MUSIC GENERATION USING RECURRENT NEURAL NETWORKS

Mr.Nikhil S. Patil	Ms. Toheed Shaikh	Prof. Seema Bhuvan
Student, Sterling Institute of Management Studies Nerul, Navi Mumbai	Student, Sterling Institute of Management Studies Nerul, Navi Mumbai	Asst. Professor, Sterling Institute of Management Studies, Nerul, Navi Mumbai
patilnikhil160399@gmail.com	toheedsk58@gmail.com	seemabhuvan@ncrdsims.edu.in

ABSTRACT

In recent years, the tempos and beats of music as well as its melodies have undergone constant evolution. A group of artists using various instruments together to create a final synchronized result in the traditional way that music is made. Harmonies and beats have always been thought to be manually created in recent years.

Nonetheless, the development of digital technologies and software has made it possible for computers to produce music automatically at a startling rate. The aim of this study is to suggest a method for producing musical notes using recurrent neural networks (RNN), specifically Long Short-Term networks for memory (LSTM). The data is represented as a MIDI file for percussion instrument digital interface (MIDI) access and interpretation in order to apply this algorithm. Techniques for receiving, processing, and storing MIDI files for use as input as well as the process of getting the data ready for input into the model are also covered. The model should be able to recall prior information about the structure and specifics of a musical sequence in order to improve its learning skills. The layered architecture employed by the LSTM model and the way in which its connections interact to form a neural network are both covered in this study.

Keyword: *MIDI, RNN, LSTM, Music Generation.*

1. INTRODUCTION

Recent research and developments in the field of music technology have made it possible for deep learning and other analogous technologies to imitate performers. These artificial intelligence models are referred to as generative adversarial networks, or GANs. They get experience handling challenging tasks as they handle enormous amounts of data. Music doesn't have a definite dimension because it's made up of notes and chords. Due to the fixed dimensionality of inputs, objectives, and outputs that typical deep neural network techniques assume, they are not capable of producing music due to output independence. Hence, it became apparent that a Discover the right finance company here. If none exist, remove this. A method that learns to map sequences to sequences might be useful. The science of pitch harmony is used in music as an art form to create beautiful sounds. It may be heard without the use of any non-musical instruments and is widely employed as a means of artistic expression. A single musical measurement unit is referred to as a "note" and is indicated by through the "" sign.

Notes may act differently when they are added to or are absent from other notes. A scale, a chord progression, or just a single note can organize music. Given that it is enjoyable to the ears, music makes sense. The fact that there are numerous musical genres available can be annoying. Building off of already existing compositions, the genetic algorithm is a technique for making music. Each fragment's strong rhythm may be highlighted by evolutionary algorithms, which then combine them into distinct musical works. On the other hand, each iteration step makes it ineffective. Contains a delay. Also, the absence of context makes it challenging to gather comprehensive and accurate rhythmic information. We require a system that can remember the prior note sequence, predict the subsequent one, etc. to resolve the problem mentioned above. Consistent neural pathways. The system must also absorb and modify the original musical patterns.

This article describes a method that can be used to automatically produce music and melodies without the need for human input. The algorithm is built on LSTM networks, a form of neural network. The basic objective is to create a model that can study a collection of musical notes, analyze them, and then produce a number of excellent musical sounds. By merging character-based neural language models, such as Character RNN and LSTM cells, which incorporate bi-directionality and attention, we extend the use of RNNs to develop a music generator. Music is expressed using the ABC notation. Textual form. There are deployed networks, specifically Long and Short-Term Memory RNNs. We described a method for creating music using algorithms.

