International Journal of Applied Sciences and Smart Technologies

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

Artificial Generation of Realistic Voices

Dhruva Mahajan[1,*], Ashish Gapat[1], Lalita Moharkar[1],

Prathamesh Sawant[1], Kapil Dongardive[1]

[1]*Department of Electronics and Telecommunication Engineering, Xavier Institute of Engineering, Mahim, Mumbai, Maharashtra, India*

[*]*Corresponding Author: dhruvam17@gmail.com*

(Received 17-07-2020; Revised 27-08-2020; Accepted 15-09-2020)

Abstract

In this paper, we propose an end-to-end text-to-speech system deployment wherein a user feeds input text data which gets synthesized, variated, and altered into artificial voice at the output end. To create a text-to-speech model, that is, a model capable of generating speech with the help of trained datasets. It follows a process which organizes the entire function to present the output sequence in three parts. These three parts are Speaker Encoder, Synthesizer, and Vocoder. Subsequently, using datasets, the model accomplishes generation of voice with prior training and maintains the naturalness of speech throughout. For naturalness of speech we implement a zero-shot adaption technique. The primary capability of the model is to provide the ability of regeneration of voice, which has a variety of applications in the advancement of the domain of speech synthesis. With the help of speaker encoder, our model synthesizes user generated voice if the user wants the output trained on his/her voice which is feeded through the mic, present in GUI. Regeneration capabilities lie within the domain Voice Regeneration which generates similar voice waveforms for any text.

International Journal of Applied Sciences and Smart Technologies

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

Keywords: Speech synthesis, speaker encoder, synthesizer, Text-to-Speech, vocoder

1 Introduction

In the near future, everything around us will be voice operated. With the growing trends of Alexa and Google home, advancements are being made to create an environment of Artificial Intelligence and its operating medium would be Voice. With the advent of signal processing, voice signals have extensive upgradation in terms of global standards and kept on challenging for better platform in all its forms of uses. Right from voice screeching out of airhorn to voice search engines in smart phones, voice applications have always been vital in any or all of their feats. Artificial generation itself is a form of voice cloning that is implemented with the help of neural nets that help with generating the Mel-spectrograms. This process intensifies on the input, where the characters present are synthesized into signal waveforms which are digitally spectograms. These spectograms are then coupled and then linked through a vocoder, which generates voice corresponding to the characters that are given as input.

The goal of the this paper is to build a TTS system which can generate natural speech for a wide variety of speakers which are absent throughout the process. Our model can run in real time by implementing the mic function. This is possible by achieving a powerful form of voice cloning. The output must be aligned in such a manner that runs on correct lines to provide a clone of the dataset that is trained within the system. The output speech must allign with the exact speaker voice picked from the dataset. This voice gets matched with the help of RNN, which cycles the implementation process wherein the dataset voice gets linked with input text. This implementation works throughout for all the speakers whether present or absent. When a speaker implements mic function, the speaker encoder does not operate.

Uncertainties do arise as we train the model with the speaker's reference speech (trained dataset):

- The output utterance is slightly composed.
- Dataset voices could be identical, depending on the dataset picked (VCTK and LibriSpeech).

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

- Voice output obtained, when implementing through the mic tends to be rough. This is because while recording in real time, it is very important to check the environment around. Mic integrated with system is highly sensitive and tends to pick smallest utterance.
- Recording a large amount of high-quality data for many speakers seems rather impractical.

The approach which we deploy, decouples speaker modeling from speech synthesis by independently training a speaker effective embedding network that traces and sequentially captures the character space of speaker characteristics and training a high-quality TTS [1], [2]. Taking findings from a 2017 research paper, Google- Tacotron, where we figure out the synthesis and voice generation process [3]. We took implementation steps from a deep learning architecture that we researched and coupled it with WaveNet [3], [4]. WaveNet is a neural network which acts as a vocoder, to convert mel-spectograms into corresponding voice signals. We now had to check the process and operation for which we used two public datasets that are Librispeech and VCTK. After the respective implementation we got our process confirmed for synthesis and started the model. To run with time and technology, we implement a neural network architecture of our research pertaining to synthesis of voice. This research model is a deep learning text-to-speech model. Text-to-Speech describes that a string of text would be converted to speech output. There are three major operations.

