

Nursultan Akhmetzhanov

Implementation of a Multispeaker Text-to-Speech Synthesis Web Application

IMPLEMENTATION OF A MULTISPEAKER TEXT-TO-SPEECH SYNTHESIS WEB APPLICATION

Nursultan Akhmetzhanov
Bachelor's Thesis
Spring 2021
Information Technology
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
Information Technology

Author: Nursultan Akhmetzhanov

Title of the bachelor's thesis: Implementation of a Multispeaker Text-To-Speech
Synthesis Web Application

Thesis examiner: Kari Jyrkka

Term and year of completion: Sprint 2021

Number of pages: 31

The main goal of this thesis was to implement a multispeaker model on a web application, additional goals were to introduce the reader to text to speech synthesis and compare the open source multispeaker models.

Most of the information was taken from the original research papers and used to explain the work of the models and compare the models of multispeaker TTS using the MOS (Mean Opinion Score) values from the papers themselves. The development of the web application with multispeaker TTS functionality was based on open-source repositories for TTS.

As a result, the goal was not met due to complexity of the chosen task, lack of knowledge and the experience in deep learning and speech synthesis. The steps in the development of the practical application and the recommendations on how to proceed with the tasks were written and explained.

Keywords:

speech synthesis, deep learning, machine learning, web application.

PREFACE

The thesis has been done in spring of 2021 and the topic was chosen by the author himself due to his interest in ML and text-to-speech synthesis. This work has been done with the help of Kari Jyrkkä who held a role of a tutoring teacher for this thesis and me.

I would like to thank my tutoring teacher for this thesis Kari Jyrkkä, who guided me during the development and helped me in the development through regular review sessions, I would not have been able to finish my thesis without him. I would also like to thank my language instructor Johanna Talvensaari who reviewed the language and the thesis writing rules. I would also like to thank Mozilla, its TTS repository on GitHub and everyone who contributed to the development of that project online and made it considerably easier for me to write this thesis and achieve my goal.

Oulu, May 15, 2021
Nursultan Akhmetzhanov

CONTENTS

CONTENTS	5
VOCABULARY	6
1. INTRODUCTION	7
2. HISTORY OF SPEECH SYNTHESIS AND CURRENT SPEECH SYNT- HESIS TECHNOLOGIES	9
2.1 The first steps in speech synthesis. Mechanical speech synthesis	9
2.2 Intelligible speech synthesis. Electronic speech synthesizers	10
2.3 Modern speech synthesis. Deep learning speech synthesizers	12
2.3.1 WaveNet, Parallel WaveNet	12
3. MULTISPEAKER TEXT-TO-SPEECH SYNTHESIS AND HOW IT WORKS	15
3.1 Multispeaker TTS models	15
3.1.1 Tacotron2 with WaveNet	16
3.1.2 Multispeaker ClariNet	17
3.1.3 Multispeaker TTS using Deep Gaussian Processing	17
3.1.4 GlowTTS	19
3.2 Comparisons of multispeaker TTS models	20
4. WEB APPLICATION WITH SPEECH SYNTHESIS	23
4.1 Implementations of multispeaker models	23
4.1.1 Implementation of Tacotron 2	23
4.1.2 Implementation of GlowTTS	23
4.2 Recommendations on the development of a multispeaker TTS application	24
5. CONCLUSION	26
6. APPENDIX	26
REFERENCES	1

VOCABULARY

MOS – Mean Opinion Score is the mean of all opinion values collected from using Likert scale (a scale from 1 to any uneven number, which is usually 5). MOS is often used to collect respondents' subjective opinion on the subject of research.

NLP – Natural Language Processing is a field of machine learning that works on understanding and synthesis of correct text, natural text by computers. NLP is meant to close the gap between computers and humans.

TTS - Text-to-speech, in this paper, referred to text-to-speech synthesis.

1. INTRODUCTION

Speech synthesis is technology associated with future yet is very common in today's world and can be seen from many places and many devices. Famous examples are:

1. Speech Plus – speech synthesis technology from 1980s that was used by Stephen Hawking and automated phone answering systems [3].
2. Google Assistant and other voice assistants – after a series of successful researches [1, 5, 6] on speech synthesis, speech synthesis has become an essential part of Google Assistant. Its simulator is accessible on the developer documentation for Google Assistant [4].

Starting with heavy machines that simulated the work of a vocal tract to generate intelligible human speech and coming to software applications consisting of several united neural networks to synthesize humanlike speech with emotions, intonation and attention to grammar.

The neural network models for speech synthesis have only been recently created and have become faster and more precise every year, resulting in unimaginable progress in speech synthesis. The pinnacle of these research being multi-speaker, meaning a model trained to synthesize speech of multiple people, models capable of synthesizing humanlike speech in a very short time.

The goal of this thesis is to achieve understanding on how text-to-speech synthesis works as well as implementing an application that can copy one's voice from a short recording with a subsequent synthesis of learned voice from the text given as an input [1].

The things to be gone through in this document are:

1. Brief history of the speech synthesis, technologies currently used for the speech synthesis and how they work.
2. Theory behind the technology of multispeaker text-to-speech synthesis and the comparison of some of the multispeaker models.

