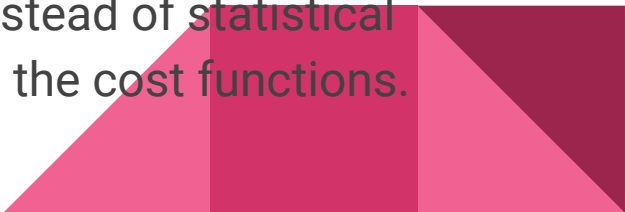
TTS Methodologies

The Statistical Parametric Approach :

No waveforms are stored. Here a speech database is used to train a set of context-dependent phone models, which are used to drive a vocoder at synthesis time. So this system will work for any context and sentence.

Hybrid Methods:

In this system both unit selection and HMM based approach is used. The statistical model guides the selection of units and then instead of statistical model driving a vocoder, its regression tree is used to get the cost functions.



What's in here ?

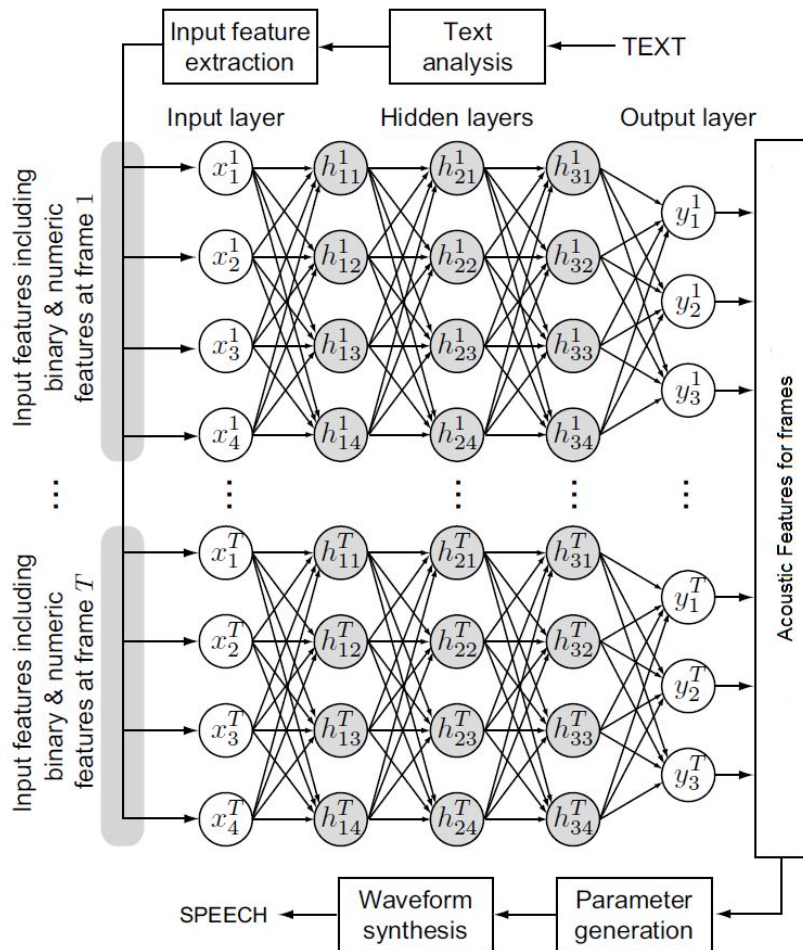
With lot of improvements in computing resources, deep learning techniques are being tried in every domain. With the advantage that it can model high-dimensional, highly correlated features efficiently.

So, All the available text to speech engines which are open-source uses HMM and Hybrid Methods. So Here I tried to Create a Base-model of DNN based Speech Synthesizer.



DNN Based System Architecture

The system starts with the Text Analysis Block which utilizes natural language processing for Part of Speech Tagging. This is followed by a Feature extraction block which analyses the context of the input text and generates features. On giving these features as input to the trained DNN model, it will give output features which can be then passed to vocoder for the actual waveform synthesis.



A DNN, which is a neural network with multiple hidden layers, is a typical implementation of a deep architecture. We can have a deep architecture by adding multiple hidden layers to a neural network.

DNN based Speech Synthesis

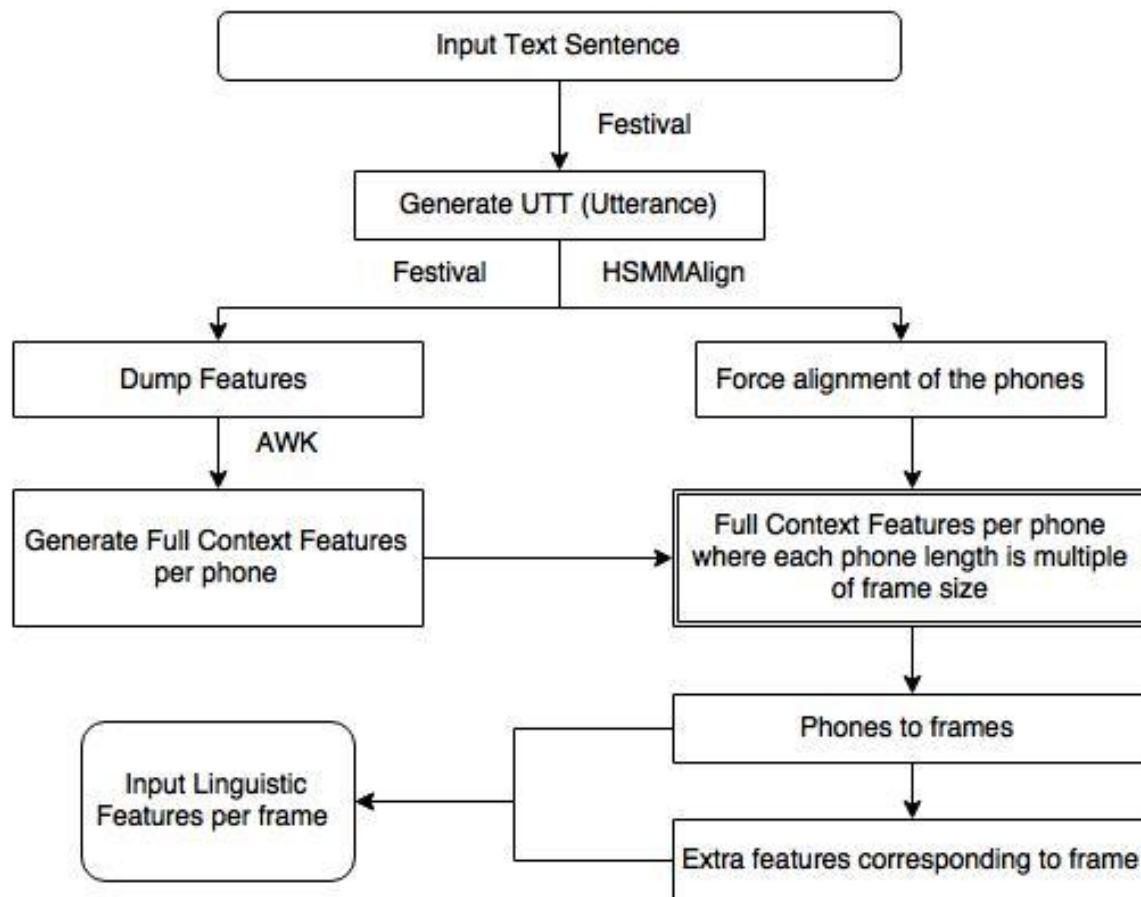
Input Feature Extraction :

A given text to be synthesized is first converted to a sequence of input features vector $\{x_{t,n}\}$ where $x_{t,n}$ denotes the n -th input feature at frame t . The input features include binary and numeric answers to questions about linguistic contexts.