- Speaker Encoder- Where the embedded text feeded in is sent to a convolution bank highway network. This convolution bank is nothing but a deep learning tool which breaks our text in separate characters. Breaking the string of characters allows the network to work on each character modulation, making it more in tune with voice rather sounding modulated. i.e. For example- GOOD MORNING in the convolution bank will be presented as G-O-O-D M-O-R-N-I-N-G.
- Synthesis- We implement trained datasets in our paper. These trained datasets arefeeded and loaded before operating the model. During the synthesis our character embedding which is disintegrated character by character is sent to attention, where the character is coupled with trained dataset voice signal.

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

- Vocoder- This is the final part of our model where the signals which are bindedas spectrogram is converted into voice. This operation uses WaveNet neural network [5]. (WaveNet was developed in deepmind labs, which is a AI research wing of google.) WaveNet as a vocoder acts as a binder, where all end results (melspectrograms obtained) are combined. These results are the voice signal modulation that take place along with each character given as input. Explaining mathematically, in theory, our input text gets converted into an algebraic equation (which consists are input characters), this equation is then simplified by taking K constant. (K constant here is the dataset selected). This simplified equation is then collected and generated into series function equation (This is the desired output). With spectrograms for these character strings, this waveform passes through the WaveNet which perform natural language processing, giving voice as the end result.

2 Methodology

Text-to-Speech implementation for a speaker model to pick up character embeddings, the model needs to be well disintegrated in three subsets as clarified. These subsets work in a sequential manner providing speech in a uniform and desired manner. Putting light on each model now will magnify each role clearly with technical overviews and details. See Figure 1 for the block diagram.



Figure 1. Block diagram

Speaker encoder illustration is shown in Figure 2. Speaker encoder plays an important role by estimating the embedding from various text characters that is feeded as input. While implementing Text-to-Speech model, it's up to the developer by what means it should be operated. To cut the unnecessary word-word translation by using a conventional Text-to-Speech system, where the text is sent in a string and the output obtained has passive speech, we use deep learning architecture, where the input text is

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

feeded as the prenet (pre-feeded information/local information). This prenet is then sent to 1D-CBHG. This is one-dimensional convolution bank highway network. This block is the place where the string is broken in separate character by character formation [5]. Example: GOOD G-O-O-D.

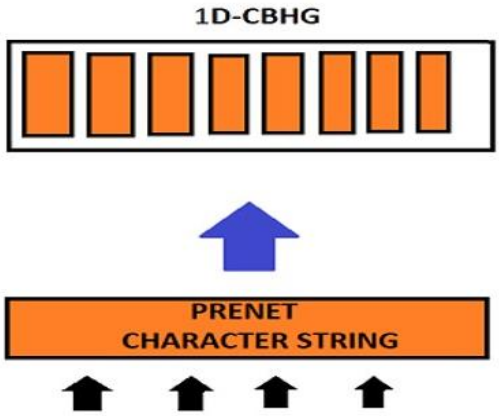


Figure 2. Speaker encoder

Synthesizer is illustrated in Figure 3. Functions of daily text-to-speech is evident in our voice search engines, smart electronics and various voice modulating devices that inherit the use of synthesizer. Text-to-speech control system employs an easy-to-use transport medium from which a user can control most text-to-speech variating functions without any prior training. Now, the synthesizer basically works in the areas of translating a plurality of discontinuous user-selected part of text in an independent form of target application into an audio output that resembles the sound of human speech. To generalize, the synthesizer is the core of the Text-to-Speech engine, where it mainly focuses on tracing and adapting the text input in a sequence and delivers audio sample of it [5].

