# Creating Music using LSTM Neural Networks

*Training LSTM based RNN to generate automated music creation*

**Aditya Bhadauriya[1], Abhishek Kumar[2], Bharat Bhushan Anand[3], Ayush Gangwar[4], Ramveer Singh[5]**

**[1234]UG Student,[5]Assistant Professor, Department of Computer Science & Engineering**

Raj Kumar Goel Institute of Technology, Ghaziabad, Uttar Pradesh, India

## ABSTRACT

*Machine learning is a widely used technique in music processing, and this article focuses on the development of an LSTM-based recurrent neural network (RNN) for music the generation. The primary objective of this network is to identify relationships within MIDI files that capture musical chords and notes. The designed network architecture is compatible with various electronic devices, allowing musicians to create diverse music compositions. The implementation of this work involved leveraging the Python Keras framework and the Music21 library, built on top of the TensorFlow framework, to process and generate the required music files. The ultimate goal of this article is to train our neural network to enable automatic music composition.*

***Keywords-*** *Recurrent Neural Network, Long Short-Term Memory, Music Generation, Musical Instrument Digital Interface, ABC Notation*

## I.INTRODUCTION

### *A.RNN*

A recurrent neural network, known as an RNN, is a specialized type of neural network designed for processing sequential data. The term 'recurrent' indicates the network's ability to apply a shared set of weights recursively to a structure, such as a directed graph. RNNs have demonstrated significant proficiency in tasks involving natural language processing, making them particularly useful in this domain.[16].

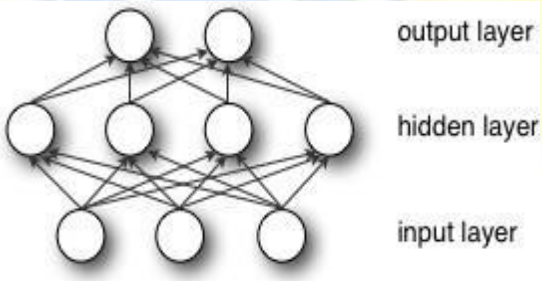


Fig. 1. Architecture of Neural Network

### *B. LSTM*

This article introduces a neural network model based on Long Short-Term Memory (LSTM) architecture. It utilizes a specialized variant of recurrent neural networks to address the issue of long-term dependency problems. LSTMs have proven to be highly effective in scenarios involving music and text generation, as they enable neural networks to capture and retain information over extended periods.

The decision made by the preceding step (i-1) in an RNN has a direct impact on the subsequent step (i). Therefore, RNNs function by combining inputs from multiple sources to produce an output.

Similar to the human brain, RNNs store information within hidden layers. The state of the hidden information at any given time depends on the current input (xt) multiplied by the weight matrix of the previous hidden layer information (ht-1). This allows RNNs to retain and process information in a sequential manner.

The process of memory forwarding can be represented mathematically as:

$$\mathbf{h}_t = \phi\left(W\mathbf{x}_t + U\mathbf{h}_{t-1}\right),$$

Fig. 2. Equation for memory forwarding. [8]

The core element of LSTM is the cell state, which can be likened to a conveyor belt. It remains relatively unaffected as it interacts with the entire chain going down, allowing information to flow through without significant alterations. To fine-tune LSTM models, specialized structures called gates are employed to selectively modify or add information to the cell state.

## LITERATURE REVIEW

1. Bob Sturm developed an LSTM-based character model [14] for generating textual representations of songs. This LSTM model consists of three distinct hidden layers, each containing 512 units.

2. In 2016, researchers introduced the Wave network, which leverages raw audio format files to generate music and voice. This network surpasses traditional text-to-speech systems by producing more natural-sounding results [15].

3. Doug Eck explored music composition using LSTMs [15]. In this approach, the network selects the same set of chords present in the sequence, and for each note, only a single node serves as input to predict the probability of playing that particular note. One notable limitation of this neural network is its incapability to accurately reconstruct notes.

## II.ABOUT LIBRARIES AND FRAMEWORKS USED

### *A.Music21*

Music21 is a popular Python-based application that is extensively utilized for managing music data. This versatile tool provides extensive functionality for describing various musical attributes and supports instrumentations that offer sheet music from diverse origins. Music21 also enables the construction of structures using nodes, allowing for flexible music representation.

