A Mini Project Report

on

"Automatic Generation of Music"

Submitted in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY (HONOURS)

IN

COMPUTER SCIENCE (DATA SCIENCE)

Submitted by

Mithun G
19BTRCR006

Myil Vaughanan V L
19BTRCR007

Sunaina Naikodi
19BTRCR016

Varshini D
19BTRCR039

Under the guidance of

Dr Mohammed Zabeeulla A N

Assistant Professor and Programme Coordinator,

Dept. of CSE

Faculty of Engineering & Technology

Jain (Deemed-To-Be University)

B. Tech (Honours) in Computer Science (Data Science)

Jain Global Campus, Kanakapura Taluk - 562112

Ramanagara District, Karnataka, India

2021-2022.

B. Tech (Honours) in Computer Science (Data Science)

Jain Global Campus, Kanakapura Taluk - 562112

Ramanagara District, Karnataka, India

CERTIFICATE

This is to certify that the project work titled **"Automatic Generation of Music"** is carried out by **MITHUN G (19BTRCR006)**, **MYIL VAUGHANAN V L (19BTRCR007), SUNAINA NAIKODI (19BTRCR016), VARSHINI D (19BTRCR039),** a bonafide students of Bachelor of Technology at the Faculty of Engineering & Technology, Jain (Deemed-to-be University), Bangalore in partial fulfilment for the award of degree, Bachelor of Technology (Honours) in Computer Science (Data Science), during the Academic year **2021-2022**.

Dr Mohammed Zabeeulla A N **Assistant Professor,** **Programme Coordinator,** **Dept. of CSE** Faculty of Engineering & Technology, Jain (Deemed-to-be University) Date: Signature:	**Prof. Mohammed Zabeeulla A N** **Assistant Professor and** **Programme Coordinator,** **Dept. of CSE** Faculty of Engineering & Technology, Jain (Deemed-to-be University) Date: Signature:	**Dr. Devaraj Verma,** **Professor and Dy. HoD,** **Dept. of CSE** Faculty of Engineering & Technology, Jain (Deemed-to-be University) Date: Signature:

Name of the Examiner Signature of Examiner

1.

2.

DECLARATION

We, **MITHUN G (19BTRCR006) , MYIL VAUGHANAN V L (19BTRCR007), SUNAINA NAIKODI (19BTRCR016), VARSHINI D (19BTRCR039)** are students of sixth semester B. Tech (Honours) in **Computer Science (Data Science)**, at Faculty of Engineering & Technology, **Jain (Deemed-To-Be University)**, hereby declare that the project work titled **"Automatic Generation Of Music"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology (Honours) in Computer Science (Data Science)** during the academic year **2021-2022**. Further, the matter presented in the project has not been submitted previously by anybody for the award of any degree or any diploma to any other University, to the best of our knowledge and faith.

Name: MITHUN G Signature

USN: 19BTRCR006

Name: MYIL VAUGHANAN V L Signature

USN: 19BTRCR007

Name: SUNAINA NAIKODI Signature

USN: 19BTRCR016

NAME: VARSHINI D Signature

USN:19BTRCR039

Place: Bengaluru

Date: 22-06-2022

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to ***Faculty of Engineering & Technology, Jain (Deemed-to-be University),*** *for providing us with a great opportunity to pursue our Bachelor's Degree (Honours) in this institution.*

In particular we would like to thank ***Dr. Hariprasad S A****,* ***Director****,* ***Faculty of Engineering & Technology****,* ***Jain (Deemed-to-be University),*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Devaraj Verma****,* ***Professor and Deputy Head****,* ***Department of Computer Science & Engineering****,* ***Jain (Deemed-to-be University),*** *for providing right academic guidance that made our task possible.*

It is a matter of immense pleasure to express our sincere thanks to ***Prof. Mohammed Zabeeulla, Program Coordinator of Data Science****,* ***Dept. of Computer Science & Engineering****,* ***Jain (Deemed-to-be University),*** *for providing right academic guidance that made our task possible.*

We would like to thank our guide ***Prof. Mohammed Zabeeulla****,* ***Dept. of Computer Science & Engineering****,* ***Jain (Deemed-to-be University),*** *for sparing his valuable time to extend help in every step of our project work, which paved the way for smooth progress and fruitful culmination of the project.*

We would like to thank our Project Coordinator ***Prof. Mohammed Zabeeulla*** *and all the staff members of Computer Science & Engineering for their support.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in completing the Project work successfully.

Signature of Students

TABLE OF CONTENTS

ABSTRACT

One of the most important aspects of the music and cinema industries is music creation. Due to the multiple components required, music production and composition, whether by hand or by computer, is time-consuming. The ability to comprehend music is dependent on the composers' and music directors' high intellectual levels. Rather of using the usual way of writing and generating music, this project uses deep learning techniques to automate music composition. Because they capture rich characteristics in the frequency domain, LSTM is a high-level deep learning sequential data modelling architecture. Here, a MIDI music file is utilised to train an LSTM network, resulting in the generation of a new musical pattern.