3. My implementation of multispeaker text-to-speech synthesis and how I came to that step by step using Mozilla's TTS application from their publicly available GitHub repository [2].

2. HISTORY OF SPEECH SYNTHESIS AND CURRENT SPEECH SYNTHESIS TECHNOLOGIES

2.1 The first steps in speech synthesis. Mechanical speech synthesis

The history of speech synthesis started in the 18th century. Researchers tried to mechanically simulate the work of human's vocal apparatus including vocal box, which is called Larynx as well, and mouth area to produce speech. The muscles in vocal box (FIGURE 1), vocal cords, vibrate the air coming from the lungs through the vocal cords to produce sounds that reach mouth area, where the position of the tongue, teeth and lips greatly affect the sounds created, more details in [25].

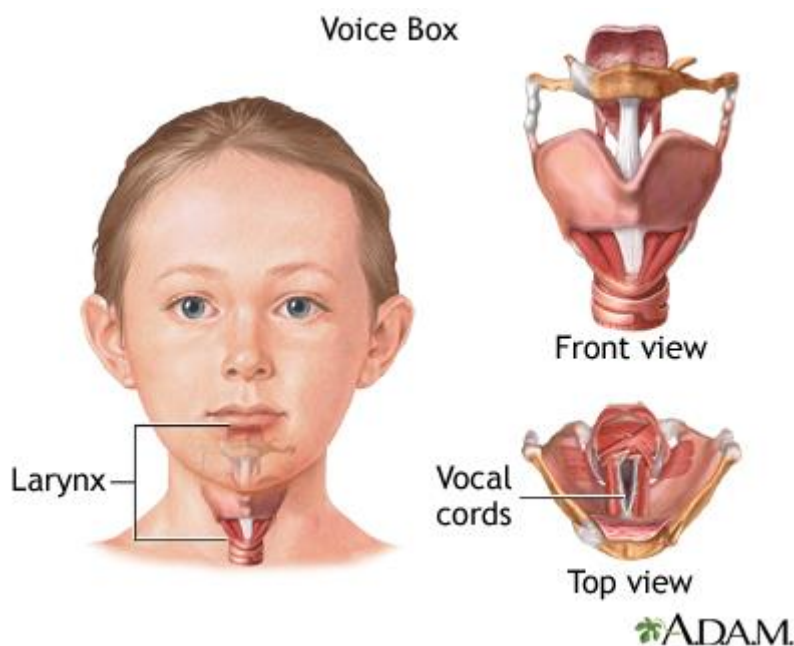


FIGURE 1. The structure of human's larynx with the location of the vocal cords.

The first attempts in speech synthesis [9, 10, 11] were made in the 18th century by Russian Professor Christian Kratzenstein who understood the differences in human physiology on pronunciation of vowels. Armed with this knowledge he constructed an apparatus that produced these sounds. Later, Wolfgang von Kempelen created a more complex design (FIGURE 2.) that imitated human's speech production and imitated the work of the main organs in the human body

that could only create vowels and consonants. The machine could produce vowels with the vibrating reed, consonants were produced with a flow of air through a passage while the reed is off (FIGURE 2.). Resonances were created by deforming the leather resonator [12].

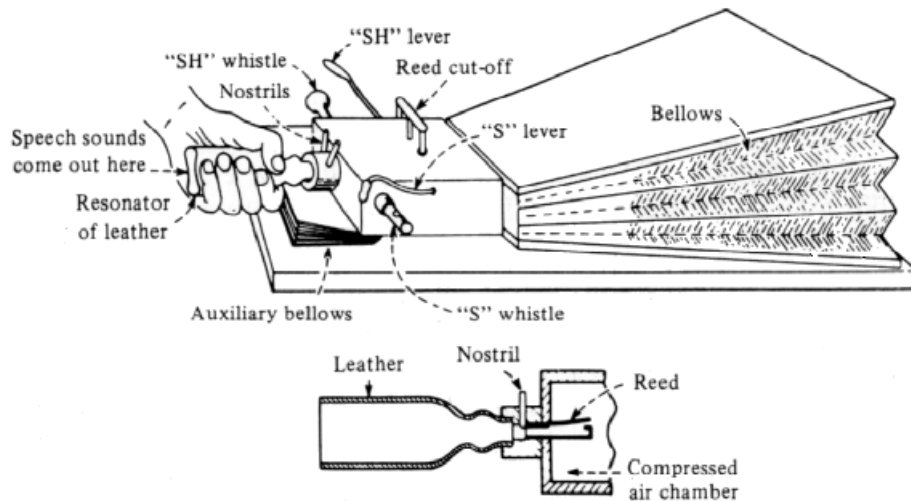


FIGURE 2. Wheatstone's reconstruction of von Kempelen's speaking machine [9].

The development of speech synthesis has not stopped here, but has only just begun, even though none of the constructions could not utter words, it was dynamic generation of sounds, not their recording.

2.2 Intelligible speech synthesis. Electronic speech synthesizers

New breakthroughs in speech synthesis were made in 1939 when a VODER (Voice Operating DEMonstratoR) was presented in World Fair in New York. VODER was inspired by VOCODER (Voice encoder). VODER looked and worked similarly to piano: a foot pedal to control the frequency of the sound and keys that could produce letters, words and short phrases if combined properly, because of that, the machine could only be operated by a trained specialist. VODER was a breakthrough because it showed the wide audience how close current speech synthesis technology was to be used in practice [12].

