Video Game Music Generation Using LSTM

Yousif Kako
yousif.kako@temple.edu

Faraz Heravi
faraz@temple.edu

Dr. Andrew Rosen
andrew.rosen@temple.edu

Abstract

Our goal is to be able to build a generative model from a Long-Short Term Memory (LSTM) in Recurrent Neural Networks. Previous work has been done using the same class of algorithm for different types of music such as classical music. Our approach was to perform music generation using LSTM on video game music. In particular we tried to narrow our goal down to generate music given midi files from a specific video game.

1 Introduction

With the rise of video games, video games music gained popularity alongside. Games such as *Super Mario Bros* and *Undertale* have notable music which is known and recognized worldwide. So the question that surrounded us was that can we resemble the great works of composers such as Koji Kondo and Toby Fox and create new music out of it using the power of a computer?
We were inspired by a Youtube video called "Computer evolves to generate baroque music!" that used an LSTM in order to reproduce the famous composer's work. Compared to classical music, video game music is much more limited due to the following reasons:

1. Each video game has a limited time of music compared to the works of classical composers.
2. Each piece of music is much shorter in length.
3. A lot of video games' music contains leitmotif, a recurrent theme, throughout the different pieces of tracks.

As a result we wanted to build a generative model that can effectively express the harmony and melody of video game music. Resemblance of these great pieces of music and usage of them towards new music would be a successful achievement.

2 Data

In order to train data and use a machine learning algorithm, we decided to use MIDI files. The reason behind it was that MIDI files can be converted into text. As a result, the usage of LSTM and Recurrent Neural Networks becomes effective since they work well on text generation.

2.1 MIDI

A MIDI files contains a series of concurrent tracks. Each track has messages and meta messages which refer to the notes and their duration. We used the music from the two video games *Super Mario Bros 2* and *Undertale*. The MIDI files for *Super Mario Bros 2* were obtained by scraping and downloading them from `www.vgmusic.com` using a Python script developed by ourselves. We downloaded the MIDI files for *Undertale* from the following youtube video: `www.youtube.com/watch?v=n138Qs-pvb8`.

2.2 Preprocess

In order to preprocess data, we used the Python library 'Music21'. The library provides an interface to acquire the musical notation of MIDI files. Additionally, it allows to create Note and Chord objects in order to make MIDI files.
The notes in MIDI files contain information about pitch, octave and offset. Music21 allowed us to collect these information and encode it into a sequence. Note that the duration of each song had to be appended at the end of the string representation of each Note and Chord.

3 Method

To approach our goal, we decided to use an LSTM model. LSTM Recurrent Neural Networks work can generate sequences simply by predicting one data point at a time. Other works such as text generation, handwriting prediction, and even music generation have used LSTM and have been successful. One example is Project Magenta which is a product of Google AI.

3.1 Super Mario Bros 2

Super Mario Bros 2 was our first shot at training a model with LSTM. We trained the soundtracks with the total length of 47 minutes and 47 seconds. After training, we generated 7 pieces of music: 1 with 100 epochs of training, 1 with 120 epochs, 3 with 143 epochs, 1 with 150 epochs, and 1 with 190 epochs.

3.2 Undertale

For *Undertale*, the training went more in depth as we tried to analyze different epochs and batch sizes. We trained a total of 2 hours, 30 minutes, and 7 seconds of *Undertale* soundtracks and we generated a total of 392 tracks. We trained on two models A and B. Some of the layers differed between these two models. We trained both of the models on batch size 128 and 1280. The music was also trained with a range of 10 to 921 epochs and was generated based on the sequence lengths [1] 50, 75, 100, and 150.

4 Results

After listening to all the tracks generated by LSTM, the results varied among the generation. Even generating music with the same number of epochs, sometimes the music turned out to be really great and sometimes not that desirable. Out of the 7 tracks generated from training *Super Mario Bros 2*, 2 tracks were really unique and melodically beautiful.

Since we generated many more tracks based on *Undertale* training, the results varied a lot. But in general, many of the melodies generated were similar to each other and sounded like *Undertale's* music. One reason could be that *Undertale* contains leitmotif consistently throughout its soundtracks.

