Music Generation with Sequential and Neural Models

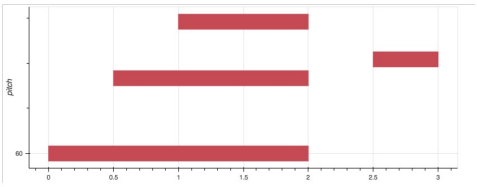
1 Introduction

This report details the development and evaluation of a music generator leveraging sequential and neural modelling to produce novel MIDI output. The system features a variable-order Markov model and a Music Transformer, allowing for flexible output generation and diversity control via adjustable hyperparameters. The models were trained using the MAESTRO V.3.0.0 dataset's 1276 MIDI recordings obtained from International Piano-e-Competition performances using Yamaha Disklaviers. The recordings include key strike velocities and pedal positions, and the data's suitability for our task is evidenced by its usage in training a Music Transformer in [1].

Preprocessing involved extending note durations based on the sustain pedal and tokenisation using the event vocabulary introduced by Oore et al. [2] containing:

- 128 note-on events
- 128 note-off events
- 100 time-shift events (10ms increments up to 1s)
- 32 velocity bins (these change the velocity of all subsequent notes)

This approach facilitates expressive timing without yielding unnecessarily long sequences, unlike a fixed-grid representation. Following this, random crops of ∼1024 events were taken from each sequence, along with the final ∼1024 events (so the models could learn to appropriately end generated passages), before applying transposition and time-scaling data augmentation. An 80/20 train-validation split was then performed. Model outputs were postprocessed into MIDI files using Python's 'mido' package.



```
SET_VELOCITY<80>, NOTE_ON<60>
TIME_SHIFT<500>, NOTE_ON<64>
TIME_SHIFT<500>, NOTE_ON<67>
TIME_SHIFT<1000>, NOTE_OFF<60>, NOTE_OFF<64>,
NOTE_OFF<67>
TIME_SHIFT<500>, SET_VELOCITY<100>, NOTE_ON<65>
TIME_SHIFT<500>, NOTE_OFF<65>
```

Figure 1: A demonstration of the event vocabulary [2].

2 Approach

2.1 The Variable-Order Markov Model (VOMM)

A VOMM with maximal order n is a sequential model that can adapt its context length $k \leq n$ during generation, unlike a fixed k-Markov model. A single VOMM was suitable since our MIDI tokens encompass pitch, dynamics, and temporal information together, whereas a fixed-grid representation would necessitate a multiple-viewpoint system.

The VOMM learns patterns offline by building a prefix forest from a corpus of sequences, similar to Pachet's Continuator [3]. Each sequence $(x_0, ..., x_{L-1})$ is parsed from left to right, and each event x_i is inserted into the forest with the preceding $k(i)$ events as context, where:

$$k(i) = \begin{cases} i & \text{if } 0 < i < n \\ n & \text{if } i \geq n. \end{cases} \tag{1}$$

x_i is referred to as a continuation of the context $(x_{i-k(i)}, \ldots, x_{i-1})$, and each node in the forest holds a set of continuations encountered in the corpus, as denoted by the superscript in Figure 2. As described at 00:14 in the demonstration, $(x_{i-k(i)}, \ldots, x_{i-1}, x_i)$ is inserted into the forest as a tree with root x_{i-1} and a path to node $x_{i-k(i)}$, with each node having continuation x_i.

During generation, potential continuations are identified by traversing the prefix forest nodes corresponding to the given context events as in Figure 3. The next event is then chosen from either the final node's continuations, considering only the full context, or the set of all continuations encountered along the path, considering all possible context lengths (00:38-01:28).

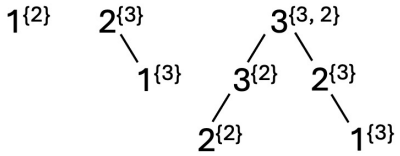


Figure 2: An example prefix forest for the input sequence (1, 2, 3, 3, 2) with maximum order $n = 3$.

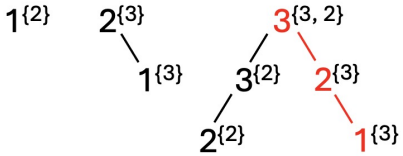


Figure 3: An example of traversing to identify the potential continuations given the context (1, 2, 3).

2.2 Music Transformer

Music Transformer [4], an attention-based neural network, operates by embedding sequences into L D-dimensional vectors, with sinusoidal position encodings representing timing information. The embeddings are passed through a Transformer decoder, in which each layer consists of a multi-head attention block followed by a feedforward network, and the output is then projected into the vocabulary space (01:29-01:34).