The next step in the progress of speech synthesis occurred when two IBM researchers John Kelly and Carol Lochbaum programmed IBM 704 to not only synthesize speech but sing as well. Interestingly, Arthur Clarke was inspired to add a song to his book 2001: Space Odyssey, after visiting IBM offices and hearing what machine sang [7].

Later, in 1980s, a company called Voltrax introduced several products with text-to-speech synthesis functionality, having the Voltrax Type'n Talk in 1978 [12]. The products from Voltrax allowed the users to connect the modules to their computers and synthesize speech without interrupting other computer processes. It required a piece of hardware with its own processing power to handle speech synthesis, speech synthesis was way too computationally heavy for the computer to handle on its own.

In 1980 a company called Texas Instruments released a product called Speak'n Spell which was inspired by Voltrax's Type'n Talk. The toy was designed to help children with spelling. It would ask a child to spell a word, one of the words in its database, and the child would have to type the word using toy's keyboard. The toy pronounced each letter and congratulated the child if the word was correct. The innovation in the toy was the fact that it did not store a recording of the words to spell them, but it stored the recording of *phonemes* which it then used to synthesize words and sentences on the go [7, 8].

Another worth mentioning breakthrough is SAM (Software Automatic Mouth), software synthesised speech without any additional hardware. SAM had a massive success on the market despite being very power consuming and synthesising speech of lower quality compared to its analogues from Voltrax [12]. Furthermore, speech synthesis was so computationally heavy it paused all other threads on computer, pausing other applications. Despite all its drawbacks, its advantage of being a purely software solution has made it to Comodore64, Apple's computers and was popular to include or to copy.

Speech synthesis has continued its development but mostly with machine learning technologies. Machine learning increased the quality of the synthesized speech making it closer to the real human speech.

2.3 Modern speech synthesis. Deep learning speech synthesizers

There are two main types of speech synthesizers at this moment:

1. Parametric – synthesizer that learns the “essence” of the speech and learns to generate speech using parameters given to the model. Tweaking pace, style, tone and other speech parameters [17].
2. Concatenative – synthesizer that has a great database of utterances of a single speaker that is used to generate phrases and sentences by concatenating, combining them with one another. Speech synthesizers of this type have a high pronunciation quality but a very low flexibility - a new database of recordings is needed for a speech with a different intonation, style or pace [17].

Recent development in parametric speech synthesis created models and architectures capable of synthesising speech of higher quality than concatenative speech synthesisers [1].

2.3.1 WaveNet, Parallel WaveNet

WaveNet model was developed by Google’s DeepMind to generate raw audio waveforms from other waveforms. The neural network can synthesize high quality speech, sometimes indistinguishable from human speech. The model is autoregressive; in short: the values it outputs are considered and used in its next inputs [16], FIGURE 3 shows WaveNet’s structure, to get full picture on the structure of the model check the animated version of it [16]. This method makes the synthesis very resource demanding. Furthermore, the speech synthesis considers the text input it was given during the synthesis to produce cohesive speech. Without any text input it outputs incohesive nonsense. Since the model is trained to produce any audio waveforms it can also produce songs [16].

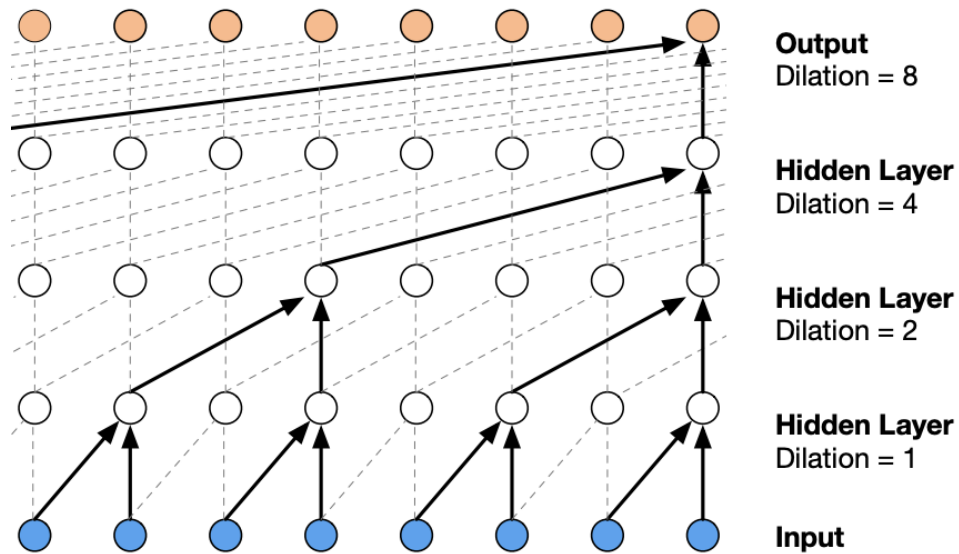


FIGURE 3. WaveNet's structure.

