Neural Speech Synthesis Using Transformers
Project Report

By

Sudheer Tati(015910674)

Paper Link: https://arxiv.org/pdf/1809.08895.pdf

Dataset: https://keithito.com/LJ-Speech-Dataset/

Code: https://github.com/sudheer997/Neural-Speech-Synthesis-Using-Transformers-

Problem Definition:

The problem I have implemented over the course of this project is based on Text-to-Speech (TTS). Text to Speech had gained a lot of prominence in various fields and had solved many real-world problems that we encounter in daily life. TTS has been a revelation right from the beginning, but it has become more powerful. With the advancement, we are able to generate intelligible and natural audios which are indistinguishable from human recordings. The main idea behind the project is to enable users to customize their lexicon to convey their thoughts, emotions, and requests in their day-to-day lives. It immensely helps users with disabilities and impacts the way they communicate by relying on this technology to express their ideas and share with those around them. There are huge applications of TTS in our daily life, Among them, the most popular ones would be in our personal voice assistants and we also come across this in various google products such as (Ok Google, Google Translate, and Google Maps).

Project Objectives:

The main objective of this project is to build a text-to-speech application, such that generated audios are indistinguishable from the human voice. The TTS system mainly focuses on the following objectives:

- For the TTS system input is Text, so first, we need to preprocess the text such as removing special characters, syllable boundaries, and punctuations are also included as special markers. Then generate phoneme sequences for text which are used for input for the model training.
- Preprocess the audio files such as normalizing the audio frequencies, trimming the audio to a specified length, and then extracting the Mel-spectrograms. Store the extracted Mel-spectrograms which are useful for the training of the TTS model.
- Design and implement the TTS model according to this paper. For this project, I have used this repo as a reference.
- Integrate HiFi-Gan vocoder[2] with the TTS model. HiFI-Gan vocoder is used to generate the high-quality audios from the Mel-spectrograms.
- Integrate Griffin-Lim vocoder with the TTS model. Griffin-Lim vocoder is also used to generate the audios from the Mel-spectrograms.
- Design and implement the TTS system that can read out the text at any speed rate that the user specifies.
- Finally, built web-app using StreamLit for this TTS system.

Analysis:

Data Analysis:

- Total Clips: 13,100
- Total Words: 225,715
- Total Characters: 1,308,674
- Total Duration: 23:55:17
- Mean Clip Duration: 6.57 sec
- Min Clip Duration: 1.11 sec
- Max Clip Duration: 10.10 sec
- Mean Words per Clip: 17.23
- Distinct Words: 13,821

Text Analysis:

- Remove any other characters other than Alphabets(ABCDEFGHIJKLMNOPQRSTUV WXYZabcdefghijklmnopqrstuvwxyzäüößÄÖ.') and special characters('!,-.:;? \'()'.)
- Instead of using general words as input to the model, phoneme sequences gives better performance. Because it is difficult to learn all the regularities when which is often the case, the training data is not sufficient enough, and some exceptions have too few occurrences for neural networks to learn.

Audio Analysis:

- Following preprocessing are done on the audio files which increases the model performance :
  - Normalize the audio frequencies.
  - Trim the long silence in the audios.
  - Generate Mel-spectrograms for the audio files. For the generation of Mel-spectrograms following hyper-parameters are used:
    - Sample rate: 22050
    - Number of mels frames: 80
    - preemphasis: 0.97
    - Minimum frame length: 0.05 seconds.

Transformer TTS Model Analysis:

- Transformer TTS architecture is same as Tacotron2[1] architecture but have some modifications on it.
- **Scaled Positional Encoding:** It is useful when inputs and outputs are on the different scale. So, in our scenario, texts and mel spectrograms may have different scales. So, scale-fixed positional embeddings may impose heavy constraints on both the encoder and decoder pre-nets so that the mapping between encoder and the decoder will happen correctly.
- **Encoder and Decoder:**
  - In this model, encoder and the decoder of Tacotron2 are replaced the LSTM with Transformers. By using transformers[5], it parallelize in training and Inference tasks and also it helps in learning long term dependencies very well[4].

- In the encoder and decoder of the transformer are originally composed of 6 layers and 8 multi-head attention. So by reducing the layers and heads, the training speed is increased but on other hand, it harms the model performance.

