South Asian Music Generator using Deep Learning

Ayesha Saeed
Computer Science
Habib University
Karachi, Pakistan
as07519

Rania Siddiqui
Computer Science
Habib University
Karachi, Pakistan
rs07494

Haya Fatima
Computer Science
Habib University
Karachi, Pakistan
sf07503

I. Introduction

Music has the incredible power to transcend cultural boundaries and evoke emotions that words alone cannot. In this era of technological advancement, artificial intelligence and deep learning have opened new horizons for music generation. Whilst various musical instruments have found their place in these innovative projects South Asian classical music especially the soul-stirring sounds of sitar remains a majorly untapped territory of musical potential. The sitar with its mesmerizing melodies is an emblem of South Asian culture. It carries centuries of tradition and artistry within its strings. Our project emerges from a compelling need to bridge this gap in the world of music generation. By harnessing the capabilities of deep learning we aspire to create a model that can generate sitar music.

II. Research Question

Our research question revolves around the utilization of deep learning techniques for the generation of south Asian music. Specifically, we aim to explore the following query:
Can deep learning techniques be harnessed to unlock the creative potential of generating south Asian music?
We will explore a variety of deep learning models to see which model is the most suitable to address this music generation challenge. The input to the model consists of MIDI data representing music and the model aims to predict and generate the subsequent sitar.
The overarching goal of our project is to not only generate music but also to promote and celebrate the rich musical heritage of South Asia. Although existing research does generate music successfully, there is a notable scarcity of projects when it comes to generating South Asian music. Therefore by addressing this research question, we endeavor to create a unique platform that showcases the beauty of South Asian instruments such as Sitar and opens the doors of creativity for musicians while helping to keep this art form alive and thriving.
Some instruments we have included are:

- Sitar
- Tabla
- Bansuri

III. Literature Review

This section delves into a selection of studies that highlight the variety and fresh ideas in this field. These studies use different methods to generate music, achieving different results.

Deep learning for music generation has traditionally relied on Recurrent Neural Networks (RNNs) where the model takes MIDI files as input. We reviewed one such paper [1] where the methodology adopted centers on a Long Short-Term Memory (LSTM) model a variant of Recurrent Neural Networks (RNN). the model utilizes MIDI format piano recordings as training data. The goal is to make the program predict the next musical note based on what came before it. In terms of results, while the model successfully grasped certain structural and stylistic conventions of music, such as proper tallying of measures and recognizing repeated segments it still required human intervention for optimal output.

Another similar study [2] that uses LSTM delves into a brief overview and intuition behind it. It explains that LSTM networks were introduced to tackle the ”vanishing gradient problem” associated with simple RNNs. Simple RNNs tend to forget previous inputs as more inputs are fed into the network. LSTMs consist of two channels: a top channel that carries state information from one iteration to the next and a bottom channel that uses the previous output to modify the state of the top channel. Additionally, gates such as the ”forget” and ”remember” control the flow of information in LSTMs. The dataset for this paper are soundtracks in MIDI format from which relevant attributes of musical notes like pitch, rest status, and duration, are extracted and the input layer is formatted using one-hot encoding. The model employs supervised learning techniques where the model generates predictions by taking a random 50-note chunk from the training data and predicting the next 500 notes and durations. This series of notes was then transformed into MIDI. Regarding the quality of the results, the best-performing models were able to produce music that was equivalent in quality to that of a human composer after having human judges evaluate each model’s outputs. However, one limitation was that the generated music still has room for improvement in terms of creating more complex and varied musical structures. Overall, the outcomes were encouraging and showed the potential of LSTM networks for producing music.

Nonetheless, there are studies that deviate from the usual approach of using recurrent neural networks (RNNs). One such is [3] which introduces MidiNet which utilizes convolutional neural networks (CNNs) in conjunction with a generative adversarial network (GAN) framework. The network takes MIDI files as input typically in the form of sequential note data which is transformed into 2-D matrices representing note sequences broken down into bars. The core of MidiNet is the CNN component which operates on these bar-level note sequences. Since CNNs capture spatial dependencies, to address temporal dependencies across different bars MidiNet introduces a conditional mechanism that leverages information from previous bars. This allows the model to ”look back” without using traditional recurrent units found in RNNs. The GAN framework consists of a generator and a discriminator. The generator's role is to create new MIDI melodies as it takes random noise vectors as input and passes them through fully-connected layers before using four transposed convolution layers to generate music. whereas the discriminator's task is to distinguish between generated and real melodies. This adversarial setup compels the generator to continually improve its ability to generate convincing melodies. To evaluate the performance of MidiNet a user study was conducted and MidiNet outperformed Google's MelodyRNN in terms of being interesting and was found to be comparable in aspects of being pleasant and realistic.

