(ISSN: 2582-5526) www.thevoiceofcreativeresearch.com

The Voice of Creative Research

Vol. 7 & Issue 2 (April 2025)

https://doi.org/10.53032/tvcr/2025.v7n2.40

Research Article

Music Creation Using a Hybrid Model of LLM And LSTM

Kashish Bhatia,
Student of Department of Computer Science,
University of Mumbai, Vidya Nagari, Kalina,
Santacruz East
Kashish2000.kb@gmail.com

Dr. Sampada Malhar Margaj,
Assistant Professor Department of Computer Science,
Kirti M. Doongursee College,
Dadar, Mumbai, Maharashtra, India
Sampada.malhar@gmail.com

Dr. Jyotshna Jagdeo Dongardive
Associate Professor,
Department of Computer Science, University of Mumbai,
Mumbai, Maharashtra, India
jyotss.d@gmail.com

Abstract

Over the years music generation using artificial intelligence and deep learning have undergone lots of progress. There are platforms such as Magneta, MuseNet, Deep Back which incorporates deep learning and LSTM models in their architecture. Along with that platform such as music, riffusion, MIDI which incorporates LLM based learning methods. These models are useful, but there are very few researches which incorporate LLM and LSTM architecture for music generation and improve their individual limitations and combine their strengths. Hence, this will be the major highlight of the research. The LLM based models offer sequential structural richness and LSTM ensures temporal consistency. The approach will involve encoding MIDI sequences into English text format, allowing the LLM to process musical structures with natural language processing techniques. The output is then refined using an LSTM-based decoder to enhance the coherence of note transitions and rhythmic consistency. The hybrid model approaches to discover the potential of combining sequence modelling for improved AI driven music composition, which can make an interesting blend between creativity and machines.

Keywords: Hybrid model, LLM, LSTM

1. Introduction

Artificial Intelligence (AI) has maintained a connection with music since the 1950s, when early experiments explored its potential in creative tasks. Recent advancements in Large Language Models (LLMs) have significantly enhanced this relationship, introducing innovative approaches to music composition. By leveraging powerful machine learning techniques, LLMs have become instrumental in transforming creative processes, including music generation, writing, and visual arts. This new wave of AI-driven creativity is redefining the boundaries of artistic expression, offering composers and producers advanced tools for generating complex and emotionally rich musical pieces.

1.1 LLM and AI – 1

Large Language Models (LLMs), such as GPT-3, are powerful deep learning models trained on extensive text datasets. This training equips them with the ability to comprehend and generate human-like language. By applying natural language processing (NLP) techniques, these models excel at tasks like text creation, translation, and summarization. LLMs are designed to analyze the structure and flow of language by identifying patterns and relationships across words, sentences, and paragraphs. This enables them to produce text that closely mirrors human writing in both style and content, making them highly versatile for various language-driven applications.

1.2 AI and music – 2

Artificial intelligence (AI) is increasingly playing a significant role in helping musicians compose, refine, and produce music. Several artists who have spoken with *TIME* have highlighted how AI is being integrated into the music creation process. Dromgoole, a musician and co-founder of the AI company Bronze, envisions AI music evolving beyond its current focus on replicating singers' voices or generating tracks instantly. Bronze has collaborated with artists such as Disclosure and Jai Paul to develop AI-driven music that constantly changes with each playback, ensuring no two performances sound identical. Rather than aiming to produce flawless, commercially static songs, the objective is to use AI as a tool to expand creative boundaries and redefine how we perceive music.

1.3 Research Question – 3

We'll briefly look at the process of music generation, the models implement and the datasets used and what all challenges did these models face and how can these models be significantly improved. There are many questions that this topic poses and a major question is about can it mimic the human emotional intelligence as creating any type of art is much more that just learning the science and skill behind it.