Input Features

- the phoneme identity before the previous phoneme ?
- the previous phoneme identity ?
- the current phoneme identity ?
- position of the current phoneme identity in the current syllable (forward) ?
- position of the current phoneme identity in the current syllable (backward) ?
- the number of syllables in this utterance ?





Features per phone :

[start_of_phone]

[end_of_phone]

[feature_string]

String Length = 300

Features per frame

Total of 9 features were added to the pre-computed vector of phone level features.

- fraction through state (forwards)
- fraction through state (backwards)
- length of state in frames
- state index (counting forwards)
- state index (counting backwards)
- length of phone in frames
- fraction of the phone made up by current state
- fraction through phone (forwards)
- fraction through phone (backwards)

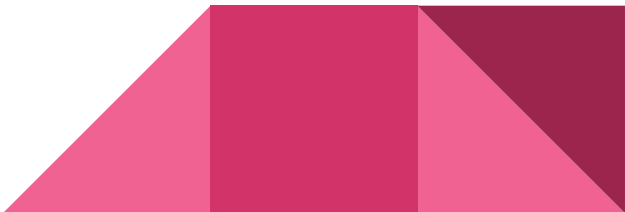
Feature Vector Length when converted to number array = $416 + 9 = 425$.



Network Architecture : Previous Model

Preprocessing : All Input feature vectors were normalized to have zero mean and Unit-variance. All output feature vectors were normalized to be within 0-0.99. A

Neural Network Framework was designed with

1. Input : 304 Nodes
 2. Hidden Layer 1 : 512 Nodes
 3. Hidden Layer 2 : 512 Nodes
 4. Hidden Layer 3 : 512 Nodes
 5. Hidden Layer 4 : 512 Nodes
 6. Output : 123 Nodes
- 

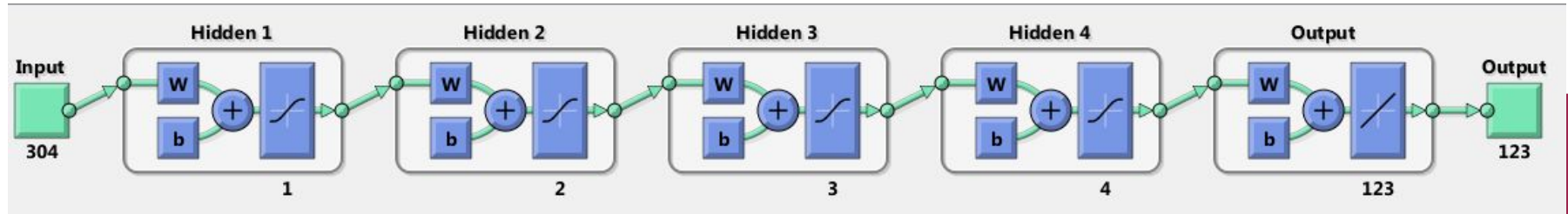
Network Architecture

In all the trainings

No. of Epochs/Iteration = 1000

Learning Rate = 0.01

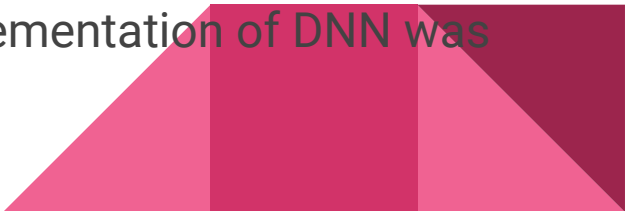
Activation Function = sigm



Network Architecture: Current Model

1. Input : 425 Nodes
2. Hidden Layers : Depending on the architecture
3. Hidden Layers Activation : TANH
4. Output Layer Activation : Linear
5. Output : Depending on the output features

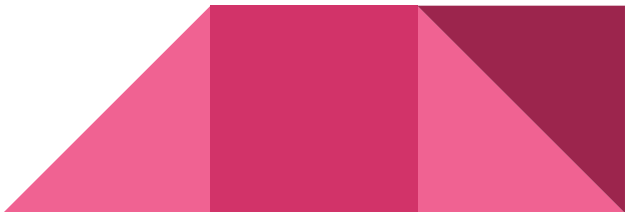
3 different systems were compared here, where the implementation of DNN was done on Theano.



Output Features

40 -> Mel-frequency Cepstral Coefficients

The main point to understand about speech is that the sounds generated by a human are filtered by the shape of the vocal tract including tongue, teeth etc. This shape determines what sound comes out. If we can determine the shape accurately, this should give us an accurate representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and the job of MFCCs is to accurately represent this envelope.



Output Features

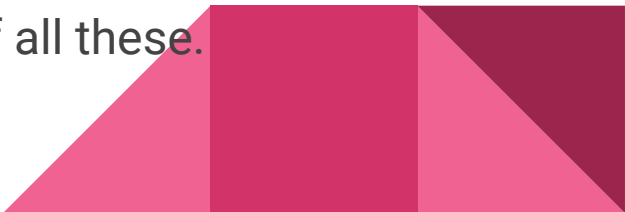
1 -> Fundamental Frequency

The fundamental frequency, often referred to as simply as the fundamental, is defined as the lowest frequency of a periodic waveform. In music, the fundamental is the musical pitch of a note that is perceived as the lowest partial present.

Deltas and Double Deltas -> differential and acceleration coefficients.

Total Output Features:

So there are 40 mfcc and 1 f0, Deltas and Delta-Deltas of all these.
That is $(40+1) + (40+1) + (40+1) = 123$.



Proposed Output features

FFT : - The Fourier transform decomposes a function of time (a signal) into the frequencies that make it up, in a way similar to how a musical chord can be expressed as the amplitude (or loudness) of its constituent notes. The Fourier transform of a function of time itself is a complex-valued function of frequency, whose absolute value represents the amount of that frequency present in the original function, and whose complex argument is the phase offset of the basic sinusoid in that frequency. The Fourier transform is called the frequency domain representation of the original signal.



Proposed Output features

Samples

These are the direct time domain samples of the input sound. When 5ms frames are considered and the sampling frequency is 16 KHz, we have $16 \times 5 = 80$ Samples



Vocoder and Speech Synthesis

Spectrogram :

A spectrogram is a visual representation of the spectrum of frequencies in a sound or other signal as they vary with time or some other variable.

Here mfcc features were used to generate the Spectrogram and then using Spectrogram and fundamental frequency, the waveform was synthesized using STRAIGHT.



