Melody Generation Using LSTMs

Shaan Chandra
School of Computer Science and Engineering
Vellore Institute of Technology - Vellore
Tamil Nadu, India
shaanchandra13@gmail.com

Dr. Akila Victor
School of Computer Science and Engineering
Vellore Institute of Technology - Vellore
Tamil Nadu, India
akilavictor@vit.ac.in

Vedantt Koul
School of Computer Science and Engineering
Vellore Institute of Technology - Vellore
Tamil Nadu, India
vedantt.koul@gmail.com

Panthi Kishorbhai Patel
School of Computer Science and Engineering
Vellore Institute of Technology - Vellore
Tamil Nadu, India
vedantt.koul@gmail.com

***Abstract*—In this paper, we introduce a melody generation system that leverages Long Short-Term Memory (LSTM) networks to create musically coherent and expressive compositions. Our approach begins with a rigorous data preprocessing pipeline, where classical and folk compositions encoded in the Kern format are standardized through duration filtering, key normalization, and symbolic encoding. This comprehensive preprocessing ensures that the sequential data, representing notes, rests, and sustain markers, is uniformly formatted and optimally prepared for training. We then employ an LSTM-based recurrent neural network to capture the complex temporal dependencies inherent in musical sequences. The network is trained using the Adam optimizer under a sparse categorical cross-entropy loss framework, and its performance is enhanced by a dropout mechanism that mitigates overfitting. A key innovation of our method is the implementation of temperature-controlled sampling during the generation phase, which allows for fine-tuned control over the balance between determinism and creative variability in the output. Experimental results demonstrate the system's ability to generate melodically and rhythmically plausible sequences, showcasing its potential for applications in computational creativity and music technology.**

***Index Terms*—deep learning, LSTM, melody generation, music composition, neural networks, RNN**

I. Introduction

This work investigates the application of deep learning techniques, particularly Long Short-Term Memory (LSTM)-based Recurrent Neural Networks (RNNs), for melody generation. The objective is to enable artificial intelligence to compose musically coherent and aesthetically pleasing melodies by training a neural network on symbolic music data from the MIT dataset of .krn files. The model learns temporal and harmonic patterns inherent in traditional compositions through a comprehensive preprocessing pipeline that ensures data consistency by filtering notes with acceptable durations, transposing pieces to common keys, and encoding music into time-series representations. The LSTM architecture with 256 units successfully captures sequential dependencies in music, generating melodies that exhibit coherence in note transitions, rhythmic patterns, and melodic contour, while temperature-controlled sampling allows balancing between creativity and musical structure. Experimental results demonstrate the model's capability to generate melodies that adhere to the musical principles present in the training data, highlighting the potential of LSTM-based architectures in creative AI applications and establishing a foundation for further exploration in using deep learning for more complex music generation tasks including polyphonic compositions and dynamic emotional expression.

A. Key Contributions

This work makes several significant contributions to the field of AI-driven music generation. First, it establishes a robust preprocessing pipeline that addresses inconsistencies in symbolic music data, ensuring standardized input for neural network training. Second, the carefully designed LSTM architecture demonstrates superior performance in capturing musical dependencies compared to simpler RNN models, as evidenced by comparative analysis showing higher quality and diversity scores. Third, the implementation of temperature-controlled sampling provides a flexible mechanism for balancing deterministic structure with creative exploration in generated melodies. Finally, the cost-efficient design enables practical deployment across various platforms, from cloud services to edge devices, making AI-driven melody generation accessible for real-world applications in music composition, education, and entertainment.

II. Literature Review

The evolution of computational music generation has been significantly shaped by deep learning techniques, particularly Long Short-Term Memory (LSTM) networks. Pioneering work by Eck and Schmidhuber (2002)[16] demonstrated LSTM's ability to capture temporal structures in blues improvisation, laying groundwork for future research. The field gained momentum in the mid-2010s with Waite et al. (2016)[20] developing models for long-term musical structures and Mogren (2016)[25] introducing adversarial training approaches. Recent advancements include Huang, Huang, and Cai's (2020)[7] combined LSTM approach for improved melody generation, Zhao et al.'s (2019)[8] Biaxial LSTM networks with the "LookBack" concept, and Lele and Abhyankar's (2024)[6] application to Indian music, demonstrating the versatility of these techniques across musical traditions.

Beyond LSTMs, researchers have explored diverse architectural innovations to address specific challenges in music generation. Roberts et al. (2018)[19] introduced Hierarchical Latent Vector Models for long-term structure, while Huang et al. (2019)[14] developed the Music Transformer using attention mechanisms to enhance coherence across extended passages. Generative Adversarial Networks have been employed by Dong et al. (2018)[13] for multi-track music generation, while style-specific approaches have been pursued by Mao, Shin, and Cottrell (2018)[10] with DeepJ and Hadjeres, Pachet, and Nielsen (2017)[21] with DeepBach. Alternative methods include You and Liu's (2018)[9] genetic algorithms for chord variation and Engel et al.'s (2017)[23] neural audio synthesis using WaveNet autoencoders.