2. LITERATURE REVIEW

We described an algorithmic method for creating music. To create a new musical pattern, musical genres could be created from any type of present instruments using a beat-to-beat rhythm. The system must be able to retrieve knowledge in order to extract musical elements and forecast future musical style trends, which is a critical goal. The system must secondly absorb and modify the original musical patterns.

a) Music Generation with Variational Recurrent Auto-Encoder

   A new architecture combining an artificial neural network with a variational auto-encoder supported by historical data and heuristics has been developed to generate pseudo- live, aesthetically pleasing and melodically diverse music. This is the first application of VRASH to music generation, and it achieves a good balance between global and local structure in the track. The proposed structure is relatively simple to implement and train, and it enables control over the style of the output, generating tracks that match the specified parameters.

b) Deep Recurrent Music Writer: Memory-enhanced Varia-tional Auto encoder based Musical Score

   The authors developed a new metric to evaluate the quality of generated music and used it to assess the outputs of a Variational Auto-encoder-based generative model for automated music composition. They applied this measure to systematically and automatically adjust the parameters and architectures of the generative model to optimize the musical outputs in terms of their similarity to a particular musical style.

c) Towards Music Generation with Deep Learning Algorithms

   The researchers developed a multi-layer Long-Short Term Memory (LSTM) Recurrent Neural Network (RNN) and a feed-forward network based on a collected data set, as well as an LSTM-based Encoder-Decoder architecture as a baseline model. However, the models did not achieve the goal of generating a 60-second-long sequence of polyphonic music. The limitations of the models were discussed, and it was determined that further refinement is needed before being able to generate actual musical sequences.

3. PURPOSE

A recurrent neural network's LSTM layer can output either a sequence or a matrix given a sequence as input. By randomly changing a portion of the input units to zero after each training update, dropout layers can be used to prevent over-fitting in neural networks. A parameter controls the percentage of units that are zeroed out. By minimizing the model's reliance on any one input unit, dropout helps to make it more regular. Each input node and each output node are coupled in a layer type called a dense layer, also referred to as a completely connected layer. As a result, every node in the dense layer's output is dependent upon every node in its input. Many neural network architectures have dense layers, which are in charge of figuring out intricate correlations between the input and output data.

Which activation function the neural network will employ to calculate each node's output is determined by the activation layer.

The ideal number of nodes for each layer is determined by the first parameter for the LSTM, Dense, and Activation layers. The percentage of input units that should be set to zero during training to prevent over-fitting is specified by the first parameter for a Dropout layer. This variable aids in regularizing the model and enhancing its capacity for generalization.

The first layer requires a special parameter known as an input shape. The goal of the parameter is to notify the network about the kind of training data set. The number of nodes in the final layer should always be the same because our system includes a variety of outputs. We can make sure that the output of the network will match the classes in the data set by utilizing an output layer with a set number of nodes that matches to the number of classes in the data set.

In this system, we employ two LSTM layers, three Dropout layers, two dense layers, and a single activation layer. Because each of our outputs only belongs to one class and we have more than two classes to work with, we will utilize categorical cross entropy to calculate the loss for each iteration of the training. In this lesson, the network will be trained for 200 epochs (iterations), with 64 samples being fed through it once every batch.

Model checkpoints allow us to cease training at any time without losing our progress. Model checkpoints enable the network nodes' weights to be saved to a file at the end of each epoch. When we stop rushing the neural network without being concerned about weight loss once the tests are happy with the loss percentage. Otherwise, saving the weights to a file wouldn't be possible until the network had completed all 200 epochs.

4. OBJECTIVE / SCOPE

The goal of our system is to develop a model that can automatically and autonomously construct melodies and rhythms. Using a single-layered LSTM model, the model is capable of remembering prior data descriptions and producing melodic music. Using MIDI files of piano notes, it is able to learn harmonic and melodic note sequences. The performance of the model will be examined in relation to the impact of increasing the number of LSTM units and experimenting with various hyper-parameter values. We believe that more research employing a significant number of calculations could further optimize this model. The main goal of autonomous music production is to support musicians in the composition and improvement of music. The level of supervision employed for the prototype. At one extreme of the range, there is complete automation and mechanization with no human involvement. With early stops built into the model to track the creation of the music, it might also be more participatory. The neural network method used in this paper is intended to be non-interactive. Since it gives a complete end output that is machine-accessible without requiring human interaction, the MIDI file format is also created for this dimension. For real musicians who can disrupt the system in the middle of content creation, the degree of autonomy is a fascinating viewpoint advancement. User input flexibility is yet another crucial essential goal.

