
MULTILINGUAL TEXT-TO-SPEECH WITH GRAD-TTS

Soklay Heng
IDMC, Université de Lorraine
Nancy, France

Albert Millert
IDMC, Université de Lorraine
Nancy, France

Shalini Priya
IDMC, Université de Lorraine
Nancy, France

Wenjun Sun
IDMC, Université de Lorraine
Nancy, France

February 7, 2022

ABSTRACT

The main objective of this work and the major part of our software project was building a web interface generating speech from a text; in our multilingual setting, we considered English and French. The end-user can benefit from the service by providing text as input in either English or French. The system automatically generates a synthesized voice uttering inputted text. We distinguish between two core components, enhancing our multilingual system - language classifier, and speech synthesis module. The former identifies text as either English or French so that the latter can generate speech accordingly. We facilitated *Grad-TTS*¹ model based on diffusion probabilistic modeling. It was used for both languages. Our multilingual speech synthesizer model was trained on *LJ Speech* dataset for English and *SIWIS* dataset for French. At the end of system development, we've utilized both objective and subjective measures for evaluating the outputs of the speech synthesis model.

keywords: *Text-to-Speech, Speech Synthesis, Multilingual, Grad-TTS, English, French, SIWIS, LJ Speech*

1 Introduction

Speech is one of the most effective ways of communication compared to other activities in humans' daily lives. It allows people better understand each other, regardless of potential barriers when using other means of communication (such as reading or writing). With progressing advancement of technology, one can facilitate computers to replace humans in many of their repetitive and arduous tasks. A speech synthesizer's objective is to work on the speech production task, which refers to generating human voice artificially. Text-to-speech (TTS) is one of the speech synthesis tasks, alongside others, such as expressive speech and talking head. Text-to-Speech synthesis consists of producing a spectrogram (speech) from a text provided as input Wu, Watts, and King, 2016. Dutoit (2001) claims that the main purpose of such a system is uttering the input text in the most familiar way to the human ear with the use of naturalness and intelligibility to measure speech quality.

There are various methods for realizing speech synthesis task (Siddhi, M., and Bhavik, 2017). They include formant synthesis, articulatory speech synthesis, concatenative speech synthesis, or - statistical-based speech synthesis. However, those methods are outdated and do not perform well in terms of naturalness. For instance, concatenative speech synthesis produces voice based on the pre-recorded speech sample units. It outputs speech by concatenating the units based on the input text. This method does not provide flexibility for producing the outputs for unstored speech samples in the database and naturalness of speech because outputs are generated by concatenating units of pre-recorded speech. TTS synthesis allows automatic phonetization of any text using phonetic rules and the language phonemes. Therefore,

¹<https://github.com/huawei-noah/Speech-Backbones/tree/main/Grad-TTS>

a text-to-speech system needs to generate human-like speech by modeling the prosody taking into account many phenomena in speech, such as intonation, style, stress, rhythm, and paralinguistic information.

Neural networks are one way of addressing this task. A common practice nowadays is to utilize a hybrid parametric end-to-end TTS deep neural network model. It consists of an acoustic model and a neural vocoder. This method offers better performance of the TTS model in terms of naturalness and intelligibility. For this purpose, we use the *Grad-TTS* model based on diffusion probabilistic modeling. The baseline of the TTS model contains a feature generator grounded in diffusion probabilistic modeling for mel-spectrograms production, which is then passed to a neural vocoder, namely - *HiFi-GAN* to generate waveform (speech) at the end. The model’s architecture is illustrated in more details in the following 4.2 section.

The beneficial qualities provided by our multilingual TTS system consist of assisting those who have visual impairment problems by improving their speaking skills both in English and French as they can learn how to pronounce a word in either English or French. Despite considerable progress in the area, generating natural speech from text remains a challenging task (BOURLARD et al., 2011).

The main objective of our software project is to create a web interface for multilingual TTS system. Multilinguality could be defined in different ways; therefore, it is important to understand what we mean by it for our system. Two main approaches can be discussed in this interpretation:

- The system in multiple languages, but for speech generation applying the selected language model to the whole text, regardless of what language it is originally written in.
- Training the system in multiple languages, detecting the language in which the whole text or different parts of the text are written, and then producing the speech using the dedicated language model for each detected part.

The second approach concerns with our work for this project. Our contribution to the original work of *Grad-TTS* by the author² which is available only for English is to train on French dataset using the same model, and the classifier is also included in our system. Our multilingual TTS system is composed of two main modules, language classifier and speech synthesizer model.

The paper is organized as follows: section 2 concerns the related work and describes other speech synthesis tasks closely related to our TTS model. Section 3 focuses on the architecture of our multilingual TTS system. The next section 4 describes the models, including language classifier model and *Grad-TTS* model used in our system. Following this, section 5 illustrates the dataset used for training our models, both for language classification and *Grad-TTS* model; we also describe data processing. Section 6 describes our training process and experiments. The next 7 section illustrates the results and how system evaluation. A final section 8 includes the conclusions.

