WILLIAM & MARY

CHARTERED 1693

GenAI for SD

Prof. Antonio Mastropaolo
Assistant: Aya Garryyeva

Fine-Tuning CodeT5 for Predicting `if` Statements

Your objective is to fine-tune the CodeT5 Transformer model from Hugging Face to automatically recommend suitable `if` statements in Python functions.

What is CodeT5? CodeT5 is a pre-trained encoder-decoder Transformer model designed for code understanding and generation. It has been trained on a large corpus of code across multiple programming languages and supports a range of downstream tasks such as code completion, summarization, translation, and generation. In this assignment, you will fine-tune CodeT5 specifically for predicting missing `if` conditions in Python functions. You will be using the **small version** of CodeT5, known as `codet5-small`, which contains approximately **60 million parameters**. This lighter variant offers a good trade-off between performance and computational efficiency.

Fine-tuning: You will fine-tune a pre-trained CodeT5 model specifically for the task of predicting `if` conditions. The model will take as input a function containing a special token masking a single `if` condition and will attempt to predict it. You are responsible for preparing the dataset by masking the `if` conditions yourself. After masking, you must tokenize the input using a pre-trained tokenizer before feeding it into the model.

Dataset: A pre-processed dataset will be provided, but you must modify it to introduce masked `if` conditions and flatten it. The dataset consists of 50,000 training samples, with validation and test sets containing 5,000 samples each. The dataset includes the following columns: `cleaned_method`, `target_block`, `tokens_in_method`. The `cleaned_method` column needs to be masked and flattened prior to fine-tuning. The `target_block` column contains the `if` condition that needs to be masked and predicted using the pre-trained model.

Steps to Complete:

1. Load the pre-trained CodeT5 model from Hugging Face `https://huggingface.co/Salesforce/codet5-small`.

   Note: Ensure you change the one in the script provided, with the model indicated here

2. Modify the dataset by masking `if` conditions appropriately and flattening it.

   Note: In practice, it is common to preserve Python's indentation by explicitly marking it with special tokens such as `<TAB>`. While this improves model awareness of block structure, we will **not enforce this requirement for this assignment**. However, you're welcome to experiment with it if you wish to explore how it affects model performance.

   Example: Given the function:

```
def check_positive(num):
    if num > 0:
        return "Positive"
    else:
        return "Non-Positive"
```

After masking and flattening, the input should be:

```
def check_positive(num): <mask>: return "Positive" else: return "Non-Positive"
```

The model should predict `if num > 0` as the missing condition.

3. Tokenize the dataset using a pre-trained tokenizer.
4. Fine-tune the model, avoiding over-fitting and under-fitting by using early stopping on the loss function of the validation set.
5. Evaluate the model's performance using multiple metrics.

Model Evaluation: You have to assess the ability of your model in producing meaningful if conditions using different evaluation metrics, including *exact match, BLEU & CodeBLEU.*

NOTE: Please refer to `https://huggingface.co/spaces/evaluate-metric/sacrebleu` for computing the BLEU score. The SacreBLEU, as per the code implementation, returns the BLEU-4 value.

This means that it evaluates the overlap between the generated output and the reference using n-grams up to 4 tokens in length (i.e., sequences of up to 4 consecutive tokens). However, a key caveat is that when the text is shorter than 4 tokens, BLEU-4 cannot be directly computed. In such cases, smoothing techniques are applied to avoid returning a score of zero, typically resulting in a very small BLEU value and allowing the evaluation process to continue without interruption.

Special Instructions for 520 Section Students

- You must collect your own fine-tuning dataset containing 50,000 Python functions.
- You are not allowed to work in groups and must complete the assignment individually.

Submission Instructions

Deadline: Tuesday, April 10th, 2025 @12:00 PM. 5 points will be deducted for each late day until after 5 days after the original deadline. After that you will get an automatic 0.

Each student must submit a Github link on Blackboard. Github repository should contain:

1. All code files and data files used as part of the assignment.
2. A ReadMe file with instructions on how to replicate your work.
3. A 2-page document describing the fine-tuning process, dataset preparation, and evaluation results, including a comparison of different evaluation metrics. Include evaluation scores from BLEU, CodeBLEU, exact matches, F1 score, and any other chosen metrics.
4. A `csv` file for the test set results with the following columns:
   - Input function with masked `if` condition;
   - Whether the prediction is correct (*i.e.,* ask is an exact match?) (`true/false`);
   - Expected `if` condition;
   - Predicted `if` condition;
   - CodeBLEU prediction score (`0-100`);
   - BLEU-4 prediction score (`0-100`);

   Name the `csv` file as `testset-results.csv`.