LIST OF FIGURES

NOMENCLATURE USED

MIDI	Musical Instrument Digital Interface.
LSTM	Long Short Term Memory.
wav	Waveform audio file format.
M4a	MPEG -4 Audio format.
RNN	Recurrent Neural Network

Chapter 1

INTRODUCTION

1.1. Overview

Automatic music generation is a process where we compose a short piece of music with the least human intervention.

Music generation and production has been in buzz recently. The major reason being the boom in Deep Learning and their powerful algorithms.

So, this paper aims to implement music generation using deep learning techniques such as LSTM.

1.2. Problem Definition

- Obtaining a good quality music for the user without the usage of human intervention.
- The generation of music which have notes, chords and octaves as its contents, differentiating between them.
- Generating the new pattern of music satisfying the required design criteria consisting of:
    1. a model to handle the variable length sequences
    2. to track long- term dependencies (memory)
    3. to maintain information about the order
    4. to share parameters across the sequence.

1.3. Objectives

- Training of the Midi files using LSTM (Long Short Term Memory) for music generation automatically.
- Using the LSTM network to satisfy the design criteria of the model.

1.4. Methodology

The MIDI (Musical Instrument Digital Interface) files are used as an input for this project since it is economical in terms of memory usage but can represent the actual notes and chords of the audio.

The project focuses on the use of LSTM network trained on MIDI files to generate artificial music. This paper covers a brief overview of the working of LSTM on sequential music modelling and how it encounters the vanishing gradient problem of RNN.

With the model being trained on MIDI files, a method (function) was pioneered called the “Malody_Generator” which generates the music as a stream and that is then converted to MIDI file format.

This generated MIDI file is then converted into “wav” format to produce an audio perceivable by human senses.

1.5. Hardware and Software Tools Used

Software:

- Google Colab
- TensorFlow 2.x

Hardware:

- Compute instances in Google Colab

Chapter 2

LITERATURE SURVEY

2.1. Related Work

The music generating challenge has been implemented and made open source by major organisations such as Google AI and Open AI.

'Magenta' was created by the Google AI and Deep Mind teams. Magenta is an open source tool that uses Deep Learning to create music and art. To train its model, the researchers gathered 200 hours of exceptional piano performance. Various deep learning approaches, such as Transformers, were used to create the model. Magenta was incorporated into Google Assistant at the Google I/O Conference in 2020.

Similarly, Open AI created JukeBox, a neural network that generates music from a wide range of genres and artist styles.

2.2. Existing System.

RNN (RECURRENT NEURAL NETWORK)

Music is a sequence of characters therefore the obvious choice will be RNN or variations of RNN like LSTMs or GRUs which can process sequence information very well by understanding the patterns in the input. A recurrent neural network is a specific type of neural network which is based on sequential information and data. Reason behind calling them recurrent because same set of weights are applied recursively over a differential graph-like structure. RNN has great application in Natural Language Processing.

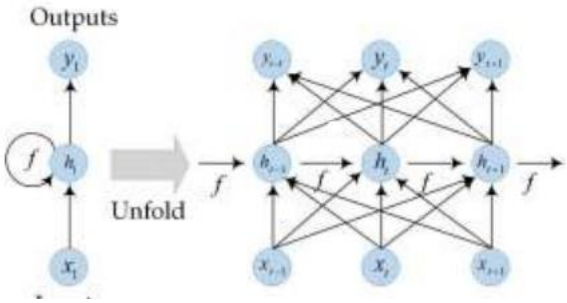


Figure 1 A simple RNN model

Char-RNN model

There is a special type of RNN called char RNN. Now our music is a sequence of characters. We will feed one after the other character of the sequence to RNN and the output will be the next character in the sequence. So, therefore, the number of output will be equal to the number of inputs. Hence, we will be using Many-to-Many RNN, where number of output is equal to the number of input

2.3. Limitation of Existing System

- An RNN could be thought of as a regular neural network with the addition of the network's input being the item of a sequence and with outputs dedicated to encoding and remembering state about the previous items sent through the network. [1]
- The reason for not using a simple RNN for the project is because of an undesirable property of simple RNNs where previous inputs 'disappear' as more inputs are fed to the network. This is known as the 'vanishing gradient' problem [1]

2.4. Proposed System

LSTM also solves complex, artificial long-time-lag tasks that have never been solved by previous recurrent network algorithms. [2]

LSTM architecture performs good with sequential data.

LSTMs were made to fix this issue, and they do this by predicting what information is important to remember, and what can be ignored.

Using a gated mechanism, LSTMs are able to recognise and encode long-term patterns. LSTMs are extremely useful to solve problems where the network has to remember information for a long period of time as is the case in music and text generation.