While WaveNet has made a breakthrough in speech synthesis with a resource demanding, slow, but high-quality audio synthesis. That is why, DeepMind has made significant improvements to WaveNet and made Parallel WaveNet that is capable of speech synthesis 20 times faster than real time [14], which means that the model synthesises a second of audio in $1/20^{\text{th}}$ of a second. Parallel WaveNet is created using teacher-student learning approach where WaveNet Teacher “trains” WaveNet Student model to return values like the Teacher WaveNet (FIGURE 4.). The Student WaveNet returns values that get fed by the Teacher WaveNet to output the score, the Student WaveNet uses that score to evaluate own performance and changes its parameters to get a higher score from the Teacher WaveNet, this goes on until its score becomes close enough to the Teacher WaveNet's score.

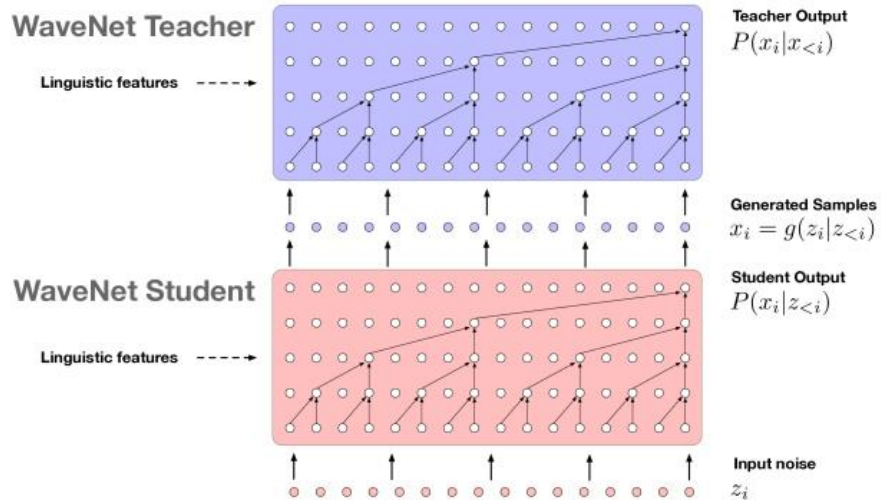


FIGURE 4. Visual representation of Parallel WaveNet’s training.

Parallel WaveNet has high synthesis speed because of its parallel structure that allows it to synthesize all pieces of the audio simultaneously - all utterances from the first to the last at the same time [14].

3. MULTISPEAKER TEXT-TO-SPEECH SYNTHESIS AND HOW IT WORKS

This chapter serves several purposes:

1. To introduce the reader to modern TTS models and architectures.
2. To compare some of the TTS models based on their Mean Opinion Score (MOS) values. MOS values are typically used in evaluation of subjective values like the naturalness of the speech.
3. To find the most suitable TTS models to implement. The most suitable model in terms of the naturalness of the synthesized speech and affordability in computing resource.

3.1 Multispeaker TTS models

Multispeaker TTS, a neural network that can synthesize speech of multiple people, is a technology that has been implemented using various methods [1,18, 19, 20, 21, 22]. The methods and the ideas behind them were so different the technologies have only one thing common with each other - the use of machine learning. Most TTS implementations include two components:

1. Text analysis – a part that analyses the input text and outputs phonemes.
2. Speech synthesiser – a part that takes a sequence of phonemes as an input and outputs waveforms that can be converted to audio files [16].

The others like [20] cannot be categorised as models with two components mentioned above. In case of [20], the architecture consists of a cascade of functions that produce sound from the text input.

Every technology happens to be different from its predecessor, some of them use entirely different, less sophisticated architectures and algorithms of machine learning.

3.1.1 Tacotron2 with WaveNet

The benchmark in the whole industry – Tacotron2 with WaveNet. Tacotron2 is based on Tacotron, an end-to-end TTS model which takes phonemes and graphemes as its input and outputs raw spectrogram from the input data [23]. Spectrogram, shortly, is an audio waveform transformed into time/frequency graph with temperature representing the loudness (amplitude) of the sounds, in more details at [23]. More efficient Tacotron, called Tacotron2 in a pair with vocoder WaveNet create a more complicated yet necessary architecture for multispeaker TTS.

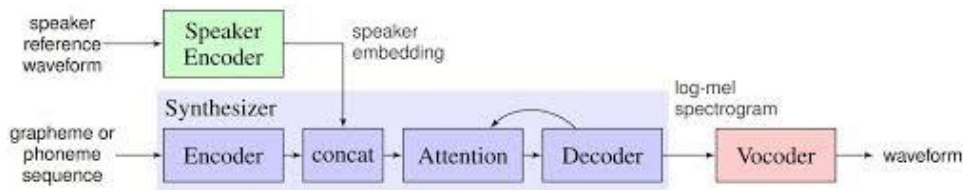


FIGURE 5. Tacotron 2 architecture model [1].

Tacotron 2 consists of 3 big parts: Speaker Encoder, Synthesizer, Vocoder (FIGURE 5). Speaker encoder generates speaker embedding (speaker’s speech fingerprint). The said speaker embedding is given to synthesizer which generates the mel-spectrogram, a spectrogram where frequencies are converted to mel scale values [23], from input grapheme sequences and the speaker embedding. The generated mel-spectrogram is provided to vocoder, integrated WaveNet , to output waveforms. All models are trained using either LibriTTS [28] or LibriSpeech [27] data.