The user can modify the number of layers, the size of hidden layers, the length of the sequence, the number of time steps, the batch size, the optimization technique, and the learning rate in our adaptable architecture.

Moreover, users can modify the amount of time steps and the window of notes fed into the note-axis LSTM model. Even if we are capable of creating music of a high caliber, there is still much potential for development. Currently, there is only one instrument used to make the music. If the model were trained to play several instruments, it would be intriguing to hear the music it could create. The model might have a method for dealing with challenging musical notes. By filtration The system can produce more robust, high-quality music by substituting unknown notes with recognized ones. In the future, we plan to employ this strategy to create mood- based music based on user input. Through investigations, it has been discovered that music can help persons with Parkinson's and dementia. To sum up, this technology might make music in response to the patient's requests.

5. RESEARCH METHODOLOGY

One of the most important steps in the model execution is creating batches. It requires sophisticated coding techniques and knowledge of the underlying principles related to the RNN character. The overall model's effectiveness with batches is what distinguishes it from earlier versions and makes it unique. Three parameters are used to generate the batches in this:

- Batch Size (16) – specifies the required number of batches.
- The number of distinctive characters. The distinctive characters that appear in the music's ABC format.
- Sequence Length (64), which indicates the length of the sequence to be sent as input.

6. IMPLEMENTATION

When given a sequence as input, the LSTM layer of a recurrent neural network can output either a sequence or a matrix. The use of dropout layers, which randomly set a portion of the input units to zero after each training update, helps minimize over-fitting in neural networks. A parameter regulates the percentage of units that are set to zero. Dropout lessens the dependency on any one input unit, which helps to regularize the model. Each input node is connected in a dense layer, often referred to as a fully connected layer, in a neural network. Each output node. This indicates that each node's output in the dense layer depends on every node's input. Many neural network architectures have dense layers, which are in charge of figuring out intricate correlations between the input and output data.

Which activation function the neural network will employ to calculate each node's output is determined by the activation layer.

The ideal number of nodes for each layer is determined by the first parameter for the LSTM, Dense, and Activation layers. The percentage of input units that should be set to zero during training to prevent over-fitting is specified by the first parameter for a Dropout layer. This variable aids in regularizing the model and enhancing its capacity for generalization.

For the first layer, a special parameter known as the input shape must be provided. The parameter's function is to inform the network about the type of training data set. The last layer of our system should always contain the same amount of nodes to represent the different outputs, which our system has.

We can make sure that the output of the network will match the classes in the data set by employing an output layer with a certain number of nodes that matches to the number of classes in the data set.

In this system, there are one activation layer, two LSTM layers, three Dropout layers, two dense layers, and two dense layers. Given that each of our outputs only belongs to one class and that we have more than two classes to consider, we will utilize categorical cross entropy to calculate the loss for each training iteration. In this lesson, the network will be trained across 200 epochs (iterations), with 64 samples being sent through it each time.

We employ model checkpoints to make sure that we are always able to halt training without losing any of our progress. Model After each epoch, checkpoints enable the network nodes' weights to be saved to a file.