2. Background and Literature Review

2.1 History of intersection of Music and AI

The concept of AI-generated music dates back to the 1960s, coinciding with the introduction of the term "Artificial Intelligence" in the 1950s. This marked a pivotal moment for both AI and music composition, although early attempts often resulted in simple melodies with limited emotional depth. The first recognized instance of computer-generated music emerged in 1957 with The Silver Scale, a brief 17-second melody created by Newman Guttman. This piece was produced using Music I, an innovative sound synthesis program developed by Max Mathews at Bell Laboratories. That same year, The Illiac Suite became the first complete musical composition created by a computer, utilizing the ILLIAC I computer at the University of Illinois at Urbana-Champaign (UIUC).

Over time, AI-driven music technology has advanced considerably, expanding into diverse applications such as generating melodies in various genres, writing lyrics, and more. In the 21st century, these advancements have accelerated, resulting in breakthroughs like AI-assisted music mastering, mixing, and even autonomous music composition. A significant example is OpenAI's Jukebox, introduced in 2020, which employs neural networks to generate raw audio, further revolutionizing the landscape of AI-generated music.

2.2 What is the data type of music?

Music is inherently sequential data, characterized by various long-term patterns such as repetition, retrograde, sequences, and call-and-response structures. Effectively capturing these long-term dependencies is vital for both music analysis and generation tasks.

Sequential data refers to information that is organized in a particular order, where each data point is influenced by the preceding ones. Unlike traditional tabular data, where the sequence of rows is often irrelevant, the order in sequential data is essential as it reflects temporal or sequential relationships between data points. This unique property makes sequence order a critical factor in analyzing and interpreting such data.

Key Features of Sequential Data

- **Temporal Dependency**
  In sequential data, future values are frequently influenced by past values. For instance, in stock price time series data, the price at time t is impacted by previous price points. Capturing this temporal dependency is crucial for accurate forecasting and pattern recognition.
- **Variable Length:**
  Sequences can vary in size. For example, text data may contain sentences of different lengths, and time series data may span varying time intervals. Managing variable-length sequences often requires techniques like padding or truncation to ensure consistency during batch processing.
- **Correlation:**
  Data points within a sequence are often closely related. For instance, words in a sentence are contextually linked, and consecutive frames in a video share similarities in motion and content. Recognizing these correlations is fundamental for effective modelling and analysis.

AI-driven music technology has advanced considerably over time, resulting in innovative approaches to music creation across multiple genres, as well as lyric writing and other related tasks. The 21st century has seen impressive progress in this field, with developments such as AI-assisted lyric generation, music mastering, and mixing. One prominent example is OpenAI's *Jukebox*, introduced in 2020, which leverages neural networks to produce raw audio data, further expanding the possibilities of AI-generated music.

2.3 Introduction to Deep Learning and LLMs

Deep learning is a specialized area within machine learning, which itself is a subset of artificial intelligence. Since neural networks are designed to mimic the functioning of the human brain, deep learning follows a similar approach. Unlike traditional programming methods, deep learning models are not explicitly coded for specific tasks. Instead, they rely on multiple nonlinear processing layers to extract and transform features from data. Each layer processes the output of the previous one, enabling the model to learn complex patterns and representations effectively.

Deep learning models are highly effective at identifying relevant features with minimal guidance from the programmer, making them particularly useful for addressing challenges related to high-dimensional data. These algorithms excel in scenarios involving a large number of inputs and outputs.

As deep learning stems from machine learning, which itself is a subset of artificial intelligence, its core objective aligns with mimicking human behaviour. Similarly, the goal of deep learning is to create algorithms that replicate the functioning of the human brain.

Neural Networks are the foundation of deep learning, inspired by biological neurons — the fundamental units of the human brain.

Deep learning leverages a set of advanced machine learning techniques designed to build feature hierarchies using artificial neural networks. In recent years, it has rapidly gained traction and is now widely applied in tasks such as image classification, speech recognition, and language translation. The field gained significant attention in 2012 when a deep learning model outperformed traditional methods relying on handcrafted features in an image classification competition.

