Text To Song Generative System

Project Vivy

Zhen Zhang
Department of Electric and Computer Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
zhenz@vt.edu

Coauthor
Affiliation
Address
email

Luis Vazquez Morales
Department of Electric and Computer Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
lvazquez@vt.edu

Coauthor
Affiliation
Address
email

Abstract

This paper proposes a novel text-to-song system. The system can generate a song that is entirely created, arranged, and sung by AI, based on user input such as lyrics and style. This research project focuses on the synthesis of singing voices, the text-to-singing-voice aspect, and develops a deep learning model named Vivy for this task. Vivy successfully synthesized the corresponding files as anticipated, and its generated results met experiment expectations. The system proposed in the paper has been preliminarily validated, making the design of text-to-song systems feasible and further advancing the application and development of AI in the field of art.

1 Introduction

In recent years, artificial intelligence has made significant strides in various domains, such as chatbot, self-driving, image generation, and music generation. However, in the field of art, especially in music, there is no generative model or multi-model system in research that can produce complete music from text. To accelerate the application of AI in the field of music, and to attempt to use AI to generate a complete piece of music, a text-to-song generative system is proposed in this paper.

In the project, the aim was to develop a complete text-to-song system where users can input lyrics, song style, BPM, and other information to obtain a fully AI-generated, complete piece of music with synthesized singing voice. This is to further accelerate the application of AI in the field of art. This novel project has not been implemented so far, thus was one of the reasons the project was chosen. Other reasons were to allow people who were previously unable to create music due to any lack of knowledge or skills can make the music they now desire. This would help accelerate new music and possible genres in the music industry due to the ease of making of music.

Due to the substantial complexity of this project and time constraints during the development process in this project, only the voice generation part of the project was able to be built. The rhythm generation and the creation of complete music will be implemented in the future.

2 Related Work

There has been no research paper so far that has tried to implement this project’s proposed idea of generating a singing voice through a lyric input. Though there are projects some projects that have generated a singing voice, most significant being one is called Diffsinger that used Diffsinger (.ds) file that contains phoneme sequence, phoneme durations, and music score parameters. Generating the .ds files to create the singing voice with the lyrics that the user wants is complex, which is something that attempted to simplify.

3 Methodologies

This project involves training a text to singing model, named Vivy. This section will introduce the overall structure of the project, as well as the approach used to train Vivy.

3.1 Project Structure

The project system consists of two parts: one is the voice generation system, and the other is rhythm generation system. Users will input information such as lyrics and style, which will then be fed into different systems for processing.

Multiple models are utilized to perform each task. For the lyric processing part, GPT, a Natural Language Processing (NLP) model, is chosen for processing. In the text-to-singing part, Vivy is developed for generation, and its output is sent to Diffsinger, a voice synthesis model based on shallow diffusion, for voice synthesis. For the rhythm synthesis part, the plan is to use musicGen, a rhythm generation model from audioCraft. The results will be sent to a note extraction model to extract all notes for Vivy's training. Simultaneously, this rhythm will also be combined with the generated voice to compose a complete song.

