# Analyzing the Effectiveness of Copyright Law on AI Output in Software Engineering and Future Developments

Surendra Jammishetti, sjammish@ucsc.edu

***Abstract*—TODO: need to do this later**

***Index Terms*—AI, Software Engineering, Copyright**

## I. Introduction

THE emergence of AI tools has, no doubt, changed many aspects of how work is approached in many professions, but the greatest impact has been on software engineering. New tools like Github Copilot, an AI assisted code generation tool thats usable in many modern code editors today, boasts a 2x speedup of feature completions and a large boost in productivity for its users [1]. However accurate those claims may be, theres no doubt that these tools are being used widely by the developer population, and they arent going away any time soon.

However the code generated by these tools must come from somewhere. Many models source their training data from the public internet, but not everything in public access allows for commercial / derivative use. Such models ignore this step, jumping straight towards using this data with any disregard for copyright protections in place. While copyright has been slow in the past, there is no doubt the legal systems around the world will regulate and form precedent for these utilites; Inevitably placing their users under risk.

**For those concerned with the legality of the software they produce, the usage of AI-generated code should be avoided as soon as possible, as they could become succeptible to copyright law in the near future.**

## II. Background

To help understand the context that this paper resides in, below are lay-person explanations of AI training, Software Licensure, AI tools, and more.

### A. Data Acquisition

The way AI development labs gather data for their models is mainly through web scrapers. Web scrapers are programs that access a immense number of websites, downloading and orgranizing the all the data they find [2]. The gathered data is then used to train whichever model they are interested in.

The modern way for a website to prevent a webscraper from downloading its contents is a file called a ‘robots.txt‘, that any honest web scraper would look for first, and obey the rules inside [3]. The problem with this ”trust me” approach is that a dishonest, or even badly programmed, web scraper can come along and look past this file and do whatever they please[3].

### B. Software Licenses

While it may come as a suprise to the average person, even code nowadays has some kind of protection against copying / un-permitted usage. The mechanism for this is called a ”Software License”[4]. The code that these licenses protect is called ”source code”. There exist many kinds of licenses, mainly meant for open use towards the public, but there are a two that are crucial to understand.

- Copy Left
  A ”Copy Left” license allows the general public to view and do whatever they would like with the source code, but they enforce that any derivative must also use the same license. In effect this is a strictly anti-commercial license as its quite hard for a buisness to use Copy Left Licensed software as anything they produce using that code must have the same Copy Left license, and as a result live in the public domain. [4]
- Proprietary
  A ”Proprietary” license are the most restrictive, as they prevent any viewer / user from copying, modifying, redistrbuting, etc [4]. They are the defaco type of license for any commercial code, as they protect code the best, and are legally viable [5].

### C. AI-Powered Tools

The most popular AI tool for developers is Github Copilot, which helps developers as they write code. Its most important capability, in the context of this work, is its power to automatically generate code when you ask. The same capability exists for Claude, OpenAI’s models, and others, where they have the capacity to take a description of what the user wants, and can write code to perform that task [6]. For example, If we want to make our own snake game, we could trivially ask a model to program the snake game.

### D. Copyright

Copyright is a mechanism used to protect created works from direct copying but not a sufficiently derivative use or one that gives adequate credit to the original author [7, p. 55]. There has been one landmark case regarding copyright and programming, Google INC v. Oracle America, Inc, which regards the copying of generic and widely known API code by Google from Oracle [5]. API in short means that its code that acts as a abstracted layer, such that the user doesnt have

to think about whats going on underneath. Think of it like me making my own laptop, but still keeping around the QWERTY keyboard so the users of my laptop dont have to learn a whole new keyboard layout. The ruling stated that its alright for google to copy this code because they were doing it in the interest of the general public and thought it wasnt possible to copyright something so generic, leading to google winning the case.

## III. AI and Code Generation

AI has been getting better and better at generating code [8], due to the vasty increasing training data these models are harvesting from the web [9]. Especially with many popular code repositories being open online, its fairly straightforward to assume that they are being used for training. While its impossible to know exactly whats being used as training data and what isnt by these large companies, they arent making it any easier with their lack of clarity and reassurance [10]. Therefore, it can be reasonably assumed that licensed code that is public purview is also being used as training data.

The issue then arises with the licensure of the code ingested by the model. Whether its under a "Proprietary" license or a "Copy Left" license, the end user could be in trouble.

A Proprietary license would bar any derivative usage, putting the user in deep trouble if found out, whether they're trying to publicize their code or not.

A Copy Left, while less troublesome, an entity trying to privatize their code would fail to uphold the conditions of the license, which demands that any derivative works also be in the public domain.

We havent even gotten to the fact that many licenses require attribution to the original author, which would be completely lost in this process of ai ingestion and generation.