Experiments

Database Used

CMU_ARTIC_SLT

US English female (slt) speaker
Carnegie Mellon University

No. of Utterances = 1132

Sampling Frequency for recorded
Waveforms = 16000 Hz

Recipe 1 (Previous)

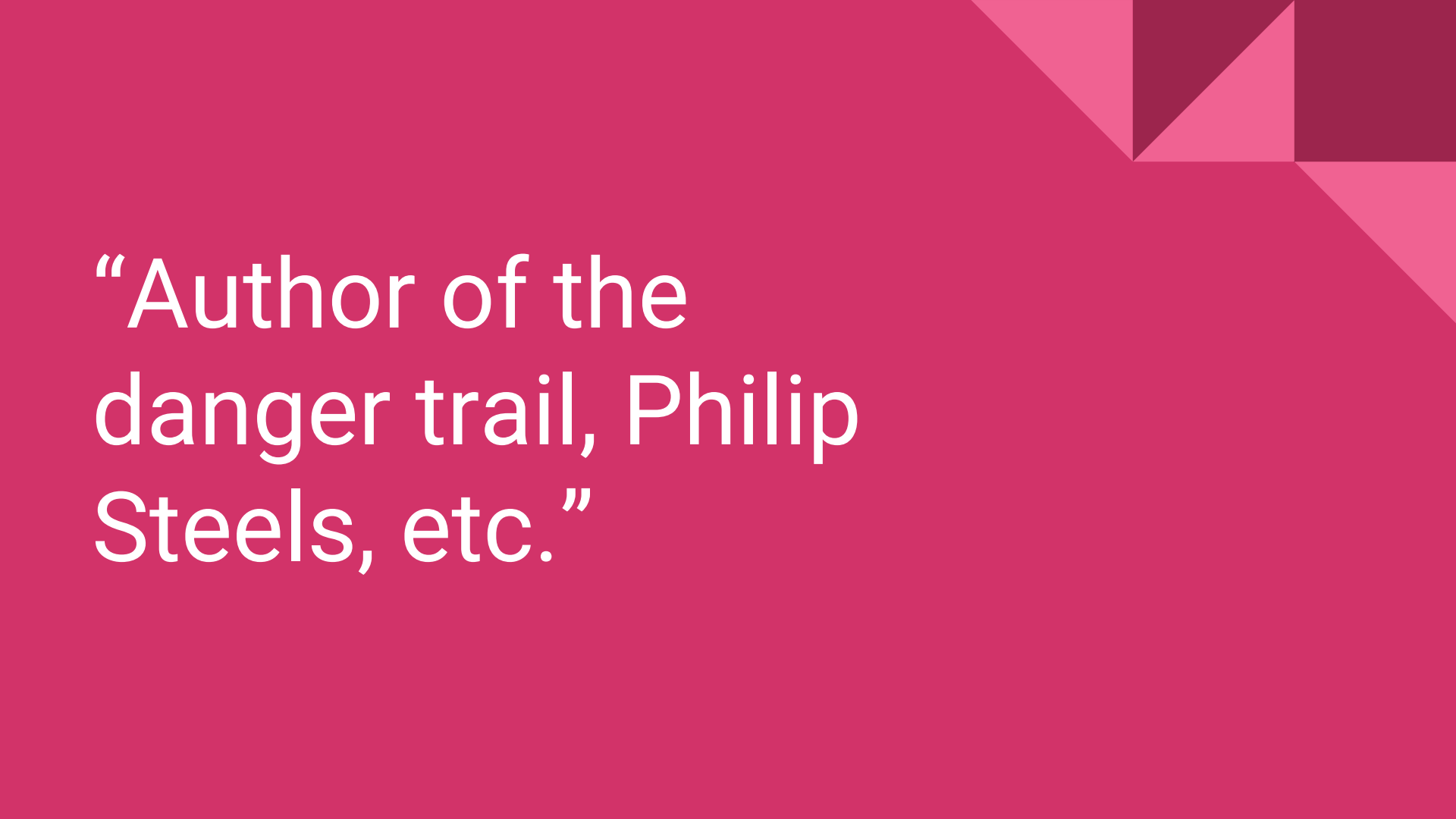
Training was done with different number of utterances.
Below given are the results with 1100 utterances.

Training Times

On 300 Utterances = 76.68 Hrs

On 1100 Utterances = 219.45 Hrs

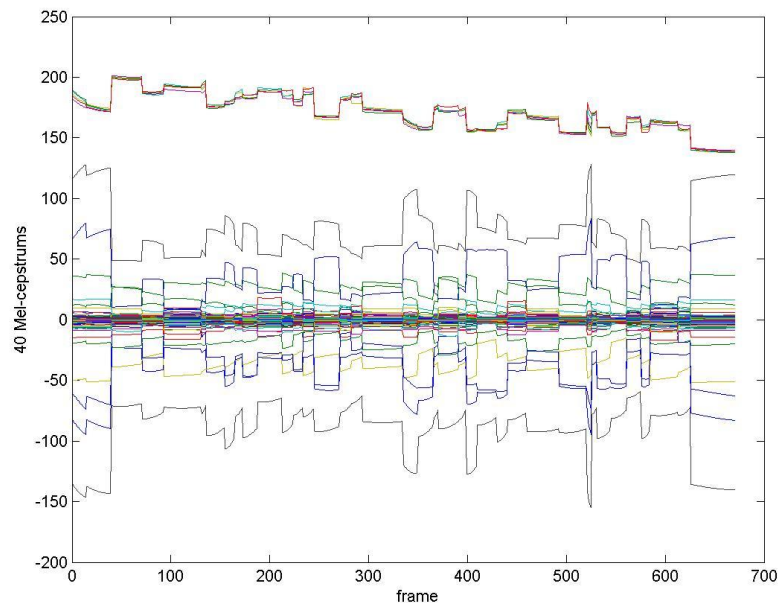
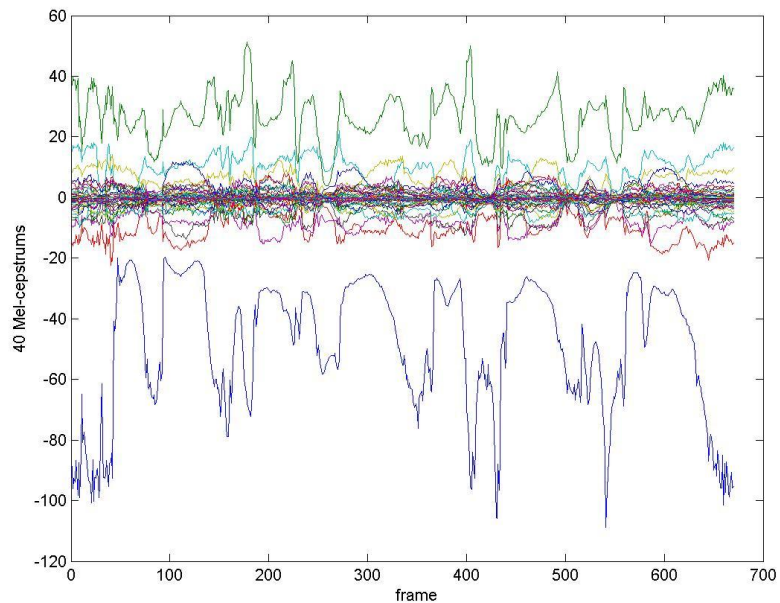


The background is a solid pink color. In the top right corner, there is a decorative pattern of overlapping triangles in various shades of pink and magenta, creating a geometric, abstract design.

