UNIVERSITY OF CALGARY

CAPSTONE PROJECT PRESENTATION:

SIGMAPILOT

(AI-DRIVEN EVALUATION OF HUMAN CODE)

May 2024

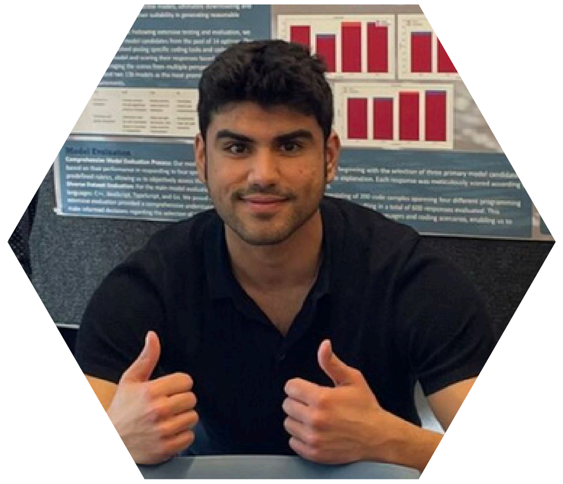
The Development Team

Sam Farzamfar
Software Developer

Tanish Datta
Software Developer

Saman Pordanesh
ML Developer

Ernest Nikolaychuk
ML Developer



Mikhail Natto
Team Manager

About The Project

Project Title:

