AI-Driven Evaluation of Human Code: Leveraging Artificial Intelligence for Review Processes

Sponsor: Alex Shaharudin
Software Sigmas: Mikhail Nathoo, Ernest Nikolaychuk, Saman Pordanesh, Tanish Datta, Sam Farzamfar
Department of Electrical and Software Engineering, University of Calgary

UNIVERSITY OF CALGARY
SCHULICH School of Engineering

Abstract

- The LM Studio & ChatGPT VSCode Extension presents an innovative solution for seamlessly integrating advanced language models into the coding workflow. In addition to enabling connections to local Language Models (LLMs) through LM Studios and OpenAI's ChatGPT API, this extension now offers the capability to utilize Google Cloud for online private computing. This enhancement extends the flexibility of the extension, allowing users to harness the power of cloud-based computing resources while maintaining privacy and security. By providing a unified interface within Visual Studio Code, developers can effortlessly access a range of language model functionalities, including code completion, documentation generation, and natural language understanding, enhancing productivity and enabling more efficient development workflows. With its expanded feature set and emphasis on user convenience, the LM Studio & ChatGPT VSCode Extension continues to lead the way in facilitating the integration of AI-powered language capabilities into the development process.

Introduction

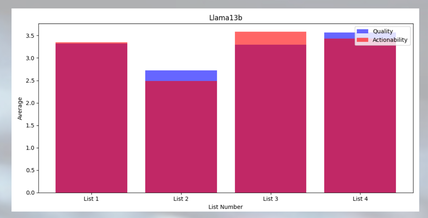
- **Privacy-Oriented Approach:** Recognizing the importance of data privacy, our project was meticulously designed with privacy in mind. The system is independent and capable of running on local machines or private servers, ensuring that sensitive code remains within controlled environments. This approach prioritizes data security and confidentiality, offering peace of mind to developers and organizations alike.
- **Innovative AI-Powered Solution:** Our project introduces an innovative AI-powered solution aimed at revolutionizing code reviews. By leveraging advanced AI algorithms, we have developed a system that significantly reduces the time required for code reviews while enhancing their effectiveness.
- **Streamlined Workflow Integration:** Configured as an intuitive add-on for Visual Studio Code, our solution seamlessly integrates into developers' workflows. It provides real-time, precise feedback directly within the coding environment, allowing developers to easily incorporate it into their regular tasks without disruption.
- **Enhanced Development Process:** With our solution, developers can focus more on intricate and imaginative projects rather than spending excessive time on routine code reviews. By automating the evaluation of code against efficiency and design standards, our technology not only expedites the software development cycle but also ensures adherence to strict coding practices, ultimately leading to improved product quality.

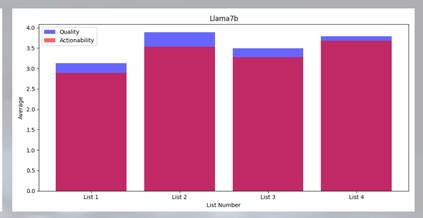
Initial Model Selection

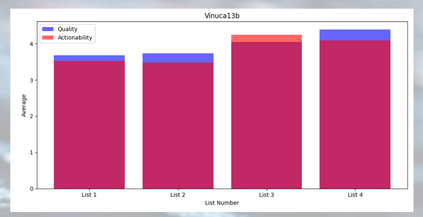
Model Exploration: We embarked on a rigorous exploration of various open-source resources such as Hugging Face, GitHub, and GPT to identify suitable NLP models for our project. This involved investigating approximately 40-50 models, focusing on compatibility with the MacBook M1 and their documented performance. Through this process, we narrowed down the selection to about 30-35 compatible models, ultimately downloading and testing 14 of them to assess their suitability in generating reasonable answers to test prompts.

Primary Model Selection: Following extensive testing and evaluation, we selected three primary model candidates from the pool of 14 options. This selection process involved posing specific coding tasks and code explanation questions to each model and scoring their responses based on predefined rubrics. After averaging the scores from multiple perspectives, we identified one 7b model and two 13b models as the most promising candidates for our project's requirements.