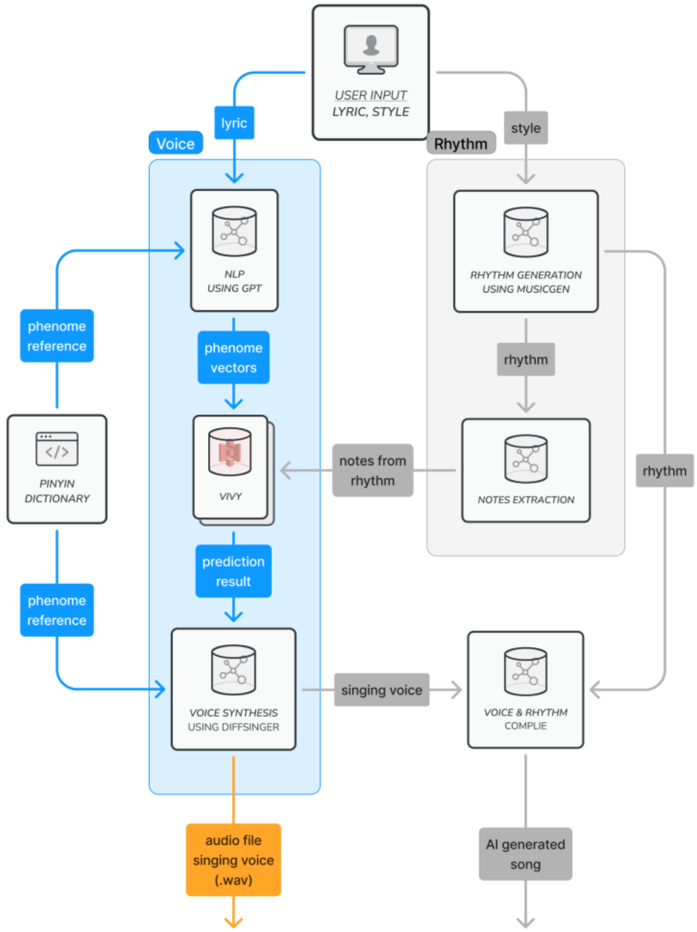

Fig 1. Project structure

3.1.1 Voice Generation

The user-input lyrics are processed through a NLP model. In this system, GPT is used for this purpose. Since the voice synthesis model employed is Diffsinger, A pre-trained model provided by them for voice synthesis. This pre-trained model is based on pinyin phonemes, hence the input for Diffsinger should be based on pinyin phonemes. Therefore, after feeding the Pinyin dictionary that is made by Opencpop into GPT, it first converts English into pinyin, then uses the pinyin-based English text to make predictions through Vivy. The results of these predictions are then further passed to Diffsinger for voice synthesis.

3.1.2 Rhythm Generation

The style information input by the user is sent to musicGen for rhythm generation. Then, a note extraction model is developed to analyze the notes within the rhythm and send these notes to Vivy for sound generation. The rhythm generation part has not been developed due to time constraints and will be the subject of further research in the future.

3.2 Data Collection

A total of 9.7 hours of singing text, 89315 entries were acquired, most of which came from Opencpop, with the remaining part from sample texts in Diffsinger. All texts are in the form of text (.txt) or Diffsinger (.ds) files, containing song phonemes, duration of each phoneme, vocal notes, and f0 time sequences. In the .ds files, they include all the elements.

In the txt files, there are phonemes, phoneme duration, and notes.

To train the model more efficiently, all the redundant parts in the .ds files, such as note duration, energy, breath time, and other numerical values, have been removed.

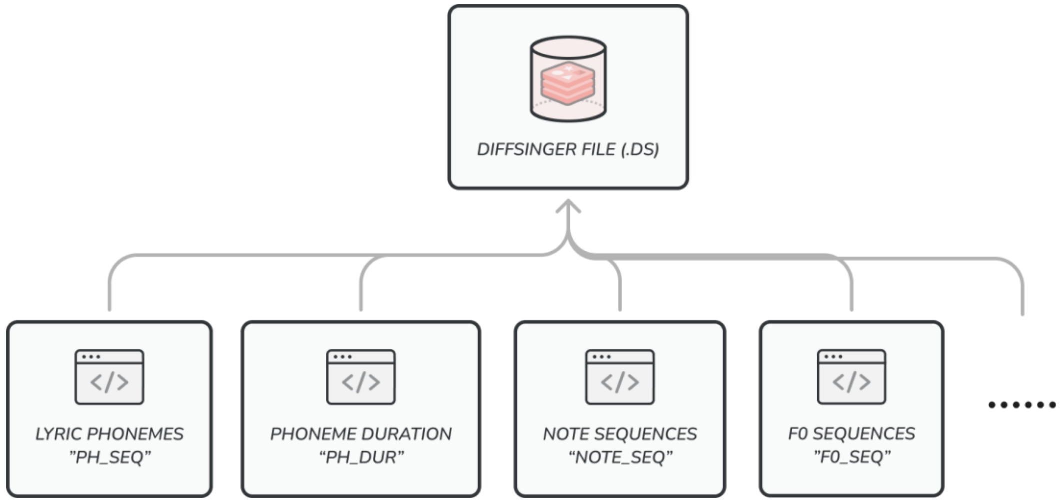

Fig 2. Diffsinger file structure

3.3 Data Processing

In this section, the process of handling all text data and normalizing them for model training is explained. First, convert the text files into Diffsinger files to facilitate batch processing later. Secondly, filter out all irrelevant numerical values, retaining only those essential for sound generation, such as ph_seq, which refers to all the phonemes of the lyrics; ph_dur, indicating the duration of each phoneme; and note_seq, which denotes the notes of each pronunciation.