AI-Driven Evaluation of Human Code

Project Purposes

I

Privacy

Importance of data privacy and security in code reviews

II

AI Solution

AI-powered solution to enhance code review efficiency and effectiveness

III

Environment Compatability

Seamless integration with Visual Studio Code for streamlined workflow

Project Motivation and Objectives

Use the most of new LLM and NLP technologies

Ensuring user privacy and data security

Enhancing code review processes with AI

Enabling offline accessibility and productivity

Dual Aspects of Our Project

AI Component

AI-Driven Code Review

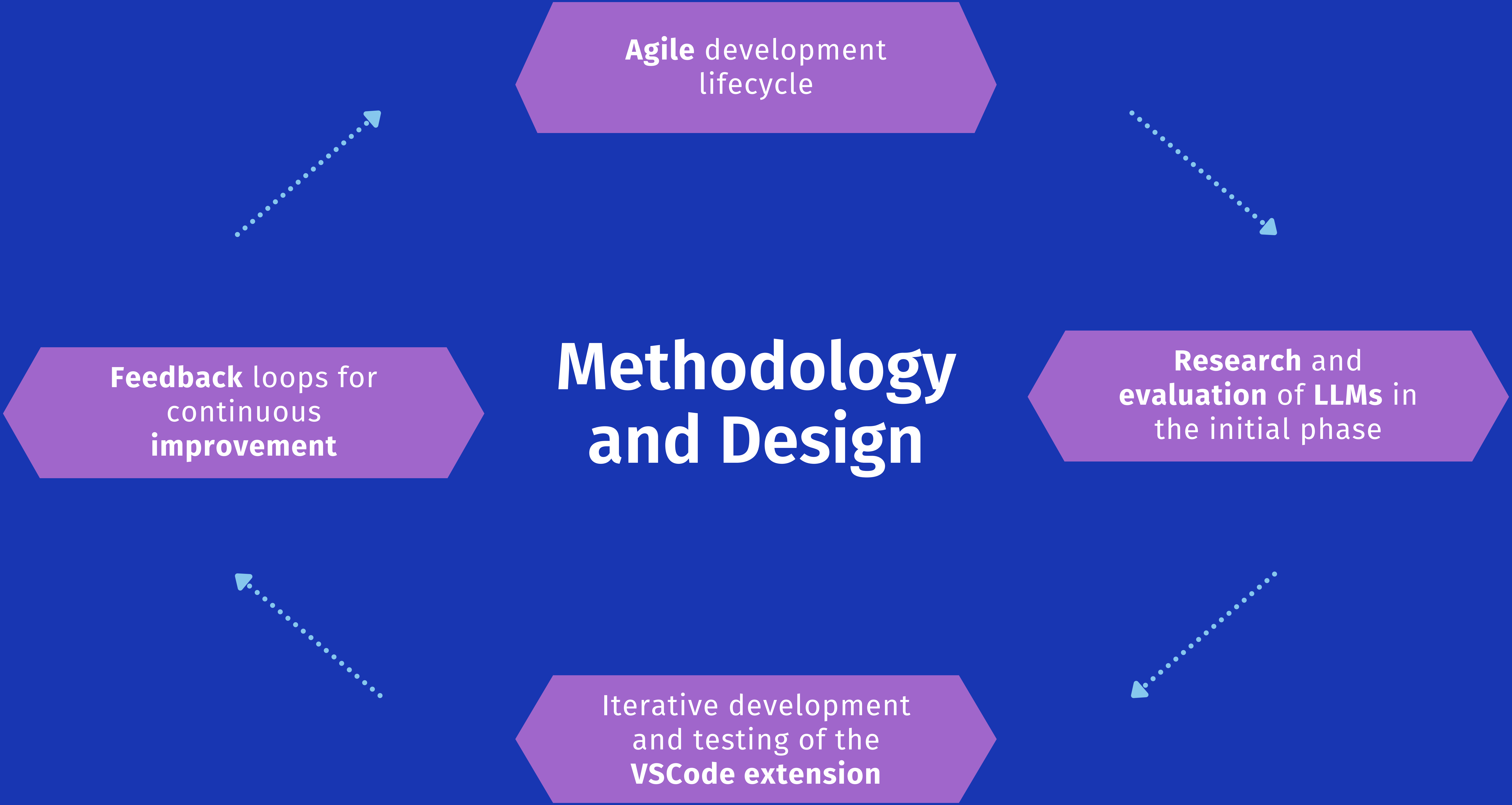
- Advanced LLM models for code analysis investigation
- Ensures model compatibility with a local machine

VS Code Extension

SigmaPilot VS Code Extension

- Seamless integration with Visual Studio Code
- Real-time code feedback and suggestions

Methodology and Design



Agile development lifecycle

Research and **evaluation** of **LLMs** in the initial phase

Iterative development and testing of the **VSCode extension**

Feedback loops for continuous **improvement**

Product Scope & Functionality

Must-haves

- LLM Research Documentation
- Working Solution
- Deployment as VSCode extension

Should-haves

- comparative analysis
- reactive GUI
- flexible model interaction

Nice-to-haves

- fine-tuning models

Initial Model Selection

- Exploration models from resources like **Hugging Face**, **GitHub**, and **GPT**