Traditionally structured MIDI files are used as inputs in music generation projects however a study [4] explores the generation of music directly from raw audio files to leverage the unstructured audio data from the internet. To achieve this the research paper makes use of both CNN and RNN specifically LSTM to enhance the quality of generated music by capturing both spatial and temporal features of the audio data. While previous research has focused on using LSTM networks to generate music based on musical features from MIDI files these approaches are limited by the low dimensionality of their input vectors. In contrast, this paper aims to address limitations in a more recent and faster method and focuses on the frequency domain but the problem is this method lacks coherent structure and has noise disturbances. The study conducts a comparative analysis between a baseline implementation consisting of an input layer, an LSTM layer, and an output layer- and several other techniques. These techniques include an additional fully connected layer with dropout, a couple with convolutional layers with max pooling and LSTM and variations with multiple LSTM layers such as stacking two LSTM layers in sequence and employing a bilinear LSTM architecture. It is observed that convolutional layers effectively capture local connections between frequencies but the quality of the generated music is compromised. This indicates that frequencies in music exhibit complex non-local dependencies. In conclusion, all models exhibited enhanced quality compared to the base model with the bilinear architecture and LSTM with 2D convolutional layers yielding the highest quality audio outputs.

Another model we came across was Transformers. The paper we looked at for this [5] used MIDI files as inputs which are first preprocessed then converted into a series of discrete tokens using a vocabulary that includes NOTE ON events for starting a note, NOTE OFF events for ending a note, TIME SHIFT events for representing forward time shifts, and SET VELOCITY events for representing the velocity of subsequent NOTE ON events. The Music Transformer model then takes these tokens as input and creates a new series of tokens that simulates a minute-long musical composition; these tokens are translated back into MIDI format. The transformer model makes use of a self-attention mechanism to analyse long-range dependencies and concentrate on various input sequence segments. It employs a technique known as ”relative position representations” to use learned parameters rather than manually computing these locations for each pair of elements. The output values of the Music Transformer are MIDI notes, and the chance of each note being played at each time step is calculated using the softmax function. The following note will be sampled using this probability distribution and supplied back into the model as input for the following time step. The Music Transformer model produced several excellent outcomes, including minute-long compositions with strong structure, continuations that cogently build on a given theme, and accompaniments that are conditioned on melodies in a seq2seq arrangement.

In addition, we came across a study [6] that discusses the use of spectrograms for generating audio using neural networks particularly CNNs for audio-style transfer. The paper highlights that unlike visual data, choosing an appropriate representation for audio is more complex and application-dependent, The study then goes on to review different audio data representations including hand-crafted features, machine-discovered features, Mel Frequency Cepstral Coefficients (MFCCs), and spectral representations. According to this paper, spectrograms are a good choice for generative applications due to their ability to retain more information than most hand-crafted features and their lower dimensionality compared to raw audio. Spectrograms present audio characteristics as 2D images with time and frequency axes and this similarity to images opens the door to applying convolutional neural network (CNN) techniques originally designed for visual data to audio. However, the paper also acknowledges the challenges posed by audio objects' non-local distribution and frequency-dependent properties within spectrograms suggesting that innovative approaches are needed to harness the full potential of spectrograms in audio-style transfer applications.

In essence, these studies collectively emphasize the multifaceted landscape of deep learning in music generation showcasing a spectrum of methodologies from LSTM-based networks to CNNs to the combination of both.

For the selection of a balanced model to suit the needs of this project, we aim to combine Convolutional Neural Networks (CNNs) and sequential processing to capture temporal dependencies. By using different filter sizes and dilations, we can capture patterns at different scales and time intervals. The sequences of notes will be fed into the network in a sliding window fashion, where each time step depends on the previous steps. This sequential processing is a form of recurrent behavior, even though it's implemented using CNNs. The model will be trained to remember the past notes in the sequence and use them to predict the next note. The most crucial part of this will be the embedding layer. It will convert the discrete note data (like "C4," "D5," etc.) into continuous, dense vectors. This is a common preprocessing step for natural language processing tasks but is also useful in music generation. These dense layers with ReLU activation are used to capture non-linear relationships in the data and make the final predictions for the next note in the sequence. Finally, the model will be trained using the sparse categorical cross-entropy loss and the Adam optimizer, whereas checkpointing will be used to save the best model weights during training.