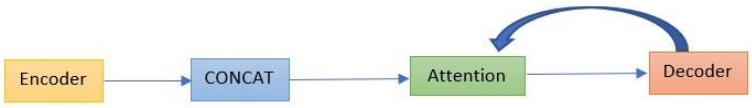


Figure 3. Synthesizer block

In the block diagram represented above, the speaker encoder gets linked to the Concat block. In this Concat block, the character broken down in the CBHG, gets

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

forwarded as a string with intervals between each character to the attention. Attention is a deep learning tool where each character is concentrated and constant is removed, and binded with the trained dataset. This trained dataset is forwarded from the Decoder, which acts as a gate for audio signal and couples it with the data in Concat. The constant removed is called 'k-constant', which is used for signal processing.

Vocoder: Simplicity in the output should be in a form such that all the speech is synthesized in the manner that the corresponding input is heard in tone that is tuned to match the tempo of the unseen speakers' voice. For this purpose, a vocoder is used where it captures the characteristic elements of the audio signal and then uses this characteristic signal to affect other audio signals. It is also dubbed as a "talking synthesizer" for its ability to fine-tune the synthesized signal in accordance to vocal frequency. See Figure 4 for Log Mel-waveforms.

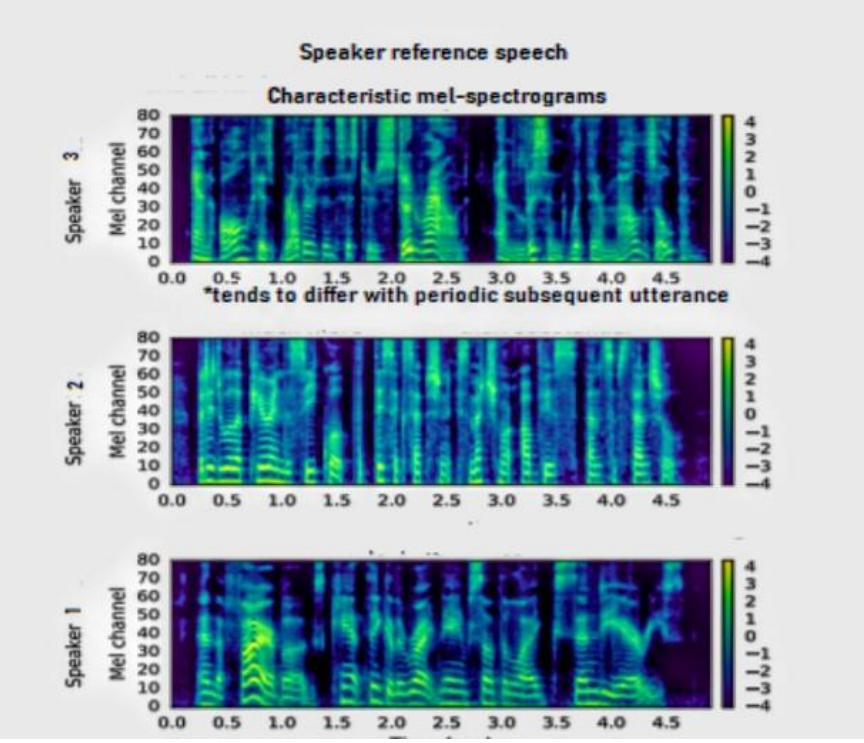


Figure 4. Log Mel-waveforms

The vocoder provides a bank of multiple bandpass filters which dissociate the input signal into narrow spectral slices. Consider, we excite channel 'k' of vocoder with the input signal $a(nT)\cos(wknT)$ for $n = 0,1,2,3,4,5, \ldots$ where wk is the centre frequency

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

of the channel in radians per second, T is the sampling interval in seconds and bandwidth of $a(nT)$ is smaller than channel bandwidth. We regard this input signal as an amplitude modulated sinusoid. The component $\cos(wknT)$ can be called the carrier wave, while $a(nT) > 0$ is the amplitude envelope.