This remarkable success and the renewed interest in neural networks can be attributed to a combination of factors, including improved algorithms, enhanced computational power, and access to larger datasets:

- availability of massive data;
- availability of efficient and affordable computing power;
- technical advances, such as:
    1. pre-training, which resolved initially inefficient training of neural networks with many layers
    2. convolutions, which provide motif translation invariance
    3. LSTM (long short-term memory), which resolved initially inefficient training of recurrent neural networks

Language models are designed to understand, summarize, translate, predict, and generate text and other forms of content. They are referred to as "large" because they are trained on extensive datasets and contain vast numbers of parameters, with some popular models reaching hundreds of billions. These models rely on deep learning techniques to process and analyse massive collections of data, including books, articles, and web content.

To address the challenge of vanishing and exploding gradients in Deep Recurrent Neural Networks, several variations were introduced, with one of the most prominent being the Long Short-Term Memory network (LSTM). An LSTM unit is designed to "remember" important information from past inputs while discarding irrelevant data. This is achieved through specialized activation layers known as "gates," each serving a distinct purpose. Additionally, every LSTM unit maintains an Internal Cell State, a vector that stores information selected to be preserved from the previous LSTM unit, ensuring effective information flow throughout the network.

A Long Short Term Memory Network consists of four different gates for different purposes as described below:-

- In the forget gate, the current input is combined with the previous output to produce a value between 0 and 1. This value determines the extent to which information from the previous state should be retained or discarded. The resulting value is then multiplied by the previous state. An activation value of 1 indicates that all previous information should be remembered, while a value of 0 signifies that all prior information should be forgotten. Interestingly, the forget gate could also be viewed as a "remember gate," as it controls how much past information is preserved.
- **Input Gate (i):** The input gate functions similarly to the forget gate, but its purpose is to determine which new information should be added to the LSTM's state. It generates an output value between 0 and 1, which is then multiplied by the output of a tanh activation block that produces the new data intended for inclusion in the state. This resulting vector is added to the previous state, contributing to the formation of the current state.
- **Input Modulation Gate (g):** Often regarded as a component of the input gate, the input modulation gate is sometimes overlooked in LSTM literature, with its functionality assumed to be integrated within the input gate itself. Its role is to refine the information that the input gate writes to the Internal Cell State by introducing non-linearity and ensuring the data has a zero-mean distribution. This zero-mean property helps accelerate the learning process by improving convergence speed. Although the input modulation gate is considered less critical than other LSTM components and is often seen as a refinement feature, incorporating it into the LSTM structure is considered a good practice.
- **Output Gate (o):** The output gate regulates the flow of information by combining the current input with the previous state to produce a scaling factor. This factor is then multiplied by the output of a tanh activation block, which processes the current state.

The resulting value is then delivered as the output. Additionally, both the output and the updated state are fed back into the LSTM unit for further processing.

2.4 Architecture of LLMs

Large Language Model (LLM) is a sophisticated artificial intelligence model designed to comprehend, produce, and engage with human language. The term "large" refers to the extensive training these models undergo, utilizing massive text datasets that often include billions of parameters. This large-scale training equips LLMs with the ability to perform various language-based tasks, such as answering questions, generating content, translating text, and more.

- **Transformers: The Foundation of LLMs**
  The Transformer architecture, introduced in 2017, serves as the foundation for most modern Large Language Models (LLMs). Its standout feature is the self-attention mechanism, which enables the model to focus on various parts of the input sequence while predicting the next token. This mechanism grants the model a comprehensive understanding of each word’s context, enhancing its ability to track dependencies across lengthy text sequences.
  Transformers are composed of multi-head attention, position-wise feed-forward networks, and residual connections. This architecture empowers them to effectively capture complex word relationships, enabling LLMs to process extensive text data with remarkable efficiency and precision.
- Key Models and Differences (BERT, GPT, and T5)Permalink Each of these models brings unique strengths to NLP tasks:
  1. BERT (Bidirectional Encoder Representations from Transformers): BERT is designed to grasp the complete context of words by analyzing both the preceding and following text simultaneously (bidirectionally). This bidirectional approach proves especially effective for tasks such as text classification and question-answering, where a thorough understanding of language is essential.