TABLE 1. MOS evaluations from the paper with research on Tacotron 2 comparing it with other models [15].

System	MOS
Parametric	3.492 \pm 0.096
Tacotron (Griffin-Lim)	4.001 \pm 0.087
Concatenative	4.166 \pm 0.091

WaveNet (Linguistic)	4.341 \pm 0.051
Ground truth	4.582 \pm 0.053
Tacotron 2	4.526 \pm 0.066

Combined strengths of Tacotron and WaveNet has made Tacotron 2 the undisputable champion in terms of quality of the speech synthesis, with a Mean Opinion Score (MOS) (TABLE 1.) higher than anyone else's.

3.1.2 Multispeaker ClariNet

Multi-speaker ClariNet differs from other multispeaker TTS technologies due to its model architecture which is basically one model, that generates mel-spectrograms and the waveforms of the final output audio. The whole model consists of 4 components, decoder, encoder, vocoder and bridge-net. All components are connected to each other by bridge-net and are added by speaker embeddings taken from the embedding lookup table as bias [19]. Such architecture allows for unified training of all models at the same time with the bias of a speaker embedding served to teach each model about unique characteristics of speaker's speech.

3.1.3 Multispeaker TTS using Deep Gaussian Processing

Multispeaker TTS using Deep Gaussian Processing (DGP) differentiates itself from other approaches and technologies with its simple and efficient architecture. The DGP based TTS model works on a cascade of GPRs - Gaussian Process Regressions [25], which are stacked one after another because of the assumption that a function can be decomposed into several functions like here [20]:

$$f = f^{(L+1)} \circ f^{(L)} \circ \dots \circ f \quad (1)$$

DGP is not suitable for complex multispeaker TTS, that is why improvements in architecture were made to enable multispeaker text-to-speech synthesis:

1. DGP based multispeaker TTS using speaker codes – uses speaker codes (S) which get applied by DGP layer and then fed to the hidden layers (h_i) of the model and outputs acoustic features (Y) - sound (FIGURE 6).

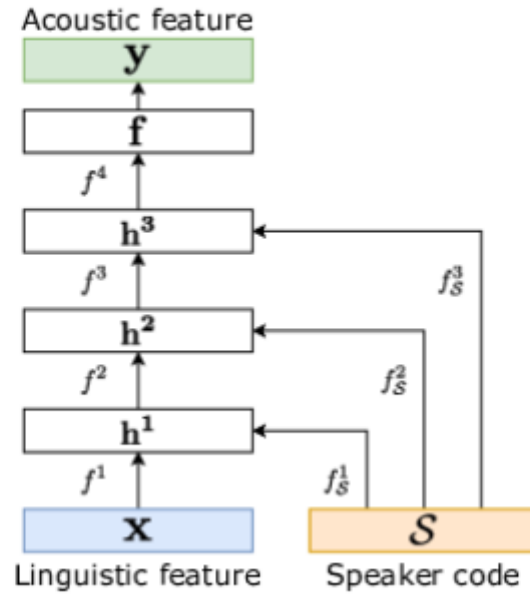


FIGURE 6. Architecture of DGP-based acoustic model for multispeaker TTS with three hidden layers. The hidden layers are fed by the speaker codes applied by single layer GPRs [20].

2. DGPLVM based multispeaker TTS is like the DGP model, but unlike it, it uses similarities and differences between the speaker codes for more efficient and cheaper TTS (FIGURE 7) by concatenating r variables which hold the similarities of speakers' acoustic features.

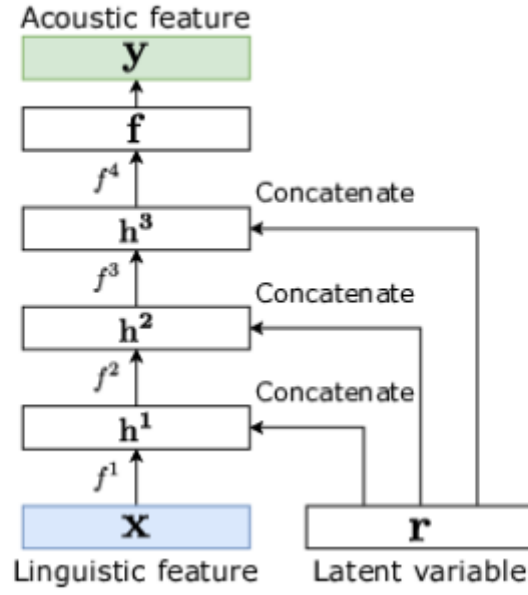


FIGURE 7. Architecture of DGPLVM based model with the hidden layers which are concatenated by latent \mathbf{r} variables [20].

According to the experiments conducted by the authors of [20] DGPLVM and DGP show very similar results, though DGPLVM generally performs slightly better than DGP and was concluded to be more effective when the amount of training data is limited [20].

3.1.4 GlowTTS

GlowTTS was inspired by the way humans read the text: in order and without skipping any words. GlowTTS was designed to produce mel-spectrograms on a monotonic and non-skipping alignments [22].