2 Related Work

We focus our literature review on text-to-speech generation related task only in this section. Deep neural network TTS is often composed with feature generator, which its function is to convert the input text to time frequency domain acoustic features, and vocoder, which is used to synthesize raw waveform conditioned on features generated by feature generator. This architecture is first contributed by the introduction of autoregressive models by *Tacotron2* (Shen et al., 2020) used for feature generation and *WaveNet* (Oord et al., 2018) used for synthesizing speech. Other popular generative modelling frameworks including *Generative Adversarial Networks* (Goodfellow et al., 2014) and *Normalizing Flows* (Rezende and Mohamed, 2015) were later used in the design of TTS system for a parallel generation producing comparable quality of the synthesized speech.

Tacotron2 (Shen et al., 2020) and *Transformer-TTS* (Li, 2019) feature generators produce speech synthesis with a high quality of naturalness because of acoustic features generation frame by frame, resulting in a perfect mel-spectrogram reconstruction from input text. *FastSpeech* (Ren et al., 2019) and *Parallel Tacotron* (Elias et al., 2020) were then introduced to solve inference speed problem in *Tacotron2* and Transformer performance and to improve pronunciation robustness. However, these models still have a drawback in terms of speed efficiency because re-computed alignment from the teacher model is required to learn character duration. *Glow-TTS* feature generator (Kim et al., 2020) based on Normalizing Flows has made a great success to deal with issues for pronunciation and computational issues typical for autoregressive solutions. It can map the input text to melspectrograms efficiently. Lately, *Diffusion Probabilistic Models* (DPMs) (Sohl-Dickstein et al., 2015) achieve a success in providing a capability to model complex data distributions. *WaveGrad* (Chen et al., 2020) and *DiffWave* (Kong et al., 2021) were shown to reproduce the fine-grained structure of human speech and and require less sequential operations. Recently, *Grad TTS* (Popov et al., 2021) motivated

²<https://github.com/huawei-noah/Speech-Backbones/tree/main/Grad-TTS>

by diffusion probabilistic vocoder has used diffusion probabilistic modelling as feature generator, making a great improvement of training speed and quality of synthesized speech.

Parallel GAN-based vocoders, such as *Parallel Wave- GAN* (Yamamoto, Song, and Kim, 2019), *MelGAN* (Kong, Kim, and Bae, 2020), and *HiFi-GAN* (Kong, Kim, and Bae, 2020) has shown a great improvement of the waveform generation performance on CPU devices which outperforms Parallel WaveNet and Wave Glow normalizing-flow-based architecture which produce faster speech generation on GPU devices only.

3 System Architecture

There are several non-functional requirements expected from the system: the application should be easily reproducible, shareable, extendable. We achieved most of the requirements by containerizing services using Docker. This is visualized in figure 1. UI service is simply a frontend application that serves as the main point of (indirect) contact between a client and a server. This service exposes port 3000, under which users can visit the website. As mentioned previously, *UI* doesn't have a direct connection to the *Server*. Instead, it sends a user request to the *Proxy* service (more specifically - reverse proxy), which accepts communication on a default *HTTP* port 80. This way, users cannot directly communicate with the server; it prevents them from performing various common server attacks.

Nginx facilitated as proxy provides a variety of flexibility. It can be extended easily with load balancing functionality. That could be useful once the application generates higher traffic, *Nginx* allows distributing the traffic automatically between several microservice instances of the server to handle the load. Containerization of the system helps in this regard. Because of it, we could easily spin up more containers of a given service; *Nginx* would take care of managing the traffic itself (respectively to the specified configuration file).

The request is then passed to a chosen server instance through *Gunicorn* python server serving *Flask* application. The former is accessible through port 8000, whilst *Flask* web API can be found (but not accessed from the outside) on port 5000. The server processes the request and provides a response that travels through the same way as it previously came to finally reach *ReactJS* website served in the user's browser.