3. **GPT (Generative Pre-trained Transformer):** GPT models follow an autoregressive approach, generating text sequentially from left to right by predicting the next word using the previous words as context. This characteristic makes them well-suited for generative tasks like text completion and storytelling.
4. **T5 (Text-To-Text Transfer Transformer):** T5 approaches tasks by converting them into a text-to-text format, where both inputs and outputs are treated as text sequences. This versatile design allows T5 to excel in a broad spectrum of natural language processing (NLP) tasks.
5. **Attention Mechanisms** The core of Transformer-based models lies in the attention mechanism, which allows the model to concentrate on words that hold contextual significance. Through the self-

attention process, each word can reference all other words in a sentence, creating more comprehensive language representations.

- Multi-head attention expands this process by running multiple attention mechanisms in parallel, with each head attending to different parts of the context. This method proves especially valuable for complex tasks that demand understanding long-range dependencies, like summarizing lengthy texts.
- **Encoder-Decoder Architectures vs. Autoregressive Models**
  Large Language Models (LLMs) are commonly built using two primary architectures:
  1. **Encoder-Decoder Models:** Widely used in tasks like translation, encoder-decoder models (e.g., T5) process an entire input sequence through the encoder and produce a new sequence via the decoder. This structure is ideal for tasks involving sequence-to-sequence transformations.
  2. **Autoregressive Models:** Models such as GPT generate text progressively, predicting one token at a time using previous tokens as context. This method is particularly effective for text generation tasks where maintaining coherence across multiple sentences is essential.

3. Methodology using Deep Learning and LLM for music generation

In this section we will look at how LSTM and LLMs work differently in generating music using different case studies for each of the methods.

3.1 Music generation using LSTM - 1

Long Short-Term Memory (LSTM) networks are highly effective in music generation because they excel at modeling sequential data and capturing long-term dependencies. Since music relies heavily on temporal structures, LSTMs are well-suited for creating compositions that are both coherent and contextually meaningful.

Studies have highlighted the effectiveness of LSTMs in generating music. For example, in the research paper *"LSTM Networks for Music Generation,"* Xin Xu examines how various LSTM architectures influence the quality of generated music.

Likewise, the research titled *"Music Generation Using an LSTM"* by Michael Conner and colleagues explores the use of LSTM networks in predicting musical elements, emphasizing their strength in managing the intricate patterns found in musical sequences.

3.2 Understanding the architecture of LLM for Music

- The architecture described in the paper "Music Generation Using an LSTM" by Michael Conner et al. utilizes a structured design featuring three LSTM layers alongside two densely-connected layers. Each LSTM layer is composed of 512 hidden units, and to mitigate overfitting, a dropout rate of 0.3 is applied following the first two LSTM layers and the initial densely-connected layer. The first dense layer contains 256 units, while the second dense layer includes a number of units that matches the total number of pitches present in the demo data. This architecture effectively models the sequential nature of musical elements, allowing for the generation of coherent and meaningful musical compositions.

- This project is designed to process single-track piano music, utilizing 70 piano demos, each lasting approximately two minutes. The researchers employ the Music21 toolkit to extract every note and chord, collectively referred to as pitches, from the demos and arrange them into a sequence stream. These sequences are then divided into groups of 81 pitches each. Within each group, the first 80 pitches serve as the input sequence, while the final pitch is designated as the expected output.

  To enable computational processing, a pitch-to-integer dictionary is created, converting pitches into corresponding integer values. The input data is reshaped into a three-dimensional array and subsequently normalized to align with the RMSProp optimizer. Since the network utilizes the Cross Entropy loss function, the output is encoded using the One-Hot encoding format to ensure compatibility and improve performance.
- In the project, the tf.keras.models.Sequential.fit() function from TensorFlow is utilized for training the network model. The network's input and output data are provided as parameters when calling this function. The training process is designed to run for 100 epochs. Additionally, a checkpoint is implemented after each epoch to record key parameters from the training process, such as the model's weights and the outcome of the loss function. This checkpointing feature allows the training process to be halted at any point if the recorded loss value meets the desired criteria.
- The following score is an example of one of the outputs generated by the network. As can be seen, the network is able to utilize a large variety of notes, steps, and techniques throughout this portion of the score. To generate the outputs after the network has created a sequence of notes, we used the same Music21 library as before but rather than extracting note data from midi files we created midi files from note data.
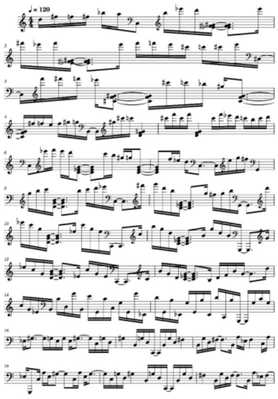
Fig. 1: Output for music according to the paper ***Music Generation Using an LSTM***