The model architecture consists of decoder, encoder and duration predictor. The models are very complex but in short, decoder is responsible for synthesis of the audio, encoder is responsible for the analysis of the input text, where at the end it outputs the mel-spectrogram after the duration predictor predicts the needed duration of the mel-spectrogram, more can be found in [22] and its appendix.

The flow based generative nature of the model enables generation of a mel-spectrogram 15.7 times faster compared to Tacotron 2 and synthesizes more natural speech with long input text, detailed explanation of that can be found here [22].

3.2 Comparisons of multispeaker TTS models

Multispeaker models mentioned above are all successful models in their own regard and serve specific purposes. Some models are more precise, more natural sounding, some are faster and less computationally heavy but synthesize less natural speech while the others wanted to test a new approach. The models from the previous subchapter are compared here.

The models were compared primarily by their Mean Opinion Score (MOS). The value used to collect subjective opinions from a big number of people, because naturalness of the speech, typically the most important metric for a speech synthesizer, can only be evaluated by humans. MOS is collected by answers given by the respondents on a 1 to 5 Likert scale, usually, from “bad” to “excellent”. The models were originally compared to either Tacotron2, ground truth data or both. The ground truth data was the MOS on recordings of real people, not the synthesized one speech.

Due to limitations in time and computing resources it was decided to compare the models using the MOS values from their respective research papers.

According to data from the research papers [15, 20, 23] Tacotron 2 with WaveNet scored the highest MOS score. Each research has evaluated their MOS values differently — different samples of data, different people, different training hyperparameters etc. Because of that, it was decided to compare the models to the MOS values of their respective ground truth data, recording of a real human speech, not synthesized. The models were compared to in original papers (TABLE 3).

TABLE 3. Mean Opinion Score (MOS) values from research papers on respective multispeaker TTS models with MOS values of Ground Truth Data.

System	Mean Opinion Score (MOS)	
	Model	Ground Truth Data
Tacotron2 with WaveNet	4.526 ± 0.066[15]	4.582 ± 0.053

Multispeaker ClariNet	$3.90 \pm 0.36[19]$	4.26 ± 0.38
GlowTTS (Highest out of all versions)	$3.45 \pm 0.11[22]$	4.54 ± 0.06

The idea is to calculate the difference between the GTD's MOS and model's MOS values, the lower the difference – the better the model can synthesize natural, believable speech (Table 3).

TABLE 4. The difference between model's MOS and GTDs MOS.

System	Difference between model's MOS and Ground Truth Data's (GTD) MOS
Tacotron2 with WaveNet	0.056 ± 0.013
Multispeaker ClariNet	0.36 ± 0.02
GlowTTS (Highest out of all versions)	1.09 ± 0.05

According to Table 4, Tacotron2 with WaveNet appears to synthesize speech the closest to GTD, with Multispeaker Clarinet right behind it and GlowTTS the last. GlowTTS showed such results compared to GTD, but showed comparative results to the Tacotron2 they build for evaluation phase $3.35 + 0.11$, which is not that different from GlowTTS's MOS. Furthermore, due to it being parallel TTS model, parallel TTS models generate mel-spectrograms faster than autoregressive TTS models like Tacotron and Tacotron 2, GlowTTS happens to generate mel-spectrograms 15.7 times faster than Tacotron 2 while obtaining comparable performance according to [22].

As you can see Tacotron 2 with WaveNet shows the best performance out of all models listed here, proving why it is used as a benchmark for TTS models. Tacotron 2 shows impressive results to this day especially since researchers keep developing the technology by replacing WaveNet with Parallel WaveNet, WaveGlow etc. Despite that, it is important to understand that Tacotron 2 is not the perfect model for multispeaker TTS: Tacotron 2 struggles to synthesize speech from a long input text without any issues, while GlowTTS can do that, but with noticeable monotonic sound of the speech.

4. WEB APPLICATION WITH SPEECH SYNTHESIS

This chapter describes the steps taken to implement multispeaker TTS using open-source repositories and speech datasets. Furthermore, this chapter includes recommendations on implementation of multispeaker TTS web application.

4.1 Implementations of multispeaker models

Implementation of TTS in a user-friendly way is a difficult task, an implementation of a multispeaker TTS as a web application is a challenge. To implement a multispeaker TTS application an open-source TTS repository from Mozilla [2] was chosen as a foundation. The repository was cloned, and steps were taken according to official build instructions on the server found in wiki section of the repository [2], the best versions of the checkpoint files were used on builds of the server for every model tried.

4.1.1 Implementation of Tacotron 2

Initially, the versions of Tacotron 2 were used to synthesize human speech due to its high MOS. Unfortunately, the web application could not run on Tacotron 2 because of the absence of `'/data/rw/home/Data/VCTK/scale_stats.npy'` document on every version of the Tacotron 2. The absent document could not be found on the issues on the repository of the application and could not be recreated. Because of that, the model of the application was switched to GlowTTS, because its MOS value were right after Tacotron 2.

