Maria Dhakal
Cover Letter for Discussion Leader
CSE-60876 (Research Methods)

Paper: Ahmed, T., & Devanbu, P. (2022, October). Few-shot training LLMs for project-specific code-summarization. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering* (pp. 1-5).

General Topic: Few-shot training of GPT(Generative Pre-Trained Transformer) Codex model for code comment synthesis.

Specific Behavior or Activity Studied: The authors investigated the effectiveness of few-shot training for code summarization.

Specific Research Questions:

- ➢ Does the few-shot learning capacity of large language models extend to the task of code summarization?
- ➢ Can this few-shot learning capacity be extended to same-project learning on this same task? How does the performance of LLMs in the above two settings compare with that of state-of-the-art models?

Challenges:

- ➢ Since LLMs are pretrained on enormous datasets, the CodeX model could have seen test data already seen during it's very large-scale pre-training.

Paradigm: The authors used a simple few-shot based methodology using the Codex model. The steps include:

- ➢ Preparing the Prompt with few-shot training examples and query function
- ➢ Sending prompt to Codex Model
- ➢ Receive output and fix the output length
- ➢ Prepare Target Comment

Problem: Project-specific code summarization using deep learning approaches requires a large (O(10^4) or O(10^5)) sample size to learn local features, which is hard to find on a project basis. Even if the project exists, retraining a big model for each new project can be cumbersome, but also necessary.

Importance: The few-shot learning capacity of very-large language models can make do with just a handful of training examples; furthermore retraining is not really cumbersome, as one can just change the prompt.

Claims: The authors found that few-shot training can significantly outform the fine-tuned model trained with thousands of samples with just ten samples. This approach performs better on same-project setting compared to cross-project setting. The authors claim this approach to be very promising and feasible.

State of Knowledge: Large Language Models like GPT-3, can perform very good tasks of prediction, generating code from docstrings and vice-versa, fine tuning, etc. The authors focus on the few-shot learning aspects of LLMs and draw attention towards using this interesting salience for software engineering.

Evidence: The authors used a four step few-shot learning approach in the Codex Model, and evaluated the performance of the approach for same-project few shot and cross-project shot. With 10 samples, Codex outperforms all fine tuned foundation models CodeT5, CodeBERT, GraphCodeBERT, Polyglot CodeBERT, and PolyGlotGraphCodeBERT in all six programming languages, even though the fine-tuned models are trained with thousands of data.
Same-project few-shot training improves the Codex model's performance for all 8 projects.

Story Structure: The authors talked about the versatility of LLMs and introduced the concept of few-shot learning, where LLMs can accomplish tasks with minimal training examples by conditioning their behavior on given prompts. Then, they discussed the challenges with project-specific code summarization in software engineering and offered a learning approach for LLM to overcome the challenge by using examples and prompts to train LLM for comment synthesis.