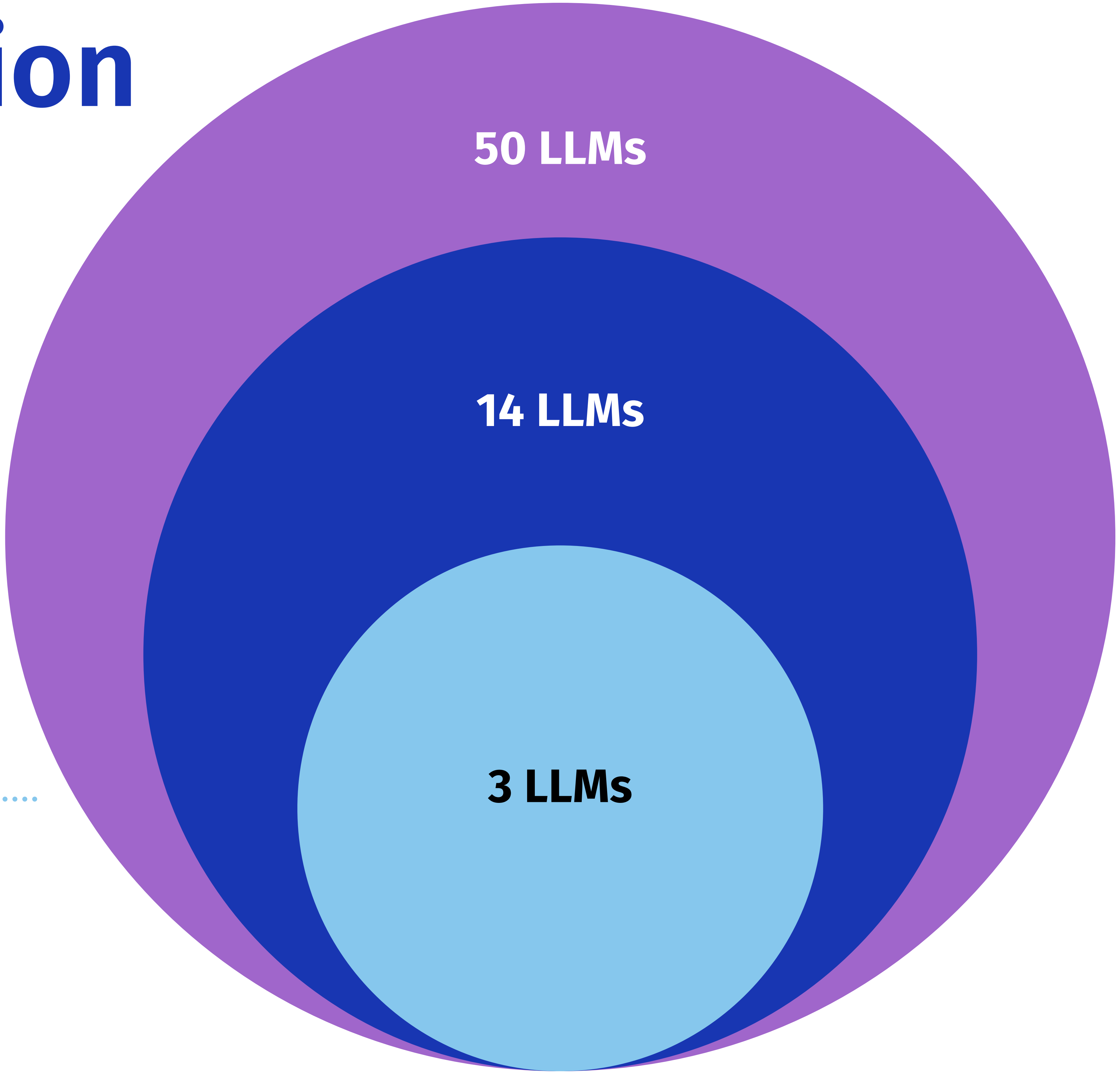
  40 - 50 Models

- Testing models for compatibility and performance

  14 Models

- Selection of three primary model candidates based on coding tasks and code explanation responses (Basic 4 question testing)

