Lab Assignment 2

Course: CS202 Software Tools and Techniques for CSE
Lab Topic: Commit Message Rectification for Bug-Fixing Commits in the Wild
Date: 11th August 2025

Objective

This lab introduces students to the basics of mining open source software (OSS) repositories. The process involves processing and analyzing commits on the GitHub version control system for popular real world projects. In the wild, what constitutes a bug and its fix is best judged by developers who contribute to those projects. Existing tools may not be able to capture such domian or project specific semantics. The overall aim of this lab assignment is to establish a framework for understanding how developers think of bug fixing commits. Potential applications include dataset creation for training automated program repair models, multi-versioned program analysis, patch generation, vulnerability localization for cybersecurity, and many more.

Learning Outcomes

By the end of this lab, students will be able to:

- ✓ Identify bug-fixing commits.
- ✓ Develop a framework for testing commit message alignment, and rectifying the same if applicable.
- ✓ Should be able to improve software maintainability for any open source project.

Pre-Lab Requirements

- Any Operating System (Windows, Linux, MacOS, etc.)
- Python 3.10 or later
- `pydriller` (tool support for mining software repositories)
- Pre-trained LLM (https://huggingface.co/mamiksik/CommitPredictorT5)
- (**recommended**) SET-IITGN-VM (https://doi.org/10.5281/zenodo.10467159)

Lab Activities

(a) **Repository Selection**:

- Choose **ONE** medium-to-large scale open-source repository to analyze. Make sure this is a **real-world** project (like flask), and not toy projects on GitHub.

(b) **Define Selection Criteria**:

- Establish your own criteria for selection (inclusion/exclusion) of repositories and include this information in your report. Basically, you need to specify how

you settled with the final selected repository. Recall the hierarchical funnel diagram from the slides from Lecture 2.

- Examples of selection criteria may include metrics such as the number of GitHub stars, forks, etc.
- You may use the SEART GitHub Search Engine to perform this task.

(c) **Bug-Fixing Commit Identification:**

- Identify bug-fixing commits from the repository. The strategy to define the notion of a bug as well as how to identify the corresponding commit, should be defined by you.
- For each bug-fixing commit, store the following information (in csv format).

Hash	Message	Hashes of parents	Is a merge commit?	List of modified files
...	...	...	...	...
...	...	...	...	...

(d) **Diff Extraction and Analyses:**

- For each modified file (in the previous step), store the following (as csv).

Hash	Message	Filename	Source Code (before)	Source Code (current)	Diff	LLM Inference (fix type)	Rectified Message
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...

(e) **Rectifier Formulation:**

Why is a rectifier needed? Developers may batch together multiple changes per commit spanning multiple files and sometimes multiple locations within the same file. This may cause the developer to use a misaligned, and/or imprecise commit message. Even if the LLM replaces the developer for generating the message, it may not be successful too. Therefore, a rectifier is needed to address this issue by *rectifying messages (if needed)* on a per file basis to better contextualize, and analyze fixes. The main creativity of the experiment lies in formulation of this rectifier, and left as an exercise.

(f) **Evaluation: Research Questions**[1]**:**

- ***RQ1 (Developer eval.)***: Do developers use a precise commit message in the fixing commit? Quantify the hit rate.
- ***RQ2 (LLM eval.)***: Does the LLM generate a precise commit message in the fixing commit? Quantify the hit rate.
- ***RQ3 (Rectifier eval.)***: To what extent were you able to rectify the message? Quantify the hit rate.

[1] Do not simply answer Yes/No. Support your findings using tables, plots, and analyses.

Resources

- Lecture 2 slides
- https://pydriller.readthedocs.io/en/latest/index.html
- https://huggingface.co/mamiksik/CommitPredictorT5