The major difference we found between the structure and others, was that with the large amount of processing power available to us via MSOE's high performance computing cluster named Rosie, we were able to make our network larger, and deeper with more stacked LSTMs at the front of the network. This allows the network to notice more patterns and characteristics that a shallower network would struggle to recognize.

Chapter 3

METHODOLOGY

3.1. Dataset

This project's dataset was scraped from Kaggle's 'Classical Music MIDI' dataset. Under the Creative Commons License, this dataset is free and open source. This collection contains 19 great composers' pieces in the form of classical piano midi files.

The following is a list of composers as well as the amount of MIDI files collected for each of them:

- Granados – 3
- Tschai – 12
- Schubert – 29
- Schumann – 24
- Mendelssohn – 15
- Liszt – 16
- Muss – 8
- Greig – 16
- Mozart – 21
- Albeniz – 14
- Borodin – 7
- Debussy – 9
- Burgm – 9
- Chopin – 48
- Balakir – 1
- Beeth – 29
- Brahms – 10

- Bach - 3

There are totally 295 number of MIDI files in the dataset

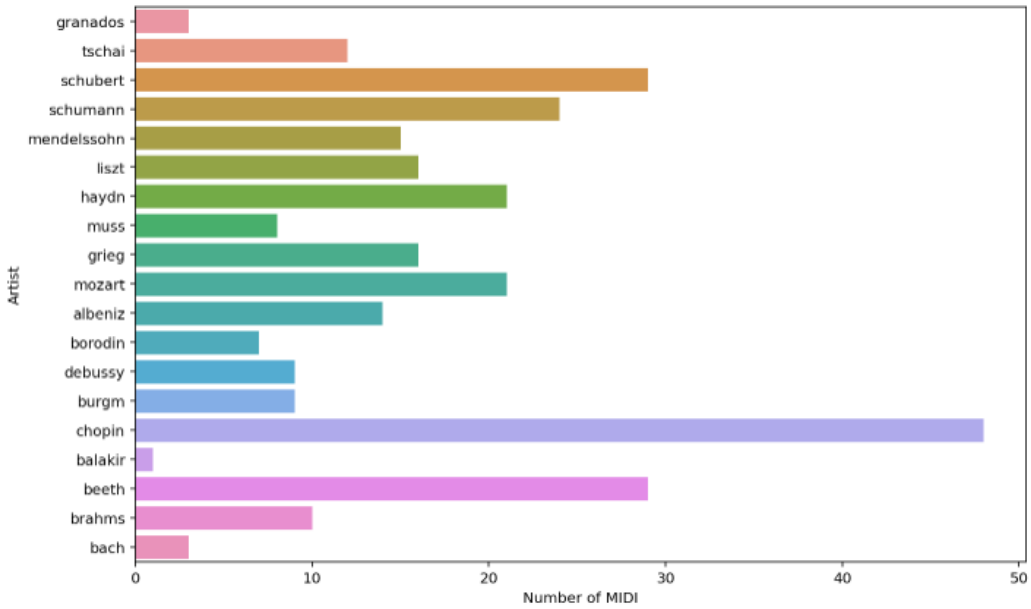


Figure 2 A count plot showing various artists and the number of MIDI files collected

3.2. Architecture

All of the requirements are installed with their respective versions at first. This project makes use of a number of dependencies, including numpy, pandas, IPython, music21, the os module, and many others.

The Kaggle dataset is downloaded and imported into the project environment. The 'Classical Music MIDI' dataset was used. Musical Instrument Digital Interface (MIDI) is a standard for connecting electronic musical instruments, computers, and other audio equipment to play, edit, and record music. Up to 16 channels of MIDI data can be carried on a single MIDI cable, each of which can be routed to a different device.

The imported MIDI files are then consolidated into a single folder and turned into a music stream type with the help of the music21 library. This step is required

since the MIDI file is separated into two object types using the stream data: Notes and Chords. Pitch, octave, and offset data can be found in note objects. Chord objects are essentially a container for a group of notes that are performed simultaneously. This pre-processing phase aids the model's extraction of rich features from the data and improves training performance.

Following the pre-processing, the appropriate data analysis is carried out in order to gain insight into the dataset in question. Albeniz's MIDI songs were subjected to such scrutiny. According to this research, his songs contained a total of 18419 notes, with 162 of them being unique. Other information includes the Average Recurrence of the notes in the Corpus, as well as the most and least frequent notes in the Corpus.

It was also crucial to remove any unusual chords from the data. Because these uncommon chords can behave as outliers in our data, they can have an impact on the model's performance later on. A total number of notes that appear less than 100 times was chosen as the criteria. The data contained a total of 14069 notes after the unusual chords were removed. This is the cleaned data Corpus that will be used to train models and create music.

This cleaned data is now subjected to generate a sheet of music. This process helps the music technicians to understand the new structure of the music after the removal of unusual chords. Later this music sheet also helps to compare with the generated music and helps to find new patterns of music being generated. Below is an image of music sheet generated for one of the cleaned data.