IV. Methodology

A. Data Set

In this section, we discuss the nature of our data, its acquisition and how it is processed to be the input of our model.

1) Summary:

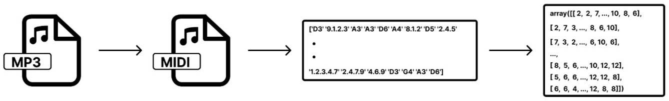
a) Positioning Figures and Tables: We collected raw mp3 audio data from YouTube after rigorous searching for clear music with audible notes. The data was downloaded and converted to '.wav' format using online converters manually. It was then divided into 15-second music segments to make processing more manageable.

These 15-second '.wav' files were converted into '.midi' files manually using online converters. We tried processing the data into '.midi' format through Python libraries, however, that was out of our scope and it proved to be a limitation on google colab as well. We also created spectrograms of each segment and RGB matrices of each spectrogram as input to a model that generates image-to-image.

<table>
<tr><th>Src</th><th>Video Name</th><th>Len</th></tr>
<tr><td>1 [7]</td><td>Morning Meditation Ragas On Sitar - B. Sivaramakrishna Rao</td><td>21:57</td></tr>
<tr><td>2 [8]</td><td>Indian Sitar Instrumental Music 10 Hours</td><td>10:00:00</td></tr>
<tr><td>3 [9]</td><td>Sitar, Bansuri Tabla trio / Rishab Prasanna & Sandip Banerjee & Nicolas Delaigue</td><td>28:41</td></tr>
<tr><td>4 [10]</td><td>Heal Ragas || flute and sitar by Hariprasad chorsia|| calm mind|| Thumri in raag bhairavi</td><td>13:35</td></tr>
<tr><td>5 [11]</td><td>Pandit Ravi Shankar - Morning Meditation Ragas On Sitar|Indian Classical Instrumental Music</td><td>52:43</td></tr>
</table>

TABLE I
Data Sources

Next, the midi data's notes and chords are extracted using music21 library. Then, the unique notes are normalized to unique integers. Frequent notes are identified and stored with their frequency to reduce vocabulary volume. This data is our main input.



```
['D3' '9.1.2.3' 'A3' 'A3' 'D6' 'A4' '8.1.2' 'D5' '2.4.5'
•
•
'1.2.3.4.7' '2.4.7.9' '4.6.9' 'D3' 'G4' 'A3' 'D6']
```

```
array([[ 2,  2,  7, ..., 10,  8,  6],
[ 2,  7,  3, ...,  8,  6, 10],
[ 7,  3,  2, ...,  6, 10,  6],
...,
[ 8,  5,  6, ..., 10, 12, 12],
[ 5,  6,  6, ..., 12, 12,  8],
[ 6,  6,  4, ..., 12,  8,  8]])
```

Fig. 1. Data Processing Timeline

2) Experimentation: Our initial attempts involved experimenting with Long Short-Term Memory (LSTM) models. In this exploration, we constructed a neural network architecture comprising two LSTM layers, along with additional dense layers. The model was designed to predict the next musical note in a sequence. However, despite its theoretical soundness, the generated audio quality did not meet our desired standards. This led us to further exploration and refinement of our model architecture to achieve improved results.

3) Model Used: Our music generation model employs a deep learning architecture that leverages convolutional neural networks (CNNs) for processing sequential data, specifically musical notes. This model architecture is quite innovative, as CNNs are traditionally used for image processing but here are adapted for sequential, time-series data i.e. music. A single input to the network is a one-dimensional vector with dimensions (1, 32), representing a sequence of 32 musical notes. This sequence acts as the input to the model, which in turn, predicts the subsequent note in the sequence. The neural network model is trained on the prepared input sequences and their corresponding output notes. Training involves optimising the model's internal parameters to minimise the difference between predicted and actual notes. The model continues to learn and adjust its internal representations of the data during training.

Let's delve into the layers and their specific roles in the model:

Embedding Layer:

1) The first layer in the model is an Embedding layer, which is crucial for handling the integer-encoded input data. This layer transforms the integer-encoded notes into dense vectors of fixed size (100 in our model). It's a way of representing discrete variables like musical notes in a continuous, high-dimensional space, which helps the model capture more complex relationships between notes.
2) The input length is set to 32, corresponding to the chosen sequence length for the model. This means the model looks at 32 notes at a time to make its next note prediction.