In our approach, we leverage the capabilities of Music21 to extract a dataset primarily consisting of MIDI files. We utilize the note and chord objects provided by Music21 to serve as input for our neural network. Moreover, Music21 plays a vital role in converting the generated output from the neural network into sheet music, facilitating the visualization and interpretation of the expected musical results.

### *B. Keras*

Keras is a high-level API built on top of TensorFlow, which facilitates working with TensorFlow [9].

We have developed and trained our LSTM model-based network using the Keras framework. By utilizing the high-level TensorFlow APIs provided by Keras, we were able to streamline the coding process and enhance the overall efficiency of our implementation.

## III. METHOD

At a high level, our methodology involves inputting MIDI files containing music, predominantly consisting of Final Fantasy soundtracks. We train our tracker offline using these MIDI files, enabling it to learn from the musical patterns within the data. Subsequently, this trained tracker can be utilized to generate music at a later stage based on the acquired knowledge.

### *A. Input Format*

To implement the neural network, it is crucial to comprehend the input format required for the network.

- In our approach, we provide multiple MIDI files as input, which are then divided into two distinct types of objects: Chords and Notes.
- A note object obtained from Music21 comprises information about three essential aspects of music: offset, pitch values, and octave.

Here is an example excerpt showcasing the input format:

<music21.note. Note F>
<music21.chord. Chord B-2 F3>
<music21.note. Note E>
<music21.chord. Chord B-2 F3>
<music21.note. Note D>

Chord objects, on the other hand, serve as containers for sets of notes that are played simultaneously, representing chords in the music composition.

- Pitch: The quality of a sound that determines its highness or lowness, and it is closely related to the frequency of the sound. In musical notation, pitch is represented by letters from A to G. Each letter corresponds to a specific musical note with a distinct pitch.
- Octave: It is the interval between one musical pitch.

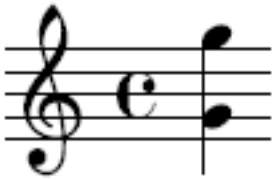

Fig. 3. An example of an octave, from G4 to G5 [11]

To generate music that flows harmoniously, our neural network must accurately predict the next note or chord to be played.

Additionally, it is crucial to consider the intervals between consecutive notes, as they can vary significantly. Musical compositions often exhibit patterns of rapid note transitions followed by moments of silence where no notes are played.

By utilizing Music21 to read MIDI files, we can analyze the intervals between successive notes. Typically, these intervals tend to be small, often around 0.5. For the purpose of our experiment, we can safely disregard such small offsets as they have minimal impact on the overall melodic structure of the music.

<music21.chord.Chord E3 A3> 77.5

<music21.chord.Chord F3 A3> 78.0

<music21.chord.Chord F3 A3> 78.5

## *B. Preparing the data*

In our data processing pipeline, we handle MIDI files that cover a wide range of musical genres. To process these MIDI files, we utilize the capabilities of Music21 library, which enables reading and parsing MIDI files, yielding note and chord objects. These note and chord objects serve as the input for our LSTM[x4] network. We employ the Music21 converter to convert each MIDI file, resulting in stream objects that contain all the individual sounds and chords present in the composition. To represent the pitch number, we encode it into string notation. When dealing with chords, we use a single string to capture the ID of each note, with a dot separating them. This encoding scheme facilitates the decoding of the network's output into meaningful notes and chords, enabling the generation of coherent music.

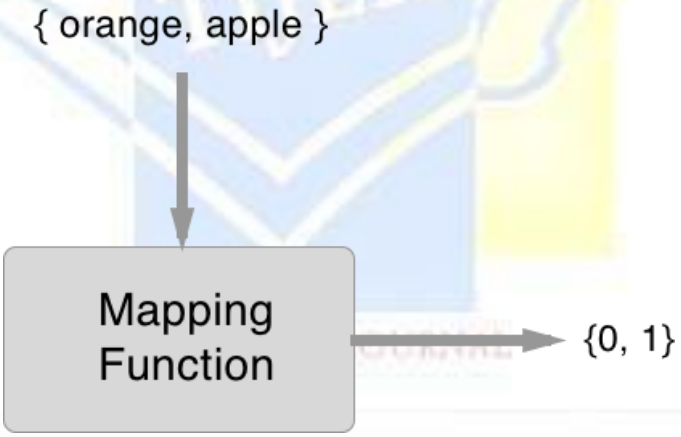


Fig. 4. Converting Categorical to Numerical data

Since our network much better with integer-based value, therefore we convert this categorical data to integer-based values using one hot encoding. [12].