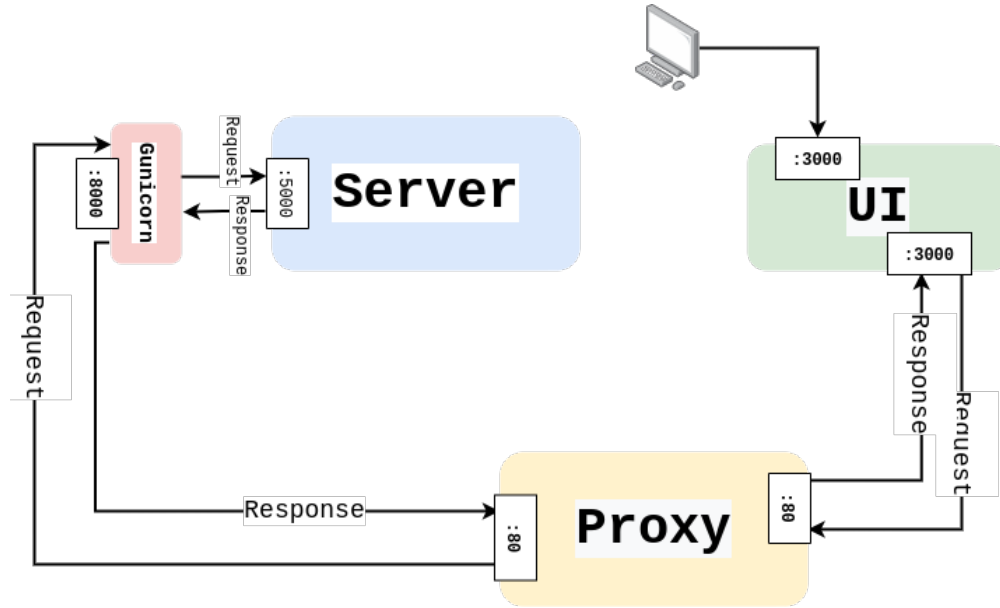


Figure 1: System containerized architecture with request travel visualization

As we've seen, it gives more flexibility while increasing overall security in the system by hiding services and their ports to a minimum; services are accessible in an underlying network in which they recognize each other with names - no hardcoding IP addresses in the code. One reason for containerization that we haven't mentioned yet - no downtime in deployment; replacing a container with another version can be achieved seamlessly: 1) deploy new service version, 2) old version keeps running, 3) when ready - split traffic between the two versions using *Nginx*, 4) take the old service version down, 5) spin up more instances of new service if needed.

3.1 API Endpoints

The minimal server web API constitutes a *POST* endpoint under route. In case the server obtains another *HTTP* method, it will respond with an audio blob file, which when played utters *Can't touch this* - as the server can't be reached this way successfully. Otherwise, the request is being processed: 1) joining all lines together, 2) language classification, 3) respective synthesis model used to generate an audio file with speech, 4) sending back attachment in *.mp3 file as a response.

4 Models

4.1 Language Classification

The role of the Language classifier is to distinguish the language of the input sentence. Through language recognition, the system is able to select the appropriate speech model for a better user experience.

To train the Language classifier, we used an English-French reference database with 200 English and French sentences respectively. The Language classifier first uses BERT to embed the words in the sentence to obtain word vectors, and then uses the LSTM neural network to train the classifier. The structure of the language classifier is shown in the following figure.

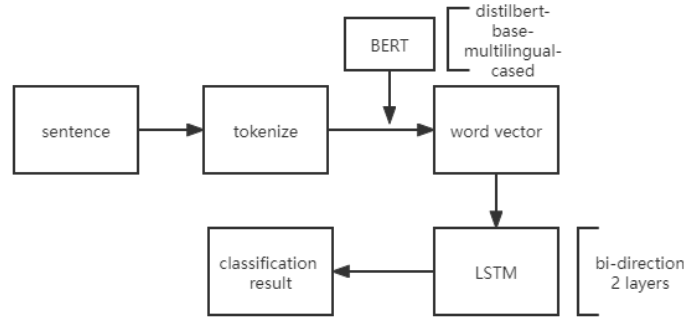


Figure 2: Pipeline of classifier

Because it is a relatively easy to implement binary classification problem, the classifier achieves good performance, it can accurately distinguish English and French sentences.

4.2 Grad TTS

Grad TTS (Popov et al., 2021) is an acoustic feature generator with a score-based decoder using recent diffusion probabilistic modelling insights. In *Grad-TTS*, MAS-aligned encoder outputs are passed to the decoder that transforms Gaussian noise parameterized by these outputs into a mel-spectrogram. To cope with the task of reconstructing data from Gaussian noise with varying parameters, we write generalized version of conventional forward and reverse diffusions has been included. One of the remarkable features of this model is that it provides explicit control of the trade-off between output mel-spectrogram quality and inference speed.

In particular, *Grad-TTS* is capable of generating mel-spectrograms of high quality with only as few as ten iterations of reverse diffusion, which makes it possible to outperform *Tacotron2* in terms of speed on GPU devices. Additionally, it is possible to train *Grad-TTS* as an end-to-end TTS pipeline (i.e., vocoder and feature generator are combined in a single model) by replacing its output domain from mel-spectrogram to raw waveform.

4.2.1 Diffusion probabilistic modelling

It is a type of stochastic process and the goal is to find a reverse diffusion such that its trajectories closely follow those of the forward diffusion but in reverse time order. The basic idea behind diffusion probabilistic models (DPMs) is to build a forward diffusion process by iteratively destroying original data until we get some simple distribution (usually standard normal), and then we try to build a reverse diffusion parameterized with a neural network so that it follows the trajectories of the reverse-time forward diffusion.

Forward diffusion process transforms any data into Gaussian noise given infinite time horizon T .

Reverse diffusion removes noise not arbitrarily, but according to the reverse trajectories of the forward diffusion. Since forward diffusion adds noise gradually, reverse diffusion has to remove noise gradually as well.

