LEVERAGING LLMS FOR AUTOMATED MUSIC GENERATION

AUTHORS

Corey Zhang Mark Santolucito

AFFILIATIONS

Eastlake High School
Sammamish, WA, USA
Email: s-cozhang@lwsd.org

Barnard College, Columbia University
New York, NY, USA
Email: msantolu@barnard.edu

ABSTRACT

This paper explores the application of large language models (LLMs) for music generation, specifically focusing on generating MIDI files, JSON representations, and music- related code. We aim to leverage LLM capabilities to automate and enhance the music composition process. We systematically examine methodologies and tools for integrating LLMs in music generation with direct music creation and code-based techniques. The challenge of automating music composition using AI remains significant due to the complexity and creativity required in the process. Our findings demonstrate the potential of LLMs to innovate and streamline music composition, offering new tools and approaches for musicians and developers.

INTRODUCTION

This research explores the transformative potential of artificial intelligence (AI) in music composition, focusing on large language models (LLMs) like GPT-3.5. By utilizing the OpenAI API to generate music in MIDI and JSON formats, the study aims to enhance the efficiency and creativity of music creation. Key contributions include developing a framework for AI-generated music, establishing benchmarks for evaluation, and comparing LLM-generated compositions with traditional methods. This work aims to advance AI-driven music composition, fostering innovation and collaboration across technology and the arts.

STATE OF THE ART

AI in music composition has evolved from rule-based systems to advanced deep learning models, such as those used in Google's Magenta and Sony's Flow Machines. Transformer models like OpenAI's MuseNet further enhance music generation across genres. Integrating MIDI and JSON formats with large language models (LLMs) like GPT-3.5 combines real-time performance data with detailed musical structures, enabling more sophisticated and flexible compositions. While LLMs have proven effective in generating text and code, their use in structured music formats is still emerging, presenting a new frontier in AI-driven music creation.

METHODOLOGY

This study investigates three methods for generating MIDI scores in JSON format:

- Direct Generation produces MIDI scores directly in JSON format, minimizing errors from format translation but limiting post-generation adjustments and requiring rigorous validation.
- Code Generation creates executable code (e.g., JavaScript) to generate MIDI scores, offering customization but introducing potential execution errors and requiring coding expertise.
- Rich Code Generation involves complex scripts with detailed instructions for intricate compositions, providing higher musical nuance but demanding substantial computational resources.

We evaluated these methods using three prompts, each tested across 10 trials, totaling 90 trials:

1. Prompt 1: Generate MIDI scores in JSON notation that can be played back by Tone.js. This baseline prompt assesses the model's ability to generate MIDI scores compatible with web-based music playback and interactive environments.
2. Prompt 2: Generate a MIDI JSON Bach chorale adhering to counterpoint rules with four voices following a chord progression and counterpoint harmony rules. This prompt evaluates the model's proficiency in classical music composition.
3. Prompt 3: Generate a MIDI JSON complex piano piece with two hands, featuring syncopated rhythms and varied notes. This prompt challenges the model's capability for intricate, multi-layered music.

The evaluation metrics include:

