A TRADITIONAL APPROACH TO SYMBOLIC PIANO CONTINUATION

Christian Zhou-Zheng[1,2] **John Backsund**[2] **Dun Li Chan**[3] **Alex Coventry**[2]
Avid Eslami[4] **Jyotin Goel**[5] **Xingwen Han**[2] **Danysh Soomro**[2] **Galen Wei**[6]
[1] Metacreation Lab, Simon Fraser University [2] EleutherAI
[3] Sixth Form College, Universiti Sains Malaysia [4] University of Toronto
[5] Indian Institute of Technology Jodhpur [6] Vanderbilt University
christianzhouzheng@gmail.com

ABSTRACT

We present a traditional approach to symbolic piano music continuation for the MIREX 2025 Symbolic Music Generation challenge. While computational music generation has recently focused on complex tasks requiring sophisticated architectural modifications, we argue that simpler approaches may be more effective for constrained, single-instrument tasks. We thus return to a simple next-token-prediction objective on tokenized raw MIDI, aiming to outperform more complicated systems with better data and better techniques, drawn from language modeling.

1. INTRODUCTION

The generation of continuations of piano music has a long history in computational music generation, largely due to the propensity of piano music readily available in symbolic formats. Recent developments in sequence modeling have allowed continuation to be viewed as an autoregressive task, to be modeled with a suitable tokenization scheme and a powerful sequence model like the ubiquitous Transformer [1]. A nonexhaustive list of prior work in this vein includes the Music Transformer [2], Museformer [3], FIGARO [4], and MuseCoco [5].

Most research in symbolic music modeling has so far focused on generalizing these techniques to—and improving performance on—long-sequence, multitrack, multi-instrument, and/or text- or attribute-controllable generative tasks. This typically requires "tricks" to be developed to allow the model to handle these harder tasks, such as fine- and coarse-grained attention for long sequences [3]; text feature extraction techniques for controllability [4]; and attribute augmentation, also for controllability [5].

However, when restricted to a simple, short-form, and single-instrument task, such as that posed by the MIREX 2025 Symbolic Music Generation challenge [6], these "tricks" may not be necessary. We conjecture that a return to the language-modeling approach taken by the Music Transformer [2], consisting of straightforward next-token prediction on tokenized raw musical data without any augmentations, will outperform the "tricks" of generalized models when trained specifically for this task.

2. SYSTEM DESCRIPTION

As indicated previously, an accurate intuition for our approach may be obtained by taking the pretraining methodology of a standard large language model and replacing domain-specific components with their symbolic music counterparts (e.g. tokenized text with tokenized MIDI). We structure the remainder of this extended abstract as a technical report on our methodology, in the interest of reproducibility for future work in this vein.

2.1 Objective

We adopt the objective of the MIREX 2025 Symbolic Music Generation challenge [6], paraphrased as follows:

> Given a 4-measure piano prompt, with an optional pickup measure, generate a musically coherent 12-measure continuation. Assume all music is in 4/4 time and quantized to a sixteenth-note resolution.

The input and output are JSON objects, containing `"prompt"` and `"generation"` keys, which in turn contain lists of objects of the form

```
{
  "start": 16,
  "pitch": 72,
  "duration": 6
},
```

where `"start"` ranges from 0-79 for the prompt and 80-271 for the generation, and `"pitch"` ranges from 0-127, corresponding to MIDI pitch numbers. We model the music in MIDI instead of JSON for ease of use with existing computational music tooling, and convert between the formats using scripts provided in the competition baseline.

2.2 Data selection and preprocessing

We used the recently released Aria-MIDI dataset [7] for training, under the premise that better data is always one of the most effective ways to improve performance. We used the `pruned` split, discarded examples with an `audio_score` of less than 0.9, and held out 10% of the remainder for validation. The dataset comprises Type 0 MIDI files automatically transcribed from solo piano performances, which are rarely quantized, so we quantized all files to a sixteenth-note resolution ahead of training.

We used the MidiTok library [8] to implement a simplified REMI tokenization [9], resulting in a vocabulary size of 228. In particular, we disabled note velocity encodings (`use_velocities=False`) and the splitting of tokens at bars or beats (`encode_ids_splits="no"`). These settings mirror the simplified musical representation specified by the competition, described in Section 2.1.

Each training sample was generated by loading a MIDI file from the dataset with the symusic library [10], encoding it with our REMI tokenizer, and selecting a random range of 16 consecutive bars that contained at least 100 tokens. This formed the sample passed to the model, which typically ranged from 200 to 1200 tokens in length.

2.3 Model architecture

We adopt a decoder-only RWKV-7 backbone [11] as our sequence model, as it provides better data efficiency and easier training in a resource-constrained environment over the quadratic Transformer [1]. We adopt the "deep and narrow" strategy suggested by Tay *et al.* [12] and Zhou-Zheng and Pasquier [13], resulting in a model with 12 layers, a hidden dimension of 384, and a feedforward dimension of 1536, for about 20 million total parameters. This small size allows remarkably fast training and inference in a resource-constrained consumer environment.

2.4 Training

We trained for 50 epochs on a single RTX 4090 using the RWKV-LM library [14], using the Adam optimizer [15], weight decay of 0.1, a batch size of 32, and a sequence length of 1024, as we found that the training sequences rarely exceeded this length. We used a cosine learning rate scheduler from 1e-4 to 1e-5 and optimized with respect to the standard cross-entropy loss. Training took approximately 2 days to complete.

For validation, we took 8 samples on each of 7 test prompts at intervals of 4 epochs and qualitatively analyzed them. We found the model from epoch 32 to perform best, as later checkpoints showed a slight decline in quality and earlier checkpoints seemed undertrained.