Loss function is used to estimate gradients of log-density of noisy data often referred to as score matching.

4.2.2 Inference

The acoustic feature generator consists of three modules: encoder, duration predictor, and decoder. An input text sequence length L typically consists of characters or phonemes. The encoder converts an input text sequence into a sequence of features used by the duration predictor to produce hard monotonic alignment. The output sequence is then passed to the decoder, which is a *Diffusion Probabilistic Model*.

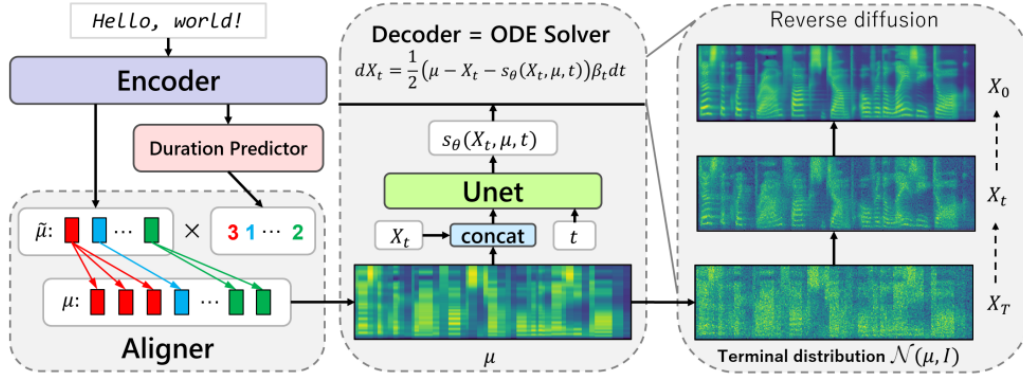


Figure 3: Grad TTS inference scheme

4.3 Auto MOS

While crowdsourcing introduces a degree of parallelism to the rating process, it is still relatively costly and time-consuming to obtain *MOS* for TTS quality testing. (Patton et al., 2016) Inspired by *AutoMOS* (Patton et al., 2016) and *MOSNet* (Patton et al., 2016) and the iflytech platform, we decided to try to build an automatic MOS evaluation model. This model takes the time-frequency map of the speech file as input, obtains the feature value through CNN, and finally inputs the feature into the LSTM network for *MOS* analysis.

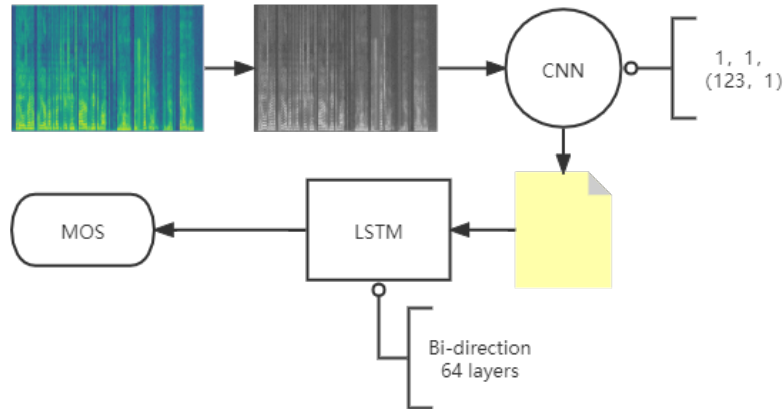


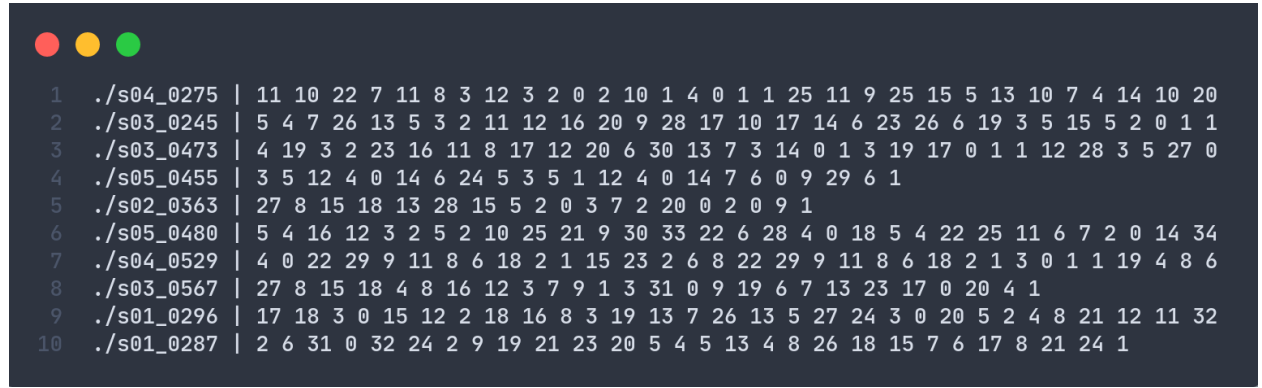
Figure 4: Pipeline of AutoMOS

5 Data Preparation

For this task we needed two datasets: one for English, and another one for French.