## IV. Detecting the Origins of Generated Code

While it may seem impossible, If there was a way to figure out where the code generated by an AI came from, any users of the generated code could be in huge trouble depending on the licensure of the source code. Such technology doesnt exist today but is tending towards that direction, as seen in this study: [11]. They dive deep into the possibility of detecting whether or not code generated by a model can be traced back / verified to be in some dataset.

Obviously this hasn't been attempted on mainstream models, but with their findings being so fruitful, its only a matter of time before new research builds off this foundation to see if generated code has its roots on public code. Logically then, source code authors could audit whether or not others have indirectly used their source code, via this AI ingestion and generation proxy. Now a solid case can be built up by authors with strict licenses, that this usage is in violation of their licenses and the derivate users now face legal trouble.

## V. Forward Compliance

While origin analysis of AI-generated code is not yet developed, preventative action must be taken now by those interested in upholding the legality of their software. As this technology develops, the risk of legal trouble only increases.

Forward compliance simply means to disassociate with AI-generation tools, and return to a "traditional" style of programming. This is doubley-so for corporate projects which would face even higher repurcussions by any future lawsuits.

While some may argue that the process of AI-generating code is transformative by defenition, but currently there exists no precedent, so claming as such is meaningless. When dealing with Copyright, its best to live on the side of pessimism, so treating AI-generation as akin to copying allows users of these tools to be protected in case future precedent is pessimistic as well.

Others can argue that the copying of code has precedent, and is allowed according to Google INC v. Oracle America Inc., but a deeper inspection of the case details must be had. The copying of the API code was only allowed because its value lay in the familiarity the general public already had with it. Additionally this API code lacked any of the deeper functionality which could be copyrighted and protected, its sematics ruled to be too abstract [5].

This reasoning cannot be used for all code however, as the AI-tools can generate anything from this API-like code, to code that performs real work, with no care for the licensure of either. If an author were to form a case claiming that their proprietary code was "copied" and now exists with an unauthorized holder, using the afforementioned origin anazlyzer to show the means of "copying, the results of this case would turn out very different.

## VI. Discussion and Summary

There exists a real legal threat, maybe not now but certainly in the near future, for those using AI-generated code within their produts, doubly-so if its commercial in nature. As of right now there is no way to tell if the code generated comes from a open-source repository or one thats heavily licensed. With the research on origin analyzers only developing, its very reasonable for AI-generated code to be back tracked to its source, leading users to fall under jurisdiction of this source code. For those concerned with the legality of their code, its best to stray away from these tools.

## Acknowledgments and Notes

I'd like to thank Dr.Kirsch and my peer reviewers for helping this essay come together. One thing I'd like to note are that the text formatter im using for this template does not allow for citations to be bolded, so note that any references to [7], [5] and [11] are my peer-reviewed sources, so sorry for the trouble.

## References

[1] E. Kalliamvakou, *Research: Quantifying github copilot's impact on developer productivity and happiness*, May 2024. [Online]. Available: https://github.blog/news-insights/research/research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/.

[2] Miquido. “What is ai data scraping? ”[Online]. Available: https://www.miquido.com/ai-glossary/what-is-ai-data-scraping/.
[3] *What is a robots.txt*, 2024. [Online]. Available: https://www.cloudflare.com/learning/bots/what-is-robots-txt/.
[4] P. Odence. “Five types of software licenses you need to understand: Black duck blog. ”[Online]. Available: https://www.blackduck.com/blog/5-types-of-software-licenses-you-need-to-understand.html.
[5] H. L. Review, *Google llc v. oracle america, inc.* Nov. 2021. [Online]. Available: https://harvardlawreview.org/print/vol-135/google-llc-v-oracle-america-inc/.
[6] M. S, *How to use github copilot: Setting up and learning various useful ai coding methods*, Jun. 2023. [Online]. Available: https://www.hostinger.com/tutorials/how-to-use-github-copilot.
[7] S. Stokes, *Art and Copyright*, Third. Hart, 2021.
[8] T. CodeSignal, *Ai vs. human engineers: Benchmarking coding skills head-to-head - codesignal*, Sep. 2024. [Online]. Available: https://codesignal.com/blog/engineering/ai-coding-benchmark-with-human-comparison/.
[9] R. LaCour, *Gpt 4o mini vs. gpt 3.5: Key differences and advantages*, Jul. 2024. [Online]. Available: https://blog.getodin.ai/gpt-4o-mini-vs-gpt-3-5-key-differences-and-advantages/.
[10] S. Willison, *It’s infuriatingly hard to understand how closed models train on their input*, 2023. [Online]. Available: https://simonwillison.net/2023/Jun/4/closed-model-training/.
[11] W. Ma, Y. Song, M. Xue, S. Wen, and Y. Xiang, “The ”code” of ethics: A holistic audit of ai code generators,” *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 5, pp. 4997–5013, 2024. DOI: 10.1109/TDSC.2024.3367737.