King Me!
(Technical Topic)

The Day the Music Died? Generative AI and its Effect on the Music Industry
(STS Topic)

A Thesis Prospectus
In STS 4500
Presented to
The Faculty of the
School of Engineering and Applied Science
University of Virginia
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Paul Wesley Stepler

November 3, 2023

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISORS

Prof. Pedro Augusto P. Francisco, Department of Engineering and Society

Introduction

Software development is almost always a communal effort. Granted, you will at times have people who are capable of developing entire pieces of software on their own, but more often than not, the software development lifecycle depends on several different teams, each comprised of several different people. And even in the case where someone might develop an entire piece of software on their own, it is almost certain that they used some sort of library or module another developer created. Taking it even further, these developers are using languages sometimes created long before they even knew how to type. The bottom line is there are little to no places in the software industry that will not require you to interact with other developers and to work as a team with those developers.

Recently, however, the rise of artificial intelligence (AI) has raised questions about whether this will continue to be the case. While at this point AI chatbots, such as OpenAI's ChatGPT or Google's Bard, have primarily served as tools to help, not replace, software developers, conversations have taken place about whether that is the road we are inevitably heading down. The code writing capabilities of these models—especially ChatGPT—raise questions about whether or not entry level or junior software developers will one day be replaced entirely by AI. There is little to no question that AI will be around for quite a while, and will have some sort of role in society. The question that must be asked, then, is how substantial will its role be? There is some data that shows AI, when used in moderation, does indeed increase productivity among software developers. A recent study compared how long it took two groups of programmers to implement an HTTP server in JavaScript. The treatment group used GitHub Copilot, a popular pair-programming AI tool, to help them complete their work, while the control group did not use any AI tools. The study found that "the completion time of the treated

group was 55.8% lower than that of the control group" (Cihon & Demirer 2023). AI, when used well, can be a very beneficial tool. But we must be wary of going too far.

In the technical topic of my thesis, I will walk through an independent project I worked on this past summer: coding the game of checkers using Python. I will explain how I implemented this game, how I could have benefitted from the use of AI, and how I did benefit from the work and research of other software developers. In my sociotechnical topic, I will take a deeper dive into the world of AI. Specifically, I will be looking at generative AI and how it will uniquely impact the music industry. The rise of various AI tools has made waves in all kinds of fields, and the music industry is among one of the more intriguing. The connection between these two topics is they each to a certain degree explore how society ought to utilize AI going forward. Its capabilities are vast, but in both the software industry and the music industry—maybe moreso in the music industry—it poses a threat to those who work in those fields to provide for themselves and for their families.

King Me! (Technical Topic)

This past summer, I took on an independent project of coding the game of checkers entirely in the popular programming language, Python. The goal of this project had two primary goals. The first, keeping my programming skills sharp. Being a Computer Science student at the University of Virginia is no joke – the curriculum is rigorous and requires you to be at your best in order to succeed. With this in mind, if you were to do absolutely zero coding over the more than three month long summer break, you would almost certainly come into the semester at least a little bit out of practice. So, rather than take those three months off from programming, I chose to do this project in the interest of keeping my skills sharp and to better prepare myself for the coming semester.

The second reason I took on this project was to learn some new things. Staying up to speed as a programmer is not simply about excelling at the concepts and algorithms you already know—you also have to be able to adapt and learn new methods and technologies you have never used before. I had never attempted developing a game, so there was a lot I had to learn along the way.

The first step in the development process was making some design decisions. Chief among them, was how the program would store and represent the checkerboard and pieces. I developed this game using a popular Python module called PyGame. Essentially, PyGame is a library of various different functions and classes available to aid developers in writing Python games. One of the classes made available via PyGame is the Sprite class. Simply put, a Sprite is an object that can smoothly and effectively interact with other Sprites within a game. In my implementation, both the checker board squares and pieces are stored as Sprites. Each set of checker pieces are stored in separate lists, so the program can easily access them if needed. Similarly, the program stores the checkerboard in a two dimensional list in order to effectively access any given square on the board. For example, if the list was called “board”, and the program needed to do something with the fourth square in the third row, it could access that square using the syntax “board[2][3]”. The code uses two and three instead of three and four because counting in computer programming starts at zero.