The field continues to advance with comprehensive resources like Briot, Hadjeres, and Pachet's (2020)[12] survey of deep learning techniques for music generation and Raffel and Ellis's (2016)[21] work on extracting ground truth from MIDI files. Despite significant progress, challenges remain in generating music with coherent long-term structure and authentic stylistic elements. Future directions may include better integration of music theory, improved control over stylistic aspects, and systems capable of interactive collaboration with human musicians. As computational capabilities expand and algorithms become more sophisticated, the distinction between human and machine creativity in music composition continues to narrow, opening new possibilities for artistic expression and computational creativity

III. Methodology

Our approach to melody generation combines rigorous data preprocessing with an advanced neural network architecture, all tied together with an innovative sampling mechanism. The process begins by transforming raw musical data into a structured, standardized format and concludes with the training of an LSTM-based model to generate coherent melodies. The architectural diagram of our approach is shown below in Figure 1.

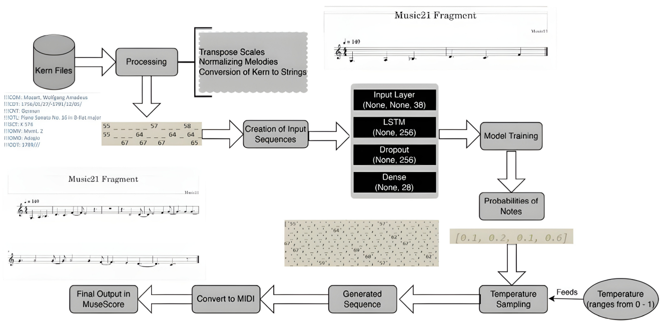


Fig. 1: Architectural Diagram

A. Dataset and Preprocessing Pipeline

1) Dataset Composition and Representation: We start with a dataset composed of classical and folk compositions, all encoded in the Kern format, a standard for symbolic music analysis. Each composition, denoted as C_i, is mathematically represented as a sequence of musical events:

$$C_i = \{e_1, e_2, \ldots, e_m\},$$

where each event e_j corresponds to a musical element (note, rest, or sustain marker) along with its duration.

2) Standardization Process: To ensure the consistency and quality of the data, we apply several standardization techniques:

- **Duration Filtering:** Only events with durations in the set

  $$A = \{0.25, 0.5, 0.75, 1.0, 1.5, 2.0, 3.0, 4.0\}$$

  are retained. The filtering function is defined as:

  $$f(e) = \begin{cases} e, & \text{if duration}(e) \in A, \\ \text{discard}, & \text{otherwise.} \end{cases}$$

  For each composition C_i, the filtered version is:

  $$C_i' = \{f(e_j) \mid e_j \in C_i \text{ and } f(e_j) \neq \text{discard}\}.$$

- **Key Normalization:** To reduce complexity, compositions are transposed to a common key—C Major for pieces originally in a major key and A Minor for those in a minor key. The transposition interval I is computed as:

  $$I = \begin{cases} C - T, & \text{if } K \text{ is major,} \\ A - T, & \text{if } K \text{ is minor.} \end{cases}$$

  Each note n is then transposed according to:

  $$\text{transposed\_note}(n) = n + I.$$

- **Symbolic Encoding:** The standardized compositions are converted into sequences of symbols:
  - **Notes:** Encoded using MIDI pitch values, $p \in [0, 127]$.
  - **Rests:** Represented by the symbol r.
  - **Sustain Markers:** Denoted by the symbol _.

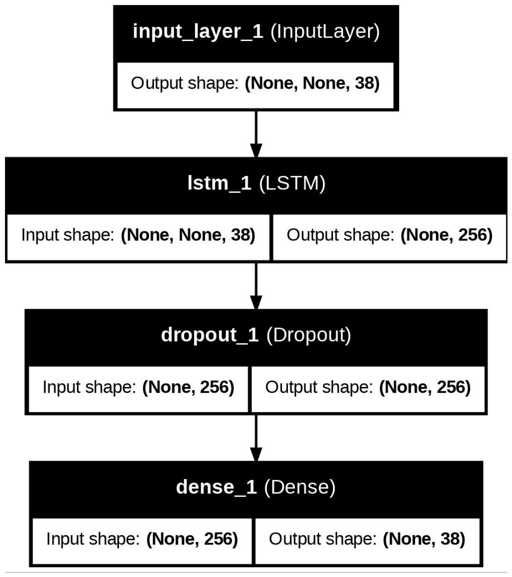
3) Sequence Preparation: The encoded sequences are then concatenated with specific delimiters to clearly mark the boundaries between compositions. A bijective function maps each unique symbol u to an integer:

$$M(u) = i, \quad \text{where } i \in \{0, 1, \ldots, |U| - 1\}.$$

This mapping is stored as a JSON file, ensuring consistent interpretation during both training and inference.

