GenAI for Software Development: Assignment 2

David Berger
dhberger@wm.edu

1 Introduction

This assignment aims to fine-tune and train a transformer model to generate appropriate if statements when given a masked method as input. The data consists of Python methods, which are flattened and masked using Pygments. The data is formatted into a dataset, run through the pre-trained RoBERTA tokenizer, and trained using the Salesforce CodeT5-Small transformer model provided by Huggingface. The model's performance is then evaluated and tested with the help of wandb.ai's API. The source code of the assignment can be found at https://github.com/dhbergerwm/CSCI420HW2.

2 Implementation

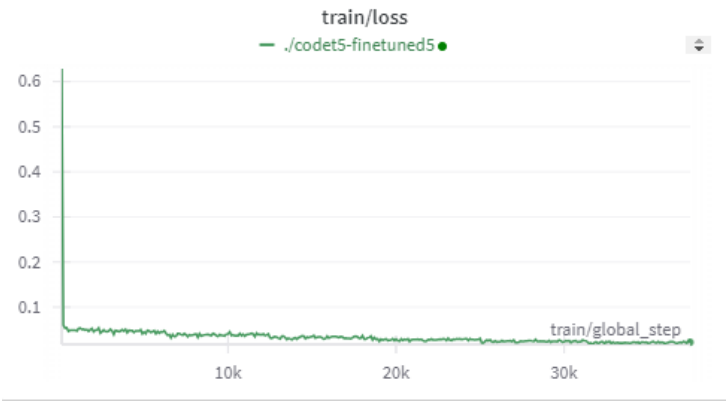
Dataset Preparation: For this assignment, a dataset of 60000 Python methods was provided. Each method contained an if statement to be predicted by the transformer model. The dataset was split into 50000 training methods, 5000 validation methods, and 5000 test methods. Due to hardware limitations, only 25000 methods were (randomly) sampled from the training set to speed up the training process.

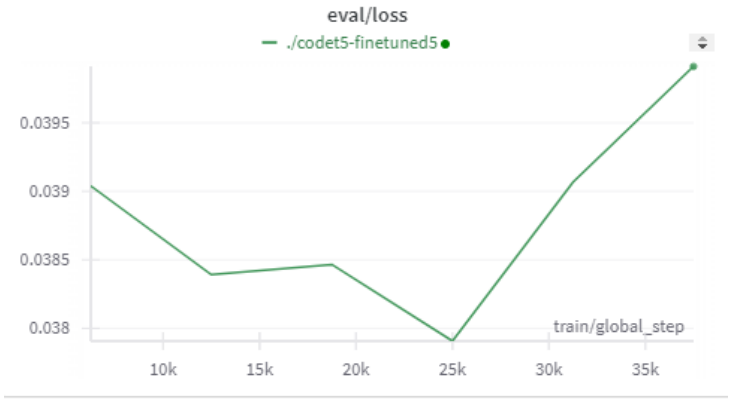
Pre-Processing and Fine-Tuning: The code was pre-processed using the Pygment module's Python lexer. Each method was flattened and its target if/elif statement was masked as an <IF-STMT> token. Tab tokens (<TAB>) were added to aid in the model's understanding of block structure. With Pandas dataframes and the dataset module, the masked methods and their respective labels were put into a dataset form. The model selected for this task was a Salesforce CodeT5-Small transformer provided by Huggingface.co. The original dataset was run through a pre-trained RoBERTa tokenizer and mapped to a new tokenized dataset to fine-tune the model.

Model Training

The CodeT5-Small model was trained on the fine-tuning dataset with a learning rate of 0.00005 in batches of four. The model was set to run for seven epochs, although it only completed six due to early stopping with a patience window of two epochs. At epoch four, the best epoch before early stopping occurred, the training loss was recorded to be 0.0303 and the validation loss was roughly 0.0379, as seen in the figure to the right. Additionally, using wandb.ai's API, the model's overall performance was displayed in live-updating charts as seen below:

Epoch	Training Loss	Validation Loss
1	0.047500	0.039040
2	0.032200	0.038391
3	0.032400	0.038466
4	0.030300	0.037907
5	0.026000	0.039066
6	0.022100	0.039385





3 Evaluation

Model Testing: To evaluate the model's performance on the test set, three metrics were chosen – exact matching, CodeBLEU, and BLEU-4. Exact matches were calculated by comparing the predicted conditional statement with its corresponding label. The percentage of exactly matching outputs was 26.62%. Using Microsoft's CodeXGlue evaluator, the model's CodeBLEU score (with all four weights set to 0.25) was 90.46. Using the SacreBLEU module to calculate the BLUE-4 score, the average turned out slightly higher than the CodeBLEU score at 97.06. The CodeBLEU and BLEU-4 scores were computed by replacing the label method's masked statement with the predicted statement and comparing it to the original, unmasked method.

Output: The primary output for this assignment is found in **testset-results.csv**. The file has nine columns related to the model's performance on the test set. The first column contains the flattened and masked method from which the model intends to produce the conditional statement. The second and third columns contain the model's prediction and the ground truth statement, respectively. The remaining columns are evaluation metrics – any exact matches are marked as TRUE in the fourth column and FALSE otherwise. The two following columns show the CodeBLEU and BLEU-4 values for individual methods. The overall percentage of these exact matches, the model's CodeBLEU score, and the average BLEU-4 score are also included.