Convulational layers (CONV 1D)

1) Following the embedding layer, the model uses multiple 1D convolutional layers (Conv1D). These layers are

adept at extracting higher-level features from the sequential data. Unlike their 2D counterparts used in image processing, 1D convolutions slide along one dimension, making them suitable for time-series data like music.

2) The model employs three Conv1D layers with increasing dilation rates. Dilation in convolutions helps in expanding the receptive field of the filters, allowing the model to incorporate a broader context of input notes without increasing computational complexity.
3) Each Conv1D layer uses the 'relu' activation function, which helps in introducing non-linearity to the model, enabling it to learn more complex patterns in the data.

DropOut Layers

1) After each Conv1D layer, there is a Dropout layer with a dropout rate of 0.2. Dropout layers randomly set a fraction of input units to 0 at each update during training, which helps in preventing overfitting. By dropping different sets of neurons, it ensures that the network doesn't rely too much on any one node, promoting a more robust learning.

Max Pooling Layers(Max pooling 1D)

1) Max pooling layers are used after some of the Conv1D layers. These layers reduce the dimensionality of the input, which helps in reducing computation and also in controlling overfitting. The pooling operation provides an abstracted form of the representation.
2) In our model, MaxPool1D with pool size 2 is used, which reduces the dimensionality of the output from the previous layers by half.

Global Max Pooling Layer:

1) A GlobalMaxPool1D layer follows the final Conv1D layer. This layer takes the maximum value over the time dimension, effectively reducing each feature map to a single number and thus flattening the output. It helps in reducing the total number of parameters and computation in the network.

Dense Layer

1) Towards the end of the model, there are Dense layers, which are fully connected neural network layers. The first Dense layer has 256 neurons and uses 'relu' activation, serving as a fully connected layer that learns from the features extracted by the previous layers.
2) The final Dense layer serves as the output layer of the network. It has a number of neurons equal to the number of unique output classes (i.e., unique notes) and uses a 'softmax' activation function. This layer outputs a probability distribution over all possible next notes, from which the next note in the sequence is predicted.

Comparison of CNN models with other models

1) RNNs: These models have been extensively used in music generation due to their ability to handle sequential data. They maintain an internal state while processing sequences, enabling them to capture long-term dependencies. However, they suffer from issues like vanishing gradients and struggle to capture very long-term dependencies.
2) LSTM/GRU: Variants of RNNs designed to alleviate the vanishing gradient problem. LSTMs and GRUs have memory cells that can retain information over longer sequences. While effective, they can still face challenges in capturing global dependencies across lengthy music compositions.
3) GANs: GANs use a generator and a discriminator to create realistic music. While they can produce high-quality samples, training GANs can be challenging due to instability issues, and controlling the generated output might be more complex.

Why did we choose CNN model?
Now that we have compared CNN with other music generation models, let's discuss why 1D CNN model is a preferred model over others. CNNs, as opposed to RNN-based models, excel in identifying local patterns. CNNs are particularly good at recognising short-term patterns in music, such as chord progressions or rhythmic themes. Moreover, due to the hierarchical nature of CNNs, with multiple layers capturing increasingly abstract features. Due to this feature, it can learn both local and global music structures effectively. Moreover CNNs also have parallel processing capabilities that are computationally more efficient compared to RNNs, especially when we deal with large scale music datasets. The biggest issue that is often encountered is that RNNs and their variants struggle with vanishing gradient problems over long sequences. CNNs on the other hand, particularly with dilated convolutions, can capture long term dependencies without suffering from any of such issues.

V. Results and Discussion

A. *Quantitative Evaluation*

Though we initially attempted to assess accuracy based on validation data, we found no direct correlation between the model's accuracy and the quality of the generated music. As a result, we shifted our evaluation towards a qualitative approach for a more comprehensive assessment.

B. *Qualitative Evaluation*

The qualitative evaluation was conducted by Shehroze Hussain, an experienced Sitar Instructor with 18 years of expertise in South Asian music.

1) Positive Feedback: Mr. Hussain provided positive feedback on the generated music, noting the following:

- Presence of notes frequently found in South Asian instruments like Sitar and Bansuri.
- Correct tempo, note arrangement, and sequencing observed in the generated music.

2) Areas for Improvement: He also highlighted areas for improvement:

- Limitations in capturing nuances of frequencies through MIDI files.
- Lack of coherence among notes and absence of layered frequencies.

