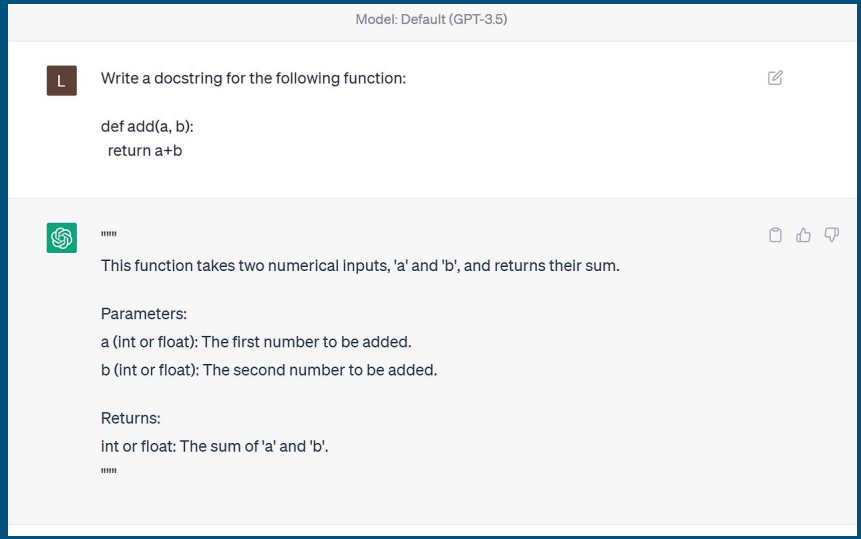
Code summarization using pre-trained LLMs

Jack Gindi and Lorenzo Meninato

Background

- Many programmers are using LLM systems such as ChatGPT and GitHub CoPilot to write, document, and test code
- The systems are not pre-trained on programming tasks, they still produce high quality output
- **Goal:** to try to reproduce a small-scale code summarization model with fine-tuning



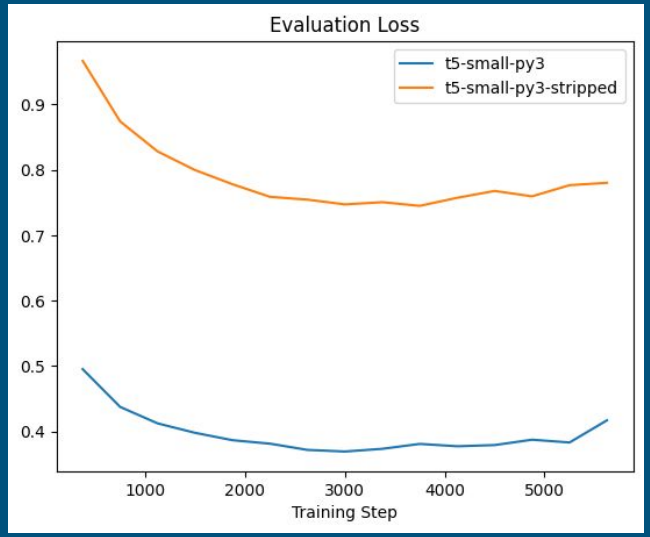
Overview

- **Data:** CodeSearchNet dataset
- **Fine-tuning training set size:** 30k training and 5k validation (function, docstring) pairs
  - Goal: “translate” code into docstrings
  - 6 languages available
- **Epochs:** 15
- **Maximum sequence length:** 256
- **(Effective) batch size:** 80
  - 10 gradient accumulation steps with batch size 8

Experiments

1. Tried a few architectures: BERT-based, T5, FLAN-T5
2. Fine-tuned T5 on python only with different prompts
    a. “Write documentation for the following code:”, “Summarize this code in English:”
    b. Stripped docstrings to mitigate overfitting
3. Tried adding other languages to see if that would improve performance
    a. Trained 3 models: python only, python/java, python/java/javascript/go

