Deep Learning for Music Generation

Abhijeet Kumar[1] Akash Kumar[2] Aman Prakash[3]
[1,2,3]Department of Computer Science and Engineering
[1,2,3]Galgotias University, India

Abstract— The use of deep learning to mimics the working of the human brain by processing data for use in detecting objects, translating languages, recognizing speech and making decisions [4]. Recently, deep learning methods have achieved state-of-the-art results on examples of this problem and therefore, I think that Artificial Intelligence as a subject play an important role in this time. My colleague and I am a student of Computer Science and for the project we decided to build a Deep Learning for Music Generation. Our project deals with the generation of music using raw audio files (midi file) in the frequency domain relying on various architectures. Fully connected and convolutional layers are used along with LSTM's to capture rich features in the frequency domain and increase the quality of music generated. The hard amount of work can be reduced. We have tried to cover various aspects inside this project, and also tried to solve the major issues.

Keywords: Neural Networks, LSTMs, Convolutional layers, fully connected layers, notes, chords

I. INTRODUCTION

In this project we can compose the music that we can't compose in real life due to not knowing the instruments. But using *Deep learning techniques* and some frameworks, we can compose our own original music score. Here, we don't have to be a music expert to generate music.

Task: First we will take some existing music data and train our model using the existing data. Our modal will be able to generate music once it will be ready. It will understand the patterns of music and generate new music. The music will be not as of professional quality, but it will be a decent quality music and melodious to hear.

A. *Long Short Term Memory (LSTM)*

It is a collection of different frequencies. So, the Automatic music generation is a process of composing a short piece of music with minimum human interaction.

Music can be compose using WaveNet and LSTM (Long Short Term Memory).

It is variant of RNN that is capable of capturing the long term dependencies in the input sequence. LSTM has a wide range of applications in sequence-to-sequence modelling tasks like speech Recognition, Text summarization, video classification and so on.

– We use LSTM because Music is sequential data.

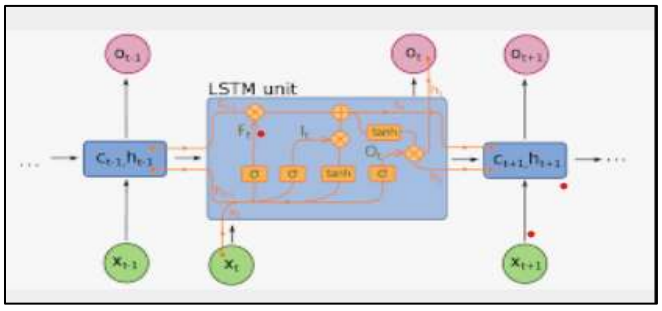

Fig. 2: LSTM structure [5].

Any Music has combination of both notes and chords in a sequence. So, if you put notes and chords in a sequence it will creates a music. Now it depends if that music would be a melodious music, how you are arranging them. How you are putting notes and chords. So, all the music has both the combination of notes and chords.

Let suppose we have a music file and we are going to extract it's content i.e. notes and chords and then we are going to feed that data in LSTM model so that model can capture the relationship how notes and chords are arrange to create a melodious song.

Fig. 3: Overview of flow of generated music

II. METHOD

A. *Training data (or) Input Data*

For this project there is requirement of input and training data so for this we have taken the midi files. It is consisting of single instrumental music like piano music. To extract the data from midi file we have used music21 library.

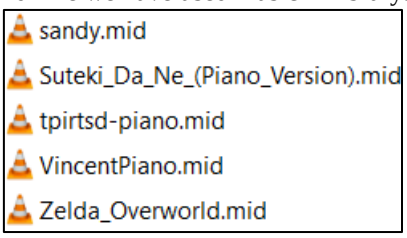

Fig. 3: Sample midi music data [1].

B. *Data Preparation*

As of now we have examined the data and we know that we want to use the notes and chords as the input and output of our LSTM network it's time to prepare the data for LSTM network.

Now we load the data into array [1].


```
notes = []

for file in glob.glob("midi_songs/*.mid"):
    midi = converter.parse(file) # Convert file into stream.Score Object

    print("parsing %s"%file)

    elements_to_parse = midi.flat.notes


    for ele in elements_to_parse:
        # If the element is a Note,  then store it's pitch
        if isinstance(ele, note.Note):
            notes.append(str(ele.pitch))

        # If the element is a Chord, split each note of chord and join them with +
        elif isinstance(ele, chord.Chord):
            notes.append("+".join(str(n) for n in ele.normalOrder))
```

After loading the each file with help of music21 library,
We got the list of notes and chords in the file. As per our project we got : Total notes- 60498

Unique notes- 359.