As shown at 01:35-02:03, the attention block transforms input X into queries $Q = XW^Q$, keys $K = XW^K$ and values $V = XW^V$ using the learnt $D \times D$ matrices W^Q, W^K, and W^V. The queries, keys, and values are then split into H $L \times D_h$ attention heads, where $D_h = \frac{D}{H}$, enabling the model to simultaneously focus on different input parts and capture dependencies. Each head utilises a learnt relative position embedding E^r to produce an $L \times L$ logits matrix $S^{rel} = \text{Skew}\left(QE^{r\top}\right)$, where the skewing function transforms the absolute-by-relative indexing to absolute-by-absolute indexing, as illustrated in Figure 4. Attention probabilities are then computed as:

$$\text{RelativeAttention} = \text{Softmax}\left(\frac{QK^\top + S^{rel} + M}{\sqrt{D_h}}\right)V, \tag{2}$$

where M is a combined padding and upper triangular mask, excluding padding tokens and future events from consideration. The attention outputs for each head are concatenated and linearly transformed before passing through the feedforward network.

The model is trained to predict the next token in a sequence using cross-entropy loss. Once trained, the model produces logits for the next token given a sequence, which can iteratively be used greedily or probabilistically to generate sequences (02:04-02:18).

3 Evaluation

While qualitative evaluation of generated music is effective for assessing aspects like realism and creativity, it requires a large, diverse group of subjects to be effective. Therefore, we focused on quantitative features capturing the pitch diversity, melodic structure and rhythm of each sample [5]:

- Pitch count: the number of different pitches.
- Pitch class histogram: the pitch class distribution.
- Pitch class transition matrix: a matrix tracking pitch class transitions between consecutive notes.
- Pitch range: the difference between the highest and lowest pitch.
- Average pitch interval: the average interval between two consecutive pitches.
- Note count: the number of notes.
- Average inter-onset interval: the average time interval between two consecutive note onsets.

We generated 500 samples for each model and compared the (element-wise) mean and standard deviation of these features with those of a random sample of 500 validation set sequences using Welch's t-test. The VOMM had a maximum order of 4 and generated continuations considering only the full context, while the Music Transformer generated continuations using the logits probabilistically. Due to memory constraints, we couldn't load the entire training set into the VOMM. Instead, we trained 500 VOMMs on random subsets of $102 \approx \frac{51{,}040}{500}$ training sequences and generated a sample from each. Alternatively, the Music Transformer was trained on the full training data for 62 epochs before generation.

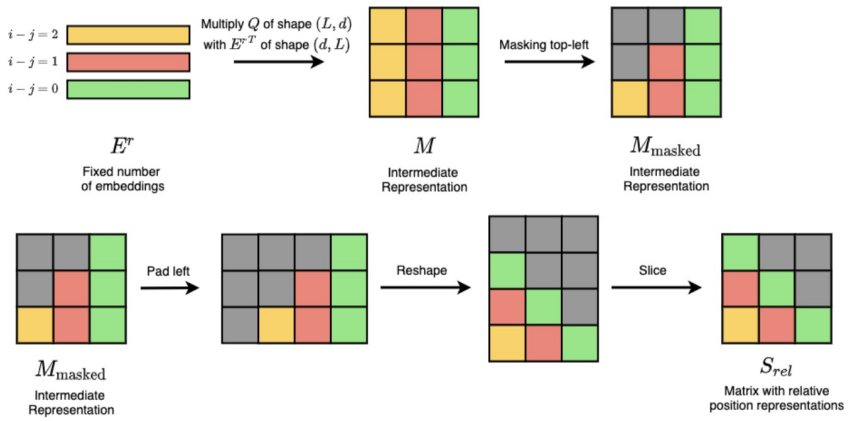


Figure 4: An illustration of the skewing function used to obtain S^{rel} [6].

Table 1 suggests that the Music Transformer samples more closely resemble real ones and exhibit fewer variations compared to the VOMM samples, particularly in note count. However, it's important to note that comparing the models is challenging due to differing training and generation methods. Despite this, some consistency is observed in the pitch class histograms and transition matrices across the sample sets, as illustrated in Figures 6 and 7.

Overall, the generated samples exhibit slight randomness, indicating the need for further training in the Music Transformer. However, the probabilistic nature of the VOMM implies limitations in further refinement, which underscores the superiority of deep learning approaches in capturing musical patterns rather than merely storing and probabilistically regenerating them. Future extensions of this work include optimising the VOMM implementation for utilisation of the full training data, and conducting more thorough evaluation with larger sample sets, qualitative analysis, and varying model hyperparameter values. Investigating online training during generation for the VOMM may also improve sample coherence and stylistic consistency.

Metric	Real samples	VOMM samples	Music Transformer samples
Pitch count	41.15 ± 10.67	46.77 ± 19.13*	42.59 ± 10.90*
Pitch range	57.48 ± 11.63	60.56 ± 14.53*	59.89 ± 11.60*
Average pitch interval	-0.02 ± 0.10	-0.04 ± 1.16	-0.01 ± 0.25
Note count	237.00 ± 26.51	283.36 ± 281.70*	232.37 ± 53.11
Average inter-onset interval	107.02 ± 68.59	104.31 ± 54.34	91.80 ± 44.49*

Table 1: The mean and standard deviation of each feature for the three sets of samples. Asterisks indicate statistically significant t-test results between the model samples and the real samples.