If the phase of each channel filter is linear in frequency within the passband (or at least across the width of the spectrum) and if each channel filter has a flat amplitude response in its passband, then the filter output will be, by the analysis of the previous section. $Yk(n) \sim a[nT - D(wk)]\cos(wk[nT - P(wk)])$

Creating a GUI is illustrated in Figure 5 as follows. With the reach of applications with software, the main over layer and the visible interaction is the Graphical User Interface (GUI) which allows dynamic ability to the user for its functioning [6]. For our model we intended the working on an interface to allow the user to interact with the model. Creation of GUI also implements the ease of functions laid out at disposal of the user within a fixed framework. GUI requires the interaction to be in a flow that does not hamper the user and neither causes any sort of imbalance within the process. Taking for consideration, our model is heavily based on synthesis, making it completely oriented to user interaction (input/character embedding). So, with this, the GUI should be in a manner that allows easy flow of task within the same framework. To create the GUI for our model, we implement tkinter library of python. Within the tkinter package, there are many functions that are used to make things more organized and presentable. Tkinter allows us to make various frameworks, buttons and organizes the functions systematically. Major properties to include in the GUI is illustrated in Figure 5.

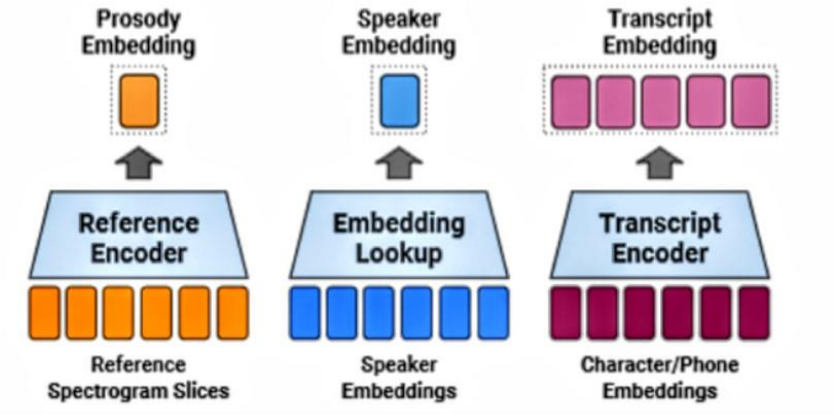


Figure 5. Main GUI components

International Journal of Applied Sciences and Smart Technologies

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

3 Implementation

Real time Voice Cloning (Using SV2TTS): The entire approach to real time voice cloning is adapted on the basis of Transfer learning from speaker. i.e. dubbed as prosody transfer (voice styling implementation). It is a speaker verification to multi-speaker text-to-speech synthesis. It essentially defines the framework for voice cloning that barely requires 4-6 seconds of reference speech. It is majorly dependent on three early works from Google: the GE2E loss (Wan et al., 2017), Tacotron (Wang et al., 2017) and WaveNet (van den Oord et al., 2016). This proposed model is three-stage pipeline, as listed above in order. The google cloud services, google search engine, or google assistant make use of these same models.

Model Architecture of Speaker encoder, Synthesizer and Vocoder are shown in Figure 6.