3.3 Looking at Google Magneta

The Magenta project by Google delves into the fusion of machine learning and creative expression, emphasizing the creation of art and music using advanced neural network models.

A key focus of Magenta's research is the implementation of Long Short-Term Memory (LSTM) networks, a specialized form of recurrent neural network (RNN) known for its effectiveness in processing sequential data.

LSTM networks are highly effective in music generation because they excel at capturing long-term dependencies in sequential data. Traditional RNNs face challenges in retaining context over extended sequences due to issues like vanishing gradients. LSTMs overcome this by introducing memory cells and gating mechanisms, enabling them to remember and apply information across longer time spans. This is especially important in music, where recurring themes and evolving motifs are fundamental.

In the early stages of Magenta's research, models like Performance RNN employed LSTM architectures to generate expressive piano performances. Performance RNN was specifically designed to capture musical nuances such as timing and dynamics by modeling sequences of events like note onsets and velocities. This method allowed for the creation of performances that reflected realistic musical expression.

Despite their effectiveness in modeling local structures, LSTMs face limitations in capturing long-term dependencies due to their reliance on fixed-size hidden states. To address this, the Magenta team explored alternative architectures to better capture the hierarchical and long-range patterns present in music. This led to the development of the Music Transformer, which uses an attention-based mechanism to efficiently model long-term musical structures. By allowing direct access to all previous events in a sequence, the Music Transformer effectively overcomes the constraints of fixed-size hidden states.

The shift from LSTM models to Transformer architectures in Magenta's research reflects a deeper understanding of how various neural network designs can represent music's complex structure. While LSTMs initially proved effective for modeling sequential data and short-term patterns, Transformer models marked a significant step forward by improving coherence and capturing broader musical relationships.

In conclusion, Google's Magenta project has extensively employed LSTM networks for music generation due to their strengths in sequential modeling and short-term dependency handling. However, recognizing LSTM limitations in tracking long-range dependencies, the project expanded its research into Transformer-based models, resulting in improved musical composition capabilities with enhanced complexity and structure.

3.4 Music using LLM

Large Language Models (LLMs) like GPT can be modified to process music data by combining architecture elements that are tailored for sequential and time-series data. This is often achieved by incorporating richer representations of music, such as MIDI or symbolic notation. A common approach for adapting LLMs to music involves merging Transformer model components (like those in GPT-2 or GPT-3) with specialized techniques for preparing musical data.

Fig. 2: Basic LLM Architecture

Structured Tokenization: This method encodes basic details like notes, note duration, and pitch, but excludes elements such as tempo, time signature, and rests. The assigned token ID range spans from 0 to 253.

TSD Tokenization: This format is similar to Structured Tokenization but additionally includes tempo, time signature, and rests, with a token ID range between 0 and 252.

MIDI-Like Tokenization: Combining features from both Structured and TSD tokenization, this method closely resembles the structure of a standard MIDI file, with token IDs ranging from 0 to 276.

Mapping Music to LLaMA: Since interactions with LLaMA require input as text strings that are decoded using LLaMA’s tokenizer, MIDI tokens are transformed into "pseudo-English" sequences. Each tokenization type results in different token ID ranges: Structured (0–253), TSD (0–252), and MIDI-Like (0–276). LLaMA’s token range extends from 0 to 30,000, where low-value tokens typically map to uncommon Unicode characters that are difficult to express as text. To resolve this, values between 2,000 and 20,000 are added to the MIDI tokens, making them more interpretable.