We figured out that when we trained with really a low number of epochs[2], the results sounded basic and not musically pleasing. When the number of epochs is high compared to the data length, the model would sound pleasing, but at the same time it would be copying many melodies and structures of the original soundtracks[3]. We do not want to copy the original soundtracks as that would make music generation purposeless. But by finding the right number of epochs for each dataset training, the results will be pleasing to the ear.

We developed a command-line based program where it is possible to input the directory of the data, number of epochs, batch size, and number of generations. As a result, generating music will be simple using this project. Of course, the computer needs to have the power to train the data. All the results we produced and the project source code is available in a Github [4] repository.

5 Discussion

5.1 Evaluation

One of the challenges of this project was the evaluation of music generated. There is no direct way to measure how good a music is generated based on a given data because there is no accuracy score. A piece of music is good when it is pleasing to ear. For this project, we only wanted to take a look at how closely can we resemble the great work of video game composers. As Dr. Rosen is a musician himself and has listened to many video game music, he was able to determine the quality of the music generated. Based on Dr. Rosen, our model did produce music that is pleasing to ear. The pieces might not be a resemblance of the great work of the composers, but they surely have the potential to be so.

5.2 Emotions

The question of how closely can generated music resemble human emotion has always existed. Many argue that music created by artificial intelligence does not contain human emotions. We will discuss why this would not be an issue in our project. The pieces of music generated by the model varied even amongst the pieces with the same amount of training. Many of the pieces had some parts that definitely contained flaws and repetition which took the song out of its course. As a result, we believe that this model is still not ready to entirely produce music by itself although it can create very pleasing sounds to ear.

5.3 Copyright

While using this program, one must be careful of the line between copying and creating original music. If the model is overfit, the probability of the piece being similar to original tracks is higher. If anyone plans

[1] Sequence length refers to the length of the randomly selected data points from the original data. The sequence gives the model a starting point such that it can build off of it. We used the original data to generated music, but it is possible to start with a custom sequence.

[2] Of course the number of epochs depends on the length of the data. If the data is less, then the number of epochs required for training will be less.

[3] In other words, the model will be overfitted. To prevent this, we need to choose the right number of epochs.

[4] The Github repository can be found in `https://github.com/f4r4z/MusicGenerator`.

on using this project, he or she has to keep in mind that at some point the model might generate samples from original soundtracks. This was clearly visible in our *Undertale* music. Many generated pieces sounded the same, and they all took samples from the original tracks. Sometimes the tempo was much higher or lower, therefore this was not obvious, but other times it was.

6 Conclusion

We were able to show that an LSTM model applied to MIDI data is capable of generating music that is at least comparable to the original tracks. This means that they pieces may not be able to fit right into the production line, but there were pieces of songs that sounded surprisingly good. These sections can give composers very good ideas in the same style as the original data. Sometimes really beautiful melody was created, or the original tracks were tweaked to create new orders of notes.
Given the opportunity, we hope that we can build a model that can separate tracks and create full songs just based on the original data. We also hope that we can perform blind experiments in order to evaluate different hyperparameters for future work. As of this paper, the project can be used to create surprisingly good melodies from training video game or possibly other genres MIDI files.[5]

References

[1] Carykh. Computer evolves to generate baroque music.
https://youtu.be/SacogDL_4JU

[2] Sigurour Skúli. How to Generate Music using a LSTM Neural Network in Keras. *Towards Data Science*.
https://tinyurl.com/y8dapg5u