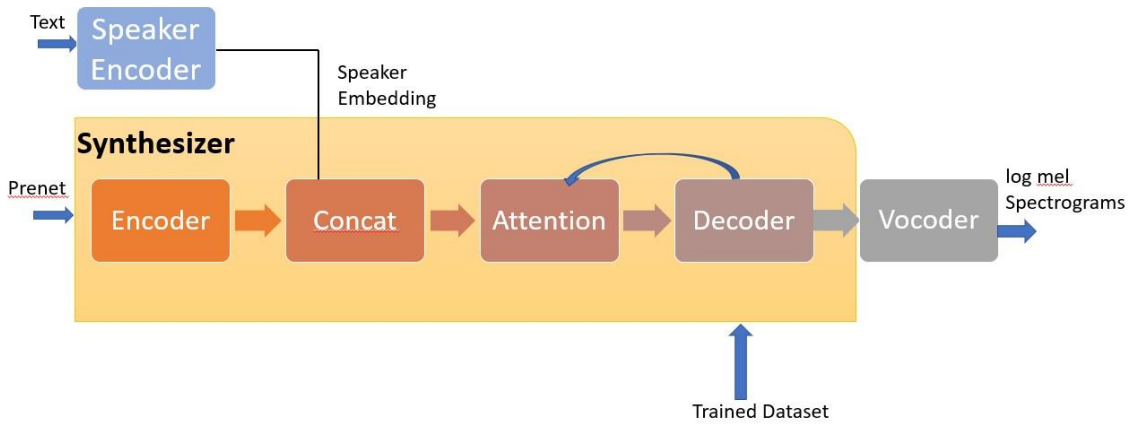


Figure 6. Three-stage pipeline for Text-to-Speech

Model Architecture of Speaker Encoder: It is a three-layer LSTM with 768 hidden nodes which is followed up by a projection layer comprising of 256 units. Although there is no reference in any of references present, as to what the projection layer defines. Hence, we round up to consider the overall function of the projection layer as a 256-output fully connected layer (per LTSM) which is repeatedly applied to each and every output of the LTSM. The inputs to this model are 40-channels log-mel spectrograms

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

with 25ms window width and a 10ms step. The desired output is the L2 normalized hidden state of the last layer, which works as a vector of 256 elements.

Model Architecture of Synthesizer: In the synthesizer implementation, the target Mel spectrograms present for the synthesizer provide more features than those used for speaker encoder which are computed from a 50ms window with a 12.5ms step and have 80 channels. We use a python implementation of LogMMSE algorithm, which is used for filtering the audio speech by erasing the noise in the early frameworks. Consequently, we train the synthesizer for 150k steps, comprising of a batch size of 144. The outputs per second is set to 2 for the decoder. While implementing, the architecture tends to provide speech synthesis to attain identical cloning to the unseen speaker. During this process there are losses that are accounted in the verge predicted and ground truth mel spectrograms. This is the L2 loss function. While training, the model is set to ground truth aligned (GTA). The reason for this is that if we do not set the model to GTA, then the synthesizer would produce different variations of the same utterance (text or embedding). Implementation of neural nets are shown in Figure 7.

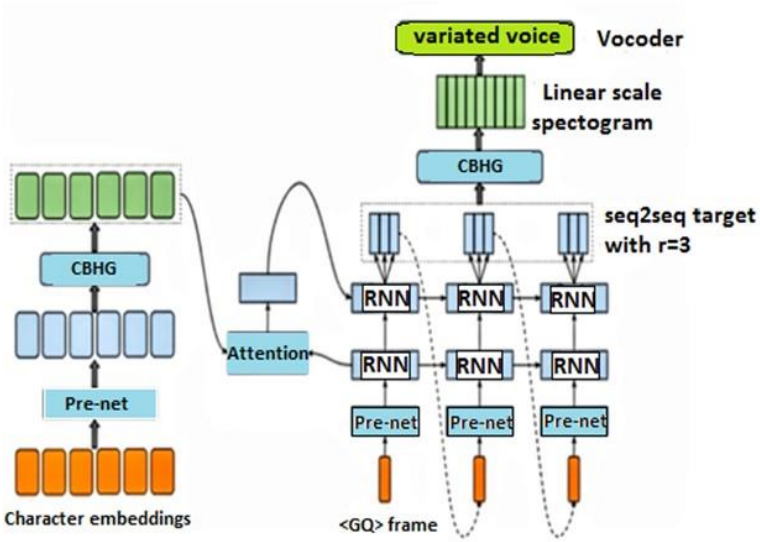


Figure 7. Implementation of neural nets

Model Architecture of vocoder The vocoder implemented in the model is WaveNet [1, 4]. WaveNet produces naturalness in TTS. This is the primary reason it is fairly used in Tacotron and SV2TTS. However, the efficiency of this neural net is too

Volume 3, Issue 1, pages 11–26
p-ISSN 2655-8564, e-ISSN 2685-9432

good coming at the cost of the speed. It is very slow and the slowest deep learning architecture at interference time. Even if this is a matter of stress on the implementation part, there are improvements that can be initiated on it. Googles own vocoder works at the rate of output of 8000 samples per second, which is by far not bad for a neural net. The model implemented is an open-source Pytorch implementation that is based on an RNN model deployed by Github user fatchord.