C. Utilization

1) Improvisation Derivation: The potential of MIDI patterns in guiding improvisation techniques in South Asian music was recognized. Suggestions were made to adjust the generated music to a specific scale to facilitate improvisation.

2) Rhythmic Exploration: Recognizing the significance of rhythm in creating groove and patterns, Mr. Hussain proposed that experimental improvisation within the domain of rhythm can be a potential use for the generated music.

VI. Conclusion

The project focused on leveraging deep learning techniques, specifically CNNs, to generate South Asian music, primarily focusing on instruments like the Sitar, Tabla, and Bansuri. Through extensive literature review and experimentation, the model architecture was refined to suit the sequential nature of musical data.

Qualitative evaluation by a music expert highlighted the model's success in capturing certain nuances of South Asian music, including note arrangement and tempo. However, areas for improvement were identified, such as the limitations of MIDI files in capturing frequency nuances and the need for coherence among notes and layered frequencies.

The model's potential utilization in guiding improvisation techniques and exploring rhythmic patterns within the realm of South Asian music was acknowledged.

In conclusion, while the model shows promise in capturing some characteristics of South Asian music, there's room for further refinement to enhance its ability to generate more nuanced and coherent musical pieces, especially in replicating the intricacies of instruments like the Sitar and Bansuri. The project serves as a foundation for further research and development in this domain, aiming to preserve and promote the rich musical heritage of South Asia through AI-generated compositions.

References

[1] A. Joshi, A. Gokhale, A. Khandekar, A. Kesar, S. Khade, and N. Tarapore, "Music Generation Using Recurrent Neural Networks," International Journal for Research in Applied Science and Engineering Technology, vol. 10, no. 12, pp. 1352–1358, Dec. 2022, doi: https://doi.org/10.22214/ijraset.2022.48200

[2] M. Conner, L. Gral, K. Adams, D. Hunger, R. Strelow, and A. Neuwirth, "Music Generation Using an LSTM." arXiv, Mar 22, 2022. doi: https://doi.org/10.48550/arXiv.2203.12105

[3] L. Yang, S. Chou, and Y. Yang, "MidiNet: A Convolutional Generative Adversarial Network for Symbolic-domain Music Generation,"Research Center for IT innovation, Academia Sinica, Taipei, Taiwan , Jul. 2017, doi: https://doi.org/10.48550/arXiv.1703.10847

[4] V. Kalingeri and S. Grandhe, "Music Generation with Deep Learning." arXiv, Dec 15, 2016. doi: https://doi.org/10.48550/arXiv.1612.04928

[5] C.-Z. A. Huang et al., "Music Transformer," arXiv, Sep. 2018, doi: 10.48550/arxiv.1809.04281.

[6] L. Wyse, "Audio Spectrogram Representations for Processing with Convolutional Neural Networks." arXiv, Jun 28, 2017. doi:.https://doi.org/10.48550/arXiv.1706.09559

[7] @GeethanjaliClassicalMusic. "Morning Meditation Ragas On Sitar - Peaceful Music for Relaxation - B. Sivaramakrishna Rao" YouTube, Mar 8, 2017. Available: https://www.youtube.com/watch?v=pG9HcnkXIB0&ab_channel=Geethanjali-IndianClassicalMusic. [Accessed: Oct, 2023].

[8] @RelaxCafeMusic. "Indian Sitar Instrumental Music 10 Hours" YouTube, Feb 8, 2020. Available: https://www.youtube.com/watch?v=D6B4xo6zYdk&ab_channel=RelaxCafeMusic. [Accessed: Oct, 2023].

[9] @SalonZaENOglasbo. "Sitar, Bansuri Tabla trio / Rishab Prasanna & Sandip Banerjee & Nicolas Delaigue" YouTube, Jul 5, 2016. Available: https://www.youtube.com/watch?v=9XERKkmrO8c&ab_channel=SalonZaENOglasbo. [Accessed: Oct, 2023].

[10] @Rang Rang Bharat. "Heal Ragas || flute and sitar by Hariprasad chorsia|| calm mind|| Thumri in raag bhairavi," www.youtube.com. https://www.youtube.com/watch?v=LLW5dVMh8oM (accessed Dec. 04, 2023).

[11] @Saregama Hindustani Classical. "Pandit Ravi Shankar - Morning Meditation Ragas On Sitar | Indian Classical Instrumental Music," www.youtube.com. https://www.youtube.com/watch?v=rQphif7Lh8Et=1014s (accessed Dec. 04, 2023).