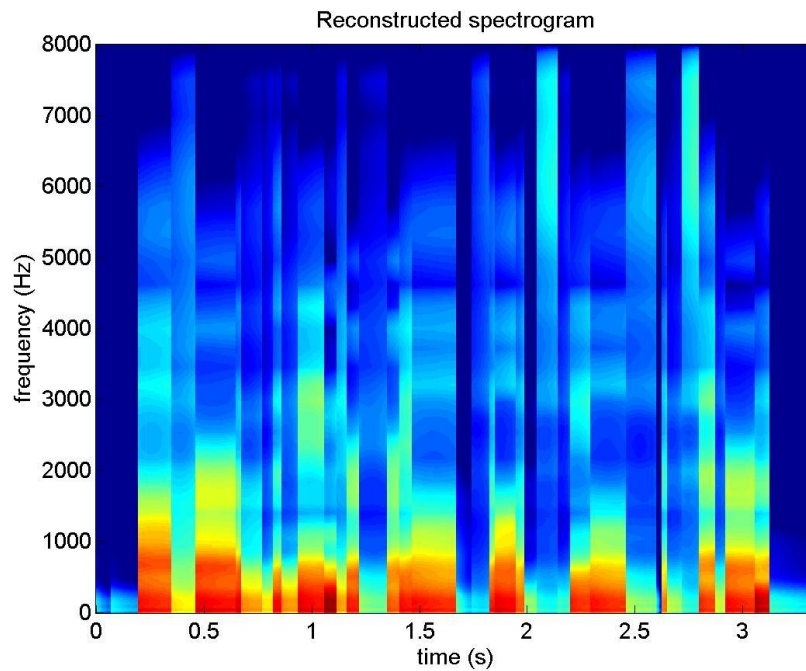
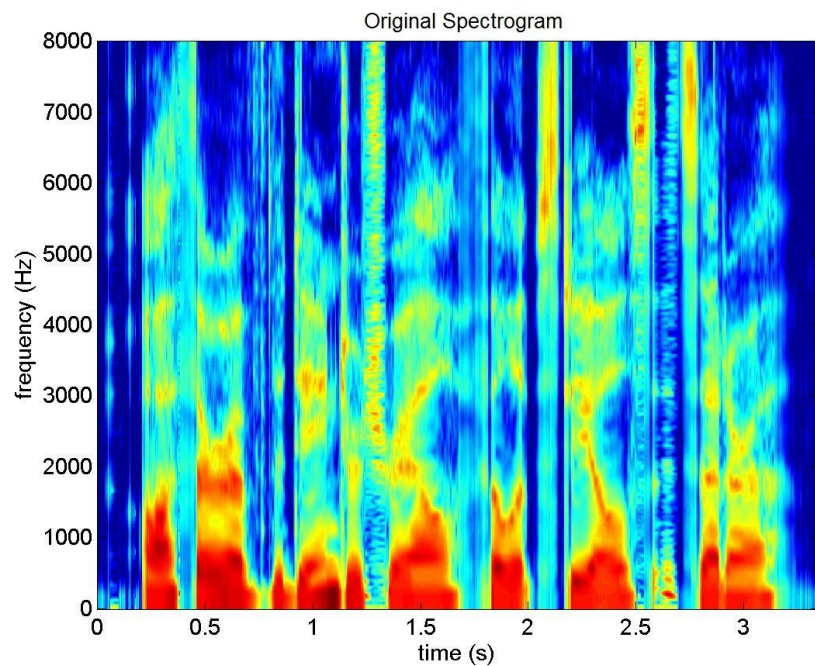
“Author of the
danger trail, Philip
Steels, etc.”

Results

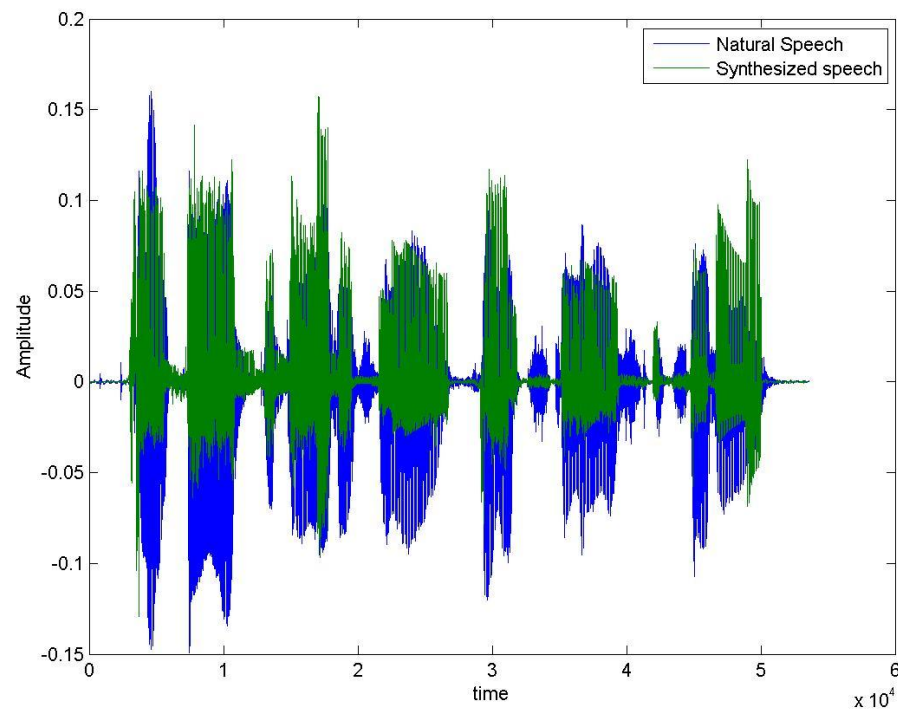
Training with 1100 Utterances



Spectrogram



Waveform



Recipe 2 : DNN (MFCC)

- **Input Features**

No. Input Features = $416 + 9 = 425$

- **The DNN Model**

Hidden Layer Size : [1024,1024,1024,1024,1024,1024]

Hidden Layer Activation : all TANH

Output Layer Activation : Linear

Learning Rate : 0.002 training epochs : 25



Recipe 2 : DNN (MFCC)

- **Output Features**

MFCC : 60

Band Aperiodicities : 1

Fundamental Frequency : 1

Voiced/Unvoiced : 1

Total Features = $3 \times (1 + 1 + 60) + 1 = 187$

- **Data Distribution**

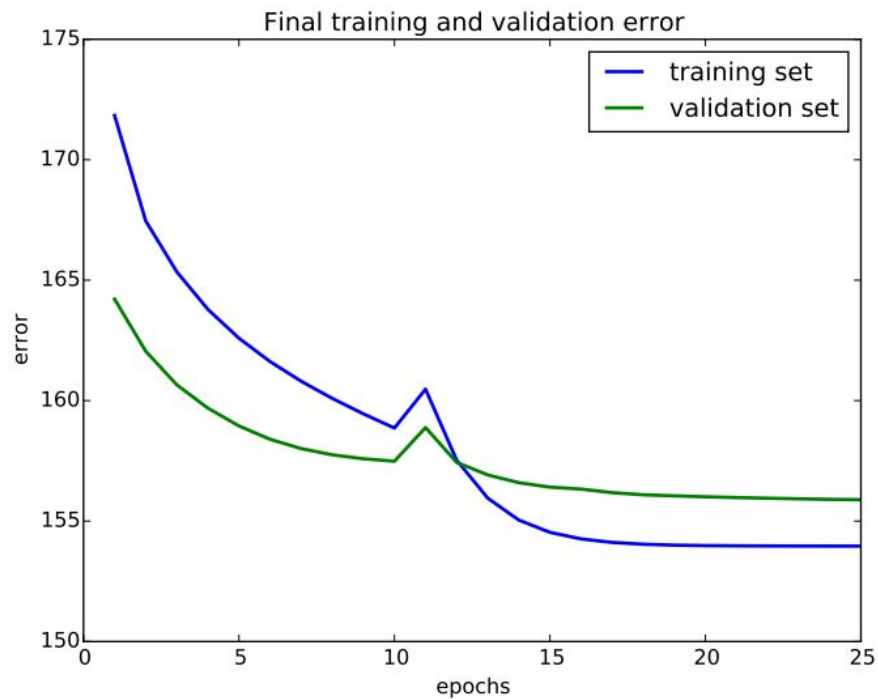
Training Files : 1000 Validation Files : 66 Testing Files : 66