Zero-shot speaker adaption the speech characteristics to be synthesized are picked up from the audio signal. Zero-shot adaption is the ease with which the model gets linked with the training data from the unseen speaker which is not present during the training. It just requires time of 4-6 seconds for the model to generate new speech by synthesizing the speaker characteristics. Interference can cause the speaker information to get synthesized without knowledge of the input fed through. However, in our model, interference occurs with arbitrary untranscribed speech audio which does need the text to match with the synthesizer, thereby making the implementation hassle free and comparatively quick.

Dataset Used Two datasets which are public by nature for synthesis and speech training for vocoder network is used for implementation. VCTK comprises of 43 hours of clean speech from 109 speakers, among which most have British accents. We down sampled the audio to 25 kHz, trimmed leading and trailing silence (reducing the median duration from 3.3 seconds to 1.8 seconds). It's been split into three subsets: train, validation which has the same speakers as the train set and test which has 11 speakers held out from the train and validation sets. LibriSpeech comprises of the union of the two "clean" training sets, consisting 436 hours of speech from 1,172 speakers, sampled at 16 kHz. The speech is US English majorly, however since it is sourced from audio books, the vocals and style of speech can differ significantly between utterances from the same speaker. We reassembled the data into shorter utterances by force aligning the audio to the transcript using an automatic speech recognition (ASR) model and breaking segments on silence, reducing the median duration from 14 to 5 seconds.

Naturalness of Speech Clearer the person's voice is, crisper is the audibility factor that follows. This might tend to differ in cases of speech synthesis, where natural speech gets encoded and decoded with help of speech synthesizers and in our case deep

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

learning networks too. So, to arrange it cordially, there is prenet data, also termed as local information that gets coupled with character embeddings from the input. There is interference that plays a major role and thereby noise gets added on while processing through the highway networks. With subsequent processing and ability to trace character by character (using Fourier transform) we get the log-mel spectrogram through vocoder. After attaining the desired output, the major question is how much of the part is natural speech. As we implement two public datasets, we tried comparing the VCTK and LibriSpeech datasets to check the speech synthesis through synthesizers and vocoder. We tried comparing by using 11 unseen and seen speakers for VCTK and 10 unseen and seen speakers for LibriSpeech. Each of the comparison was conducted independently [2]. Comparison of time taken for synthesis is listed in Table 1.

Table 1. Comparison of time taken for synthesis

System	Speaker Information	VCTK	LibriSpeech
Ground truth	Same Speaker	4.67 ± 0.04	4.33 ± 0.08
Ground truth	Same Gender	2.25 ± 0.07	1.83 ± 0.07
Ground truth	Different Gender	1.15 ± 0.04B	1.04 ± 0.03
Embedding Table	Seen	4.17 ± 0.06	3.70 ± 0.08
Proposed Model	Seen	4.22 ± 0.06	3.28 ± 0.08
Proposed Model	Unseen	3.28 ± 0.07	3.03 ± 0.09

Speaker Similarity and Verification To check the speech having cleaner detail, we check the results of the above comparison of the two datasets. This is done to check whether the desired output is identical to the input given. From the comparison table, we get to know that the values delivered by VCTK tend to be an edge above the LibriSpeech dataset. The speech from the VCTK dataset is cleaner. This can be also understood by higher ground truth baselines in the VCTK. That makes the VCTK better, but on using LibriSpeech on VCTK model, it was visible that the output was better than that of VCTK model. This means that depending on the dataset and type of groundtruth, the similarity can be accomplished. For the part of speaker verification, on LibriSpeech, the synthesized speech is at most similar to the ground truth voices. The

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

LibriSpeech synthesizer obtains similar EERs of 5-6 % using reference speakers from both datasets, whereas the one trained on VCTK performs much worse, especially on out-of-domain LibriSpeech speakers.