3.3.1 Text to Diffsinger File

Since the training texts provided by OpenCpop primarily consist of text files and raw audio files, and because these texts from OpenCpop make up over 80% of all training samples, using their files for training can significantly improve the model's output. It is essential to prioritize converting them into .ds files for subsequent batch processing. After reading the text files, all texts are segmented and tagged, finally outputting them in the .ds file format and incorporating them into the dataset.

3.3.2 Data Filtering

All .ds files are read, and any data unnecessary for audio generation is removed. Additionally, this part also generates some statistical data to assist researchers in their analysis, such as the total count of each value and the total duration of each phoneme duration.

3.3.2 Normalization

After importing the organized filtered dataset, the ph_seq and note_seq are first converted into numerical values using integer encoding technology. Since notes in music follow a regular pattern, it's important for the model to learn this pattern. Therefore, before converting, a manually organized dictionary was prepared, in which all notes are sorted by pitch from low to high. In this project, 54 notes ranging from "C#2/Db2" to "A#5" are used for conversion. A special note, 'rest,' which represents a brief pause during singing, is assigned the number 55.

Subsequently, after obtaining the encoded results, to ensure uniformity in the length of all data, padding is applied to all data based on the length of the longest data in the set. This padding is done using the mean value.

3.4 Model Development

Due to the time-sensitive nature of this project, several models adept at learning sequences were selected for training. Two variants of Recurrent Neural Networks (RNN) were used: Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU). Additionally, the Transformer model was also employed for training. All models selected are simple standard models, with a training epoch set to 1000.

3.5 Testing and Evaluation

To train the model, the training and validation data are randomly split in a 70/30 ratio, using the validation data to check and analyze the results. Researchers hope the model will exhibit creativity, so no specific expectations are set for the model's predicted results. However, it is expected that the model should learn to pause at AP or SP, a type of rest symbol commonly found in Diffsinger, typically occurring between sentences or at the end, thereby generating a 'rest'.

4 Results

The results of the project were a model called Vivy that generated prediction results from phoneme vectors, a Pinyin Dictionary that converted English lyrics to Chinese phoneme, a framework that connected the Diffsinger model with the previous results to form the input to singing voice audio file, and three audio files generated by using three different model implementations.

The model implementation used were LSTM, GRU, and Transformer. Each model had 1000 epochs for the dataset to pass through the model. Each model was trained on the same computer, which is equipped with a Ryzen 9 7000 series 16-core CPU, 64GB of RAM, and an NVIDIA RTX 4080 graphics card. The duration and size for each model are described in Table 1.

Model Name	Training duration	Model size
GRU	44 min 40 s	15.98 KB
LSTM	47 min 33 s	19.86 KB
Transformer	1 hr 13 min 22 s	23.47 KB

Table 1. Prediction result

In this experiment, a line from Rick Astley's *Never Gonna Give You Up* was used: *Never gonna give you up, never gonna let you down.* This lyric was tokenized by GPT, resulting in: 'AP n ei f a g e n a j i f u y u a p u AP n ei f a g e n a l e y u d ao en AP'. The experiment aims to observe if the model can generate a reasonable sequence of notes and durations for Diffsinger to synthesize audio. Here, 'AP' represents a brief pause, and its corresponding note should be 'rest'

Since this study hopes that the audio generation model will be more creative, it did not set too many expectations. However, the trained model should correctly identify that a rest note should be output when an AP or SP symbol is encountered. After observing the outputs of three models, only the GRU model successfully predicted a rest note in the aforementioned situations. Both LSTM and Transformer models failed to make this prediction. Additionally, the duration of the GRU's output is closer to that of normal singing.

Based on listening to the output audio files for each model, it was concluded that the audio file from the GRU model gave the best result based on its tempo and flow throughout the audio file. The audio files from the LSTM and Transformer had a slow tempo and felt somewhat to drag on. These audio files were ranked based on the results shown in Table 2.