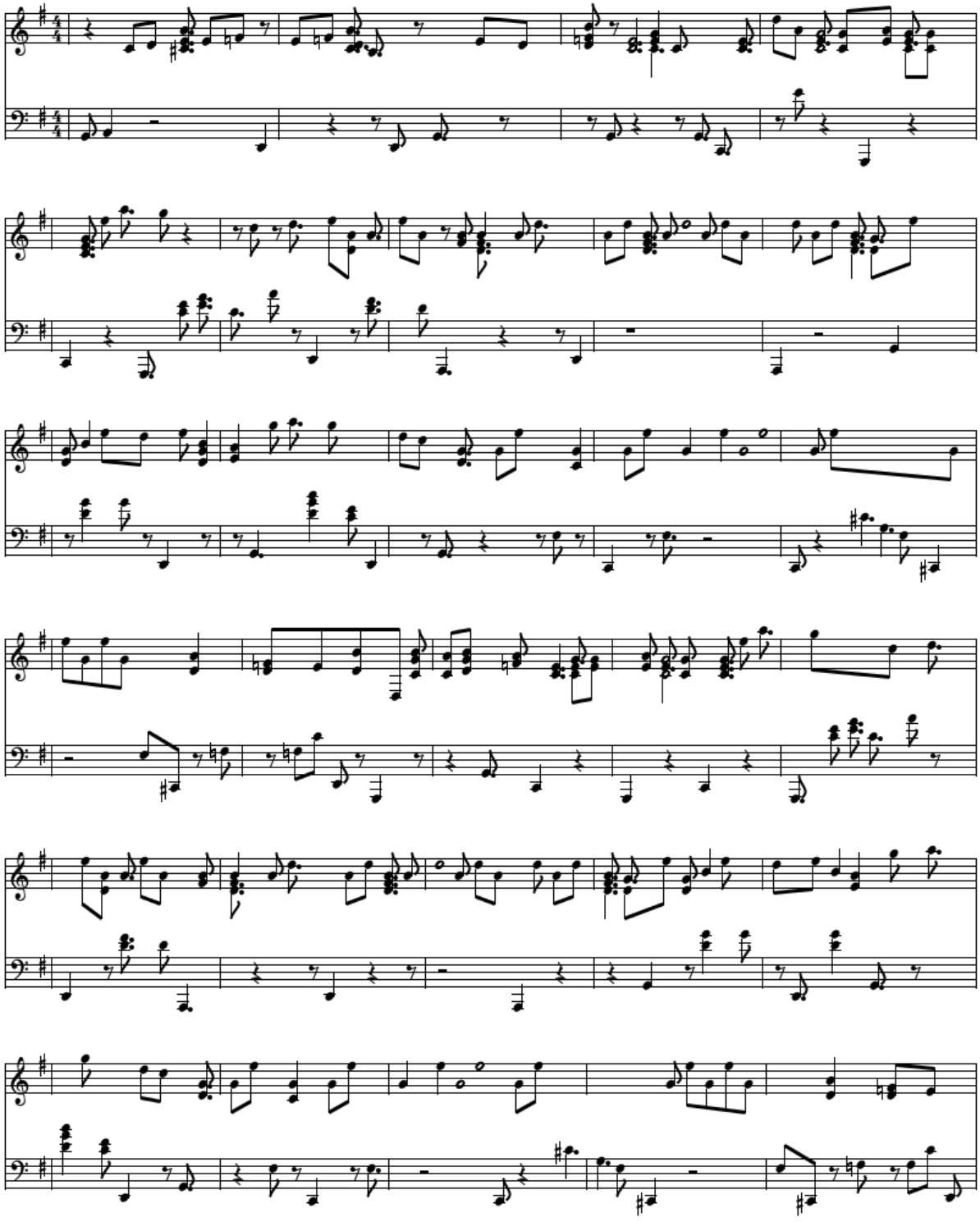
[3] Alex Issa. Generating Original Classical Music with an LSTM Neural Network and Attention. *Towards Data Science*.
https://tinyurl.com/y9eetsal

[4] Allem Huang and Raymond Wu. Deep Learning for Music. *Stanford University*.
https://cs224d.stanford.edu/reports/allenh.pdf