4 Results and Discussion

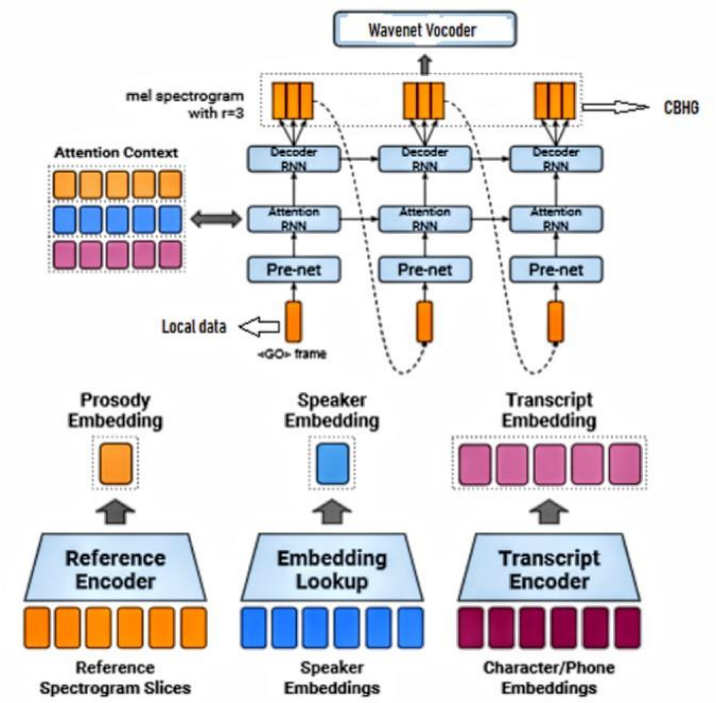


Figure 8. Proposed-method

Having all the details and implementation incorporated, we got our proposed design (see Figure 8) that would be implemented by us. We had to take in consideration every part as sequential support that will provide proportional synthesis and deliver the desired output. We had two procedures followed to obtain results.

Method 1: In the first method we used the train datasets for speech synthesis, which were LibriSpeech and VCTK respectively. The output obtained was clear and audible with the help of headphones. In open environment, the voice felt a little light, and required a speaker with an equalizer.

Method 2: This method is an advancement to our paper which could be extended further successfully with correct implementations. We tried feeding in our voice through

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

the mic, and tried cloning. The speech did get synthesized but the output obtained wasn't as clear as the previous case and was little gibberish in nature.

Audio Quality Analysis: The quality audio can be better heard by using speakers with better bass and equalizer in an open environment. There is a 2-3 seconds lag in setting the audio sample from the dataset which, again can be improved by trimming the audio samples. Within all testing periods, the model was reliable and highly efficient in terms of user interface. Further extension is possible by implementing user generated voice (refined), and miscellaneous dataset implementation of various accents and dialects.

Now, in accordance to the above model, we had to implement a GUI (see Figure 9), that can provide subsequent functioning of the above model and also should be user friendly to operate. We implemented Python tkinter package in accordance to the system model and created the model that equips and provides most of the proposed system through easy flow of framework design.