Running a game on a computer relies entirely on a section of the code called the event loop, part of a broader concept called event-driven programming. Event-driven programming simply means “the flow of the program is determined by events triggered by user actions or system occurrences” (Krisnamughni 2023). This runs continuously, checking if certain conditions have been met, until the game ends and the program breaks out of the loop. Another

useful feature in PyGame is its ability to track when a user has clicked the mouse. In this game of checkers, the event loop waits until it detects a click of the mouse. More specifically, it waits until the user clicks on a piece they are allowed to move. Once this takes place, the program calculates the user's possible, legal moves

When a piece is only able to move one space forwards or make a single jump, the calculations are simple: the program simply checks if there are any other pieces blocking this piece's possible paths. If there are not, the program highlights the squares this piece is allowed to move to, and waits for the user to decide where they want to move and click on one of the highlighted squares. When there are more complex moves on the board—such as several different possible double, triple, or even longer jumps, the code likewise grows more complex. In order to correctly calculate all possible moves, the program executes what is called a depth-first search algorithm—a common solution to a coding problem such as this one.

Throughout all of these processes, the event loop continues to run, waiting for a click of the mouse or other conditions to be met. The loop will continue to run until the game ends. It checks for the end of the game by checking the size of the respective lists that store the pieces. A piece is deleted from its list when it is taken off the board, so once the event loop sees one of the lists is empty, meaning all pieces have been taken off the board, the program exits the event loop and the game is over.

The Day the Music Died? Generative AI and its Impact on the Music Industry (STS Topic)

AI has the capability of enhancing and improving a great number of areas. Some of those areas are fairly inconsequential, like adding a simple AI model to the aforementioned implementation of checkers to allow a player to play against the computer. But others bear much more significance, in some cases even impacting one's livelihood. One such field where this

rings true is the music industry. The sociotechnical portion of my thesis will explore the impacts that a subset of AI, called generative AI, will have on the music industry. Throughout this research process, I will frame and support my arguments with the evidence acquired through literary analysis of a wide variety of sources.

Generative AI is still fairly new to the scene, but has already become a hot-button issue regarding how it will affect the music world, and there are a wide range of opinions on the matter. It is generally agreed upon that Generative AI will inevitably influence the music industry, with some going as far to say we are "witnessing the dawn of a whole new music business" (Kimbrell 2023). Another common opinion is that AI can be a helpful tool in the creative process. Gideon Kimbrell, in his article for *Rolling Stone,* points out that artists can "use AI to overcome writer's block, ensure that style and content tone is unique in the final stages of editing, and fill skillset gaps (e.g., musicians making album art, songwriters generating vocals, etc.)" (Kimbrell 2023).

A third argument centers around the inherent humanity of music, and how we ought to be cautious with how much power AI is given. Jessica Powell, chief executive of the start-up Audioshake, points this out, making the point that "Taylor Swift is far more than a Taylor Swift song" (Hunter-Tilney). These questions raise some compelling points that are worth considering. Inherently, humans desire connection with one another, and music serves as a unique medium in which this can occur. When a variable such as Generative AI enters the equation, it destabilizes that connection. This may have lasting consequences for the music industry writ large.

Additionally, Generative AI raises several concerns about copyright issues. In Aril 2023, "Universal Music Group—the music company representing superstars including Sting, The Weeknd, Nicki Minaj and Ariana Grande…sent urgent letters…to streaming platforms, including

Spotify and Apple Music, asking them to block artificial intelligence platforms from training on melodies and lyrics of their copywritten songs" (Yurkevich 2023). Shelly Palmer, a Professor of Advanced Media at Syracuse University, says in response to this issue: "You can flag your site not to be searched. But that's a request—you can't prevent it" (Yurkevich 2023). This reality raises several ethical questions about what data AI has trained on to create its "original" music, and what data it should be allowed to train on.

