Arranging an Audio Track to other Genres by using CycleGAN-based Deep Learning Model *

Alex DongHyeon Seo
dseo22@wisc.edu

Hyecheol (Jerry) Jang
hyecheol.jang@wisc.edu

Stella Kim
ykim736@wisc.edu

Abstract

Changing the genre of a song is one of the methods used when compositing music. To the best of our knowledge, musicians usually add their new ideas to the song, while trying to keep most of the special characteristics of the original song when arranging music. Similar to other artistic tasks that require human creativity, converting the genre of a song takes a significant amount of time and effort. In this project, we propose a method to translate a music genre by using machine-learning, which can generate a new song with comparably less amount of time than humans. Specifically, we utilized cycleGAN based model to translate a soundtrack to another soundtrack.

Due to the complexity and difficulties of dealing with audio data, our model is able to handle files written with MIDI (Musical Instrument Digital Interface) specification only with specific characteristics. In the near future, we expect to expand our project to use regular audio files rather than MIDI to do our tasks to generalize our model. By doing so, we hope our model to be used for the general public without further modifications.

1. Introduction

Recent advancements of Artificial Intelligence grant computers the abilities to mimic the noble creativity of the most intellectual lives. One of the advancements is computer-vision which allows computers to understand images just like human beings. However, unlike images, music has not been studied as extensive in the field of deep learning. Thus, we thought it would be interesting to do a project using data that represent audio.

Today, we are given many resources to share and access to all different types of music through apps like Spotify or YouTube. Due to those music apps and newly invented comfortable ear buds such as airPods, many people seem to be always listening to music on a daily basis. People listen to music while studying, driving, working and doing all other different types of tasks. Because of such increasement in number of music listeners and people who are interested in music, musicians are always trying hard to provide new musical experience to the listeners. One of the ways that musicians create a song is changing a song to a different style of music while keeping the main characteristics of the original song. This is different from creating a totally new song from scratch and this way of changing a song to a new song by adding some new ideas is called music arrangement. Music arrangement can be done differently by making some different changes and one of the changes that could be made to a song is the genre. Only changing the genre of a song while keeping the main theme of the song might sound trivial but it is actually a very difficult task which takes quite a long time and much effort. Therefore, we would like to see if changing the genre of music could be done using machine-learning methods.

Our goal is very similar to AmazonWeb Service's DeepComposer [1], a keyboard that allows people to create a new song or arrange to a different genre with a simple combination of notes of their own. DeepComposer is based on deep learning models using generator and discriminator to update the music. Slightly different from DeepComposer, our project will focus on transferring the genre of the song to the user's desired genre.

*Project proposal for Spring 2020, University of Wisconsin-Madison STAT453 Introduction to Deep Learning and Generative Models course (Instructor: Sebastian Raschka);
All authors contributed equally

[1] AWS DeepComposer Product Description Page: https://aws.amazon.com/deepcomposer/

2. Literature Review / Backgrounds

2.1. Audio Files

2.2. Model Architecutures

2.3. Baseline Code

3. Proposed Method

4. Experiments

4.1. Dataset

Finding a large amount of music having paired labels(the arranged/mixed music based on the same track) is very challenging and even considered as impossible task, since there is a limited number of arranged tracks for each music(with high possibility, there is no arranged track). To use cycleGAN [2], we at least need to have unpaired labeled data. More specifically, we need music sources which have genre information as their label. Also, because the music industry is one of the most sensitive markets toward copyright issue, it is also important to only use the music tracks which do not have any restrictions on the usage for research purposes.