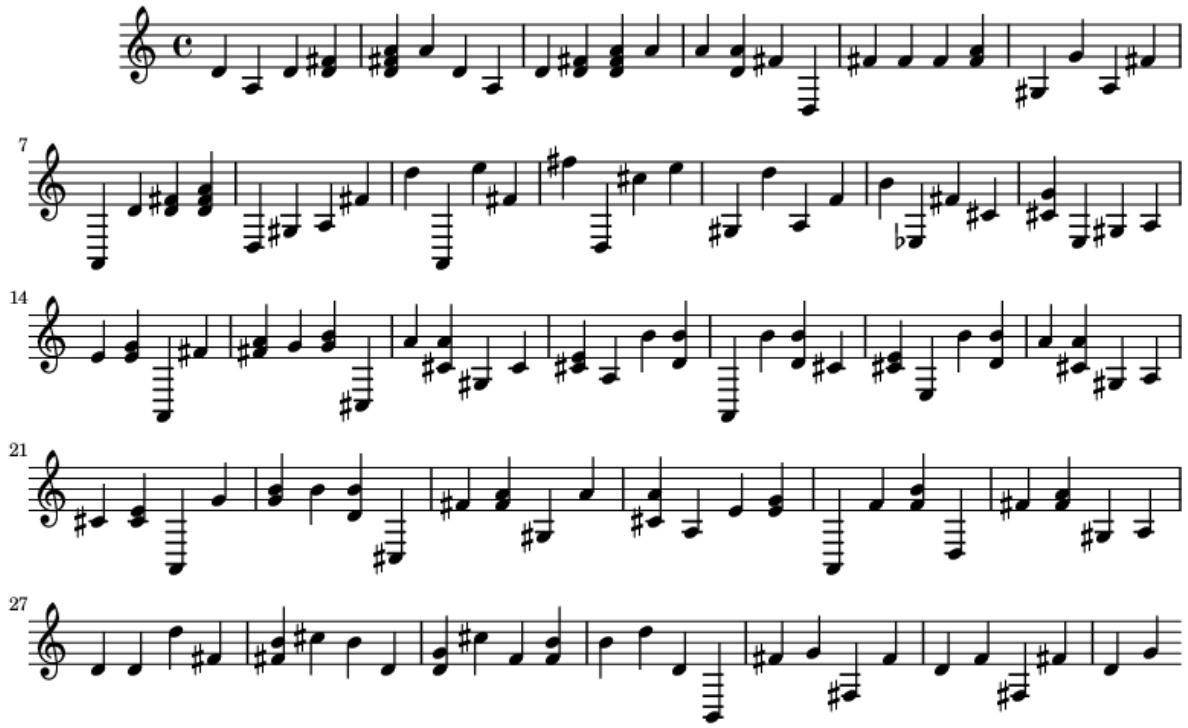


Figure 3 Music sheet for Albeniz's song

Encoding and breaking the corpus into smaller, equal-length sequences is the next step: The Corpus already has notes in it at this point. This corpus will be encoded, and tiny sequences of features and targets of identical length will be created. The mapped index in the dictionary of the unique characters that each feature and target represent will be included in each feature and target. In addition, independent and dependent variables X and y are formed, resized, normalised, and hot encoded.

Finally, after cleaning and pre-processing the data, LSTM architecture is implemented. LSTM stands for Long Short-Term Memory, they rely on gated cell to track information throughout many time steps. They are well suited for maintaining long term dependencies in the data and tracking information at multiple time steps to overcome vanishing gradient problem. There are many interactive layers which controls the flow of information through LSTM cell. Information is being added or removed through structures called gates (*generally they are sigmoidal or tanh*).

These gates help in modulating and capturing how much of the input should be passed through. There are 4 steps in LSTM process:

- Forgetting irrelevant history – *FORGET*
- Storing most relevant new information – *STORE*
- Updating their internal cell state – *UPDATE*
- Generating an output – *OUTPUT*

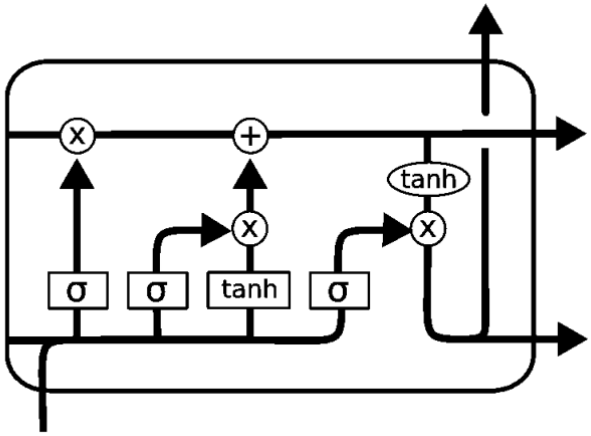


Figure 4 A simple LSTM cell

Sequential model is used to generate the LSTM model in TensorFlow 2.x. Later layers such as LSTM, Dropout, and Dense are added. Dropout layers are made up of 10% of the neurons; the first LSTM layer comprises 512 neurons, while the second LSTM layer has 256 neurons. A 256-neuron Dense layer is also added. The SoftMax activation function is used in the output layer because it anticipates the multinomial probability distribution. The Adam optimizer is employed.