4.1.2 Implementation of GlowTTS

Web application on GlowTTS could synthesize speech from its default voice, without any speaker embeddings. Speaker embeddings were generated according to instructions from `'/TTS/TTS/speaker_encoder/README.md'` file, the only change done in `'/TTS/TTS/speaker_encoder/config.json'` file was the change in `model.input_dim` to 80. Speaker embedding was generated from a short recording of an author's voice saying: "This is the recording of my voice. I'm seeing

things that should be said and I like seeing things”, calling the `‘/TTS/TTS/speaker_encoder/bin/compute_embeddings.py’`. The recording was stored in .wav format and put in a folder structure imitating the structure of LibriTTS dataset with a text file with a transcription of the recording. The generated speaker embedding was stored in application directory for speaker embeddings - `‘/TTS/TTS/speaker_encoder/voice_embeddings’`.

Speaker embeddings could not be used by the model to synthesize the speech, the files and functions in the application without modifications in the code which were suggested by one of the main contributors of the repository and discussed in this issue [29]. The changes were done in files:

1. `‘TTS/TTS/bin/compute_embeddings.py’`
2. `‘TTS/TTS/server/conf.json’`
3. `‘TTS/TTS/speaker_encoder/config.json’`
4. `‘TTS/TTS/tts/models/glow_tts.py’`
5. `‘TTS/TTS/speaker_encoder/model.py’`
6. `‘TTS/TTS/server/server.py’`
7. `‘TTS/TTS/utils/synthesizer.py’`

The changes done in these files were done to debug modify the functions in the files to get the needed results. The files were chosen from python’s error messages. Debugging did not yield any results, speech synthesis from the speaker embedding was not realized. The further development of the application was stopped because the application could not be developed from the scratch with the time left for the deadline.

4.2 Recommendations on the development of a multispeaker TTS application

First, the general recommendation is to not take the chosen task lightly, understand your skills and the knowledge needed to accomplish the chosen task. Specifically:

1. Understanding the basic machine learning concepts like training, models, algorithms, datasets, supervised learning, unsupervised learning, reinforcement learning, classification, regression etc.
2. Understanding of the model, technology chosen to be implemented. Poor or superficial understanding of the concepts of the technology will jeopardize the development of the application. The developer should understand what the chosen model does and how it does that.

While the above-mentioned recommendations there are recommendations of minor importance:

1. Experience in the development of applications. Whether you decide to develop web, mobile or desktop application, the understanding of the nuances of the platform and experience in its development can decrease the difficulty of the development.
2. Experience in speech synthesis and NLP. Experience in both will mitigate the risks of failure of the understanding and the development of the application. Having previous experience in a similar or the same field is beneficial but not necessary.
3. Regular feedback from developers with experience in the speech synthesis field, preferably – the chosen technology. The connection with developers working on similar projects will be beneficial for the problem solving. This can be accomplished by making your application open source and by through resources like Stack Overflow or issues on GitHub.
4. The project's foundation should not be a third-party repository, unless the author understands the codebase of the application, can freely modify the code of the application and the repository has an active community capable of answering the questions and issues appeared.

5. CONCLUSION

The history of speech synthesis is old and had many large breakthroughs to get to the current level of speech synthesis. That is why the implementation of a TTS web application should not be taken lightly since the task requires understanding and experience in machine learning, deep learning and NLP. This paper failed to implement the web application with multispeaker TTS, primarily because of the lack of needed knowledge and experience in machine learning, NLP and TTS, and because of the shortage of time. Such a task is recommended to be taken only with enough time and knowledge, furthermore, the development of the application should be not heavily based on a third-party open-source repository unless the repository has a strong documentation, active community, well written and modifiable codebase.

REFERENCES

- (1) Y. Jia, Y. Zhang, R. J. Weiss, Q. Wang, J. Shen, F. Ren, Z. Chen, P. Nguyen, R. Pang, I. L. Moreno, Y. Wu, 2018-2019. "Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis". *Google funded research*, [Accessed: April 18, 2021], <https://research.google/pubs/pub47019/>
- (2) Mozilla, 2018-2021, "TTS". [Accessed: April 5, 2021], <https://github.com/mozilla/TTS>
- (3) M. Joao, "How Intel Gave Stephen Hawking a Voice", *Wired UK issue 01.15*, [Online], [Accessed: April 25, 2021] Available: <https://www.wired.com/2015/01/intel-gave-stephen-hawking-voice/>
- (4) TTS Simulator, SSML, *Google Assistant for Developer*. [Accessed: April 25, 2021], https://developers.google.com/assistant/conversational/df-asdk/ssml#tts_simulator
- (5) Y. Wang, R.J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Zongheng Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, R. A. Saurous, 2017, "TACOTRON: TOWARDS END-TO-END SPEECH SYNTHESIS", [Accessed: April 26, 2021], <https://arxiv.org/pdf/1703.10135.pdf>
- (6) A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, 2016, "WAVENET: A GENERATIVE MODEL FOR RAW AUDIO", [Accessed: April 30, 2021], <https://arxiv.org/pdf/1609.03499.pdf>
- (7) "The Evolution of Computer Speech", 28.02.2018, *The Science Elf*, [Accessed: April 30, 2021], <https://www.youtube.com/watch?v=wQjTgvUEOrY>
- (8) "How Speech Synthesizers Work", 10.03.2019, *The 8-bit guy*, [Accessed: April 30, 2021], <https://www.youtube.com/watch?v=XsMRxNSDccc>
- (9) Flanagan J. (1972), "Speech Analysis, Synthesis, and Perception", *Springer-Verlag, Berlin-Heidelberg-New York*.
- (10) Flanagan J., Rabiner L. (Editors) (1973), "Speech Synthesis", *Dowden, Hutchinson & Ross, Inc., Pennsylvania*.