In the figure shown above we have a mapping function which primarily act as one-hot encoder where the categorical values like orange, apple are one-hot encoded into integer values 0 and 1 respectively. These encoded values make the Machine learning to work efficiently than the string based categorical values.

```
encoded = to_categorical(data)
```

Fig. 5. One-hot encoding using keras

### *C. Network Architecture*

Our model consists of these layers.

● LSTM: A Recurrent Neural Network Traditional human brains learn by doing tasks repeatedly. Instead of starting from scratch every time, we learn skills that build on what we already know. This cannot be done by a conventional neural network. For instance, to comprehend the actions taking place at each frame in a video. Recurrent networks contain loops and are networks.

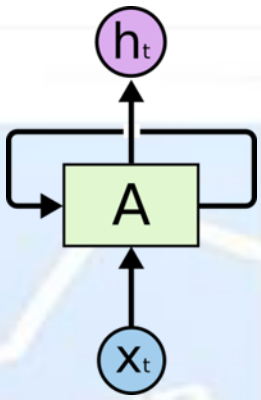


Fig. 6. A RNN's loop [13]

- Dropout layers: To reduce overfitting in neural network we are using Dropout technique.
- Dense layers: Is nothing but a fully connected neural layer connecting each input node to output node.
- The Activation layer.

### *D. Generating Music*

To create the soundtrack, we will use the same code that was used to train the model. The only change is that we will load the model with weights file instead of loading the notes and chords objects. The network will be built up in the same manner as before.

Although the starting point for our music is chosen at random from the list, if one wishes to control the starting point, a function can be made to replace the existing random function.

When creating music, you can choose any number of notes. We selected 1000 notes, which results in approximately 4 minutes of music. For each note we want to create, we give the network the sequence. A longer piece of music can be produced by choosing more notes, but it will take longer to create.

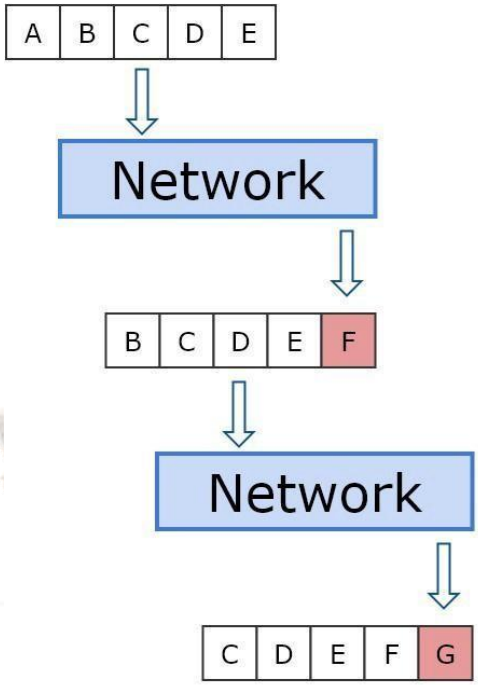


Fig. 7. Network selecting notes to generate music.

## IV.RESULT

A test_output.mid file is generated by the network which can be played on different applications like Apple Garage band. Some sample music generated can be listen from here https://soundcloud.com/poush12/music-lstm-1 [17]. There are some weird notes which can be seen in the output sheet. This is because of neural network incapable of making perfect melodies.