2.5 Inference

We use the rwkv.cpp library [16] to perform inference. The sampling parameters are set as follows: temperature=1.0, top-p=0.95, repetition penalty=1.0, top-k=40. We publish an end-to-end system as a Docker image, which can be run with the following bash script with command `bash script.sh in.json out/ n_sample`:

```
#!/bin/bash

IN_ABS=$(realpath "$1")
OUT_ABS=$(realpath "$2")
mkdir -p $OUTPUT_FOLDER_ABS

USER="christianzhouzheng"
IMAGE="$USER/rwkv-mirex:latest"
docker pull "$IMAGE"
docker run --rm \
    -v "$IN_ABS:/app/input.json:ro" \
    -v "$OUT_ABS:/app/output" \
    "$IMAGE"  "/app/input.json" \
    "/app/output" "$3"
```

3. RESULTS

We were not able to get any objective or subjective metrics in time for the competition deadline, but we plan to update this section upon receiving the results of the competition evaluation. We are also in the process of archiving our generated evaluation samples, which will remain private until the evaluation closes in the interest of fairness.

4. CONCLUSION

We presented a simple, traditional approach to piano music continuation, inspired by recent advances in language modeling. By leveraging state-of-the-art resources and techniques, such as the RWKV-7 architecture [11] and Aria-MIDI dataset [7], we aim to demonstrate that a sophisticated system can be outperformed by a simple system with a stronger foundation.

Future work may increase the size of the model or training corpus or increase the sequence length of the model, scaling performance in the same vein as in language modeling. Our approach can be extended to multitrack, multi-instrument pieces rather easily by using the REMI+ tokenization scheme [4], and longer sequence lengths can be handled by leveraging byte-pair encoding [17], both of which are already supported by MidiTok [8] and both of which future work may investigate.

5. ACKNOWLEDGMENTS

We thank the MIREX 2025 organizers and the Symbolic Music Generation task captains for providing us with this opportunity. We also thank the members of the Metacreation Lab for their feedback and support.

This extended abstract was produced by a project under the inaugural EleutherAI Summer of Open AI Research (SOAR), led by project captain Christian Zhou-Zheng. We thank the anonymous Discord user `genetyx8` for coordinating SOAR, and to Stella Biderman and EleutherAI for generously providing compute for other experiments.

6. REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2017, p. 6000–6010.

[2] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. M. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer: Generating music with long-term structure," in *International Conference on Learning Representations*, 2018.

[3] B. Yu, P. Lui, R. Wang, W. Hu, X. Tan, W. Ye, S. Zhang, T. Qin, and T.-Y. Liu, "Museformer: transformer with fine- and coarse-grained attention for music generation," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2022.

[4] D. von Rütte, L. Biggio, Y. Kilcher, and T. Hofmann, "FIGARO: Controllable music generation using learned and expert features," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=NyR8OZFHw6i

[5] P. Lu, X. Xu, C. W. Kang, B. Yu, C. Xing, X. Tan, and J. Bian, "MuseCoco: Generating symbolic music from text," 2023. [Online]. Available: https://arxiv.org/abs/2306.00110

[6] G. Xia, J. Jiang, A. Maezawa, Z. Wang, Y. Zhang, R. Yuan, and J. S. Downie, "MIREX (music information retrieval evaluation eXchange)," https://www.music-ir.org/mirex/wiki/MIREX_HOME, Future MIREX Team, 2025, music Information Retrieval Evaluation Competition, part of ISMIR Conference.

[7] L. Bradshaw and S. Colton, "Aria-MIDI: A dataset of piano MIDI files for symbolic music modeling," in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=X5hrhgndxW

[8] N. Fradet, J.-P. Briot, F. Chhel, A. El Fallah Seghrouchni, and N. Gutowski, "MidiTok: A python package for MIDI file tokenization," in *Extended Abstracts for the Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference*, 2021. [Online]. Available: https://archives.ismir.net/ismir2021/latebreaking/000005.pdf

[9] Y.-S. Huang and Y.-H. Yang, "Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proceedings of the 28th ACM International Conference on Multimedia*, New York, NY, USA, 2020, p. 1180–1188.

[10] Y. Liao and Z. Luo, "symusic: A swift and unified toolkit for symbolic music processing," in *Extended Abstracts for the Late-Breaking Demo Session of the 25th International Society for Music Information Retrieval Conference*, 2024. [Online]. Available: https://ismir2024program.ismir.net/lbd_426.html#lbd

[11] B. Peng, R. Zhang, D. Goldstein, E. Alcaide, X. Du, H. Hou, J. Lin, J. Liu, J. Lu, W. Merrill, G. Song, K. Tan, S. Utpala, N. Wilce, J. S. Wind, T. Wu, D. Wuttke, and C. Zhou-Zheng, "RWKV-7 "Goose" with expressive dynamic state evolution," 2025. [Online]. Available: https://arxiv.org/abs/2503.14456

[12] Y. Tay, M. Dehghani, J. Rao, W. Fedus, S. Abnar, H. W. Chung, S. Narang, D. Yogatama, A. Vaswani, and D. Metzler, "Scale efficiently: Insights from pretraining and finetuning transformers," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=f2OYVDyfIB

[13] C. Zhou-Zheng and P. Pasquier, "Personalizable long-context symbolic music infilling with MIDI-RWKV," 2025. [Online]. Available: https://arxiv.org/abs/2506.13001

[14] B. Peng, "RWKV-LM," https://github.com/BlinkDL/RWKV-LM, 2025.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[16] RWKV, "rwkv.cpp," https://github.com/RWKV/rwkv.cpp, 2025.

[17] P. Gage, "A new algorithm for data compression," *The C Users Journal*, vol. 12, no. 2, pp. 23–38, 1994.