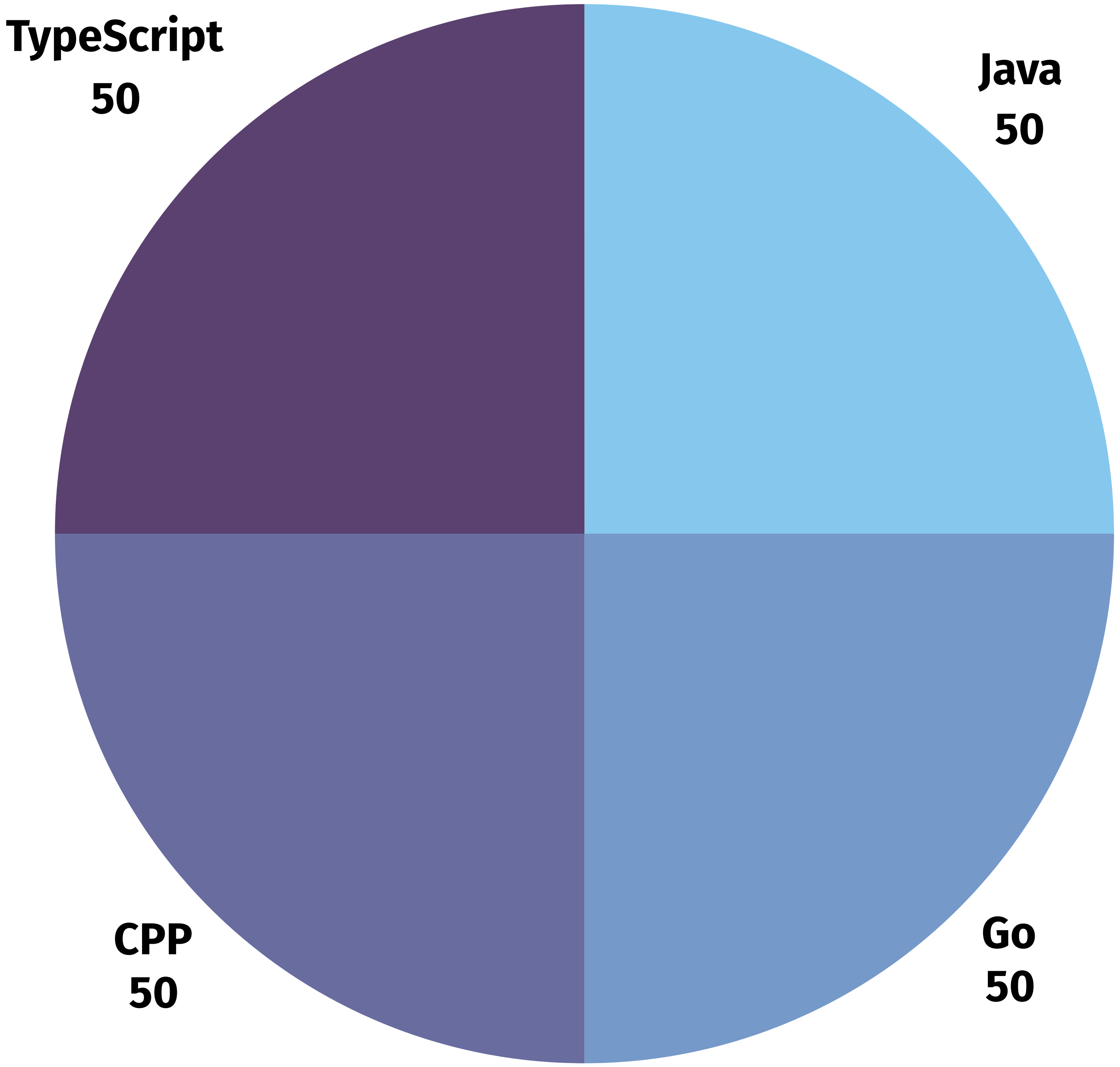
  3 Modles



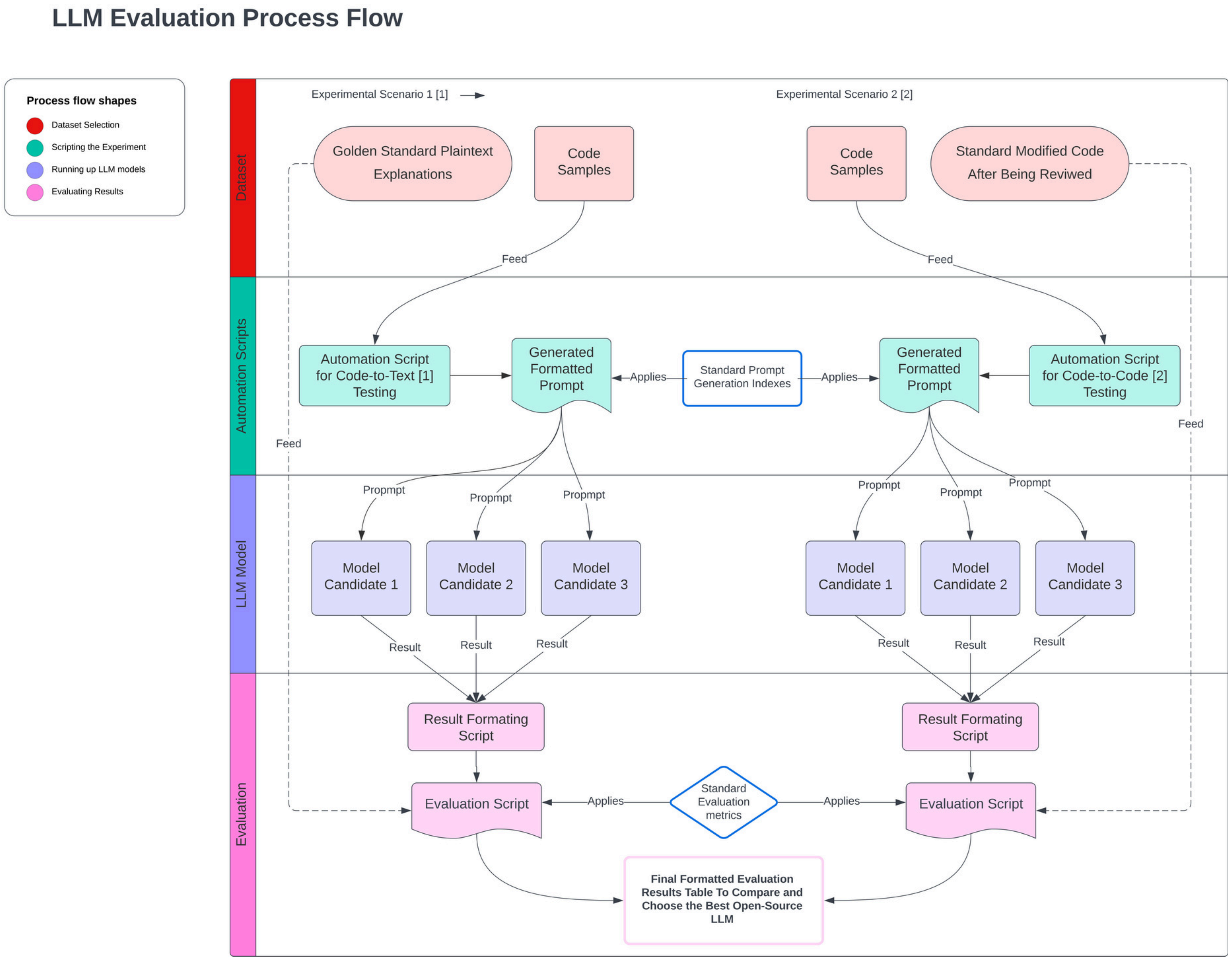
Primary Model Evaluation

Main Model Selection Experiment

- Evaluation of models on **200 code samples** across **C++**, **JavaScript**, **TypeScript**, and **Go**
- Comprehensive scoring based on accuracy, clarity/actionability of feedback
- Final selection based on the best cumulative score out of this experiment