*LJ Speech*³ dataset is a publically available speech dataset in English provided by (Ito and Johnson, 2017). It contains 13100 audio files with the utterances based on seven books. The female voice from the recordings was recorded by a single speaker. Each audio file has a respective transcription file. Authors inform that recordings range from 1 to 10 seconds and altogether constitute around 24 hours worth of speech data. The dataset comes preprocessed and ready to use for the model.

In order to satisfy the functional requirements of the application, we needed to extract and preprocess French dataset. We opted for another publically available *SIWIS*⁴ dataset. It was also recorded by a female voice. (Honnet et al., 2017) claim that the dataset is "aimed" for building text-to-speech tools as it provided high-quality recordings alongside the transcriptions. The raw dataset contains 9750 recordings, altogether constituting over 10 hours worth of speech data. Utterances come from various sources - debates, fiction books, etc. A subset of the data contains words emphasised in different context, but we didn't take it into account. The data is more limited in terms of data size than *LJ Speech* but in our work we had to take the available resources and time constraints into account.



1	./s04_0275	11 10 22 7 11 8 3 12 3 2 0 2 10 1 4 0 1 1 25 11 9 25 15 5 13 10 7 4 14 10 20
2	./s03_0245	5 4 7 26 13 5 3 2 11 12 16 20 9 28 17 10 17 14 6 23 26 6 19 3 5 15 5 2 0 1 1
3	./s03_0473	4 19 3 2 23 16 11 8 17 12 20 6 30 13 7 3 14 0 1 3 19 17 0 1 1 12 28 3 5 27 0
4	./s05_0455	3 5 12 4 0 14 6 24 5 3 5 1 12 4 0 14 7 6 0 9 29 6 1
5	./s02_0363	27 8 15 18 13 28 15 5 2 0 3 7 2 20 0 2 0 9 1
6	./s05_0480	5 4 16 12 3 2 5 2 10 25 21 9 30 33 22 6 28 4 0 18 5 4 22 25 11 6 7 2 0 14 34
7	./s04_0529	4 0 22 29 9 11 8 6 18 2 1 15 23 2 6 8 22 29 9 11 8 6 18 2 1 3 0 1 1 19 4 8 6
8	./s03_0567	27 8 15 18 4 8 16 12 3 7 9 1 3 31 0 9 19 6 7 13 23 17 0 20 4 1
9	./s01_0296	17 18 3 0 15 12 2 18 16 8 3 19 13 7 26 13 5 27 24 3 0 20 5 2 4 8 21 12 11 32
10	./s01_0287	2 6 31 0 32 24 2 9 19 21 23 20 5 4 5 13 4 8 26 18 15 7 6 17 8 21 24 1

Figure 5: File - phoneme indices input data format

SIWIS dataset wasn't enough as we couldn't pass it to the model in this form. Authors provided a script using which we were able to send request with the transcription to their web API. Then, given a response, we could finally extract a list of phonemes corresponding to the text. Going through all text we compiled a dictionary mapping phonemes to indices, and vice versa. This dictionary was used to turn textual transcriptions into numeric values which could then be inputted to the model. The reverse dictionary could be used for transforming model's predictions back to the human-readable form. A simplified and truncated format obtained from the preprocessing step is exemplified by figure 5. The left side informs about the sample taken into account; the right side lists sequentially indices corresponding to the phonemes.

6 Experiment

Our models were trained by using Grid5000⁵, which is a large-scale and flexible server for experiment-driven research in most of the computer science related fields and high computing in Cloud, Big Data and AI. The server enabled us to successfully train our models, which demand a lot of memory usage, storage and powerful performance – exceeding by far the capabilities of a regular computer.

The input phonemized *SIWIS* text (as described in 5) is passed to the encoder. The output were an 80-dimensional mel-spectrograms. *Grad-TTS* was trained for 5797 iterations with a mini-batch size of 16. *Adam* optimizer was used and the learning rate was set to 0.0001.

The model has been trained for 13 epochs so the output is not very accurate. Figure 6 shows the diffusion loss during training. Long training is essential to get a good model.