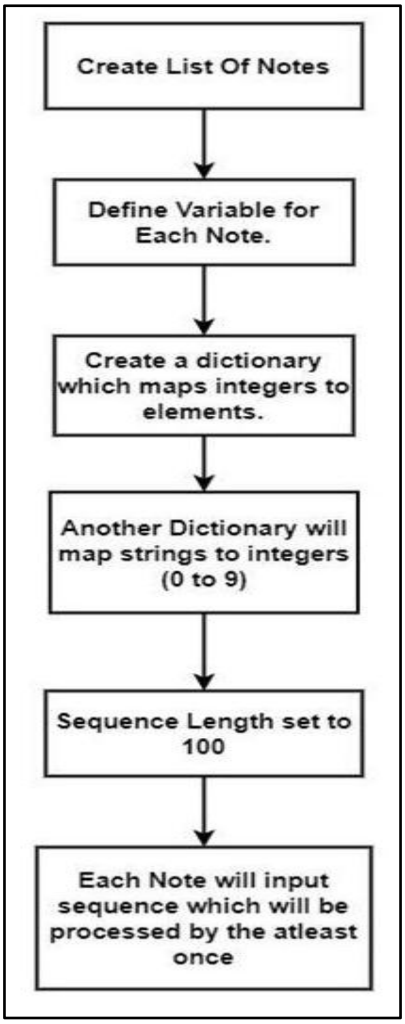

Figure 1: Basic Flow Diagram

If the tests are happy with the loss number, we can stop running the neural network without worrying about losing the weights. Otherwise, we wouldn't be able to store the weights to a file until the network had completed all 200 epochs.

Figure 1 illustrates the proposed system's basic flow (1). A list of notes is first created by the solution using information from the practice MIDI files. It then declares a variable for every note. All notes in all octaves are given individual integers in a dictionary that is built. Then, a new dictionary is made that converts text into previously stated integers between 0 and 9. A 100-note sequence is then produced.

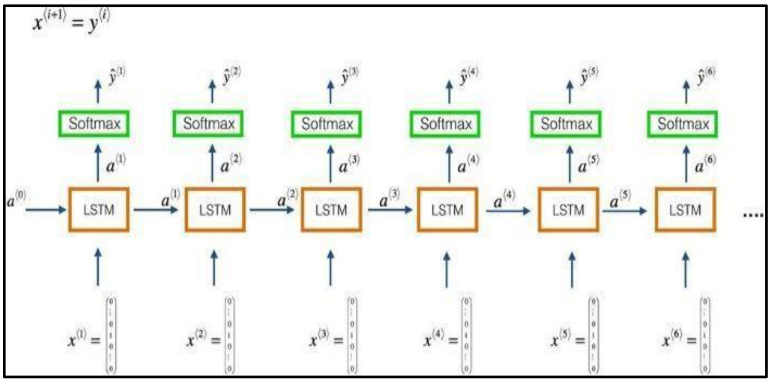

Figure 2: One to Many LSTM architecture

Fig (2): We use a character level-based architecture to train a model. Each LSTM cell therefore accepts the previous time step activation (at1) and the prior layer actual output (yt1) as input at the current time step 'tt' in order to forecast the next note in the sound stream. This is seen in the diagram below (Fig 2).

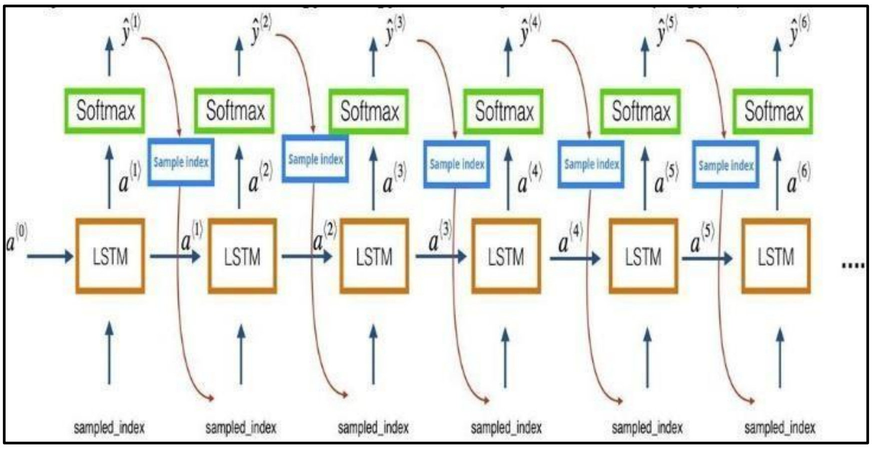

Figure 3: Sampling from a trained Network

Figure 3 The activation and cell state from the previous LSTM cell will propagate into the subsequent cell at each sampling step, where they will be used to produce a new output.