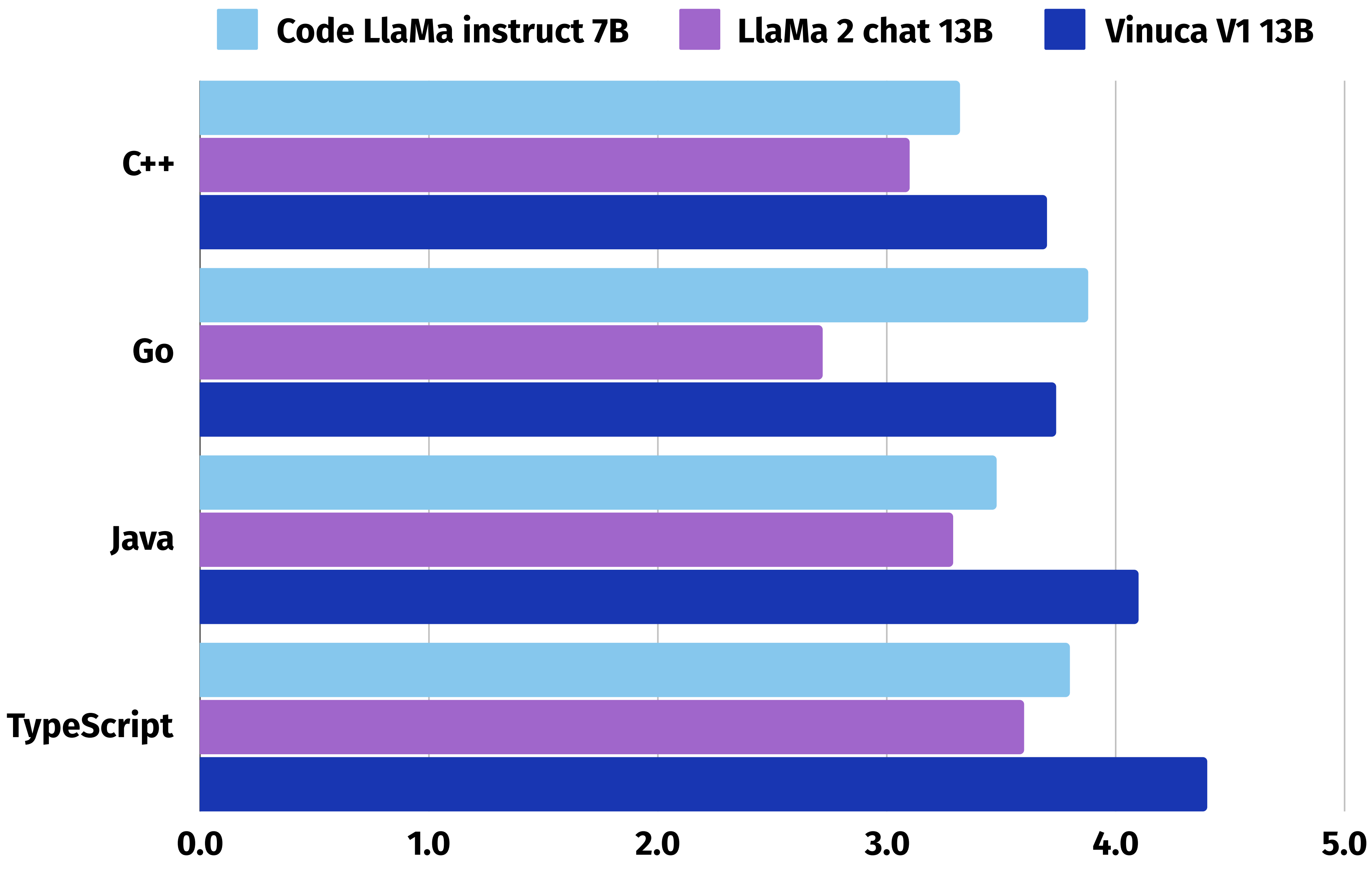


Primary Model Evaluation Workflow Diagram

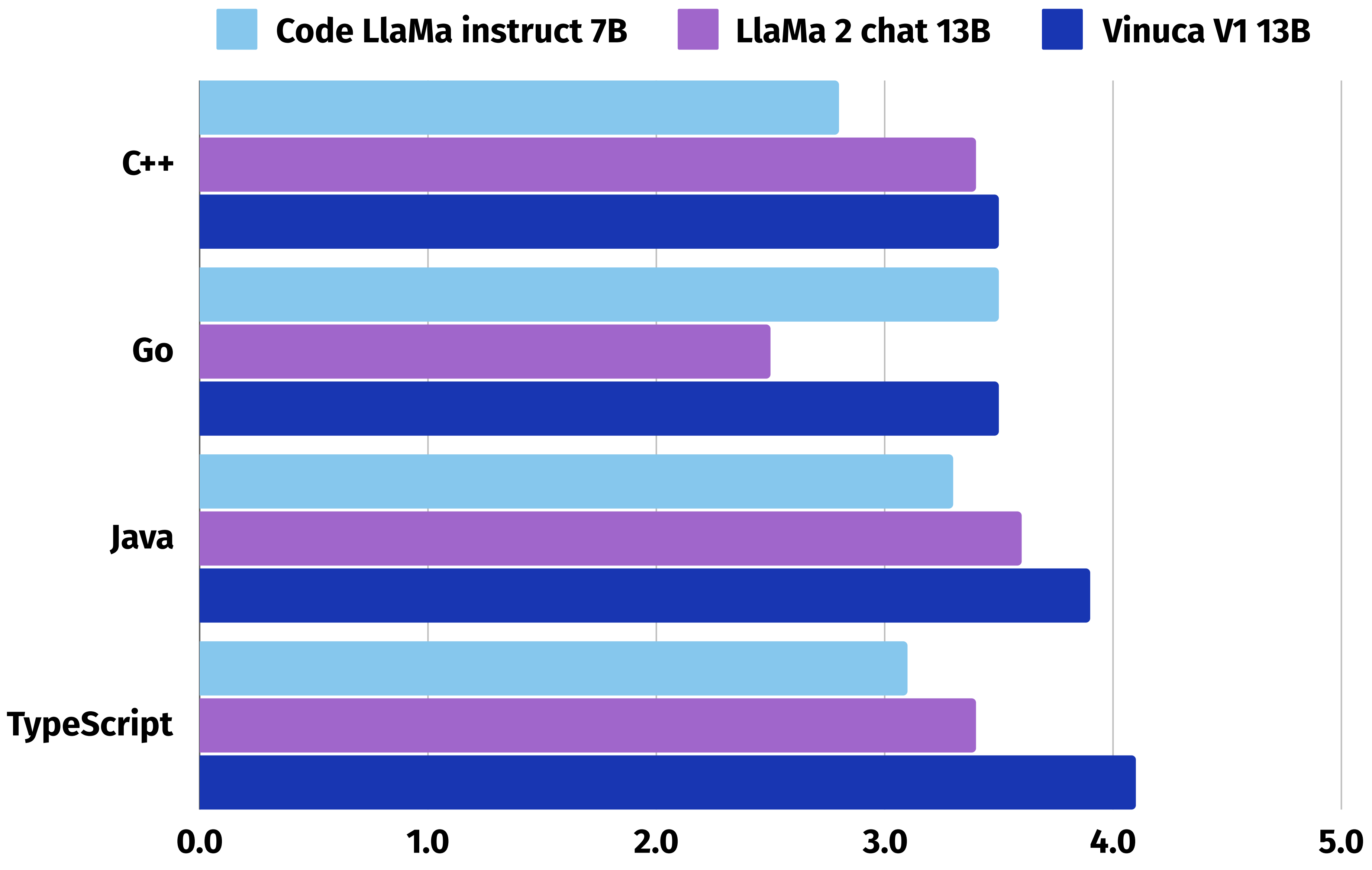

LLM Evaluation Process Flow
Process flow shapes
● Dataset Selection
● Scripting the Experiment
● Running up LLM models
● Evaluating Results
Dataset
Automation Scripts
LLM Model
Evaluation
Experimental Scenario 1 [1]
Experimental Scenario 2 [2]
Golden Standard Plaintext Explanations
Code Samples
Code Samples
Standard Modified Code After Being Reviwed
Feed
Feed
Automation Script for Code-to-Text [1] Testing
Generated Formatted Prompt
Applies
Standard Prompt Generation Indexes
Applies
Generated Formatted Prompt
Automation Script for Code-to-Code [2] Testing
Feed
Feed
Propmpt
Propmpt
Propmpt
Propmpt
Propmpt
Propmpt
Model Candidate 1
Model Candidate 2
Model Candidate 3
Model Candidate 1
Model Candidate 2
Model Candidate 3
Result
Result
Result
Result
Result
Result
Result Formating Script
Result Formating Script
Evaluation Script
Applies
Standard Evaluation metrics
Applies
Evaluation Script
Final Formatted Evaluation Results Table To Compare and Choose the Best Open-Source LLM

Experiment Results

Corrected **Quality** of Review by Model and Language



Corrected **Actionability** of Review by Model and Language



SigmaPilot VSCode Extension

The SigmaPilot VSCode Extension seamlessly integrates with Visual Studio Code, providing real-time code review capabilities. It offers customizable prompts, multiple prompt categories, direct code input, and real-time AI feedback to enhance the coding workflow.