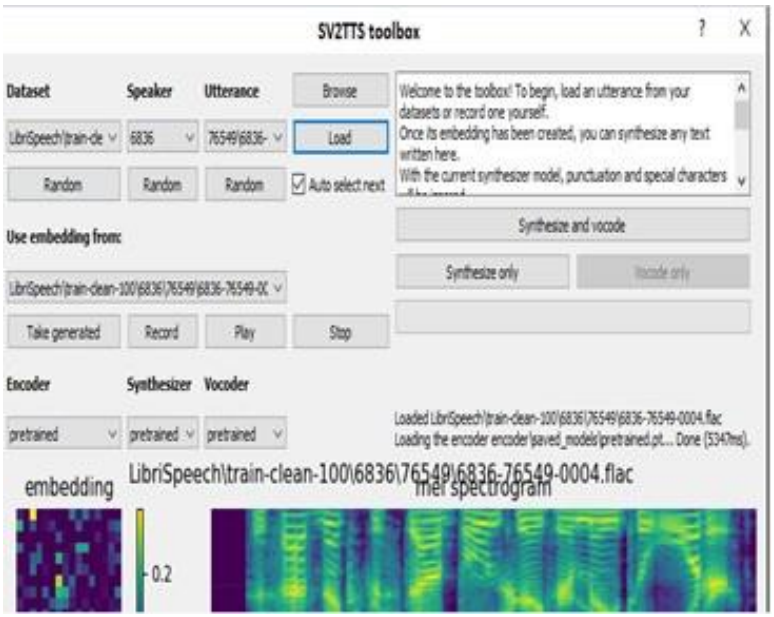


Figure 9. Output GUI

5 Conclusion

The entire concept of having voice cloning has always had some advancements tending to always grow for the better. Incorporating all the factors, we try instill the

Volume 3, Issue 1, pages 11–26
p-ISSN 2655-8564, e-ISSN 2685-9432

model with all datasets and pretrained data through which the model infers and tries to pullout the desired output claimed. To attain the desired output, we need to verify the speech to be identical to the trained unseen speech, so as to be sure. If the speech synthesis is not identical, we cannot term the output as the desired output. Such models have implications and applications mostly in the armed forces for providing stealth mode a new edge. Voice cloning can help make communication between borders, across seas and even on telephone a rounded-up mystery, which is the reason it benefits the armed forces greatly. The application also lies with media. People tend to use voice cloning for media concentration applications or entertainment media such as Dubsmash. This model also helps with regeneration of voice wherein, a certain person can communicate if he/she is disabled or lost ability of speech under certain circumstances. The voice cloning works for the future, wherein with on-time upgradation, and correct use we can use regeneration factor for taking over music and pop culture with a wave. With a sytem having a multi-area application domain. It is very important to place certain regulations and boundaries bounded by legal clauses for implementation. Major part of the model lies within the use of the Text-to-Speech system, which is the backbone of the model. Many systems integrated in artificial intelligence pick Text-to-Speech system as prime domain because of its broad area of implementation, right from voice assist, voice detection to voice cloning. Our model works with voice cloning and moves in direction of voice regeneration, which will be a major breakthrough in the near future. The proposed model does not attain human-like naturalness, despite the use of a WaveNet vocoder (along with its very high inference cost), in contrast to the single speaker results. This is a consequence of the additional difficulty of generating speech for a variety of speakers given significantly less data per speaker, as well as the use of datasets with lower data quality.

Acknowledgements

The corresponding author acknowledges all the co-authors and group members for co-operating and working with an optimistic mindset. Every work related to the paper required dedicated and devoted attention from the department in association, and

personal guidance from the project guide, Prof. Lalita Moharkar who stood by all along the buildup of the research model. Trying to walk the entire path from scratch required detailed reference help which acted as a walking stick, which were journal papers and technical papers present in international journals and publications.

References

[1] L. Wan, Q. Wang, A. Papir and I. L. Moreno, "Generalized end-to-end loss for speaker verification." *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference*, 2018.

[2] Y. Jia, Y. Zhang, R. J. Weiss, Q. Wang, J. Shen, F. Ren, Z. Chen, P. Nguyen, R. Pang, I. L. Moreno and Y. Wu, "Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis." *Advances in neural information processing systems*, **31**, 4485–4495, 2018.

[3] Artificial Intelligence at Google – Our Principles. https://ai.google/principles/, 2018.

[4] A.V.D. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior and K. Kavukcuoglu. "Wavenet: A generative model for raw audio." *arXiv preprint 1609.03499*, (2016).

[5] J. Shen, R. Pang, Ron J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis and Y. Wu. "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions." *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018.

[6] M. Grechanik, Q. Xie and C. Fu, "Creating GUI testing tools using accessibility technologies." *Proceedings of the IEEE International Conference on Software Testing, Verification, and Validation Workshops*, 243–250, 2009.

Volume 3, Issue 1, pages 11–26

p-ISSN 2655-8564, e-ISSN 2685-9432

This page intentionally left blank