³<https://www.keithito.com/LJ-Speech-Dataset>

⁴<https://datashare.ed.ac.uk/handle/10283/2353>

⁵<https://www.grid5000.fr/w/Grid5000:Home>

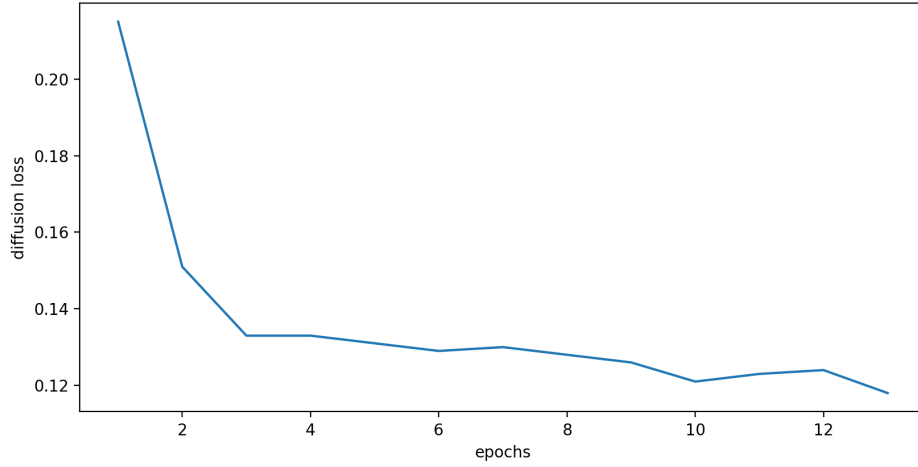


Figure 6: French : Diffusion loss at training

7 Result

Regarding the evaluation of our TTS model performance, two features were taken into account, naturalness and intelligibility. Naturalness concerns with how natural an artificial speech is; a natural speech generation is not just a combination of each word speech together, but it should produce a smooth human-like sound. Intelligibility is the most important feature of speech generation because the main goal of speech production is the comprehension of the speech waveform perceived by humans; speech waveform will be useless if the listeners do not understand what the speech waveform is representing for. Objectivity and subjectivity are two approaches used to measure the two criteria of the TTS.

7.1 Intelligibility

Objective evaluation is used to measure the quality of waveform; the idea is that we use automatic speech recognition (ASR) python library developed by Google for transcribing our synthesized speech waveform. With the transcription achieved by *Google ASR*⁶, we can use this hypothesis text to make a comparison with the ground truth text by computing the word error rate (WER)⁷. However, this approach does not guarantee that there is no bias in evaluating our TTS model. This is due to the fact there will be a degrading quality of speech signal when passing to the ASR.

Table 1: Results obtained from performing the experiments using ngram-based approach

Language	Sentences	WER compared with Google ASR
English	10	0.56
French	10	0.30

Subjectivity evaluation is a useful evaluation method for speech synthesizer in addition to the objective evaluation because only humans have much better performance in listening to speech; also, speech synthesizer is implemented for serving the purpose of making an interaction between humans. Therefore, it is essential to ask people whether or not they understand the speech produced by the synthesizer. This will provide a reliable and more accurate result for evaluating our TTS model. The task involves asking 10 to 15 English and French native speakers for doing an online survey. The English online survey can be found here: <https://www.questionpro.com/a/TakeSurvey?tt=X9oMaZgBU/8%3D> The French online survey can found here: <https://www.questionpro.com/a/TakeSurvey?tt=HKC1n98eV8A%3D> In order to avoid bias for measuring the intelligibility, we have randomly select 5 synthesized speeches for the participants to perform task. We would like to access whether or not people understand what the synthesized speeches are about. Only speeches are provided for the participants to listen and to answer what they hear. Once the collected data are enough, we analyze data by using WER corresponding to hypothesis (data collected from targeted participants) and the ground truth (the real texts of synthesized speeches). It is interesting to note that when analyzing French TTS

⁶<https://pypi.org/project/SpeechRecognition/>

⁷<https://pypi.org/project/jiwer/>

model with subjective approach, most of the words' pronunciations provided by participants correspond to the ground truth French text. However, the WER in this case is higher than that of comparing with Google ASR is because of the spelling variants of French word, for example, *Herton* vs. *Erton* vs. *Ayrton*.

Table 2: Results obtained from performing the experiments using ngram-based approach

Language	Participants (Native Speakers)	WER
English	10	0.55
French	15	0.42

7.2 Naturalness

For the naturalness of synthesized speech, we use *Mean Opinion Score* (MOS) to measure. It is a manual evaluation method that divides the naturalness of speech into five score segments: 1-5, with higher scores indicating better naturalness of speech. Each listener is assigned a unique voice, and each voice is played only once. The test data used by the system is the test data set from *The Voice Conversion Challenge 2018* (VCC2018), with a total of 36 English sentences. For each sentence, we invited three raters to score, and finally took the average of these three scores as the final result. The distribution of the length of test sentences is shown in figure 7.