- **Encoder pre-net**: 3-layer CNN
- **Decoder pre-net:** 2-layer Fully connected network
- **Mel linear:** a fully-connected layer, generates mel spectrogram frames.
- **Stop linear:** a fully-connected layer, predicts the stop token for each frame.
- **Post-net:** a 5-layer CNN with residual connections, refines the mel spectrogram.

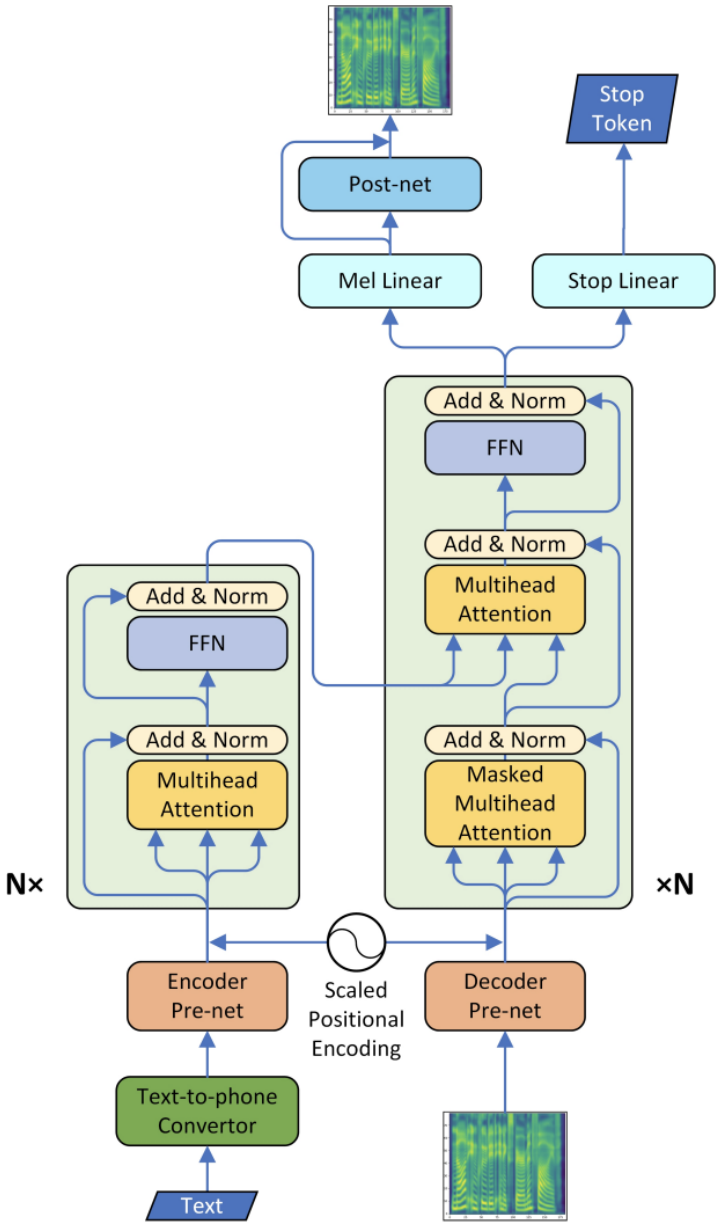
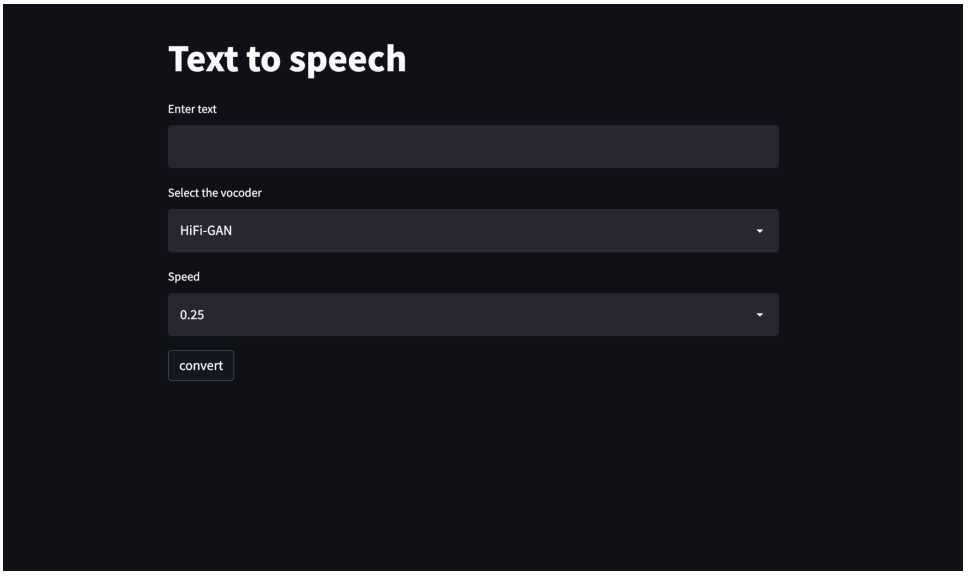