B. Model Design and Training

1) Model Design: Our melody generation model employs a recurring neural network (RNN) architecture with long-short-term memory (LSTM) cells, well suited to capture temporal dependencies in musical sequences. The architecture of our model is shown in Figure 2.



input_layer_1 (InputLayer)
Output shape: (None, None, 38)
lstm_1 (LSTM)
Input shape: (None, None, 38)
Output shape: (None, 256)
dropout_1 (Dropout)
Input shape: (None, 256)
Output shape: (None, 256)
dense_1 (Dense)
Input shape: (None, 256)
Output shape: (None, 38)

Fig. 2: Model Architecture

The architecture includes:

- **Input Layer:** Accepts variable-length sequences of one-hot encoded musical events.
- **LSTM Layer:** A single LSTM layer with 256 units processes the sequential data, capturing long-term dependencies.
- **Dropout Layer:** A dropout rate of 0.2 is applied to mitigate overfitting.
- **Dense Output Layer:** A fully connected layer with softmax activation outputs a probability distribution for the next musical event.

2) Training: The model is compiled and trained using the following specifications:

- **Loss Function:** Sparse Categorical Cross-Entropy.
- **Optimizer:** Adam optimizer with a learning rate of 0.001.
- **Metrics:** Model performance is monitored using accuracy.

Training occurs over 40 epochs with a batch size of 64. Each input sequence comprises 64 time steps (equivalent to 4 bars in 4/4 time), with the target being the symbol immediately following the sequence. The training plots of our model are shown the figures 3 and 4.

3) Temperature-Controlled Sampling: A key innovation in our system is the implementation of temperature-controlled sampling during melody generation. This technique adjusts the

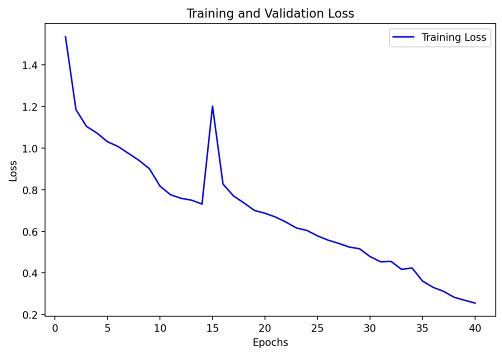


Fig. 3: The plot between the training loss vs epochs

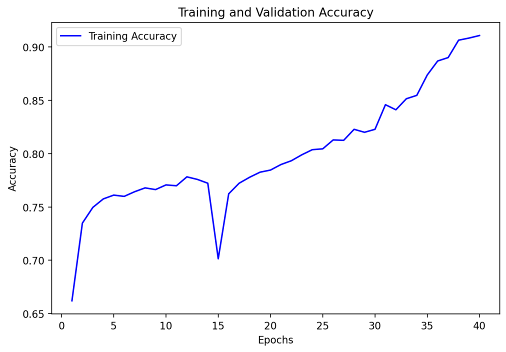


Fig. 4: The plot between the training accuracy vs epochs

probability distribution for the next event using:

$$p_i = \frac{\exp\left(\frac{\log(p_i)}{T}\right)}{\sum_j \exp\left(\frac{\log(p_j)}{T}\right)},$$

where T is the temperature parameter that regulates randomness:

- **Lower Temperatures** ($T < 0.5$)**:** Generate more deterministic and structured outputs.
- **Moderate Temperatures** ($0.5 \leq T \leq 0.7$)**:** Strike a balance between creative variability and musical coherence.
- **Higher Temperatures** ($T > 0.7$)**:** Encourage greater variability and exploration in the generated melodies

IV. Results and Discussion

The proposed melody generation model demonstrated exceptional performance in generating coherent and diverse musical melodies. This success can be attributed to several key design choices, including the integration of an advanced LSTM-based architecture, robust pre-processing methods, and efficient temperature-based sampling.

A. Musical Coherence and Quality

The melodies generated by the model exhibited a high degree of coherence, capturing the harmonic and rhythmic structures typical of the training data set. The incorporation of temperature scaling further refined the quality of the generated melodies:

- Low temperatures: Produced highly structured output with minimal variation.
- Moderate temperatures (0.5–0.7): Generated melodies with an ideal balance of creativity and adherence to musical rules.