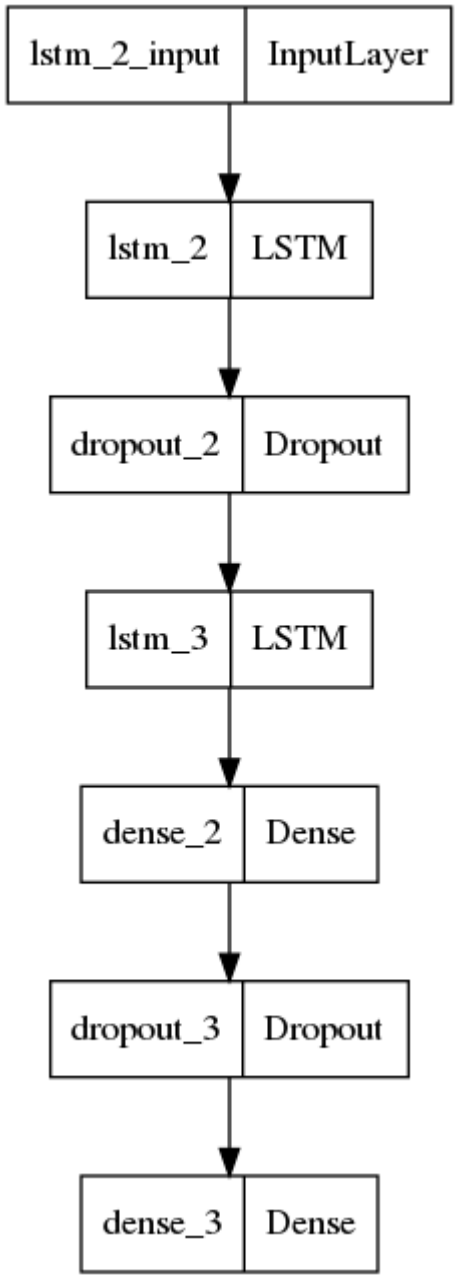


Figure 5 Model Architecture diagram

The model is trained until the Early Stopping approach is used for 200 epochs, after which the loss is determined.

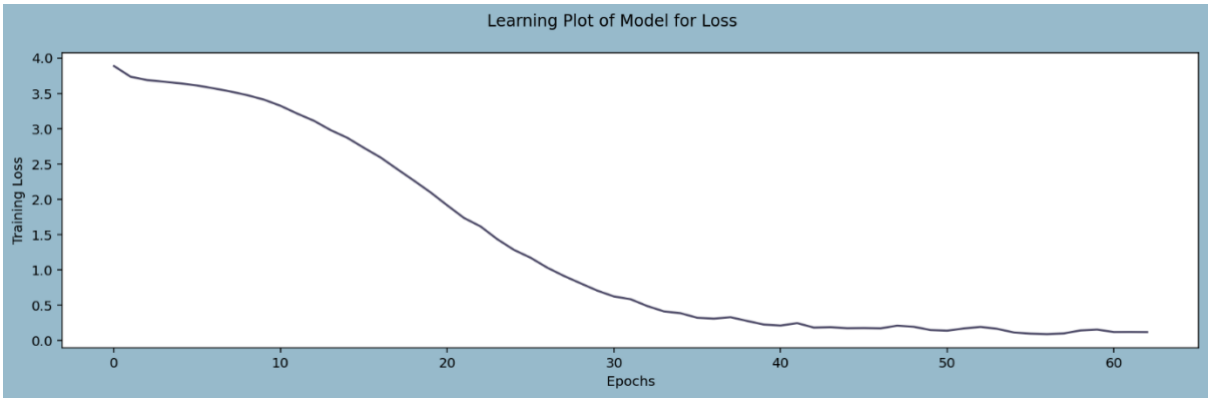


Figure 6 Training loss vs Number of epochs

To generate the tune, a function is written from scratch. This function may forecast and normalise data from a trained model using exponential and logarithmic prediction. The Music21 module is used to transform music that has been predicted or created into a stream file format.