- **Training Time**

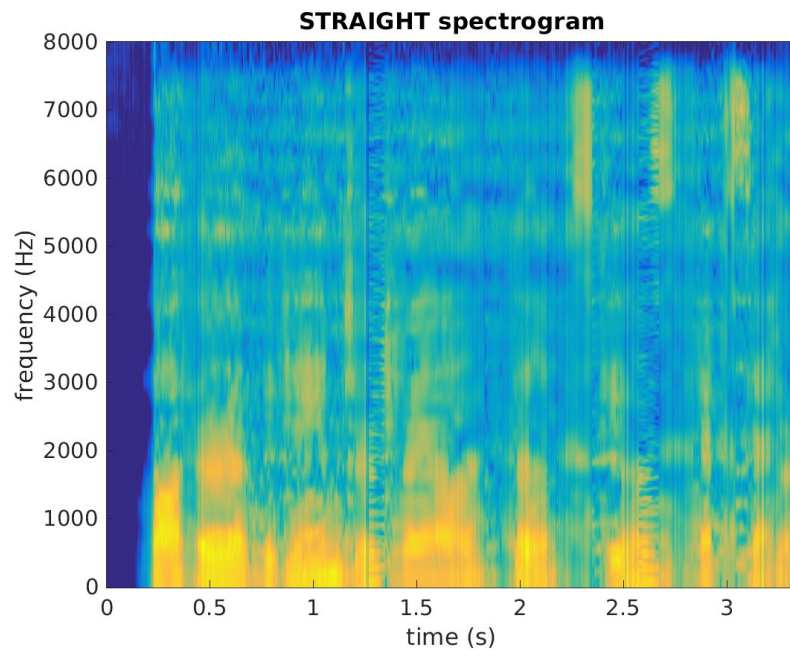
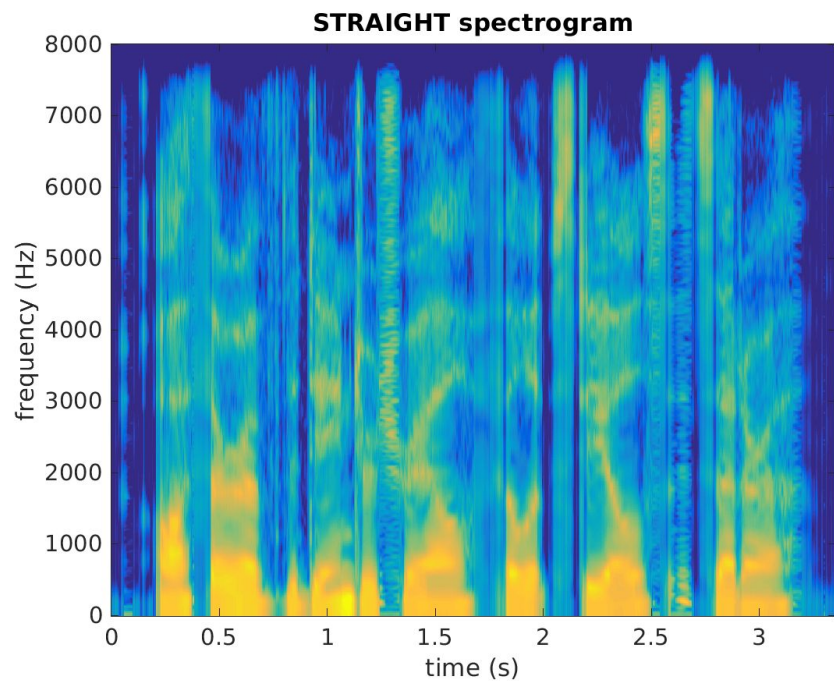
6.3 Hrs



Training Error Plot



Spectrogram



Recipe 2 : DNN (samples)

- **Input Features**

No. Input Features = $416 + 9 = 425$

- **The DNN Model**

Hidden Layer Size : [1024,1024,1024,1024,1024,1024]

Hidden Layer Activation : all TANH

Output Layer Activation : Linear

Learning Rate : 0.002

training epochs : 25



Recipe 2 : DNN (Samples)

- **Output Features**

Band Aperiodicities : 1

Fundamental Frequency : 1

Voiced/Unvoiced : 1

Samples per frame : $16 \times 5 = 80$

Total Features = $3 \times (1+1) + 1 + 80 = 87$

- **Data Distribution**

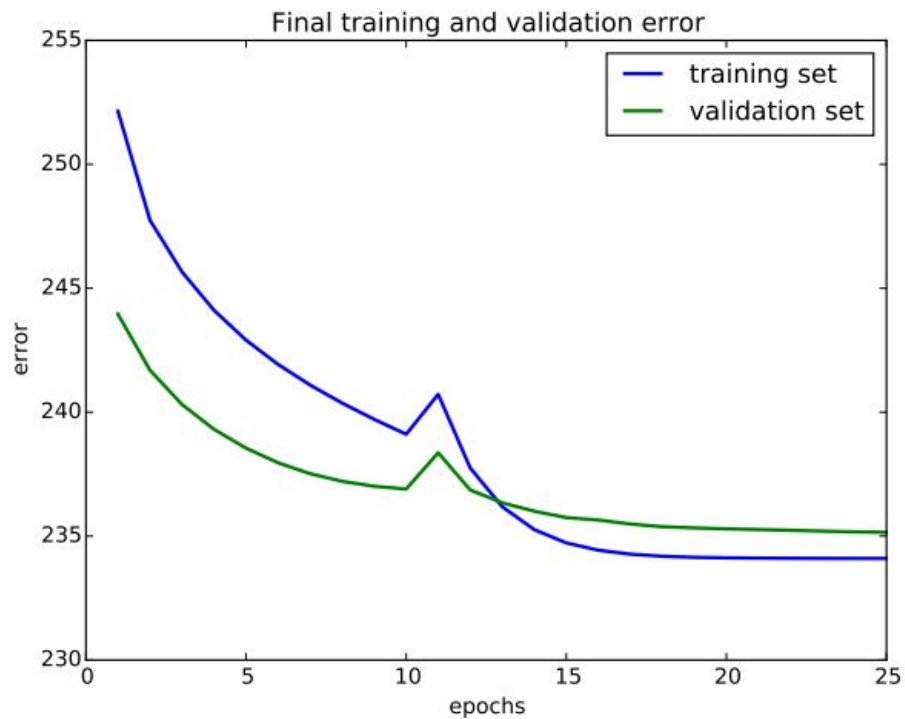
Training Files : 1000 Validation Files : 66 Testing Files : 66