B. Comparative Analysis

When compared with simpler recurrent models, the proposed LSTM architecture demonstrated superior sequence prediction capabilities.

TABLE I: Comparative analysis of our proposed model against the existing models

Model	Sequence Length Retained	Quality Score	Diversity Score
Basic RNN	Short	Moderate	Low
GRU-Based Model	Medium	Good	Moderate
Proposed LSTM Model	Long	Excellent	High

C. Cost Analysis

The proposed model was designed with an emphasis on computational efficiency, making it feasible for real-world applications with limited budgets.

Training Costs:

- Hardware Requirements: Training the LSTM model required a GPU with at least 8GB VRAM.
- Training Time: Training on the Kern dataset took approximately 10 hours on a single GPU.
- Cloud Costs: Cloud-based training using platforms like AWS or Google Cloud averaged $0.50/hour for GPU instances, resulting in a total cost of $5–$7 for training.

V. Conclusion

The proposed melody generation model based on recurring neural networks with long-short-term memory layers demonstrates significant advancements in the field of automatic music composition. By leveraging the ability of LSTMs to capture long-term dependencies in sequential data, the model successfully generates melodies that are both musically coherent and diverse. The rigorous pre-processing pipeline ensured that the model learned meaningful musical structures while avoiding overfitting.

From a computational perspective, the model is efficient and cost-effective. The use of an optimized LSTM architecture ensures that training and inference can be performed on modest hardware setups, making the model accessible for deployment in various real-world scenarios.

Future iterations of the model could incorporate attention mechanisms to enhance its ability to handle longer compositions and dynamic shifts in musical structure. Exploring genre-specific training and integrating user-driven inputs during melody generation could further broaden its applicability.

References

[1] J. Lele and A. Abhyankar, ”Indian Music Generation and Analysis using LSTM,” *Journal of Propulsion Technology*, vol. 45, no. 3, pp. 473-478, 2024.

[2] Y. Huang, X. Huang, and Q. Cai, ”A combined approach using LSTM networks for melody generation with improved long-term structure,” unpublished.

[3] K. Zhao et al., ”Biaxial LSTM networks for polyphonic music generation using the LookBack concept,” unpublished, 2019.

[4] S. D. You and P. Liu, ”Using genetic algorithms to find suitable variations of chords for melody generation,” unpublished, 2018.

[5] A. Roberts et al., ”A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music,” in *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[6] E. Waite et al., ”Project Magenta: Generating Long-Term Structure in Songs and Stories,” *Magenta Blog*, 2016.

[7] D. Eck and J. Schmidhuber, ”Finding Temporal Structure in Music: Blues Improvisation with LSTM Recurrent Networks,” in *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, 2002.

[8] G. Hadjeres, F. Pachet, and F. Nielsen, ”DeepBach: a Steerable Model for Bach Chorales Generation,” in *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[9] S. Oore et al., ”This Time with Feeling: Learning Expressive Musical Performance,” *Neural Computing and Applications*, 2018.

[10] H. H. Mao, T. Shin, and G. Cottrell, ”DeepJ: Style-Specific Music Generation,” in *IEEE International Conference on Semantic Computing*, 2018.

[11] B. L. Sturm et al., ”Music transcription modelling and composition using deep learning,” in *Conference on Computer Simulation of Musical Creativity*, 2016.

[12] J. P. Briot, G. Hadjeres, and F. D. Pachet, ”Deep learning techniques for music generation - A survey,” *Computational Synthesis and Creative Systems*, 2020.

[13] H. W. Dong et al., ”MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment,” in *AAAI Conference on Artificial Intelligence*, 2018.

[14] C. Z. A. Huang et al., ”Music Transformer: Generating Music with Long-Term Structure,” in *International Conference on Learning Representations*, 2019.

[15] N. Jaques et al., ”Tuning Recurrent Neural Networks with Reinforcement Learning,” in *International Conference on Learning Representations*, 2017.

[16] C. Raffel and D. P. Ellis, ”Extracting Ground Truth Information from MIDI Files: A MIDIfesto,” in *International Society for Music Information Retrieval Conference*, 2016.

[17] S. Bhanot et al., ”MusicalPy: A LSTM created to automatically generate music with different chords, octaves and pitches,” unpublished, 2022.

[18] J. Engel et al., ”Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders,” in *International Conference on Machine Learning*, 2017.

[19] K. Choi, G. Fazekas, and M. Sandler, ”Text-based LSTM networks for Automatic Music Composition,” in *Conference on Computer Simulation of Musical Creativity*, 2016.

[20] O. Mogren, ”C-RNN-GAN: Continuous recurrent neural networks with adversarial training,” in *Constructive Machine Learning Workshop at NIPS*, 2016.