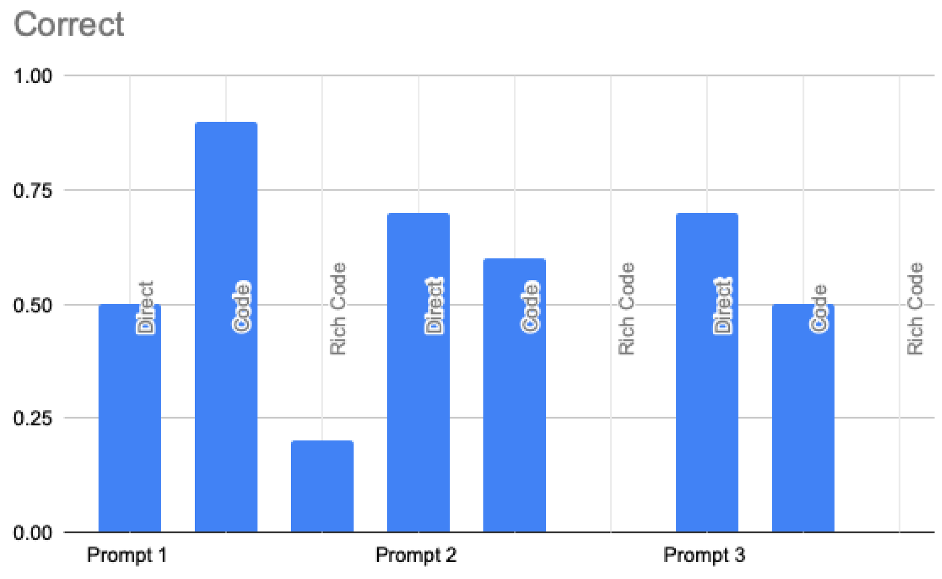
- Correctness: Measures adherence to predefined criteria.
- Compiled: Assesses whether MIDI scores compile successfully.
- Complexity: Quantifies the number of notes to gauge musical detail.

Results were analyzed based on these metrics to determine the effectiveness and consistency of each method using the OpenAI API's GPT-3.5-turbo model.

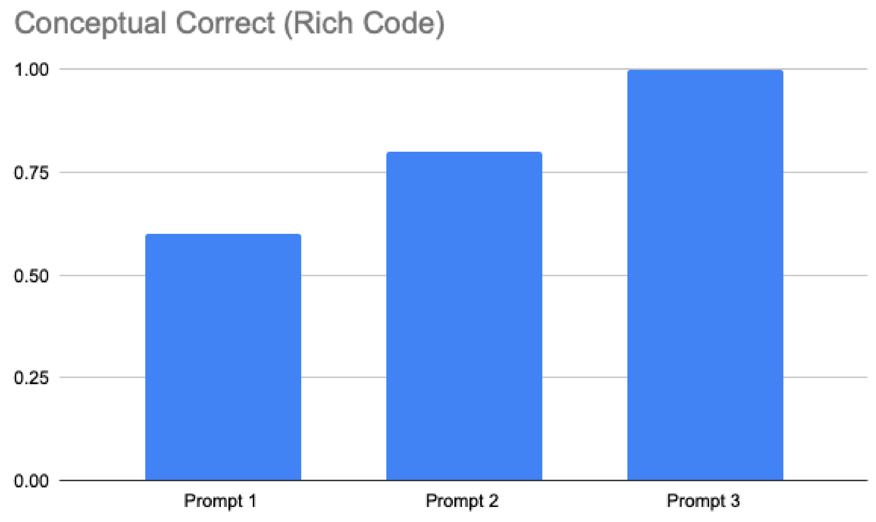
RESULTS

	Correct	Conceptual Correct	Compiled	Complexity
Direct	Moderate consistency, scoring between 0.5 and 0.7 across prompts.		Consistently produces compilable outputs but with moderate success.	Produces simpler, less intricate compositions.
Code	High score of 0.9 for one prompt but variable scores for others (0.6 and 0.5).		Achieves high compilation success but varies by prompt, with a high score of 0.9.	Generates moderately complex outputs but lacks consistency.
Rich Code	Consistently low scores (0.2, 0, and 0), indicating challenges in producing musically accurate outputs with complex data structures.	Effectively meets procedural and structural guidelines, ensuring adherence to compositional instructions.	Excels in compilation success, especially in complex prompts, achieving a perfect score of 1.	Highest complexity scores (13.5, 40.4, and 52.4), indicating the ability to create intricate compositions.