- **Training Time**

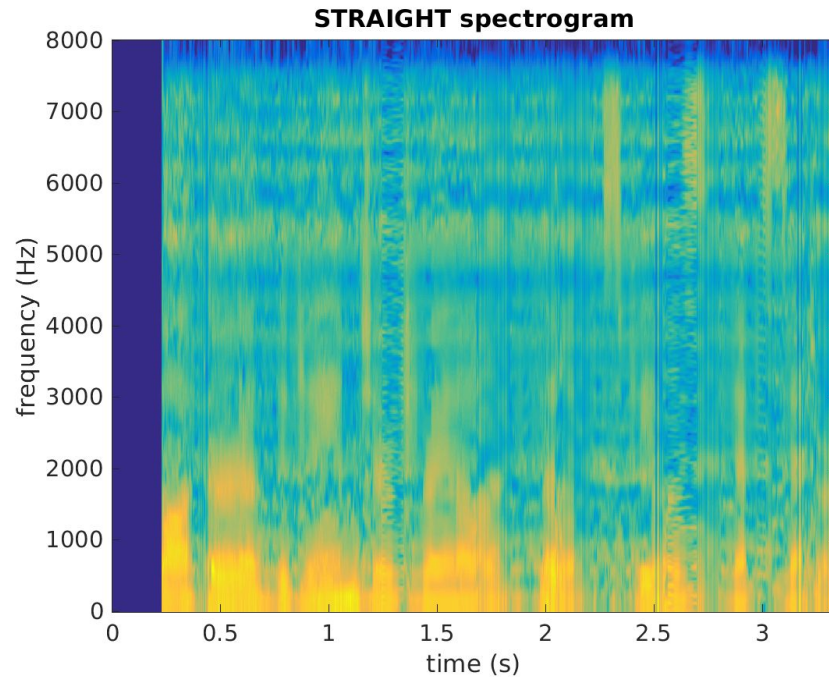
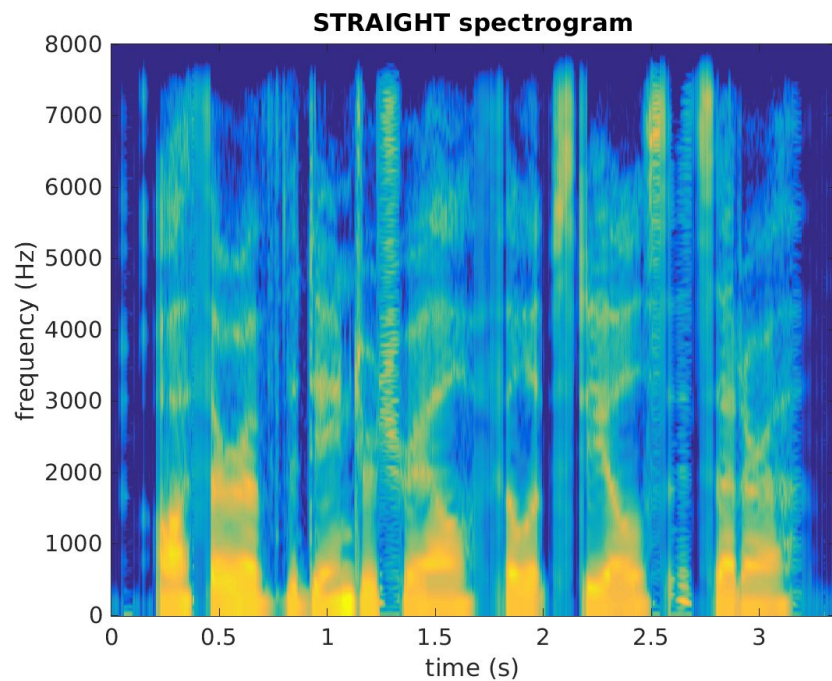
6.21 Hrs



Training Error Plot



Spectrogram



Recipe 2 : DNN (FFT)

- **Input Features**

No. Input Features = $416 + 9 = 425$

- **The DNN Model**

Hidden Layer Size : [1024,1024,1024,1024,1024,1024]

Hidden Layer Activation : all TANH

Output Layer Activation : Linear

Learning Rate : 0.002 training epochs : 25



Recipe 2 : DNN (FFT)

- **Output Features**

FFT Features : 512 (Real Part and Imaginary Part concatenated)

Band Aperiodicities : 1

Fundamental Frequency : 1

Voiced/Unvoiced : 1

Total Features = $3*(1+1)+1+512 = 519$

- **Data Distribution**

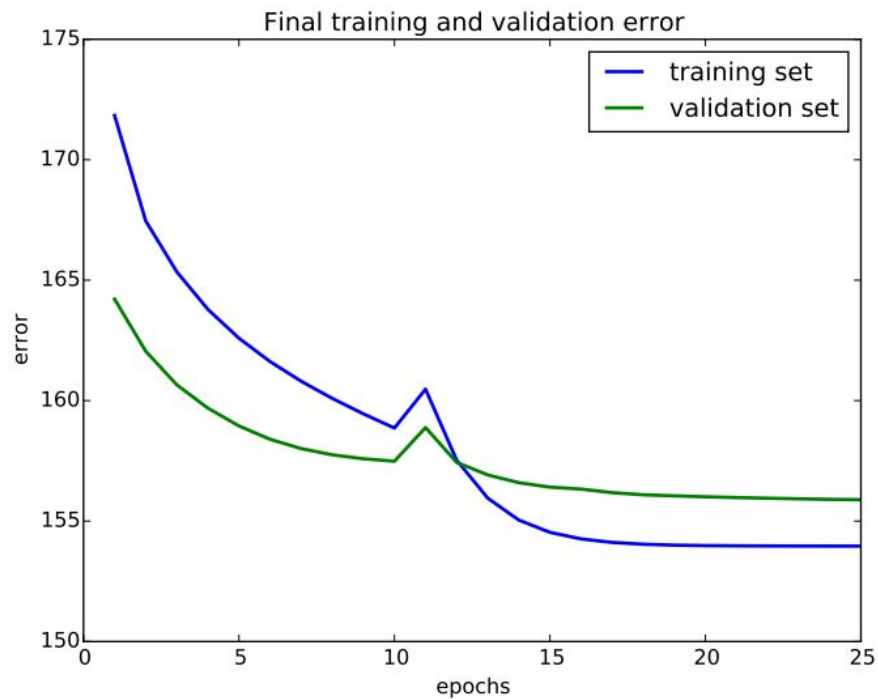
Training Files : 1000 Validation Files : 66 Testing Files : 66