For this project, we collected MIDI files of different genres, including Classical, Pop, Rock, and Jazz. We first tried collecting MIDI files of all four different genres from the same database to ensure the fairness and equality across all the files; however, we could not find a good database that has good MIDI files of all the four genres we intended to use for the project. Thus, we allowed ourselves to collect the files of each genre from different databases. At the end, we collected MIDI files of classical music from mfiles[2] while MIDI files of Pop and Rock are from midiworld [3]. However, we had some difficulties finding a good set of MIDI files of Jazz music so we used the same data with Brunner et al. [1]

For classical, pop and rock MIDI files, we did data crawling since those files are from websites. To do data crawling, we first used import.io, [4] which is a web application that gets all the text contents of a webpage including links. Then we used R to cut off all the unnecessary text data we got from import.io to only select the download links for files. After cleaning up the data, then we used for-loop in R to go through the iterations and download all the files that are linked to addresses that we found earlier. For Jazz, we simply downloaded the files from GitHub.

[2]Classical Database: https://www.mfiles.co.uk/classical-midi.htm

[3]Pop Database: https://www.midiworld.com/search/?q=classic
Rock Database: https://www.midiworld.com/search/?q=rock

[4]Import.io product website: https://www.import.io/

4.2. Software

The baseline code was written in TensorFlow 1.4, whch is very out-dated. We updated code so that it is at least runs without any warning or errors on the latest TensorFlow 1.x. As of now, though the TensorFlow 1 is outdated, we decided to use TensorFlow 1 as we are more familiar to.

To read and manipulate MIDI files properly, we utilizes pretty_midi [5] and Pypianoroll [6].

4.3. Hardware

For local testing before uploading the code and start train our model, we used Dell XPS 9550, which equipped Intel's i7-6700HQ @ 2.6Ghz, DDR4 2133Mhz, and NVIDIA GeForce GTX 960M with vRAM of 2GB.

After testing locally, for our main training, we used Google's Colaboratory Pro. We used standard instance (runtime), which have 12GB of RAM with NVIDIA NVIDIA Tesla P100 Graphic Cards having vRAM of 16GB.

5. Results and Discussion

6. Conclusions

7. Acknowledgements

Our project is part of Spring 2020 semester's Statistics 453 [7] (Introduction to Deep Learning and Generative Models) course of the University of Wisconsin-Madison. We specially appreciate with Prof. Raschka for all of his effort toward lectures and support toward our project.

8. Contributions

All of our team member decide to work on all steps and tasks together, as we believe it is important for everyone to at least understand how and what we are doing to achieve the common goal. However, as each individual has his/her own specialties and strength, some tasks has been designated to specific person.

As Hyecheol (Jerry) Jang has strength on understanding and fixing the codes written on python, he mainly worked on analyzing and fixing the baseline code. To do so, he went through the tensorflow API to understand what's going on inside the code and to remove depressed and unsupported code parts from the baseline code.

[5]pretty_midi Documentation: http://craffel.github.io/pretty-midi/

[6]Pypianoroll Documentation: https://salu133445.github.io/pypianoroll/

[7]Course Website: http://pages.stat.wisc.edu/~sraschka/teaching/stat453-ss2020/
GitHub Repository: https://github.com/rasbt/stat453-deep-learning-ss20

Alex studied similar studies and papers that are about creating music in various ways. He did research on the papers to find out which method worked and why specific methods worked better than the others. Also, he reviewed different models to find out which model would fit the best for the purpose of project.

Stella worked on finding the right MIDI files for the project. She went through various websites and databases to select the correct MIDI files that could be used for the purpose of our project. Also, she worked on understanding the code of the baseline model for the data processing part, to understand why the data was processed in such way, she had to study the MIDI file characteristics and music as well.

Even though we had some specific tasks assigned to each person, overall, as mentioned above, we helped each other and worked togehter on most of the tasks. Also, we divided the work on the presentation and the report evenly as well.

9. Codes

All the codes are posted on the GitHub repository. [8] The repository ipynb file that used as an interactive workspace to run the code. All the data preprocessing codes are embedded on the notebook as we want to run those code only when needed. The model has been implemented in the python files, and it is called inside the notebook.

References

[1] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao. Symbolic music genre transfer with cyclegan. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 786–793. IEEE, 2018.

[2] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[8] https://github.com/hyecheol123/CycleGAN-Music-Style-Transfer