C. Preparing sequential data for model

Now appending the pitch of all the notes object by using its string notation because we can recreate the most significant parts of note into string notation of the pitch. We can append every chord by encoding with the id of every note in the chord together into a single string with each node being separated by a comma. This encoding will easily decode the output generated by the network into the correct notes and chords. Now we will create mapping between string-based categorical data to integer-based numerical data. We do this since neural network perform much better with integer-based numerical data than string-based categorical data. Now we will create input sequences for the network and with their respective outputs. Below code to prepare sequential data for our model LSTM [1].

```
sequence_length = 100

# get all pitch names
pitchnames = sorted(set(item for item in notes))

# create a dictionary to map pitches to integers
note_to_int = dict((note, number) for number, note in
enumerate(pitchnames))

network_input = []
network_output = []

# create input sequences and the corresponding outputs
for i in range(0, len(notes) - sequence_length, 1):
    sequence_in = notes[i:i + sequence_length]
    sequence_out = notes[i + sequence_length]
    network_input.append([note_to_int[char] for char in sequence_in])
    network_output.append(note_to_int[sequence_out])

n_patterns = len(network_input)

# reshape the input into a format compatible with LSTM layers
network_input = numpy.reshape(network_input, (n_patterns,
sequence_length, 1))
# normalize input
network_input = network_input / float(n_vocab)

network_output = np_utils.to_categorical(network_output)
```

1) Model

Here, we have used four different types of layers.

1) The Activation layer will be responsible for determining the activation function of our neural network that will use to calculate the output of a node.
2) Fully connected layers is a type of fully connected neural network layer in which each input node is connected to each output node.
3) LSTM layers is a type of RNN which takes input as a sequence and return a matrix [2].

D. Music Generation

Now, we have trained our LSTM model with data and we found the best weights. Now our model is ready to make the prediction. Since we have full list of note sequences at our disposal we will pick a random index in the list as our starting point, and this will allow us to rerun the generation code without changing anything and get different results every time. But if we wish to continue from starting point we can simply replace random function with a command line argument. We choose 200 notes using the network to generate one or two minutes of music. Now we create a stream of object from generated notes. And now we have output midi music file contain generated music and with the help of music21 function we will be able to get the music.

III. FURTHER SCOPE

We have achieved wonderful result in this project by Generating some melodious music by using simple LSTM network and 352 classes. However, there is always a scope of improvement in some areas [2].

First, the implementation of project which we have at the moment doesn't support if we vary the duration of notes and different offsets between notes. To achieve that we could add more classes for different duration and add rest classes that represent the rest period between notes.

Secondly, to achieve some more good results with more classes Added. We could have also Increase the depth of the LSTM network, which would require a more powerful computer because we can't train a model which have more depth with our ordinary computer. Also, to train my LSTM model on my computer which I use at my home it took more than sixteen hours. So, we can imagine now.

Third, the model can be trained with multi-instrument tunes. Because we know model is only trained with one type of Instrument. So, if we add some different instrument music To the model. It will be interesting to listen what type of music is getting generated.

Finally, a method can be added into the model which can handle unknown notes in the music file. The model can filter the unknown notes and can replace it by known notes to get more clarity in the generated music [2].

IV. CONCLUSION

We have now finally generated music with the help of LSTM neural network. The result may not be as perfect but we have done some great work with what system, technology we have. But we have potential to generate some complex music with multi-instrument. Future work can be done on the variants of LSTM and may be on different type of neural network which Will require high powered GPUs.

REFERENCES

[1] https://github.com/abhijeetgu/Deep-Learning-for-Music-Generation
[2] https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786834d4c5

[3] https://www.youtube.com/watch?v=5tvmMX8r_OM&list=PLtBw6njQRU-rwp5__7C0oIVt26ZgjG9NI
[4] https://deeplearning4j.org/lstm.htm
[5] https://medium.com/@vinayarun/from-scratch-an-lstm-model-to-predict-commodity-prices-179e12445c5a