Through experimentation, adding offsets of 10,000 and 4,000 yielded the best results for generating accurate musical output. After prediction, these offsets are reversed by subtracting the added values. If invalid tokens emerge, the model generates replacements. The decoded tokens are then reconstructed into a new MIDI file, allowing the quality of the generated music to be compared with the original input to evaluate the model’s ability to manage non-standard sequences like musical data.

3.5 Music Transformers - LLM based model

The Music Transformer (Huang et al., 2018) adapts the Transformer architecture — originally designed for natural language processing (NLP) — to generate music. By utilizing the self-attention mechanism, it effectively learns long-term dependencies in musical structures such as harmony, melody, and rhythm, enabling the creation of lengthy and coherent compositions.

Key Features:

- Transformer Framework: Unlike LSTMs that process sequences sequentially, the Transformer’s self-attention mechanism allows it to analyze the entire sequence simultaneously. This feature enhances the model’s ability to identify long-range connections between distant notes, chords, or musical elements more efficiently than RNN-based approaches.

- Enhanced Contextual Awareness: The Transformer's self-attention mechanism enables it to selectively focus on different parts of the composition as needed, improving its ability to recognize and maintain thematic patterns, motifs, and structural elements in music. This results in more coherent musical outputs.
- Improved Handling of Temporal Information: The Music Transformer effectively models musical data with precise timing details, such as 1/16th note intervals. This capability makes it ideal for generating intricate and detailed musical pieces.

4. Current Challenges with music generation

- Temporal Dependencies: Music follows a sequential pattern, where each note's significance is shaped by the notes preceding and following it. This introduces a complexity similar to language processing but with added intricacies.
- Tonality: Accurately representing musical keys, scales, and the emotional depth conveyed through melodies poses a significant challenge.
- Limitations: One notable constraint of this project lies in the fast-evolving nature of computer science and artificial intelligence. While the study explores cutting-edge algorithms and techniques, advancements in technology could rapidly alter the landscape for musicians, developers, and consumers. Additionally, the Turing test survey faced limitations due to the small number of available music samples and a restricted pool of participants.
- Creativity: While models can effectively learn and replicate patterns, true creativity often requires deviating from established patterns. Striking a balance between predictable structures and innovative ideas remains a challenge even for advanced systems.

5. Conclusion

This paper explored the theoretical foundations of Long Short-Term Memory (LSTM) networks and Large Language Models (LLMs) in the context of music generation. LSTMs, known for their ability to capture sequential patterns and short-term dependencies, offer valuable insights into modeling musical structures such as melodies and rhythms. On the other hand, LLMs excel in understanding complex patterns and generating coherent outputs by leveraging extensive training data and advanced architectures like Transformers. The study also proposed a hybrid model concept, combining the strengths of both LSTM and LLM architectures. While LSTMs effectively handle sequential music data at the note or phrase level, LLMs can enhance global coherence and structure across longer compositions. This hybrid approach presents a promising theoretical framework for improving the quality and creativity of AI-generated music. Although this research is theoretical and lacks practical implementation, it lays the groundwork for future exploration. By merging LSTM's temporal precision with LLM's broader contextual understanding, future models may overcome current limitations in music generation. Further empirical testing will be essential to validate this approach and refine the integration of these models for enhanced musical creativity.