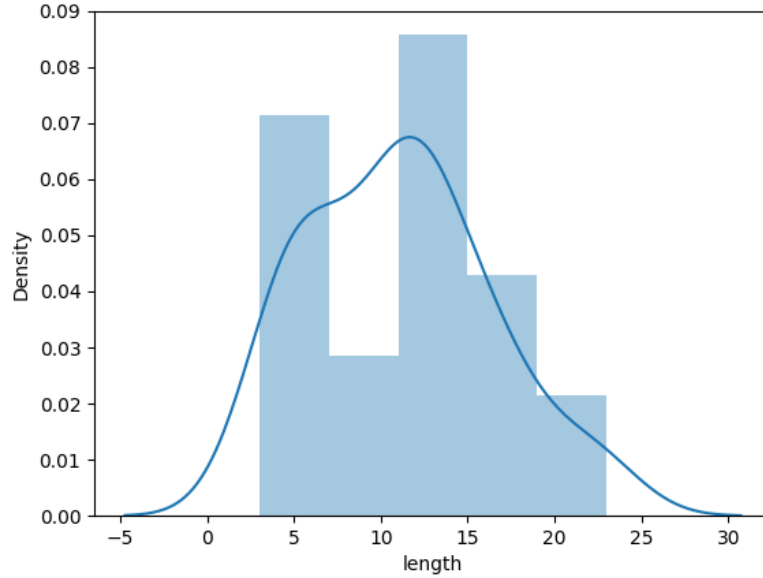


Figure 7: Length distribution

The final artificial MOS score is visualized in figure 8.

Through the correspondence between sentence length and *MOS* score, we found that there is no direct relationship between the performance of the model and the length of the sentence. This observation can be found in figure 9.

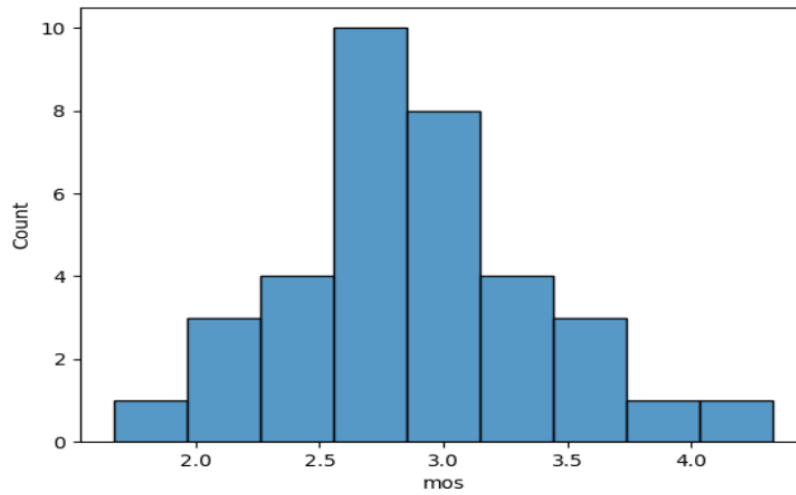


Figure 8: Final MOS distribution

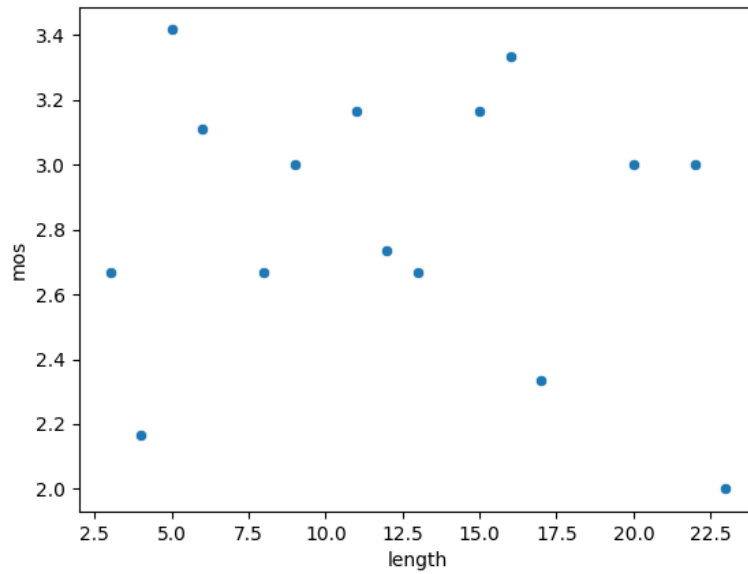


Figure 9: Relationship between length and MOS

Regarding a subjective evaluation of French model, we used online survey form to ask participants to rate the quality of naturalness of French speech synthesizer from 0 (less likely) to 5 (most likely). This is due to the fact that crowd-sourcing is costly, and we have spent some money on doing evaluation for English TTS model; therefore, we decided to collect data by ourselves by sharing it to our peer networks as we are currently living in France. Based on the data we have collected from 19 French native speakers shown below 10, it can be clearly seen the quality of naturalness of our French speech synthesizer is fairly good, meaning the sound quality is acceptable.