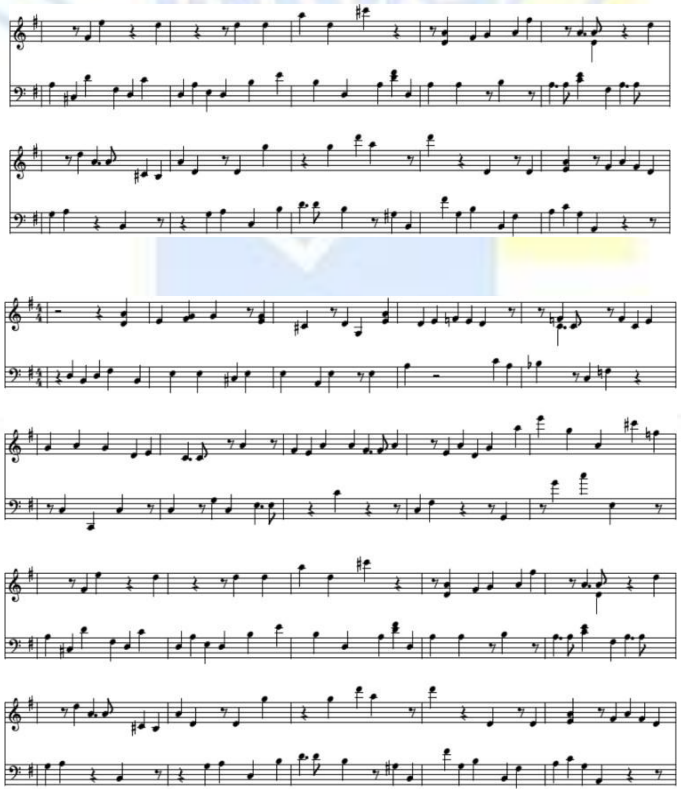

Fig. 8. Notes of the music generated by network

## V. LIMITATIONS

We achieved good results thanks to our LSTM network and 352 classes. However, it can be improved in many ways.

First of all, in order to protect our main communication base and be efficient at the same time, we have not taken into account the right time. To do this, there must be an additional class added for always and an additional class to mark the time.

Second, our network must understand how to handle unknown messages. Now, if our network encounters a document it does not recognize, it crashes. Find the letters that are equal to the unknown letters that will be the answer.

Many instruments can be added to the file to create different music, so far we've only tested it with one instrument.

## VI.CONCLUSIONS

To automate song creation, we use a simple LSTM based network to generate the Song. The results are very good, if not perfect, suggesting that neural networks can be used to make music and have the ability to create more music. A good model that has been found to record long-term prospects is the LSTM. Our network learns the music process and then collects the music to be recorded. Future research may focus on the differences between the lstm and ensemble models, which require powerful GPUs.

## VII.REFERENCES

[1] Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 (2014).

[2] Kleedorfer, F., Knees, P., Pohle, T.: Oh oh oh whoah! towards automatic topic detection in song lyrics. In: ISMIR. pp. 287–292 (2008).

[3] Hiller, L., Isaacson, L.M.: Experimental Music. Composition with an Electronic Computer. McGraw-Hill Book Company (1959).

[4] K. Choi, G. Fazekas, and M. Sandler, “Text-based LSTM networks for Automatic Music Composition”, 1st Conference on Computer Simulation of Musical Creativity, (2016).

[5] Laden, B. and Keefe, D. H. (1989). The representation of pitch in a neural net model of chord classification. Computer Music Journal, 13(4):44{53.

[6] Rothstein, J. (1992). MIDI: a comprehensive introduction. Oxford University Press, Oxford.

[7] Gers, F. A. and Schmidhuber, J. (2000). Recurrent nets that time and count. In Proc. IJCNN'2000, Int. Joint Conf. on Neural Networks, Como, Italy.

[8] https://deeplearning4j.org/lstm.htm

[9] https://keras.io/

[10] https://en.wikipedia.org/wiki/Octave#/media/File:Octave_example.png

[11] https://ahmedhanibrahim.wordpress.com/2014/10/10/data-normalization-and-standardization-for-neural-networks-output- classification/

[12] http://colah.github.io/posts/2015-08-Understanding-LSTMs

[13] Sturm, Santos and Korshunova, "Folk Music Style Modelling by Recurrent Neural Networks with Long Short-Term Memory Units", Late-breaking demo at the 2015 Int. Symposium on Music Information Retrieval.

[14] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," CoRR, vol. abs/1609.03499, 2016.

[15] Douglas Eck , Juergen Schmidhuber, A First Look at Music Composition using LSTM Recurrent Neural Networks, Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 2002.

[16] Socher, Richard; Lin, Cliff; Ng, Andrew Y.; Manning, Christopher D., "Parsing Natural Scenes and Natural Language with Recursive Neural Networks" (PDF), 28th International Conference on Machine Learning (ICML 2011).

[17] Jean-Pierre Briot, Gaetan Hadieres and François-David Pachet, "Deep Learning Techniques for music generation – A survey". [Accessed: September, 2017].