APS360 - Applied Fundamentals of Machine Learning

Project Proposal

Beat Saber Map Generator

Submission Date: 12th February 2021

Group 57:

Vedant Prajapati	1005137610
Kyle Blackie	1004831113
Smit Patel	1004734497
Hayden Goodfellow	1004068386

Document Word Count (Excluding Cover Page, Table of Contents, Figures and References): **1120**

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING

Table of Contents:

1.0 - Introduction

The recent emergence of virtual reality technology has brought about whole new levels of immersion in modern gaming. One game that makes effective use of this technology is Beat Saber.

Beat Saber is a VR rhythm game that allows players to hack and slash through blocks, avoid bombs and dodge walls all in time with the music.

These rhythm focused game mechanics combined with the immersion provided by VR provide for a refreshing and exciting experience.

The only problem with Beat Saber is that once you've played through the most popular songs, you begin to search for other songs to play. Once you do this, you quickly come to realize that a lot of your favourite songs have not yet been made into a Beat Saber map.

Building the map for a song yourself can be very time consuming and even then, you might not be capable of creating one that will live up to your expectations set by the more popular map creators.

But what if we were able to create these maps with the simple click of a button.

This is where our project comes in. Through the power of machine learning we will develop a model that allows a user to load any song of their choosing into the model which will then output a fully functioning, high-quality Beat Saber map ready to play.

This task of generating a custom map using a song's audio data is a feat that would be extremely challenging for an ordinary program to accomplish however, with machine learning, this task becomes very feasible. Machine learning allows us to train a model with existing data of maps created by community members. The model will learn various patterns between the song data and the layout of the maps to generate its very own maps from new song data.