Conclusion

The overaching issue this thesis attempts to address the question: how much AI is too much AI? When utilized in the context of something simple like a game of checkers, it is harmless. But once it crosses into threatening humans' jobs, the stakes get significantly higher. Excessive utilization of AI has the potential to cause problems in society as a whole, and the music industry is no exception. "While AI-generated music can be impressive, it lacks the depth of emotions and personal experience that human musicians bring to their compositions" (Clarke 2023). If AI continues at the pace it is currently developing at, "some music industry jobs could disappear" entirely (Clarke 2023). This would be catastrophic for those in the music industry trying to make a living off the jobs AI replaced. The goal of this research paper is to shed more light on this issue, and how important it is to prevent it from ever becoming a reality.

References

(2023, May 24). *AI in Music Production: Enhancing Human Creativity or Replacing It?* Musicians Institute; Musicians Institute College of Contemporary Music. https://www.mi.edu/in-the-know/ai-music-production-enhancing-human-creativity-replacing/.

(2023, September). *The Musical Journey of Luke Richard Powers*. UVA Arts; University of Virginia. https://magazine.arts.virginia.edu/stories/the-musical-journey-of-luke-richard-powers.

Berry, A. (2023, August 29). *How AI "Voice Fakes" Are Changing Music*. UVA Today; University of Virginia. https://news.virginia.edu/content/how-ai-voice-fakes-are-changing-music.

Cihon, P., & Demirer, M. (2023, August 1). *How AI-Powered Software Development May Affect Labor Markets*. Brookings. https://www.brookings.edu/articles/how-ai-powered-software-development-may-affect-labor-markets/.

Clarke, L. (2023, June 22). *Artificial Intelligence: The Future of the Music Industry?* Recording Arts. https://recordingarts.com/artificial-intelligence-the-future-of-the-music-industry/.

Hunter-Tilney, L. (2023, July 20). *AI in the Music Industry*. Financial Times; Columbia University. https://www-ft-com.ezproxy.cul.columbia.edu/content/2c1c2016-69b7-48aa-b333-4c1380bb9102.

Kimbrell, G. (2023, May 9). *How Generative AI Can Impact Music and Content Creation*. Rolling Stone. https://www.rollingstone.com/culture-council/articles/how-generative-ai-can-impact-music-content-creation-1234731365/.

Krisnamughni. (2023, July 14). *Event-driven programming in simulation games*. Medium. https://krisnamughni24.medium.com/event-driven-programming-in-simulation-games-16cbd266680b.

Lynch, S. (2021, February 16). *Enhance, not Replace: AI's Potential to Make Our Work – and Lives – Better*. Stanford University Human Centered Artificial Intelligence; Stanford University. https://hai.stanford.edu/news/enhance-not-replace-ais-potential-make-our-work-and-lives-better.

Martineau, K. (2023, August 18). *What is Generative Ai?*. IBM Research Blog. https://research.ibm.com/blog/what-is-generative-AI.

Simon, F., Altay, S., & Mercier, H. (2023, October 18). *Misinformation Reloaded? Fears About the Impact of Generative AI on Misinformation are Overblown*. Harvard Kennedy School Misinformation Review; Harvard University.

https://misinforeview.hks.harvard.edu/article/misinformation-reloaded-fears-about-the-impact-of-generative-ai-on-misinformation-are-overblown/.

Yurkevich, V. (2023, April 18). *Universal Music Group calls AI music a "fraud," wants it banned from streaming platforms. Experts say it's not that easy*. CNN. https://www.cnn.com/2023/04/18/tech/universal-music-group-artificial-intelligence/index.html