Aspect	1-3	4-6	7-9	10
Accuracy	Incorrect or irrelevant	Partially correct, but has significant errors or omissions	Mostly correct, minor errors or omissions	Completely correct and comprehensive
Clarity	Confusing and poorly structured	Somewhat clear but with noticeable issues in structure or explanation	Clear and well-structured, with minor room for improvement	Exceptionally clear and well-explained

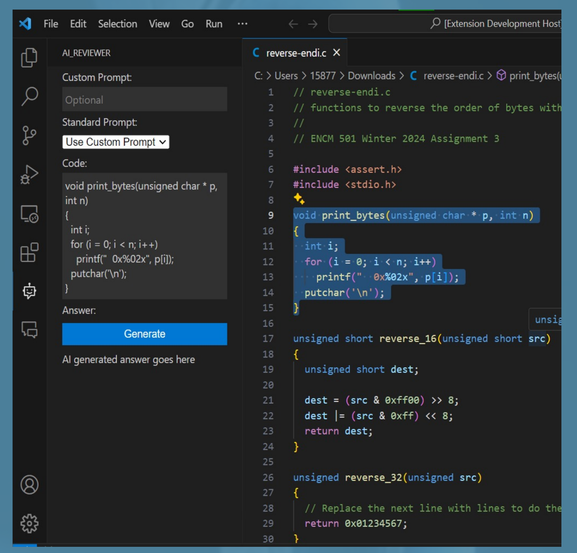


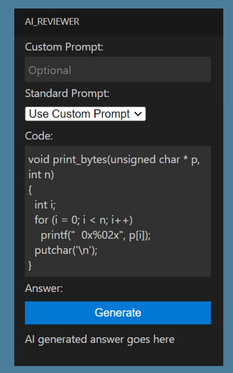


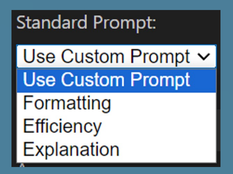


VSCode Extension

- **Seamless Integration:** The extension seamlessly integrates with VSCode, offering a familiar environment for code review. Developers can effortlessly incorporate code review tasks without switching applications.
- **Custom Prompt:** Users can customize prompts to meet project requirements, empowering developers to focus on specific aspects of code evaluation and streamline the process.
- **Prompt Options:** A drop-down menu offers various prompt categories like efficiency or design standards, ensuring comprehensive code evaluation with flexibility and ease of use.
- **Code Input:** Developers can input code directly within the extension, initiating review sessions without leaving their workspace, promoting efficiency.
- **AI Generation:** Leveraging advanced AI, the extension provides real-time feedback and suggestions based on code input and prompts, facilitating informed decisions and enhancing code quality.







Model Evaluation

Comprehensive Model Evaluation Process: Our model evaluation process involved a multi-step approach, beginning with the selection of three primary model candidates based on their performance in responding to four specific questions - two on coding tasks and two on code explanation. Each response was meticulously scored according to predefined rubrics, allowing us to objectively assess the models' capabilities from different perspectives.

Diverse Dataset Evaluation: For the main model evaluation experiment, we utilized a diverse dataset consisting of 200 code samples spanning four different programming languages: C++, JavaScript, TypeScript, and Go. We posed code explanation questions to each model 200 times, resulting in a total of 600 responses evaluated. This extensive evaluation provided a comprehensive understanding of the models' performance across various programming languages and coding scenarios, enabling us to make informed decisions regarding the selection of the main model for our project.

Requirements

- The extension and model require a minimum of an M1 MacBook or equivalent Windows system for optimal performance.
- Alternatively, for larger-scale projects or specific company needs, utilizing a local industrial server can provide sufficient computational power.
- Cloud computing options such as Google Cloud Platform, AWS, or Firebase can also be utilized to leverage scalable resources for running the extension and model efficiently.