2.0 - Illustrations/Figures

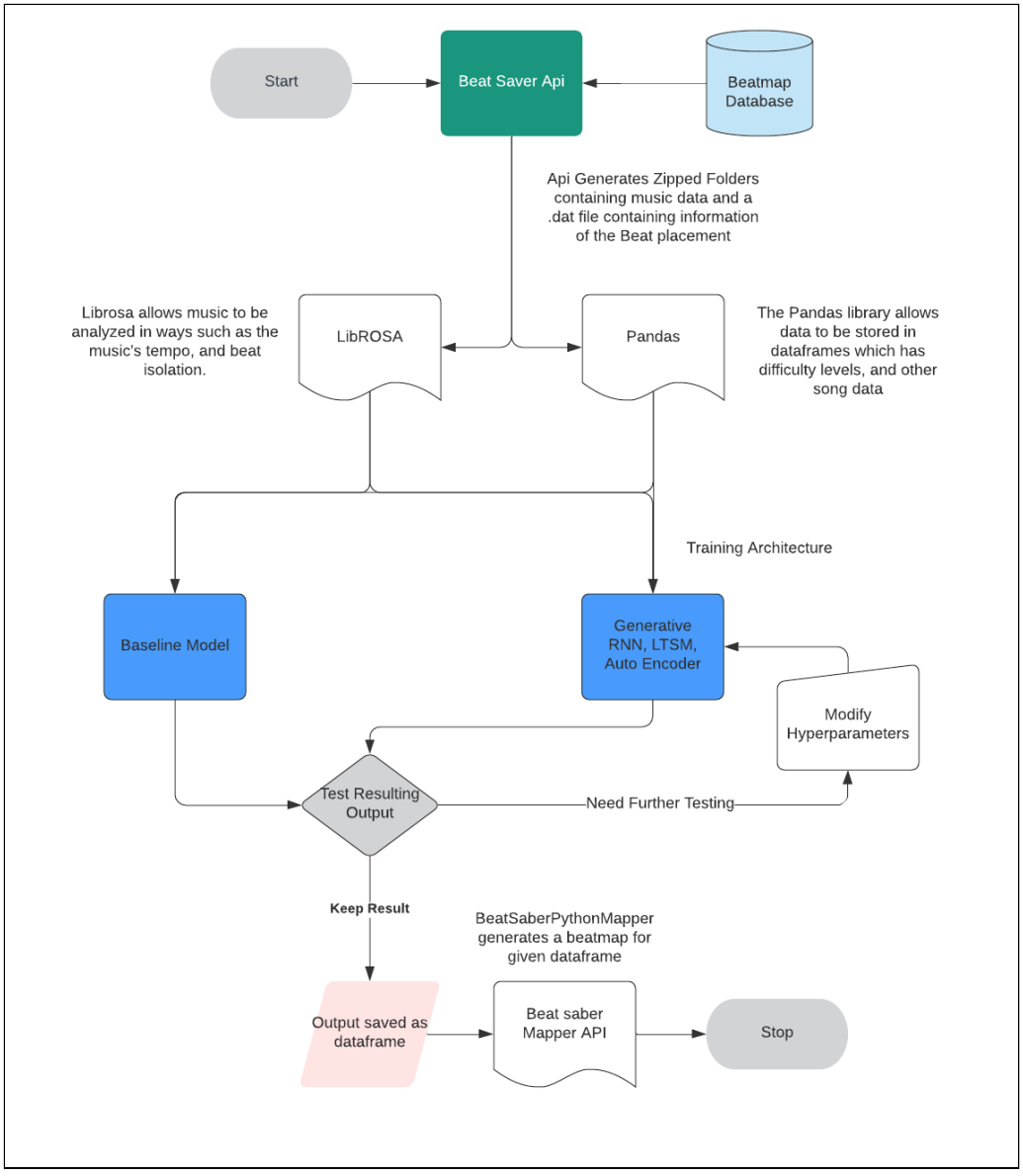

Figure 1 - Beat Saber Map Generator Model

3.0 - Background and Related Work

There currently exists multiple different Beat Saber automatic map generators, both commercially and free[1][2]. The main commercial generator, Beat Sage, is generally considered to be the best among the map generators[3]. It is different from many of the free generators in that it maps bombs and walls in addition to the usual notes[1]. Furthermore, it utilizes multiple neural networks to accomplish this: one for determining the times blocks/bombs/walls should be placed and one for determining

which types of blocks/bombs/walls should be placed at those timestamps[3]. Arguably the largest free project is the map generator created by the Oxford Artificial Intelligence Society, Deepsaber[2]. Well this project has less features than Beat Sage, it is still a fairly popular mapping tool thanks to it being completely free.

4.0 - Data Processing

The main source for our data will be from one of the largest online Beat Saber custom song database websites, BeatSaver, which contains over 20,000 custom songs[4]. Specifically, we plan to utilize their BeatSaver Reloaded API to download custom songs which meet certain parameters[5]. For instance, we don't want to train our models on songs which are extremely short "joke" maps. Additionally, we want to ensure our model is only learning on maps created by humans and not other machine learning models. By filtering our training data with a minimum song duration, requiring it be made by a human, and requiring a minimum user rating (e.g. 70% positive), we can ensure that our model is only being trained on high-quality, human-created, custom maps.

Once we have the maps downloaded, we will then utilize the LibROSA library to have it isolate the beat timings as these times would be where we want our model to place blocks. Furthermore, we will parse the map data files into a Pandas dataframe so it contains all the information on note placements. This data is what we will feed into our models.

Finally, once the models are trained we will have a set of songs that the models have not been trained on which we can use to test our models and tune our hyperparameters.

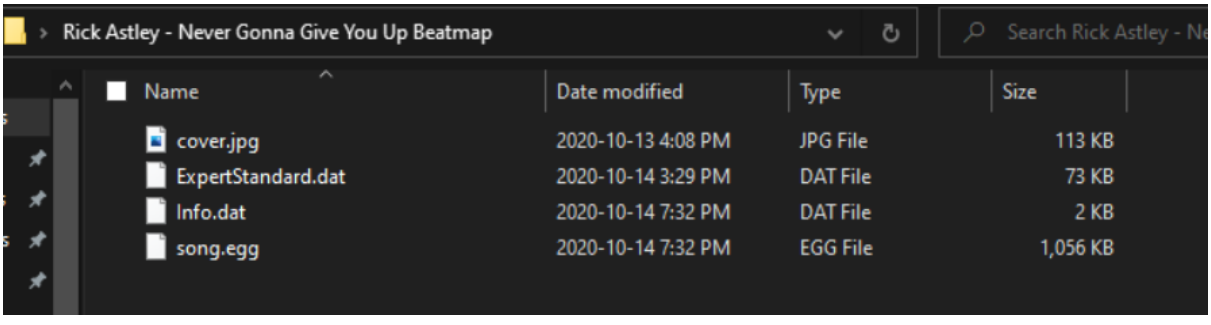

Figure 2 - Beatmap Zipped Folder

Figure 3 - Beatmap Info.dat File (Stores Song Information)

Figure 4 - Beatmap Block Placement .dat File (Stores Block Locations)

5.0 - Architecture

Due to the nature of our project, we need to use a model that is capable of learning from lots of pre-existing data and identifying patterns to develop new maps given music data.

After some research we believe that a generative recurrent neural network (RNN), more specifically a generative long short term memory network (LSTM) would best fit our needs.