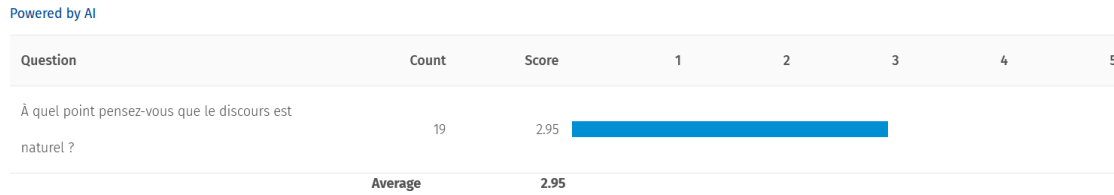


Figure 10: Opinion Mining Score from French Native Speakers for Naturalness of French TTS

8 Conclusion

We believe our work can serve as an easily extendable framework for future development. Currently, the project constitutes English and French models; however, it could be extended by more languages. One could either add a new speech synthesis model and train language information into the language classifier. This way, another language can be easily integrated into server’s functionality. As our project is publicly available online, and docker images are easily reproducible on any machine; the tool could be adopted in other works easily. Currently, English TTS model produces accurate speech while as per our results for French TTS model, on training to more number of epochs, could improve our results. As the models are pre-trained, fine tuning it on our own dataset will help in faster learning and will produce better results with small datasets.

References

- BOURLARD, HERVÉ et al. (2011). “Current trends in multilingual speech processing”. In: *Sadhana* 36.5, pp. 885–915. DOI: 10.1007/s12046-011-0050-4.
- Chen, Nanxin et al. (2020). *WaveGrad: Estimating Gradients for Waveform Generation*. arXiv: 2009.00713 [eess.AS].
- Dutoit, Thierry (2001). *An Introduction to Text-to-Speech Synthesis (Text, Speech and Language Technology, 3)*. Softcover reprint of the original 1st ed. 1997. Springer.
- Elias, Isaac et al. (2020). *Parallel Tacotron: Non-Autoregressive and Controllable TTS*. arXiv: 2010.11439 [cs.SD].
- Goodfellow, Ian et al. (June 2014). “Generative Adversarial Networks”. In: *Advances in Neural Information Processing Systems* 3. DOI: 10.1145/3422622.
- Honnet, Pierre-Edouard et al. (2017). *The siwis french speech synthesis database? design and recording of a high quality french database for speech synthesis*. Tech. rep. Idiap.
- Ito, Keith and Linda Johnson (2017). *The LJ Speech Dataset*. <https://keithito.com/LJ-Speech-Dataset/>.
- Kim, Jaehyeon et al. (2020). *Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search*. arXiv: 2005.11129 [eess.AS].
- Kong, Jungil, Jaehyeon Kim, and Jaekyoung Bae (2020). *HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis*. arXiv: 2010.05646 [cs.SD].
- Kong, Zhifeng et al. (2021). *DiffWave: A Versatile Diffusion Model for Audio Synthesis*. arXiv: 2009.09761 [eess.AS].
- Li, Naihan (July 2019). *Neural Speech Synthesis with Transformer Network* | *Proceedings of the AAAI Conference on Artificial Intelligence*. URL: <https://ojs.aaai.org//index.php/AAAI/article/view/4642>.
- Oord, Aaron van den et al. (2018). “Parallel WaveNet: Fast High-Fidelity Speech Synthesis”. In: *Proceedings of Machine Learning Research* 80. Ed. by Jennifer Dy and Andreas Krause, pp. 3918–3926. URL: <https://proceedings.mlr.press/v80/oord18a.html>.
- Patton, Brian et al. (2016). “AutoMOS: Learning a non-intrusive assessor of naturalness-of-speech”. In: *CoRR* abs/1611.09207. arXiv: 1611.09207. URL: <http://arxiv.org/abs/1611.09207>.
- Popov, Vadim et al. (2021). *Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech*. arXiv: 2105.06337 [cs.LG].
- Ren, Yi et al. (2019). *FastSpeech: Fast, Robust and Controllable Text to Speech*. arXiv: 1905.09263 [cs.CL].
- Rezende, Danilo and Shakir Mohamed (2015). “Variational Inference with Normalizing Flows”. In: *Proceedings of Machine Learning Research* 37. Ed. by Francis Bach and David Blei, pp. 1530–1538. URL: <https://proceedings.mlr.press/v37/rezende15.html>.
- Shen, Jonathan et al. (Oct. 2020). “Non-Attentive Tacotron: Robust and Controllable Neural TTS Synthesis Including Unsupervised Duration Modeling”. In.

- Siddhi, Desai, Jashin M., and Desai Bhavik (2017). “Survey on Various Methods of Text to Speech Synthesis”. In: *International Journal of Computer Applications* 165.6, pp. 26–30. DOI: 10.5120/ijca2017913891.
- Sohl-Dickstein, Jascha et al. (2015). *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. arXiv: 1503.03585 [cs.LG].
- Wu, Zhizheng, Oliver Watts, and Simon King (2016). “Merlin: An Open Source Neural Network Speech Synthesis System”. In: *9th ISCA Workshop on Speech Synthesis Workshop (SSW 9)*, pp. 202–207. DOI: 10.21437/ssw.2016-33.
- Yamamoto, Ryuichi, Eunwoo Song, and Jae-Min Kim (2019). *Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram*. arXiv: 1910.11480 [eess.AS].