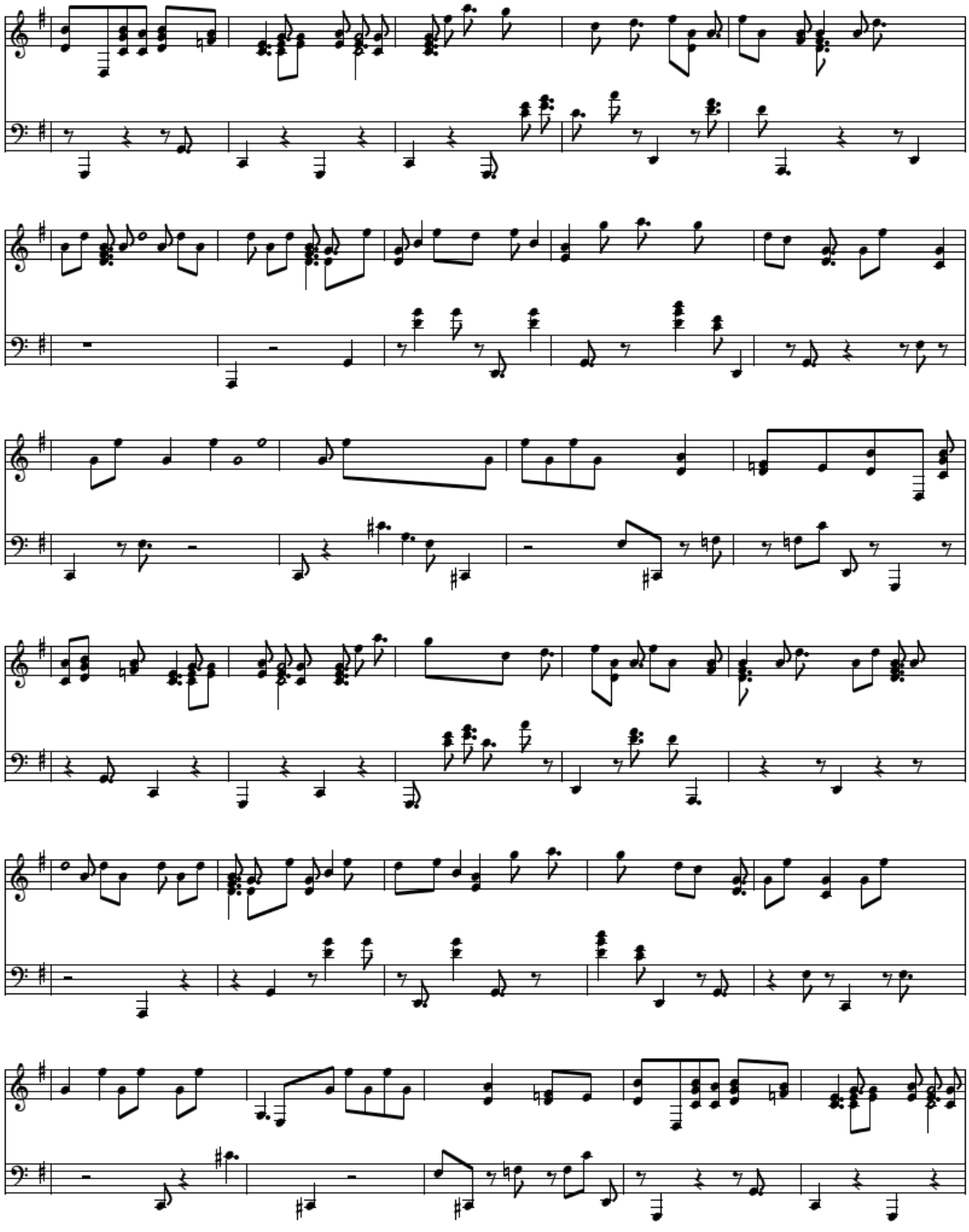
[5] Alex Graves. Generating Sequences With Recurrent Neural Networks. *Department of Computer Science University of Toronto*.
https://arxiv.org/abs/1308.0850

[5] Attached is one of our favorite pieces of music created by our program trained on *Super Mario Bros 2*. You can find the MIDI file on the Github page.

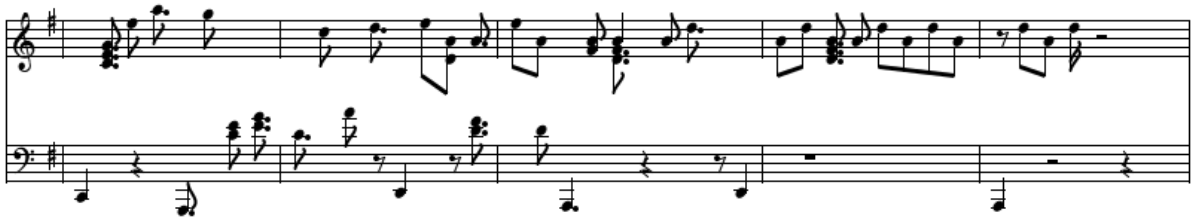
1587766991 Fi6kdDPwekqW9ngr7K0E



1



2



3