This is because LSTM models are capable of learning the general structure of our data and when given new input, will be able to generate an output using what was learned from the training data. [6]

6.0 - Baseline Model

For our baseline model, we have decided to use a random forest classifier from the sklearn library. Along with this we will also be using sklearn's classifier chain. This baseline model will be used to map notes onto fixed beat timings which are determined by the LibROSA library. As there is no note rate modulation, this model will not output high quality songs; however, it will help with benchmarking our more advanced model's note placements.

7.0 - Ethical Considerations

This project primarily deals with music and the Beat Saber Map that is generated by our model. Some possible ethical concerns with building this AI are listed below:

- Fraudsters could potentially advertise to the Beat Saber community claiming to create manually built beat maps while using our model. This issue could be exacerbated if the fraudsters attempt to earn commissions for creating custom maps of a buyer's choosing. This would obviously be false advertising as the buyer would believe that they were receiving a manually designed beat map while only receiving what our model created. This is a major concern as commissioning for maps to be created is currently very commonplace and passing off a machine-generated map as custom-built would be unethical.

8.0 - Project Plan

Week of:	Tasks	Attributions
Feb 15 **(Reading Week)**	- Research LTSM and Unsupervised Learning	Hayden
	- Implement song downloading	Smit
	- Parse songs using LibROSA for beat timings	Vedant
	- Get song data into Pandas dataframe	Kyle
Feb 22	- Code baseline model	Hayden
	- Begin training baseline model	Smit
Mar 1	- Adjust hyperparameters for baseline model	Vedant
	- Finish training baseline model	Kyle
	- Test performance of baseline model	Hayden
	- Begin coding main model on block timing	Smit, Hayden
Mar 8 **Progress Report Due (5%)**	- Finish coding main model on block timing	Vedant, Kyle
	- Begin training main model on block timing	Hayden
	- Begin coding main model on note types	Smit
	- Create Project Progress Report	Vedant
Mar 15	- Test model on block timing	Hayden
	- Finish coding main model on note types	Kyle
	- Begin training main model on note types	Hayden
Mar 22	- Test model on note types	Smit
	- Begin testing of overall model	Kyle
	- (Optional) implement sequence repetition for chorus etc.	Vedant, Smit, Kyle, Hayden
Mar 29	- Get feedback on model performance	Vedant

	- Adjust model based on feedback	Hayden
	- (Optional) implement rate modulation	Vedant, Smit, Kyle, Hayden
Apr 5 **Project Presentation (10%)** **Final Deliverable Due (20%)**	- Prepare for Project Presentation	Vedant, Smit, Kyle, Hayden
	- Finish final deliverable for project by fine tuning model	Vedant, Smit, Kyle, Hayden

Our team will communicate and conduct weekly/bi-weekly meetings via Discord to follow up on internal deadlines and accomplish the tasks without procrastinating.

How we plan to avoid overwriting each other's code:

Our google collaboratory is shared, and we have a github repository storing the code so that there is version control and no mistakes when modifying code.

9.0 - Risk Register

It's important to account for potential risks that can arise through the course of this project.

Firstly, we will make sure as a team that every individual updates each other on the work they did in the weekly/bi-weekly meetings. This way, if someone drops the course mid-way, the others would be able to pick up with ease.

If the model training takes much longer than expected on Google Collaborate, we plan to use a GPU to train the model. We have a GPU at hand.

10.0 - Github

Github Link: [vedantprajapati/Beatmap-Generator-APS360-Group-57 (github.com)]

As our project progresses, we will include all files in above github link.

11.0 - References

[1] “Custom Beat Saber Level Generator,” *Beat Sage*. [Online]. Available: https://beatsage.com/. [Accessed: 13-Feb-2021].

[2] Oxai, “oxai/deepsaber,” *GitHub*. [Online]. Available: https://github.com/oxai/deepsaber. [Accessed: 13-Feb-2021].

[3] H. Baker and J. Feltham, “AI Tool Turns Any Song Into Custom Beat Saber Map, And It Really Works,” *UploadVR*, 18-Aug-2020. [Online]. Available: https://uploadvr.com/beat-sage-ai-beat-saber-custom/. [Accessed: 13-Feb-2021].

[4] BeatSaver. [Online]. Available: https://beatsaver.com/. [Accessed: 13-Feb-2021].

[5] BeatSaver Reloaded. [Online]. Available: https://docs.beatsaver.com/. [Accessed: 13-Feb-2021].

[6] J. Brownlee, “Gentle Introduction to Generative Long Short-Term Memory Networks,” *Machine Learning Mastery*, 14-Aug-2019. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-generative-long-short-term-memory-networks/. [Accessed: 13-Feb-2021].