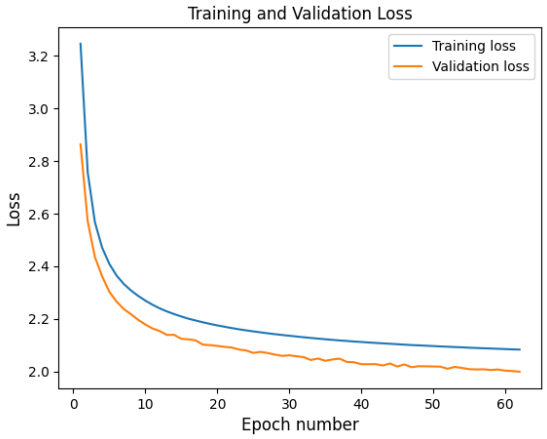


Figure 5: The training and validation loss curves for the Music Transformer.

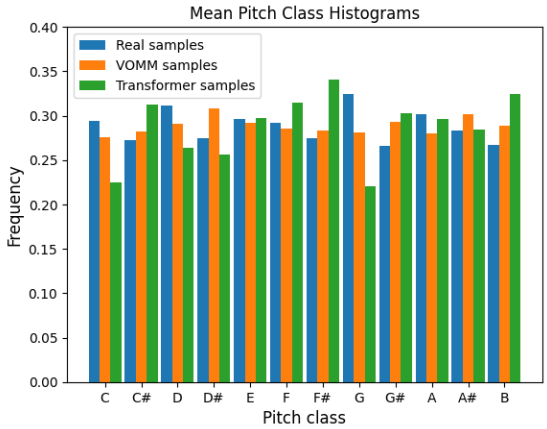


Figure 6: The (normalised) mean pitch class histograms for the three sets of samples.

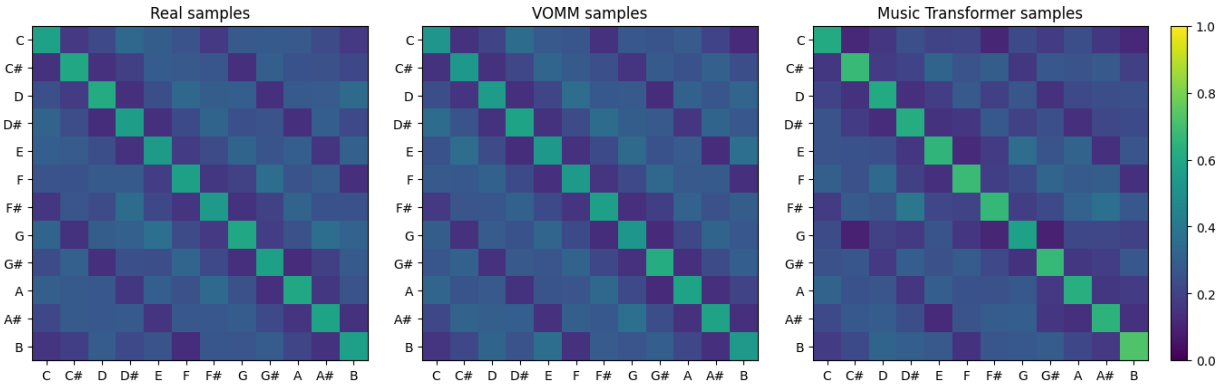


Figure 7: The (row-normalised) mean pitch class transition matrices for the three sets of samples (rows = from, columns = to).

References

[1] C. Hawthorne, A. Stasyuk, A. Roberts, *et al.*, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=r1lYRjC9F7.

[2] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan, "This time with feeling: Learning expressive musical performance," *Neural Computing and Applications*, vol. 32, pp. 955–967, 2020.

[3] F. Pachet, "The continuator: Musical interaction with style," *Journal of New Music Research*, vol. 32, pp. 333–341, Aug. 2010. DOI: 10.1076/jnmr.32.3.333.16861.

[4] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, *et al.*, "Music transformer: Generating music with long-term structure," *arXiv preprint arXiv:1809.04281*, 2018.

[5] L.-C. Yang and A. Lerch, "On the evaluation of generative models in music," *Neural Computing and Applications*, vol. 32, no. 9, pp. 4773–4784, 2020.

[6] H. H. Tan, *Understanding music transformer*, 2020. [Online]. Available: https://gudgud96.github.io/2020/04/01/annotated-music-transformer/.

[7] V. Lageard, *Melodendron*, 2021. [Online]. Available: https://github.com/valentinlageard/melodendron.

[8] A. Gomatam, *Music-transformer*, 2024. [Online]. Available: https://github.com/spectraldoy/music-transformer.