Model Name	Duration	Is phoneme duration making sense?	Is AP note predict correctly?
GRU	12s	Yes	Yes
LSTM	16s	Yes	No
Transformer	19s	No	No

Table 2. Prediction result

4 Conclusion & Discussion

This project proposes a novel text-to-song generation system, providing a viable solution for future developments in the field of art. By implementing a model that is able to generate prediction results from phoneme vectors were able to produce a working project that allows for lyric input to be used to produce a singing voice that allows normal people to generate a singing voice. By testing different implementation of the model the results showed that a GRU implementation produced the best results for a singing voice. Thus, a singing voice was able to be produced to show proof of concept that the common person is able to produce songs that they were previously unable to do so before.

4.1 Discussion

Due to time constraints on the project, the research team was only able to implement some of their ideas, and it was not possible to deliver the complete project design within the limited time. Additionally, due to the limited data resources collected, all training data was based on Chinese text, making it more challenging to generate English lyrics. This issue could be addressed by manually creating more English text data for more comprehensive training. To simplify the training process, many important aspects were omitted, such as the duration of notes and the sequence of f0. If these values are considered in further training, the results would be more pleasing to the ear.

4.2 Future Work

This project should continue to optimize the Vivy model to achieve better outputs, which includes training with the vocal and rhythm parts of existing audio files. At the same time, it's necessary to implement rhythm generation, extract the note sequence from the generated results, and use it as input for Vivy to predict. This way, the rhythm and vocals can be more in sync. Additionally, designing a user-friendly graphic user interface can further facilitate the general public in using this system for creative purposes, thereby further promoting the application and development of AI in the field of art.

Dataset, prediction examples and audio results can be found in project repository: https://github.com/nikmomo/Song-Generation-Model-with-Vocal-Project-Vivy

References

[1] "Music generation," Papers With Code, https://paperswithcode.com/task/music-generation (accessed Sep. 12, 2023).

[2] J.-P. Briot, G. Hadjeres, and F.-D. Pachet, "Deep learning techniques for Music Generation

-- A survey,” arXiv.org, https://arxiv.org/abs/1709.01620 (accessed Sep. 12, 2023).

[3] J.-P. Briot and F. Pachet, “Deep learning for music generation: Challenges and directions - neural computing and applications,” SpringerLink, https://link.springer.com/article/10.1007/s00521-018-3813-6 (accessed Sep. 12, 2023).

[4] J.-P. Briot, “From artificial neural networks to deep learning for music generation: History, concepts and trends - neural computing and applications,” SpringerLink, https://link.springer.com/article/10.1007/s00521-020-05399-0 (accessed Sep. 12, 2023).

[5] V. Kalingeri and S. Grandhe, “Music Generation with Deep Learning,” arXiv.org, https://arxiv.org/abs/1612.04928 (accessed Sep. 12, 2023).

[6] H. H. Mao, T. Shin and G. Cottrell, "DeepJ: Style-Specific Music Generation," 2018 IEEE 12th International Conference on Semantic Computing (ICSC), Laguna Hills, CA, USA, 2018, pp. 377-382, doi: 10.1109/ICSC.2018.00077.

[7] Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., & Wang, Y. (2021). Transformer in transformer. Advances in Neural Information Processing Systems, 34, 15908-15919.

[8] Liu, J., Li, C., Ren, Y., Chen, F., & Zhao, Z. (2022, June). Diffsinger: Singing voice synthesis via shallow diffusion mechanism. In Proceedings of the AAAI conference on artificial intelligence (Vol. 36, No. 10, pp. 11020-11028).

[9] Copet, J., Kreuk, F., Gat, I., Remez, T., Kant, D., Synnaeve, G., ... & Défossez, A. (2023). Simple and Controllable Music Generation. arXiv preprint arXiv:2306.05284.

[10] Wang, Y., Wang, X., Zhu, P., Wu, J., Li, H., Xue, H., ... & Bi, M. (2022). Opencpop: A high-quality open source chinese popular song corpus for singing voice synthesis. arXiv preprint arXiv:2201.07429.