Figure 3: System architecture of our model.

Results:

- For this project, I have used the pretrained model, because training the model from scratch required a lot of computation power and training time taken by this model is approximately 25 hr(according the paper).
- Generated Audio Samples:
    - Example Text: Transformers were introduced in 2017 by a team at Google Brain and are increasingly the model of choice for NLP problems, replacing RNN models such as long short-term memory
    - Griffin-Lim Vocoder: sample audio ouput
    - HiFi-Gan Vocoder: sample audio output
- Screenshot of the Webapp:



- The above web-app screenshot contains Text area where text input for the TTS model entered, select the vocoder(HiFi-Gan and Griffin-Lim), and speed rate(0.25 to 2.00) of the generated audio file. After entering all the required fields then click on the convert. It will generate the audio and display over there.

Discussion:

- The training and Inference time is better compared to the previous models(Tactoron2) because of the transformers in encoder and decoder, which enables parallel training, and the capability of learning long-distance relationships in the text.
- After using the HiFi-GAN, it reduce the noise and static in the audio sample, audio quality is high and the voice is very similar to the human voice.
- Inference time using HIFi-GAN vocoder is very less when compared to the Griffin-Lim.
- Model generates audio samples quality is very closed to human voice recordings, and the audio prosody is much more smooth.
- Model is preforming good on the rare words, because of using phonemes instead of words, it using phonemes can able to learn the all regularites with less amount of data.

- After using transformers in the model architecture, the model can able to learn attention between encoder and decoder better than the RNN's structures. So, this helps in appropriate mapping the between the text and Mel-spectrograms.

Evaluation and Reflection:

- Audio quality generated by this models are very good compared to previous models (Tactoron & Tactoron2). This models achieved removing the noise completely but still produces minute static.
- Although the transformer has enabled parallel training, the model suffers from slow inference.
- The reason for the slow inference is mainly due to the dependency of the generation of current MEL frames on the previous MEL frames. Later in autoregressive models(Ex: One paper proposed by Fast.ai) this problem is solved.
- Using human like voice in IVR applications(chatbots) would incur confidence in among the service providers and the end users.

References:

[1] Shen, Jonathan & Pang, Ruoming & Weiss, Ron & Schuster, Mike & Jaitly, Navdeep & Yang, Zongheng & Chen, Zhifeng & Zhang, Yu & Wang, Yuxuan & Skerry-Ryan, RJ & Saurous, Rif & Agiomyrgiannakis, Yannis & Wu, Yonghui. (2017). Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions.

[2] Kong, Jungil & Jaehyeon, Kim & Bae, Jaekyoung. (2020). HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis.

[3] Sharma, Ankit & Kumar, Puneet & Maddukuri, Vikas & Madamshetti, Nagasai & KG, Kishore & Kavuru, Sahit & Raman, Balasubramanian & Roy, Partha. (2020). Fast Griffin Lim based Waveform Generation Strategy for Text-to-Speech Synthesis.

[4] The Illustrated Transformer – Jay Alammar – Visualizing machine learning one concept at a time." Jay Alammar, 27 June 2018, https://jalammar.github.io/illustrated-transformer/ Accessed 16 May 2022.

[5] Vaswani, Ashish & Shazeer, Noam & Parmar, Niki & Uszkoreit, Jakob & Jones, Llion & Gomez, Aidan & Kaiser, Lukasz & Polosukhin, Illia. (2017). Attention Is All You Need.

[6] Dobilas, Saul. "LSTM Recurrent Neural Networks — How to Teach a Network to Remember the Past." Towards Data Science, https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e