Some results



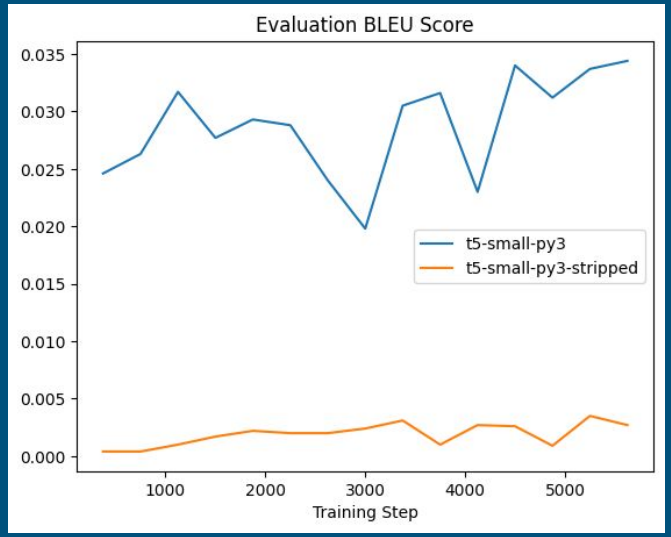
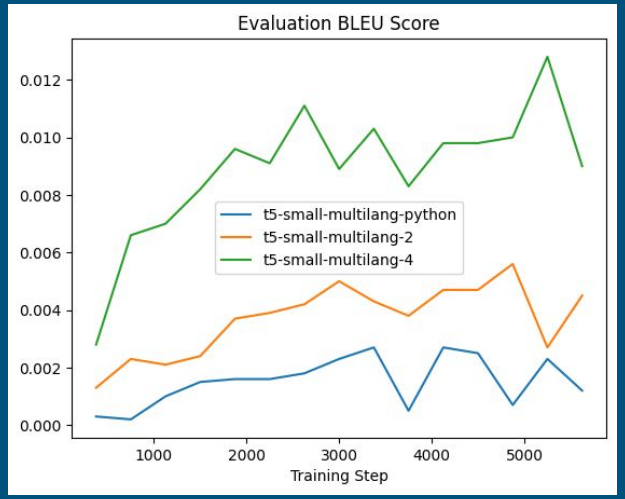
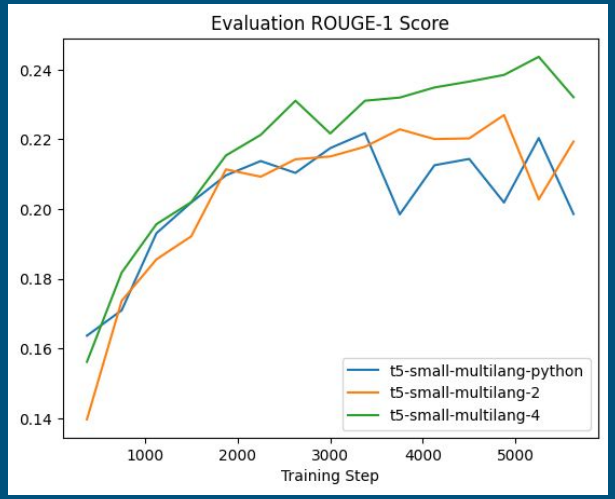


Figure 1. Difference when stripping python docstrings

Some results





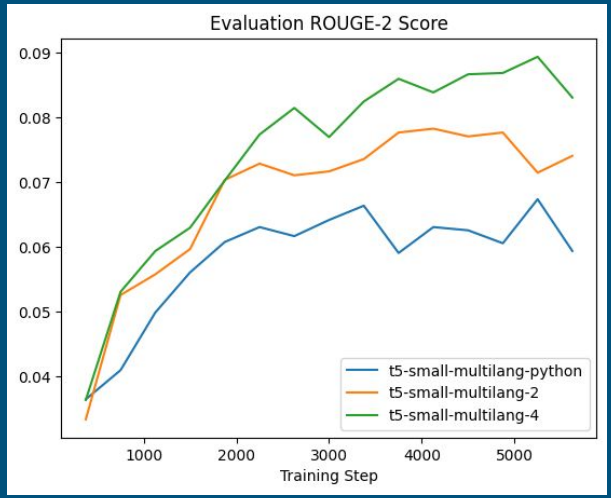


Figure 2. Adding other languages seems to actually produce better results!

Analysis: What were our main issues?

- Differences between code and natural language structures
- Data quality
- Data size/training duration/model size
- Building NLP models from scratch is hard!

References/Models

- https://huggingface.co/t5-small
- https://huggingface.co/google/flan-t5-small
- https://huggingface.co/prajjwal1/bert-small
- https://huggingface.co/datasets/code_search_net
- https://chat.openai.com/