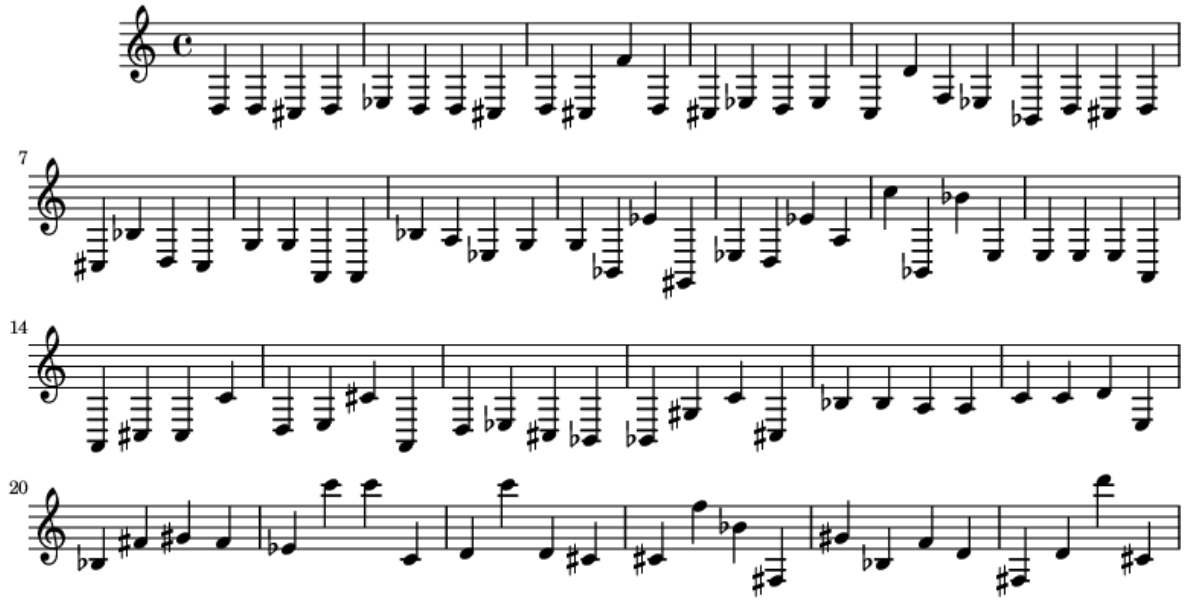
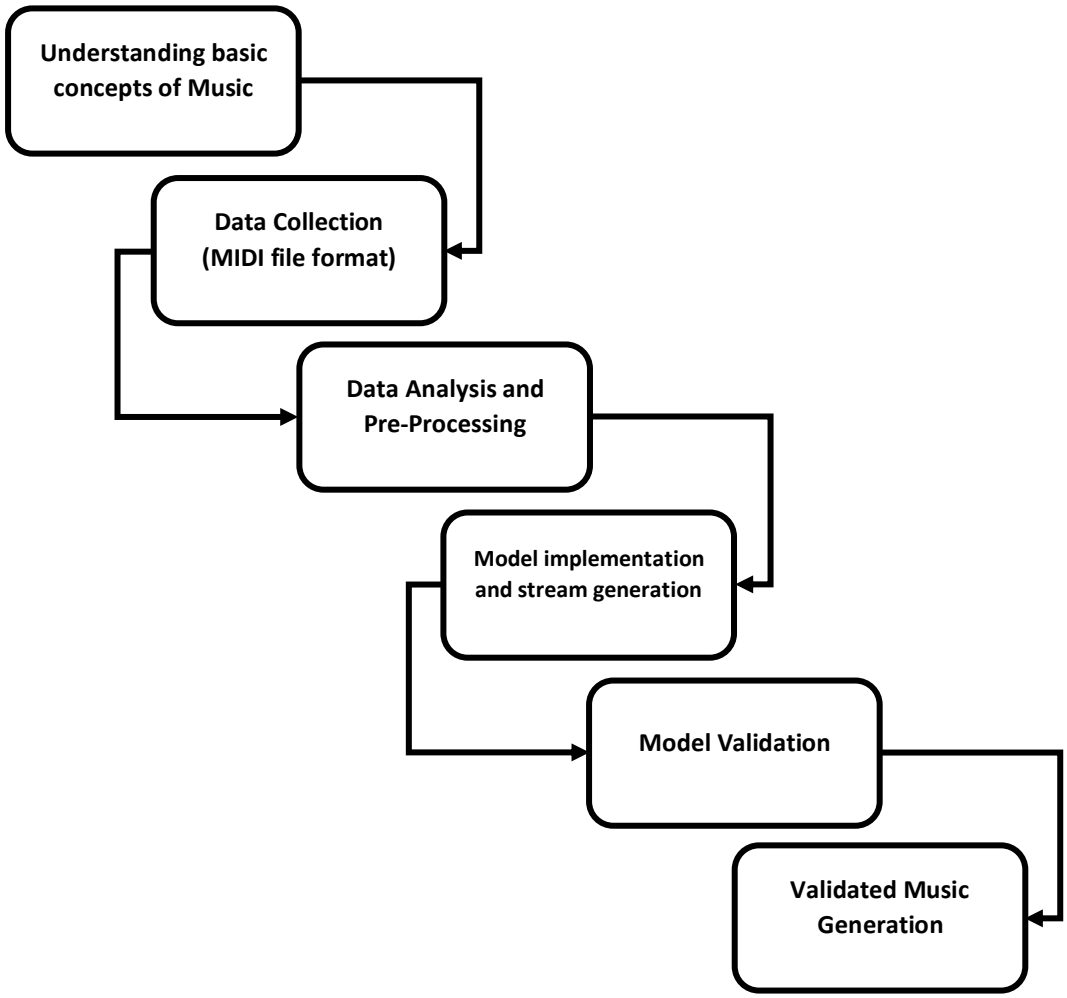


