

Comparative evaluation of traditional and neural models for multi-class classification of GitHub commit messages

Tharsa S.^{1*} and Wijerathine P.M.A.K.¹

¹Department of Software Engineering, Faculty of Computing,
Sabaragamuwa University of Sri Lanka, Sri Lanka

*stharsa@std.appsc.sab.ac.lk

Commit messages are a critical source of information within software repositories, documenting changes made to code entries. While they are important to empirical software engineering and development cycles, these messages are often highly heterogeneous, informal, and brief. This results in a variety of challenges, especially when analyzing large scale repositories with thousands of commits. The lack of standardization makes it difficult to analyze the content of these repositories, leading to a need and opportunity to automate the processing of commit messages. Automated processing would enable the development of release notes, code change impact analyses, and maintenance analytics. This study evaluates the use of several text classification models aimed at analyzing messages within the software development life cycle (SDLC). In particular, these models attempt to classify the commit messages from the repository hosting service GitHub into ten SLDC tasks: build, chore, continuous integration (ci), documentation (docs), feature (feat), fix, performance (perf), refactor, style, and test. After identifying a public repository containing a balanced dataset of 2000 commit messages, several hours of data cleaning and manual labeling were conducted to ensure a dataset of high quality. Each of the ten classes was assigned 200 commit messages after which the dataset was fully categorized. Across the different model architectures, a stratified split was applied to each. These included the Optuna-tuned TF-IDF + LinearSVM, the Bi-LSTM, and the Fine-tuned BERT, and the Hybrid BERT-SVM model. As for the transformer models, I customarily encountered convergence problems with this domain-specific data, and for this reason I applied the additional simplifying techniques of Layer-wise Learning Rate Decay (LLRD) and a weighted loss for BERT as part of the loss function. BERT (or any transformer model) for the task of sequence classification is known to suffer from convergence problems. The findings indicate that, of all the BERT architectures, the Optuna-tuned TF-IDF + LinearSVM secured the maximum validation accuracy of 0.635 and macro F1 of 0.631. The primary challenge was the generalized language of the developers that blurred the semantic distinctions between chore and refactor and confused all of the models. This sets a solid scholarly foundation for subsequent work that aims to incorporate source code differences to address the existing gaps in classification.

Keywords: *BERT, Commit Messages, Software Repositories, Support Vector Machine, Text Classification*