- **Training Time**

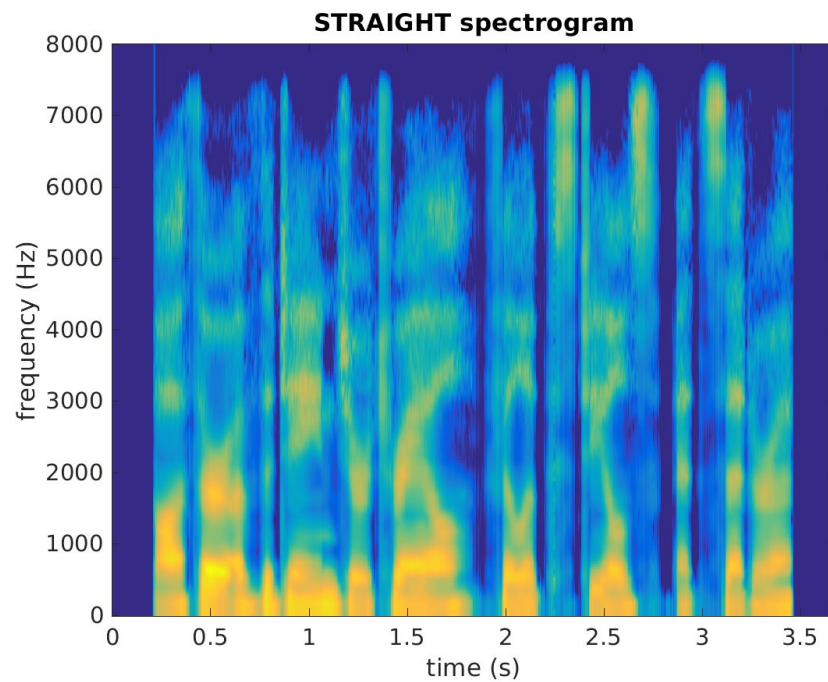
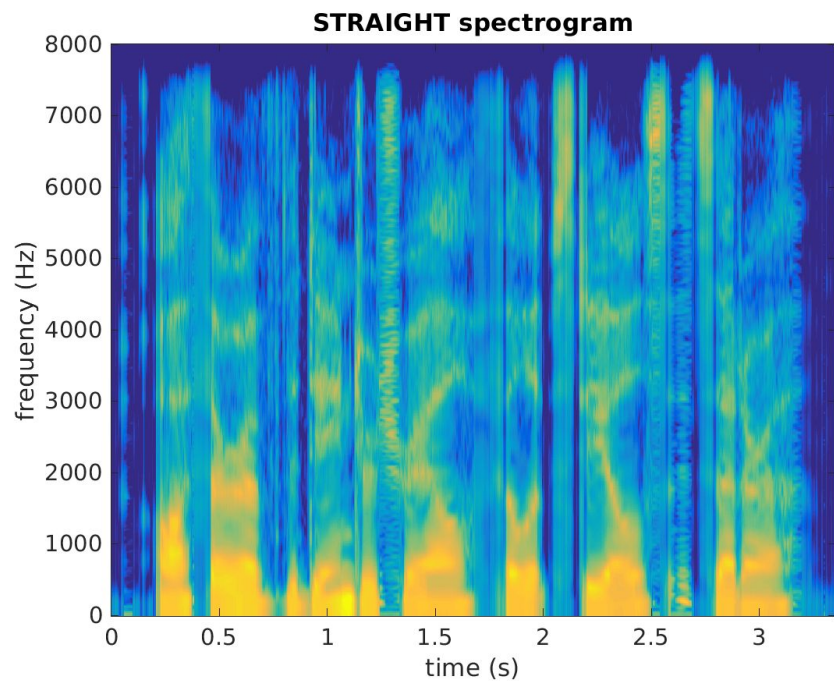
8.46 Hrs



Training Error Plot



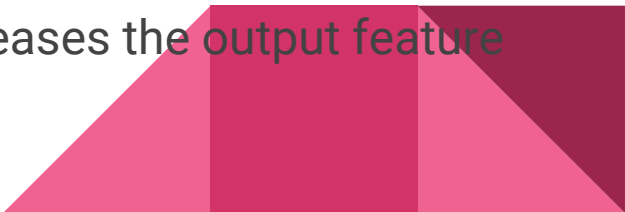
Spectrogram



Objective Evaluations

Mel-Cepstral Distortion

A MCD method of evaluation represents a distance measure calculated between mel-cepstral coefficients of original signal and synthesized signal. Below figure 5.11 shows the comparison between Mel-Cepstral Distortion values for 3 different systems.

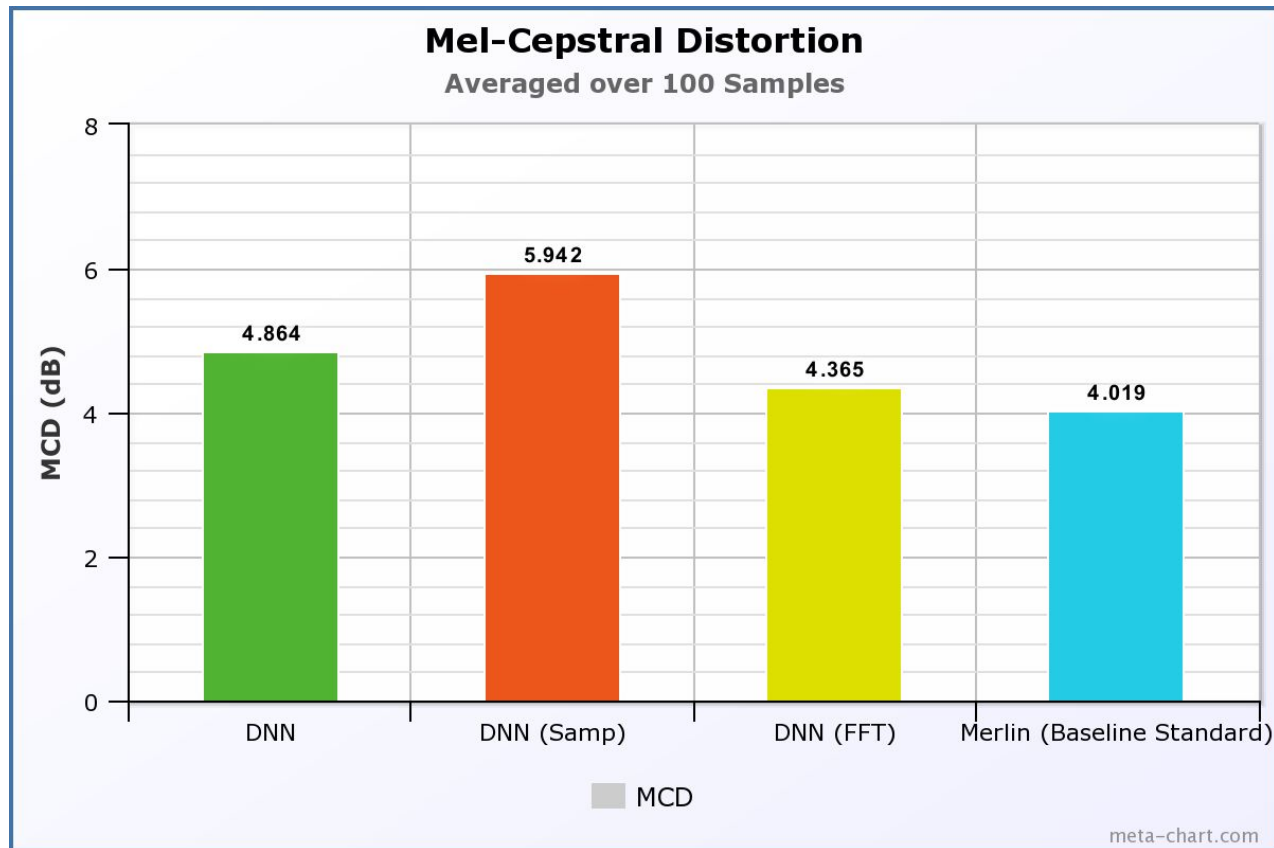
- It was seen that the system with FFT as output features performed a bit better than a system with MFCC features as output, this accounts for the FFT features that were used. But using FFT features increases the output feature vector dimension.
- 

Objective Evaluations

- The System with FFT features was also trained only on the network with 512 nodes at the hidden layer, increasing these nodes increases the training time by a big factor.
- The system with Samples at the output did not perform well, this is mainly because just samples dose not carry any time data, when passed frame by frame. This may require using RNN, on which performance could get better. This will the target for further development