Figure 7 Music Sheet for model generated music

3.3. Sequence Diagram:



i. **Understanding basic concepts of music:** A thorough research on the problem statement was made before it was taken up as a project. The conceptual understanding of music was laid out by understanding the definitions of notes, chords and octaves and how they function in order to produce a tune cordially. The application of MIDI file was also learnt and perceived, which basically contained all the information of the audio in a compressed "lite" format. Music stream concept was a key element of the research, stream format is the representation for computer understandable musical notes.

ii. **Data Collection:** The MIDI files were collected from Kaggle MIDI dataset. MIDI is the representation of the audio file, but not the actual audio itself, which resulted in a reduced dataset size. This in-turn produced a space economical model to generate music.

iii. **Data Analysis and Pre-Processing:** In this step, the data which was obtained from the dataset was analysed by observing various parameters of an audio file by plotting various graphs using various Python libraries. Succeeding the analysis, the data was converted to music stream format to make it more machine understandable.

iv. **Model Implementation and Stream Generation:** The pre-processed data was passed into the LSTM network model to understand the repeating patterns of the notes, chords and octaves. This results in the production of likable tunes which is in the form of music stream format. Since the music stream data cannot be perceived by human auditory system, a module was pioneered to convert this stream data to a MIDI file format.

v. **Model Validation:** The MIDI file that we obtained from the LSTM model is validated with every epoch. It was observed that the error value tends to zero with a greater number of epochs as the model tends to learn more.

vi. **Validated Music Generation:** Succeeding the model validation, when satisfactory error value was obtained, the MIDI file that was validated was converted to human perceivable audio format i.e., ".wav" or "m4a" format.

Chapter 4

TOOL DESCRIPTION

This section gives a detailed description about the hardware tools and software tools involved in developing this system and how they are used.

4.1. Hardware Requirements

Computational resources such as CPU, GPU (Mandatory with minimum 4 GB memory,12 GB RAM).

4.2. Software Requirements

- Google colab – this is used to run the whole code.
- Lilypond- it is a Linux package for music engraving. It converts musical notes to musical sheet.

File dependencies:

- python -3.7.13 version.
- numpy –this is for the numerical calculation.
- pandas-for creation of data frame and handling meta data.
- matplotlib- this is for exploratory data analysis.
- os-files assessing in the directory and dir. manipulation
- tensorflow-2.x- for the sequential model and lstm layer
- random- for random split
- IPython- interactive python- to listen audio and display imaged.
- music 21- it is a module tool kit for computer aided musicology used to it allows us to create Note and Chord objects so that we can make our own MIDI files easily and also view notes and chords of already existing MIDI files.

Chapter 5

IMPLEMENTATION

STEPS TO RUN THE PROGRAM:

Step 1: Scrap the Classical Music MIDI dataset from Kaggle

Step 2: Install and import all the dependencies

Step 3: Perform data analysis on the imported dataset

Step 4: Perform pre processing steps like normalisation, encoding, splitting, cleaning unwanted chords.

Step 5: Generate music sheets using music21 library

Step 6: Build a LSTM architecture using TensorFlow 2.x

Step 7: Model is trained and losses are validated.

Step 8: A function from scratch is written to generate the music from the trained model

Step 9: Music sheet is generated for the generated music from the model.

Chapter 6

RESULTS AND ANALYSIS

6.1. Result Discussion

Music stream was generated from the LSTM model and the function defined which converts this stream file into MIDI file.

The generated MIDI file is later converted into human perceived audio file (mp3, m4a, wav).