Seamless integration with VSCode	**For real-time code review**
Customizable prompts	**Multiple prompt categories**
Direct code input	**Real-time AI feedback**

Final Design: A LLM integrated UI for any user in charge for code reviewing

SigmaPilot Features

The SigmaPilot extension offers a range of features designed to enhance the developer's workflow within Visual Studio Code. It provides flexible connections to both local and cloud-based Large Language Models (LLMs), ensuring seamless integration and secure interactions.

Feature	Description
Flexible Connection	Connect to both local and cloud-based LLMs for versatile usage.
Graphical Interface	Provides an intuitive graphical interface within Visual Studio Code for easy interaction.
Configurable Parameters	Supports various configurable parameters like URL, API key, model name, and max token count for customization.
Secure and Efficient API Interactions	Ensures secure and efficient interactions through well-defined API protocols.

SigmaPilot Implementation Details

The implementation of SigmaPilot involves a combination of modern development tools and techniques to ensure seamless integration and functionality. Developed with TypeScript and Svelte, SigmaPilot connects to both local and remote AI models efficiently.

Feature	Description
Developed using TypeScript and Svelte	Utilizes TypeScript for robust backend and Svelte for dynamic frontend development.
Integration with LM Studio	Enables connections to local AI models using LM Studio for enhanced performance and privacy.
Support for OpenAI API	Facilitates remote interactions with AI models via the OpenAI API, expanding the flexibility of the tool.
Detailed Sequence and Use Case Diagrams	Provides comprehensive diagrams to illustrate the data flow and use cases within the extension.

Future Works

Fine Tuning

The selected local LLM can be more robust in analysis and more tailored for a specific task, like code review, through fine-tuning with desired data.

Requirements

- **Appropriate Dataset**
- **Data Scientist and ML Engineer**
- **Computation Power (Cloud, etc.)**

Extension Expansion

The extension can be more featured and cover more code-review chat bot tasks, such as more built-in prompt scenarios, a chat history database, better UI, etc.

Requirements

- **Designed Scenarios**
- **TypeScript Developer**
- **Prompt Engineer**

Project Transition

1. VS Code Extention Installation

Search "**SigmaPilot**" on the VS Code extension search bar and install the extension. (Link)

2. LM Studio Configuration

- Install the "**LM Studio**" application. (link)
- On the model search bar, search "**TheBloke - vicuna v1 5 16k 13B**" and download it. (link)
- Run the Local Inference Server on the LM Studio. (Tutorial)

3. Connection and Use

Connect the VS Code extension to the running local model by setting the given interface URL by LM Studio.

Thank you!

Feel free to approach us if you have any questions.