CodeMMLU Challenge Report

Huy Nguyen Quang

1 Introduction

Recent improvements in Code Large Language Models (CodeLLMs) have shown great ability in handling various software engineering (SE) [1]. CodeMMLU [1] is a test designed to measure how well large language models (LLMs) understand coding and software concepts. It uses multiple-choice questions to cover different areas, like writing code, finding bugs, and software engineering principles. In this challenge, nearly three thousand questions with answers are provided, and the test set contains around 1,200 questions. In this report, I describe my approach to this problem and suggest some future work.

2 Methodology

2.1 Base model selection

I evaluated five models using 5% of the training set: OpenAI's GPT-4o, Meta-Llama-3-70B Instruct, Meta-Llama-3.1-70B Instruct, Meta-Llama-3.3-70B Instruct, Qwen3-32B because GPT-4o and the Meta-Llama family achieve the best results on CodeMMLU Leaderboard[1], while Qwen3 shows excellent performance in a wide range of code-related tasks, as reported in [2]. I experimented with these in two settings: Zero-shot and Chain-of-Thought (CoT) [3]. Table 1 shows that GPT-4o demonstrates the best performance in both settings. Meta-Llama-3.3 tends to perform the best among the Meta-Llama family. Qwen2.5-Coder produces results close to GPT-4o in the Zero-shot setting but performs worse in the CoT setting compared to Meta-Llama-3.1 and Meta-Llama-3.3. Therefore, I decided to further experiment with Meta-Llama-3.3-70B Instruct and Qwen 2.5-Coder as it gives performance close to GPT-4o and is an open-source model.

2.2 Construct a dataset for fine-tuning

In the initial dataset, only multiple-choice questions and answers are provided. Motivated by [4], I created a data set for fine-tuning by leveraging the reasoning skills of GPT-4o. In detail, for each question in the training set, I use GPT-4o to generate a response based on a CoT prompt. If the response from ChatGPT

[1] `https://fsoft-ai4code.github.io/leaderboards/codemmlu/`

Table 1: The evaluation results (accuracy %) of different language models

Model	Zero-shot	CoT
OpenAI's GPT-4o	**71.72**	**77.78**
Meta-Llama-3-70B Instruct	63.13	65.15
Meta-Llama-3.1-70B Instruct	69.19	73.23
Meta-Llama-3.3-70B Instruct	70.71	74.75
Qwen3-32B	71.21	71.21

Table 2: Private leaderboard accuracy (%) of two models before and after fine-tuning

Model	Unfine-tuned	Fine-tuned
Meta-Llama-3.3-70B Instruct	71	73
Qwen3-32B	**74**	**78**

produces the correct answer, I add the question-response pair to the new dataset. If it produces the wrong answer, I provide the ground truth along with the question and prompt GPT-4o to generate the correct answer. If it still produces the wrong answer, those questions are rejected.

2.3 Fine-tuning base models and make submission

Meta-Llama-3.3-70B Instruct is fine-tuned using LoRA [5] (a parameter-efficient finetuning technique) on a new dataset built as described in the previous section. Then, I use Together AI[2] to create a dedicated endpoint for deploying my fine-tuned model. After deployment, I use it to generate answers based on questions in the test set and submit the results for evaliation. Next, I fine-tune a Qwen3-32B model on Google Colab's A100 GPU using a similar fine-tuning procedure with the Unsloth library [6]. The table 2 summarizes the results from both experiments:

Fine-tuning improved the performance of both models on the private leaderboard. Meta-LLaMA-3.3-70B Instruct increased from 71% to 73%, while Qwen3-32B improved more significantly, from 74% to 78%. Qwen3-32B performed better than LLaMA in both the unfine-tuned and fine-tuned versions, showing that it responds better to fine-tuning and works well for this task.

3 Future works

Although after the fine-tuning process, the performance of Meta-Llama-3.3-70B Instruct is not very strong, as it only achieves around 71% accuracy. While Qwen3-32B achieved better results overall, further alignment could still boost its

[2] `https://www.together.ai/`

performance. One suggestion for improvement is utilizing test-time scaling techniques, as recent advancements in these techniques have significantly enhanced reasoning skills in LLMs, especially for smaller models. Another potential improvement is enhancing the fine-tuning dataset, as in the current approach, the responses generated by GPT-4o are not carefully evaluated. Furthermore, Group Relative Policy Optimization (GRPO) [7] can be used to improve reasoning capabilities in these models.

References

[1] Dung Nguyen Manh, Thang Phan Chau, Nam Le Hai, Thong T Doan, Nam V Nguyen, Quang Pham, and Nghi DQ Bui. Codemmlu: A multi-task benchmark for assessing code understanding capabilities of codellms. *arXiv preprint arXiv:2410.01999*, 2024.

[2] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[3] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[4] Zhaowei Liu, Xin Guo, Fangqi Lou, Lingfeng Zeng, Jinyi Niu, Zixuan Wang, Jiajie Xu, Weige Cai, Ziwei Yang, Xueqian Zhao, et al. Fin-r1: A large language model for financial reasoning through reinforcement learning. *arXiv preprint arXiv:2503.16252*, 2025.

[5] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[6] Michael Han Daniel Han and Unsloth team. Unsloth, 2023.

[7] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.