References

1. Time. (2023, December 6). How AI could transform music in 2023. Time. https://time.com/6340294/ai-transform-music-2023/ uscutia, V. (2023). Large language models at work.
2. Empress. (2023, September 21). Music and technology: A brief history of AI in music. Empress. https://blog.empress.ac/music-and-technology-a-briefhistory-of-ai-in-music-clq1eh8fd361731wr3j9m23ehi/
3. Javatpoint. (n.d.). Deep learning for sequential data. Javatpoint. https://www.javatpoint.com/deeplearning-for-sequential-data 13
4. Liu, X., and Wang, J. (2020). Pre-training transformers for language modeling: A survey. arXiv. https://arxiv.org/abs/2006.09838utm
5. Berton, M., et al. (2022). Improving music generation using transformerbased models. arXiv. https://arxiv.org/pdf/2203.12105
6. Huang, A., and Yang, Z. (2018). Music Transformer: Generating Music with Long-Term Structure. arXiv preprint arXiv:1809.04281.
7. Predicting Music with LLM https://medium.com/@carneyr98/predictingmusic-with-an-llm-d349296a2dd9
8. Berton, M., et al. (2022). Improving music generation using transformerbased models. arXiv. https://arxiv.org/pdf/2203.12105
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. A., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. arXiv. https://arxiv.org/pdf/1706.03762
10. Noorfo, M. (2023, April 26). Noorfo music AI: A brief history. Machine Learning for Music. https://mct-master.github.io/machine-learning/2023/04/26/noorfomusic-ai-a-brief-history.html
11. Data Scientists Diary. (2023, March 3). Deep learning for music composition. Medium. https://medium.com/data-scientists-diary/deep-learning-formusic-composition-ed21ef1fccf7
12. A Study of Artificial Intelligence for Creative Uses in Music:A Research Paper submitted to the Department of Engineering and Society
13. Eck, D., Schmidhuber, J. (2002). ”Finding temporal structure in music: Blues improvisation with LSTM recurrent networks.” Neural Networks in the Arts and Humanities.
14. Dong, H. W., Hsiao, W. Y., Yang, L. C., Yang, Y. H. (2018). ”MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment.” Proceedings of AAAI Conference on Artificial Intelligence.
15. Hadjeres, G., Pachet, F., Nielsen, F. (2017). ”DeepBach: a steerable model for Bach chorales generation.” Proceedings of the 34th International Conference on Machine Learning (ICML).

16. Huang, C. Z. A., Vaswani, A., Uszkoreit, J., et al. (2018). ”Music Transformer: Generating Music with Long-Term Structure.” arXiv preprint arXiv:1809.04281.
17. Payne, C. (2019). ”MuseNet: Composing with Large-Scale Transformer Models.” OpenAI Blog.
18. Dhariwal, P., Jun, H., Payne, C., et al. (2020). ”Jukebox: A generative model for music.” arXiv preprint arXiv:2005.00341.
19. Exploring Hybrid GRU-LSTM Networks for Enhanced Music Generation Authors: Suman Maria Tony, S. Sasikumar Citation: Tony, S. M., and Sasikumar, S. (2024). Exploring Hybrid GRU-LSTM Networks for Enhanced Music Generation. SSRG International Journal of Electronics and Communication Engineering, 11(7), 150-162. https://doi.org/10.14445/23488549/IJECEV11I7P115 Indian Classical Music Generation Using LSTM and RNN Authors: Bhavanasree, Krishnan, Shanaz, Krishnan Citation: Bhavanasree, K., Krishnan, K., and Shanaz, K. (2024).
20. Indian Classical Music Generation Using LSTM and RNN. IJCRT, 3(4), 1-10. https://ijcrt.org/papers/IJCRT2403917.pdf
21. MRBERT: Pre-Training of Melody and Rhythm for Automatic Music Generation Authors: Shuyu Li, Yunsick Sung Citation: Li, S., and Sung, Y. (2023). MRBERT: Pre-Training of Melody and Rhythm for Automatic Music Generation. Mathematics, 11(4), 1-14. https://doi.org/10.3390/math11040114
22. Xu, X. (2020). LSTM Networks for Music Generation. arXiv preprint arXiv:2006.09838. https://arxiv.org/abs/2006.09838
23. Kotecha, N., Young, P. (2018). Generating Music using an LSTM Network. arXiv preprint arXiv:1804.07300. https://arxiv.org/abs/1804.07300
24. Conner, M., Gral, L., Adams, K., Hunger, D., Strelow, R., Neuwirth, A. (2022). Music Generation Using an LSTM. arXiv preprint arXiv:2203.12105. https://arxiv.org/abs/2203.12105
25. Ingale, V., Mohan, A., Adlakha, D., Kumar, K., Gupta, M. (2021). Music Generation using Three-layered LSTM. arXiv preprint arXiv:2105.09046. https://arxiv.org/abs/2105.09046
26. Mangal, S., Modak, R., Joshi, P. (2019). LSTM Based Music Generation System. arXiv preprint arXiv:1908.01080. https://arxiv.org/abs/1908.01080