Abstract

Although Large Language Models (LLMs) have been widely adopted for code generation, the generated code can be semantically incorrect, requiring additional efforts for validation and refinement. Test-driven prompt (TDP) engineering systematically incorporates executable specifications through test cases within the prompt structure provided to the LLMs. In this paper, we conduct a systematic and comprehensive empirical evaluation of TDP across eight diverse models (GPT-4o, GPT-4o-mini, Claude 3.5 Sonnet/Haiku, and Qwen 32B/14B/7B/3B Coder) using three benchmark datasets (HumanEval, MBPP, and Code Contests). Our methodology compares TDP and normal prompting with remediation loops through statistical analysis. Results demonstrate the effectiveness of TDP in code generation, where TDP approaches achieved superior performance in 100% of 16 model-dataset combinations, with 7.74% average improvement (95% CI: [6.30, 9.18], p ¡ 0.0001, Cohen's d = 1.83). Our analysis results suggest that smaller models that generate code based on TDP outperform larger variants that use normal prompts. Furthermore, our results show that TDP provides computational efficiency, particularly for coding-specialized models by achieving superior first-attempt accuracy. Finally, experiments to uncover explainability of TDP improvements shows that the largest gains were for problems that has implicit specifications. These findings demonstrate that TDP is a robust, model-agnostic strategy that can be incorporated in prompt engineering to enhance the quality of the generated code.