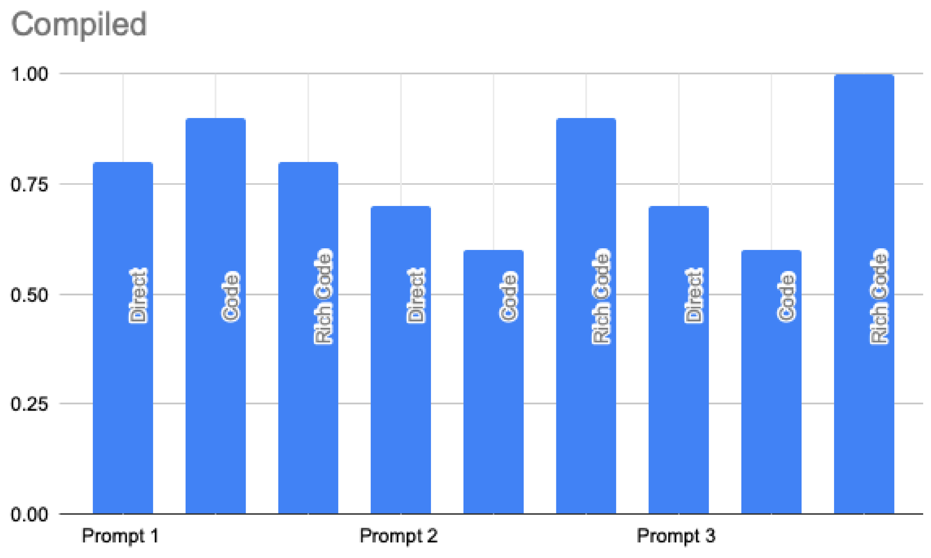
Correct



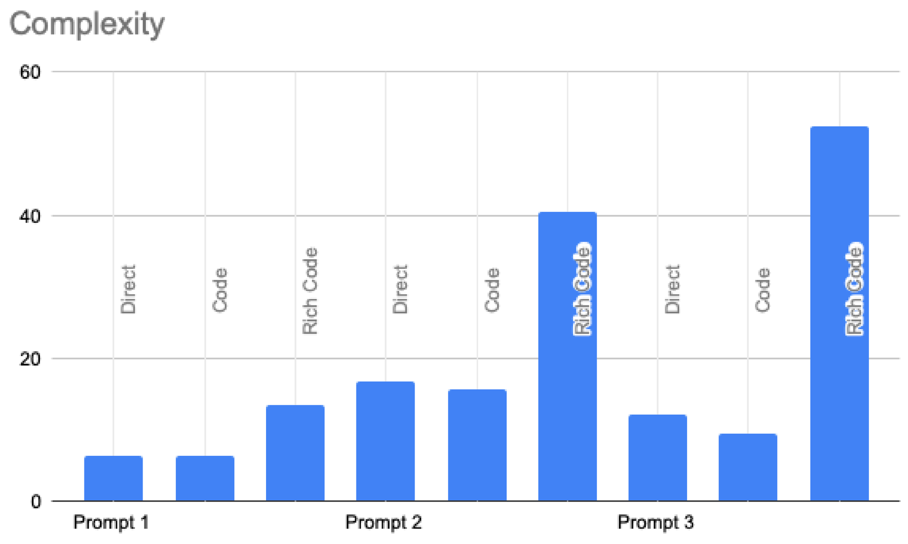
Conceptual Correct (Rich Code)



Compiled



Complexity



CONCLUSION

Our evaluation highlighted distinct characteristics of each generative method. Direct Generation maintains harmonic consistency but struggles with sequence length and pitch accuracy. Code Generation is functional but often results in harmonic clashes and simplistic structures. Rich Code Generation produces complex results but frequently lacks coherence and musicality. The Conceptual Correctness metric enhances our assessment by ensuring adherence to compositional instructions, emphasizing the need for further refinement to balance complexity, correctness, and musicality in generative music systems.