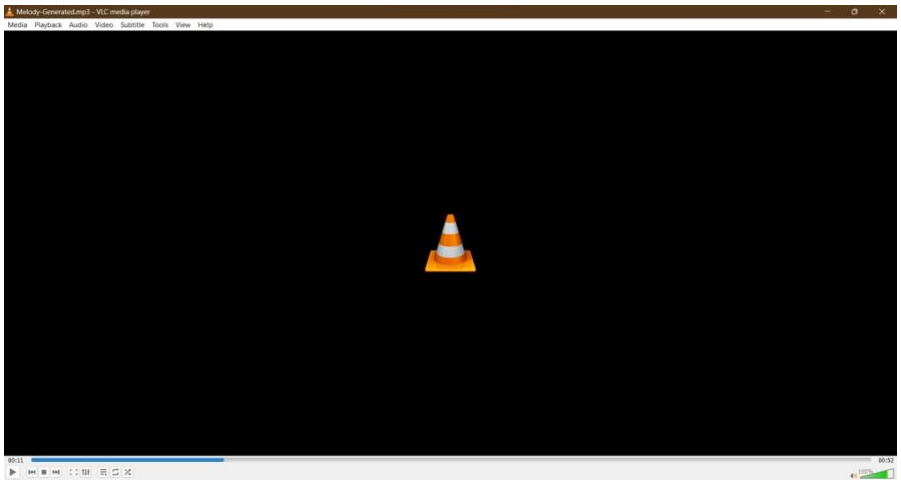


Figure 8The generated audio file playing on VLC

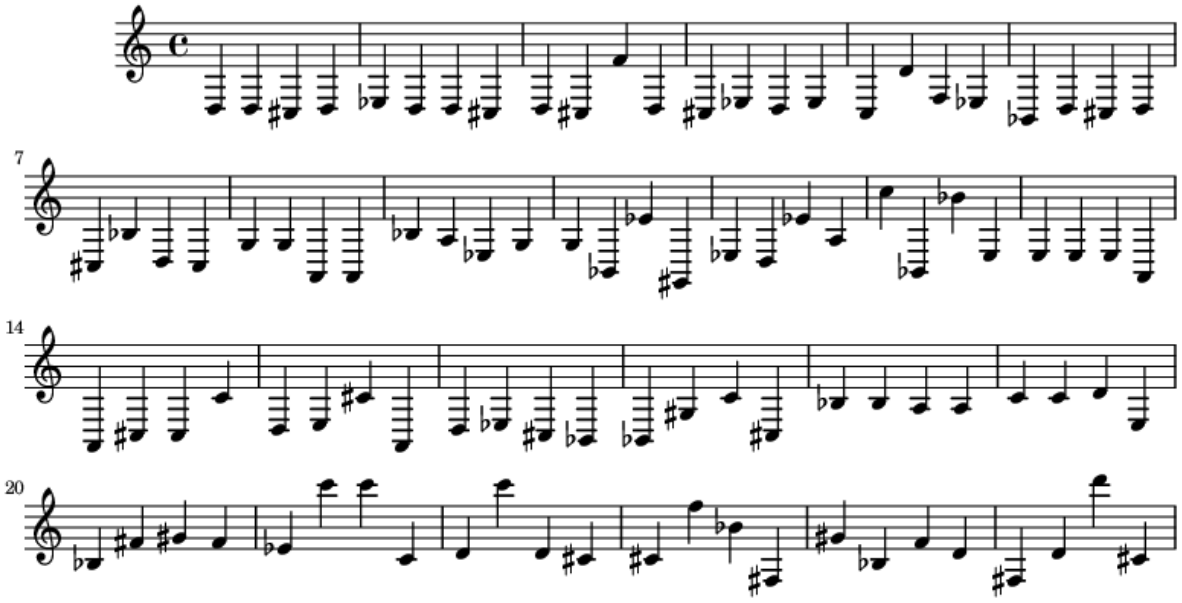


Figure 9Music sheet model generated music

6.2. Analysis

At the end of the 176th epoch, the training error of the LSTM model was found to be 0.1204. The training began with random weights, resulting in a large training error of 3.88. The model trains further down the lane to reduce this error until the callback. The music is generated using this trained model as a predictor.

Chapter 7

CONCLUSIONS AND FUTURE SCOPE

LSTMs are a very promising solution to sequence and time series related problems.

As of now, the music generated is of only one piece of instrument. It would be interesting to listen what music the model would produce if it is trained on multi-instrument music.

Also, when provided with genre, and lyrics as input, our model will try to output a new music sample produced.

To try implementing the similar kind of model using Transformers. LSTM or RNN models are sequential and need to be processed in order, unlike transformer models. Due to the parallelization ability of the transformer mechanism, much more data can be processed in the same amount of time with transformer models.

REFERENCES

- https://doi.org/10.48550/arXiv.2203.12105
- https://doi.org/10.1109/ACCESS.2020.3031155
- https://doi.org/10.48550/arXiv.2201.00052
- https://doi.org/10.1109/ACCESS.2020.3031155
- https://analyticsindiamag.com/a-hands-on-guide-to-automatic-music-generation-using-rnn/
- https://www.analyticsvidhya.com/blog/2020/01/how-to-perform-automatic-music-generation/
- https://magenta.tensorflow.org/datasets/maestro
- MIT 6.S191 – Alexandra Amini

APPENDIX:

SOURCE CODE:

https://colab.research.google.com/drive/1oPtwxR1aPMyacJJ-oPmZX3DpgXvA6iic#scrollTo=hrL2tZYOY97X

GITHUB: https://github.com/Mithun162001/Mini-Project/blob/main/Auto_Music_generation.ipynb