Thesis Report

Abstract
In recent years, Artificial Intelligence has become more sophisticated, useful, and almost necessary for a wide range of tasks that are performed in many very different aspects of our society.

It is improving at such a fast rate that many professions are partially or totally being replaced by it, and this raises the questions: How similar is Artificial Intelligence's work to human work? What are the differences between them?

In the field of informatics, these questions have become very interesting as a considerable amount of code is today written or guided by Artificial Intelligences such as ChatGPT, Copilot, or DeepSeek.

In the last months, I have been researching this topic by analyzing different types of code from the GitHub platform, written in these 9 common programming languages: Python, JavaScript, Java, Typescript, C++, Golang, Php, C and C#.

By using the "AI code detector" tool from OpenAI, I then compared Artificial Intelligence-written code with human-written code. By analyzing patterns such as indentation, variable naming, and code structure, the detector was able to provide a confidence level about the probability of the code being written by Artificial Intelligence.

The main objective of my work is to have a deeper understanding of the similarities and differences between Artificial Intelligence and human written code and to provide a valuable dataset of code written by Artificial Intelligence.

1. Introduction

Nowadays, Artificial Intelligence is increasingly important in our everyday tasks, and coding is no exception. Artificial Intelligences like ChatGPT are widely used in this field; it's mainly used to generate code or automate some repetitive tasks and processes, but it can also be used to guide and help during code writing and/or understanding.

This innovation, of course, does not only simply improve the developers' life as it also comes with some risks. Using AI to generate code may bring bugs or inefficiencies in the code as AI usually works better with shorter and technical tasks rather than with longer chunks of code that require semantic understanding.

Building on this growing presence of AI in software development, it becomes essential to evaluate how AI-generated code compares to code written by human developers. As AI tools are increasingly integrated into professional and educational workflows, understanding their coding patterns, strengths, and limitations is crucial. This thesis aims to explore the extent to which AI-generated code differs from human-written code in terms of structure, clarity, maintainability, and best practices. By analyzing both qualitative and quantitative aspects of code, this study hopes to shed light on the evolving dynamics between human programmers and artificial intelligence and the potential implications for future coding standards, education, and software engineering practices.

1.1. Report Structure

In Section 2: “Related Work,” I will explore and critically evaluate existing research that investigates the similarities and differences between human-written and AI-generated code. Several recent studies have attempted to assess the quality, readability, and maintainability of code produced by large language models. Some works focus more on patterns and common programming idioms, while others investigate code correctness and vulnerability rates. Furthermore, I will examine studies from the software engineering and machine learning communities that discuss the implications of AI assistance in professional and educational contexts.

In Section 3: “Study Design,” I will dive deeper into the question: “To what extent is AI-generated and manually written code different?” To approach this, I first curated two datasets—one consisting of code written entirely by human developers, and another composed of code generated by AI systems (primarily ChatGPT and Copilot). I focused on commonly used programming languages and ensured comparable code functionality across both sets. I then identified specific dimensions of analysis, including code structure, naming conventions, comment usage, indentation and formatting consistency, function length, and others. The study also considers contextual factors such as code origin (e.g., educational, professional, open source) and task complexity to ensure a fair comparison.

In Section 4: “Results Discussion,” I interpret the results of the comparative analysis between human-written and AI-generated code. Key patterns emerged that underscore the distinct tendencies of each source. For instance, AI-generated code tended to use more generic variable and function names, often closely reflecting the prompt given, while human-written code showed more context-aware and meaningful naming conventions. In terms of structure, AI code exhibited consistent indentation and formatting due to the model’s training on style-conformant code, whereas human code showed greater stylistic variability. Furthermore, AI-generated code was more likely to include inline comments. However, some AI-generated solutions also demonstrated elegance in handling edge cases, suggesting that, in certain scenarios, the AI mimics best practices learned from large codebases. The discussion also touches on the limitations of AI-generated code, including occasional lack of modularity, redundancy, and superficial understanding of context. Overall, the findings highlight differences that have implications for code review practices, AI tool adoption, and programming pedagogy.

2. Major Contribution

The major contribution to my thesis project was:

A Comparative Analysis between AI-Generated Code and Human-Written Code: A Preliminary Study.

3. Related Work

[To be filled: 825 words of related work content]