7. CONCLUSION

The RNN (LSTM) paradigm was used to construct a neural network that automates music creation. Long-term dependencies can be effectively captured by the LSTM model. By providing our network with musical note objects, which were subsequently utilized to create music, we were able to teach it a particular musical style. We created a prototype in which the algorithm composes music on its own. Building on earlier work, we set up the data to use one-hot encoding and gave the network a variety of musical notes. We intend to expand our network in the future so that it can recognize song beginnings and finishes, learn more song attributes, and enable multi-track inputs and outputs.

Furthermore covered in this research are the viability of enhancing the model and historical and current possibilities. To determine how effectively this approach scales, future study will make use of the Million Song Data-set. Flexibility and generalizability were key functional elements in the model's design. The fundamental concept and methodology of algorithmic music generation and training were discussed.

Our neural network model and algorithm can be utilized to develop a variety of goods that are both marketable and profitable. The technique can be used to produce mood-based fresh music for people based on their environments and interests. Also, it can jolt people's creative impulses and the process of writing music. Moreover, it can be utilized to create Indian classical music in accordance with the 'Ragas' regulations. (A raga, or musical mode, is a set of melodic guidelines for improvisation in Indian classical music. There is no parallel in Western classical music, but it is a crucial and important component of Indian musical tradition.)

8. REFERENCES

[1] Agrawal, Shipra, and Navin Goyal. ”Analysis of Thompson sampling forthe multi-armed bandit problem.” In Conference on Learning Theory , pp. 39-1. 2012.

[2] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high- dimensional sequences: Appli- cation to polyphonic music generation and transcription. ICML, 2012..

[3] Douglas Eck and Jurgen Schmidhuber. A first look at music compositionusing lstm recurrent neural networks. IDSIA.

[4] Sigurur Sku´li “Music Generation Using a LSTM Neural Network in Keras” https://towardsdatascience.com/how- to-generate-music-using-a-lstm-neural-networkin- keras-68786834d4c5 (2017). (Accessed on 16/10/22-22:00)

[5] Hochreiter, S., Bengio, Y., Frasconi, P., and Schmidhuber, J. (2001). Gradient flow

in recurrent nets: the difficulty of learning long-term dependencies. In Kremer, S. C. and Kolen, J. F., editors, A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press.

[6] P erez-Ortiz, J. A., Gers, F. A., Eck, D., and Schmidhuber, J. (2002). Kalman filters improve lstm network performance in problems unsolv- able by traditional recurrent nets. Neural Networks. Accepted pending minor revisions.

[7] Christopher Olah. "Understanding LSTM Networks." https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (2015) (Accessed on 16/10/22-22:00)

[8] Akshay Sood "Long Short-Term Memory" https://pages.cs.wisc.edu/

[9] shavlik/cs638/lectureNotes/Long(percent- sign)20Short Term(percent-sign)20Memory(percent-sign)20Networks.pdf (2016) (Accessed on 16/10/22-22:00)

[10] PDillon Ranwala "The Evolution of Music and AI Technology" https://wattai.github.io/blog/music-ai-evolution (2020) (Accessed on16/10/22-22:00)

[11] Other piano data source: http://www.piano-midi.de/midi-files.htm (Ac- cessed on 16/10/22-22:00)

[12] https://github.com/vishnubob/python-midi (Accessed on 16/10/22-22:00)

[13] N. Rafal Jozefowicz, Ilya Sutskever, and Wojciech Zaremba. "An Empirical Exploration of Recurrent Network Architectures." (2015)

[14] P. Y. Nikhil Kotecha, "Generating Music using an LSTM Network," arXiv.org,vol. arXiv:1804.07300, 2018.