- (11) Schroeder M. (1993), "A Brief History of Synthetic Speech", *Speech Communication vol. 13*, pp. 231-237.
- (12) S, Lemmetyy, 06.1999, "Review of Speech Synthesis", *Technology*, [Accessed: May 3, 2021], http://research.spa.aalto.fi/publications/theses/lemmetyy_mst/chap2.html
- (13) D. Jurafsky & J. H. Martin, "Speech and Language Processing", *Hidden Markov Models*, 12.2020, [Accessed: May 6, 2021], <https://web.stanford.edu/~jurafsky/slp3/A.pdf>
- (14) G. van den Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, D. Hassabis, 2017, "Parallel WaveNet: Fast High-Fidelity Speech Synthesis", [Accessed: May 7, 2021], <https://arxiv.org/pdf/1711.10433.pdf>
- (15) J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, RJ Skerry-Ryan, R. A. Saurous, Y. Agiomyriannakis, and Y. Wu, 2018, "NATURAL TTS SYNTHESIS BY CONDITIONING WAVENET ON MEL SPECTROGRAM PREDICTIONS", [Accessed: May 7, 2021], <https://arxiv.org/pdf/1712.05884.pdf>
- (16) A. van den Oord, S. Dieleman, 09.2016, "WaveNet: A generative model for raw audio", [Accessed: May 8, 2021], <https://deepmind.com/blog/article/wavenet-generative-model-raw-audio>
- (17) J. Singh, 11.2018, "WaveNet: Google Assistant's Voice Synthesizer", *Towards Data Science*, [Accessed: May 9, 2021], <https://towardsdatascience.com/wavenet-google-assistants-voice-synthesizer-a168e9af13b1>
- (18) M, Chen, X, Tan, Y. R. J. Xu, H. Sun, S. Zhao, T. Qin, T. Liu, 06.2020, "MultiSpeech: Multi-Speaker Text to Speech with Transformer", [Accessed: May 10, 2021], <https://arxiv.org/pdf/2006.04664.pdf>
- (19) J. Park, K. Zhao, K. Peng, W. Ping, 07.2019, "Multi-Speaker End-to-End Speech Synthesis", [Accessed: May 10, 2021], <https://arxiv.org/pdf/1907.04462.pdf>

- (20) K. Mitsui, T. Koriyama, H. Saruwatari, 08.2020, "Multi-speaker Text-to-speech Synthesis Using Deep Gaussian Processes", [Accessed: May 11, 2021], <https://arxiv.org/pdf/2008.02950.pdf>
- (21) Z. Cai, C. Zhang, M. Li, 05.2020, "From Speaker Verification to Multispeaker Speech Synthesis, Deep Transfer with Feedback Constraint", [Accessed: May 12, 2021], <https://arxiv.org/pdf/2005.04587.pdf>
- (22) J. Kim, S. Kim, J. Kong, S. Yoon, 2020, "Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search", [Accessed: May 12, 2021], <https://arxiv.org/pdf/2005.11129.pdf>
- (23) Y. Wang, RJ Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, R. A. Saurous, 04.2017, "TACOTRON: TOWARDS END-TO-END SPEECH SYNTHESIS", [Accessed: May 12, 2021], <https://arxiv.org/pdf/1703.10135.pdf>
- (24) "How Does My Voice Work?", *Temple Health, Temple Head & Neck Institute*, 11.04. 2018, [Accessed: May 19, 2021], [https://www.temple-health.org/about/blog/how-does-my-voice-work#:~:text=When%20you%20breathe%2C%20the%20vocal,\(cyclic%20opening%20and%20closing\)](https://www.temple-health.org/about/blog/how-does-my-voice-work#:~:text=When%20you%20breathe%2C%20the%20vocal,(cyclic%20opening%20and%20closing)).
- (25) L. Roberts, "Understanding the Mel Spectrogram", *Analytics Vidhya*, [Accessed: May 20, 2021], <https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53>
- (26) H. Sit, "Quick Start to Gaussian Process Regression", *Towards data science*, [Accessed: May 20, 2021], <https://towardsdatascience.com/quick-start-to-gaussian-process-regression-36d838810319#:~:text=Gaussian%20process%20regression%20is%20nonparametric,functions%20that%20fit%20the%20data>.
- (27) "LibriSpeech ASR corpus", *Open SLR*, [Accessed: May 15, 2021], <https://www.openslr.org/12>
- (28) H. Zen, V. Dang, R. Clark, Y. Zhang, R. Weiss, Y. Jia, Z. Chen, Y. Wu, "LibriTTS", *Google Research*, 2019, [Accessed: May 15, 2021], <https://research.google/tools/datasets/libri-tts/>

- (29) “Can we not plug-in embeddings if synthesizing on a single-speaker model?”, 10.03.2020, *Mozilla/TTS*, [Accessed: May 1, 2021], <https://github.com/mozilla/TTS/issues/378>
- (30) DubstepKnight, glow-tts, May of 2021, [Accessed: May 20, 2021], <https://github.com/DubstepKnight/glow-tts>