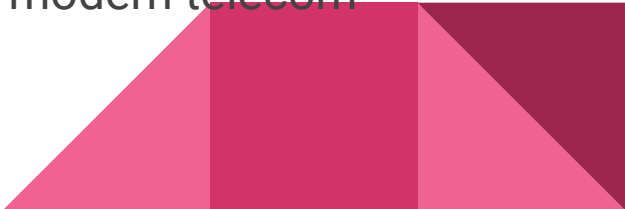
MCD



Objective Evaluations

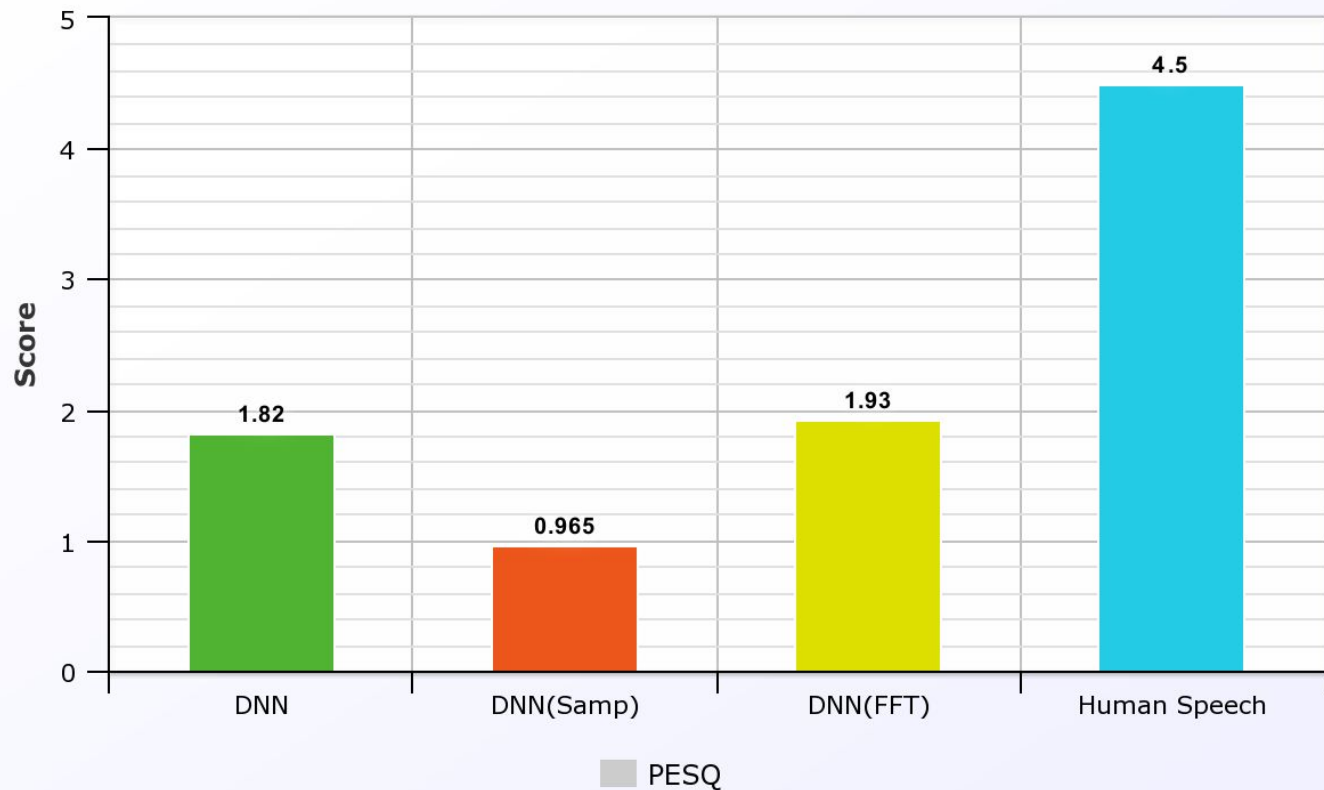
Perceptual Evaluation of Speech Quality

PESQ, Perceptual Evaluation of Speech Quality, is a family of standards comprising a test methodology for automated assessment of the speech quality as experienced by a user of a telephony system. PESQ was particularly developed to model subjective tests commonly used in telecommunications (e.g. ITU-T P.800) to assess the voice quality by human beings. Consequently, PESQ employs true voice samples as test signals. In order to characterize the listening quality as perceived by users, it is of paramount importance to load modern telecom equipment with speechlike signals.



PESQ


Perceptual Evaluation of Speech Quality



Conclusions

Among the three type of output features tested in this project the system with FFT features performed a bit better than others. This is mainly because we have all the spectral data available with FFT and unlike MFCC we consider all the values here. But this increase the no. of output features and affect the training time.

Going Forward

1. For the FFT feature based system, try it out with DNN architechure of Higher nodes in the hidden layers, This will increase the training time but can also give higher performance.
 2. For the system with Samples as features DDN system without memory are not suitable as seen from the current results. RNN implementation will be tried out for this.
 3. Build a complete standalone system with full documentation, so that this can be used as the base-model for any further developments.
- 

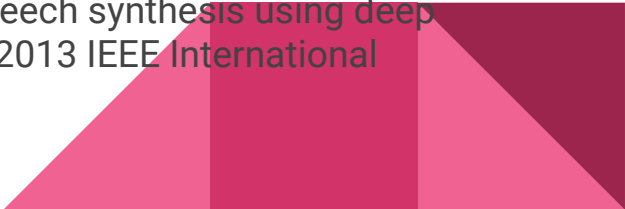
Going Forward

4. Observing the output at intermediate layers of the neural network and try to make decisions about optimal no. of layers and activation functions.
5. Analyze the performance for different type of neural networks (DNN,RNN,CNN).
6. Using Bottleneck features at the output.



References

- [1] Yoshua Bengio. “Learning deep architectures for AI”. In: Foundations and trends
- [2] Wu Chou and Wolfgang Reichl. “Decision tree state tying based on penalized Bayesian information criterion”. In: Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on. Vol. 1. IEEE. 1999, pp. 345–348.
- [3] Steven B Davis and Paul Mermelstein. “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences”. In: Acoustics, Speech and Signal Processing, IEEE Transactions on 28.4 (1980), pp. 357–366.
- [4] Saher Esmeir and Shaul Markovitch. “Anytime learning of decision trees”. In: The Journal of Machine Learning Research 8 (2007), pp. 891–933.
- [5] CH Lee et al. “Acoustic modeling for large vocabulary speech recognition”. In: Computer Speech & Language 4.2 (1990), pp. 127–165.
- [6] R. B. Palm. Prediction as a candidate for learning deep hierarchical models of data. 2012.

- [7] Koichi Shinoda and Takao Watanabe. “Acoustic modeling based on the MDL criterion for speech recognition”. In: Proc. EuroSpeech-97. 1.1997, pp. 99–102.
- [8] Takahiro Shinozaki. “HMM state clustering based on efficient cross-validation”. In: Acoustics, Speech and Signal Processing, 2006. ICASSP2006 Proceedings. 2006 IEEE International Conference on. Vol. 1. IEEE 2006, pp. I–I.
- [11] Zhizheng Wu et al. “Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis”. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). 2015. URL: [http : / / www . zhizheng . org / papers / icassp2015_dnn_tts.pdf](http://www.zhizheng.org/papers/icassp2015_dnn_tts.pdf).
- [12] Heiga Ze, Alan Senior, and Martin Schuster. “Statistical parametric speech synthesis using deep neural networks”. In: Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE. 2013, pp. 7962–7966.
- 



Thank You