REFERENCES

[1] OpenAI, "Gpt-3.5 turbo," https://beta.openai.com/docs/ guides/gpt, 2021.
[2] D. M. Huber, Modern Recording Techniques, 9th ed. Routledge, 2020.
[3] "Json: Javascript object notation," Available at https:// www.json.org/, accessed: Date accessed.
[4] A. Barate`and other authors, "Advances and perspectives in web technologies for music representation," DigitCult- Scientific Journal on Digital Cultures, vol. 1, no. 2, pp. 1–18, 2016.
[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Ka- plan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, J. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," Advances in Neural Infor- mation Processing Systems, vol. 33, pp. 18 714–18 732, 2020.
[6] A. DuBreuil, Hands-On Music Generation with Ma- genta: Explore the Role of Deep Learning in Music Generation and Assisted Music Composition. Packt Publishing, 2020.
[7] F. Ghedini, F. Pachet, and P. Roy, "Creating music and texts with flow machines," in Multidisciplinary Contributions to the Science of Creative Thinking, ser. Creativity in the Twenty First Century, G. Corazza and S. Agnoli, Eds. Singapore: Springer, 2016. [Online]. Available: https://doi.org/10.1007/978-981-287-618-8\ 18
[8] T. Sexton, "Musenet," Music Reference Services Quarterly, vol. 26, no. 3–4, pp. 151–153, 2023. [Online]. Available: https://doi.org/10.1080/10588167. 2023.2247289
[9] D. Herremans and C.-H. Chuan, "The emergence of deep learning: new opportunities for music and audio technologies," Neural Computing and Applications, vol. 32, pp. 913–914, 2020. [Online]. Available: https://doi.org/10.1007/s00521-019-04166-0
[10] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, C. Hawthorne, A. M. Dai, M. D. Hoffman, and D. Eck, "Music transformer: Generating music with long-term structure," in Proceedings of the International Confer- ence on Machine Learning (ICML), 2018, pp. 2410–2419.
[11] F. Colombo, N. Chen, C. Donahue, Z. Li, and B. Pardo, "Lakhnes: Improving multi-instrumental music genera- tion with cross-domain pre-training," Transactions of the International Society for Music Information Retrieval (TISMIR), vol. 2, no. 1, pp. 97–108, 2019.
[12] L.-C. Lu et al., "Llm discussion: Enhancing the creativity of large language models via discussion framework and role-play," arXiv preprint arXiv:2405.06373, 2024.
[13] OpenAI, "Openai codex," https://openai.com/index/ openai-codex/, 2021.
[14] OpenAI, "Openai musenet," https://openai.com/index/ musenet/, 2019.
[15] Y. Zhang, "Utilizing computational music analysis and ai for enhanced music composition: Exploring pre- and post-analysis," Educational Administration: Theory and Practice, vol. 30, no. 5, pp. 269–282, 2024. [Online]. Available: https://doi.org/10.53555/kuey.v30i5.2841
[16] E. V. Epure, G. Salha, M. Moussallam, and R. Hen- nequin, "Modeling the music genre perception across language-bound cultures," in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 4765–4779. [Online]. Available: https://aclanthology.org/2020.emnlp-main.386
[17] J. Sun, J. Yang, G. Zhou, Y. Jin, and J. Gong, "Un- derstanding human-ai collaboration in music therapy through co-design with therapists," in Proceedings of the CHI Conference on Human Factors in Computing Systems, 2024.
[18] S. Oramas and et al., "Multimodal deep learning for mu- sic genre classification," Transactions of the International Society for Music Information Retrieval, vol. 1, no. 1, pp. 4–21, 2018.
[19] M. Newman, L. Morris, and J. H. Lee, "Human-ai music creation: Understanding the perceptions and experiences of music creators for ethical and productive collabora- tion," in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), 2023.
[20] E. Kaasinen et al., "A user-centric view of intelligent en- vironments: User expectations, user experience and user role in building intelligent environments," Computers, vol. 2, no. 1, pp. 1–33, 2012.