GenAI for Software Development: Assignment 2

Joshua Vasquez jvasquez01@wm.edu

Cathy Gao ngao@wm.edu

1 Introduction

Code prediction aims to automatically generate or complete missing pieces of code within a function or class. CodeT5 is a pre-trained encoder-decoder transformation model designed for code understanding and generation. In this assignment, we fine-tune codet5-small, a smaller version of CodeT5 which, for the purposes of the class, offers a better trade-off between performance and computational efficiency. We fine-tune the model to predict if statements for python functions. In order to do this, we utilize a dataset given to us by the Professor, mask the target if statement, and flatten and tokenize the input. We then train codet5-small on our tokenized dataset, choosing the model with the lowest evaluation loss, and evaluate the model using a number of evaluation metrics. The source code for our work can be found at https://github.com/cathieg/CodeT5.

2 Implementation

2.1 Dataset Preparation

Flattening and masking: We start by using the dataset which the Professor gave us (50,000 training samples and 5,000 testing and validation samples) and mask the if statements the model is trying to predict with "<mask>". We then flatten the input. This is done through a function we created.

Code Tokenization: We first convert our dataframes into datasets. This then allows us to use the pre-processing function that the professor provided us to tokenize the data.

2.2 The Model

Model Fine-Tuning, Training, & Evaluation: We train and evaluate our codet5-small model on 50,000 and 5,000 inputs, respectively. We utilize a training batch of 32 and evaluation batch of 64 in order to maximize efficiency with model performance. To evaluate the model's performance, we use evaluation loss, where lower values indicate better model performance. We implement an early stopping of 2, meaning that if the model's evaluation loss continues to increase after 2 epochs, the model will stop training. This helps prevent both overfitting and underfitting. Eventually, we find that 7 epochs leads to the best model with the lowest evaluation loss, achieving an evaluation loss of 0.03263428807258606.

Model Testing: We test our fine-tuned codet5-small model on 5000 testing samples, using the flattened, masked code as input. The model, given the masked input, then predicts what it believes the masked if statement to be, comparing it to the actual if statement. We evaluate the model based on 3 main metrics: exact match, Code_Bleu score and Bleu-4 score. Afterwards, we output our results into a csv file. This csv file contains 6 columns, with each column containing 5,000 rows for all the testing inputs. The columns are: The input function with the masked if condition, whether it was an exact match, the expected if condition, the predicted if condition, the CodeBleu score from 0-100, and the Bleu-4 score from 0-100. In general, the Bleu-4 score was always higher than the CodeBLEU score, with almost no prediction obtaining a perfect 100 for the CodeBleu score. This indicates that, while the model is good at predicting if statements with good n-gram precision, it is not great at capturing the overall structure, syntax, and/or semantics of the code.

While we have the .py files for the script, all our outputs were produced using google colab. The google colab code will be in the repository.

AI disclaimer: We utilized ChatGPT for this assignment, mainly in combining the CodeBleu code from piazza with the evaluation function we had created (test_code), as well as translating some aspects of our google colab code into .py files.