

2026 졸업프로젝트 주제

다중 LLM 기반 Python Unit Test 자동 생성 및 품질 평가 시스템 개발

프로젝트의 목적

Python 코드에 대해 여러 LLM을 활용하여 unit test를 자동 생성하고, 생성된 테스트를 pytest로 실행한 뒤 coverage 및 mutation score 등의 기준을 이용하여 테스트 품질을 평가할 수 있는 통합 시스템을 개발한다. 또한 서로 다른 LLM과 prompt engineering 기법이 생성한 unit test의 품질을 비교 분석할 수 있도록 한다.

Research Questions

RQ1: Prompt engineering 기법(CoT, ToT, Few-shot, Zero-shot)이 LLM 기반 unit test 생성 품질에 어떤 영향을 미치는가?

RQ2: LLM이 생성한 unit test는 manual unit test와 비교했을 때 충분한 테스트 품질을 제공하는가?

RQ3: Test execution 결과를 feedback으로 제공할 경우 LLM이 생성하는 unit test의 품질이 개선되는가?

시스템 주요 기능

1) Multi-LLM unit test generation

- 여러 LLM API를 동시에 호출하여 Python 코드에 대한 unit test를 자동 생성
- 다양한 prompt engineering 기법(CoT, ToT, Few-shot, Zero-shot)을 적용한 테스트 생성 지원

2) Test execution

- 생성된 unit test를 pytest를 이용하여 자동 실행
- 테스트 실행 결과 (pass/fail, coverage, failure log) 수집

3) Test quality evaluation

- 생성된 unit test를 평가하기 위한 criteria는 사전에 설정되어야 함
- Line coverage, branch coverage, mutation score, fault detection, assertion quality가 기반이 되어야 함

4) Iterative test improvement

- 테스트 실행 결과 (pass/fail, coverage, failure log)와 예상 결과 및 실제 테스트 결과를 LLM에 feedback으로 제공
- LLM이 unit test를 반복적으로 개선하도록 지원
- Iteration을 3회 반복

5) Comparison and reporting

- 서로 다른 LLM을 통해 생성된 unit test 간의 비교
- Manual unit test (baseline)와의 비교
- Unit test 평가의 criteria를 가지고 한 가지 LLM이 생성된 전체 unit test를 평가하여 그 결과를 화면에 출력할 수 있어야 함
- Unit test 생성을 위한 LLM과 평가를 위한 LLM을 별도로 사용하여야 함

Process

- 1) Github에서 repository 내에 unit test가 포함되지 않았으며 기능이 명확한 5~10개의 python project를 수집
- 2) 각 project에 대하여, manual unit test를 생성하여 baseline 구축
- 3) Baseline test를 실행하여 테스트 결과 확보

- 4) 다양한 prompt engineering 기법을 사용하여 unit test 생성을 위한 prompt 설계
- 5) 시스템 내의 모든 LLM에 하나의 입력창을 통해 동일한 prompt를 입력하여 unit test 생성
- 6) 각 LLM이 생성한 unit test를 pytest를 통해 실행하고, 테스트 결과 (pass/fail, coverage, failure log)를 수집
- 7) 테스트 결과에 따라 unit test를 개선하도록 LLM에 feedback으로 제공
- 8) Unit test 생성 → 실행 → 평가 → 개선을 3번 반복
- 9) 최종 산출물을 실행하여 도구 상에서 unit test 결과 report
- 10) Report된 결과를 활용하여 각 도구 및 prompting 기법을 criteria에 따라 평가 및 비교분석

최종 산출물

- 1) Multi-LLM 기반 Python unit test 자동 생성 및 평가 시스템
- 2) LLM별 unit test 생성 성능 비교 결과
- 3) prompt engineering 기법에 따른 테스트 품질 분석 결과
- 4) LLM 기반 unit test 생성의 한계 및 개선 방향 분석

기존 연구들

- [1] An Empirical Evaluation of Using Large Language Models for Automated Unit Test Generation
 - JavaScript를 위한 LLM 기반의 test generator인 TestPilot을 활용하여, 25개의 npm packages 내에 있는 API function에 대한 test를 진행함
 - Statement coverage와 pass rate 평가
- [2] Automated Unit Test Improvement using Large Language Models at Meta
 - Kotlin test class를 확장하여 Android code에 대한 UT를 자동 생성함 (자동 test

class improver인 TestGen-LLM 개발)

- Meta의 두 가지 LLM을 사용
- Test equivalence를 자동으로 평가할 수 없으므로 coverage를 proxy로 사용하여 coverage가 증가하는 경우만 수용

[3] Automatic Unit Test Generation for Programming Assignments Using Large Language Models

- GPT-4를 사용하여, 여러 source로부터 정보를 통합하여 prompt를 주고 LLM이 어느 정도 수준의 semantic complexity를 가지는 syntactically correct test를 생성하도록 함 (AugGPT)
- AugGPT를 사용해 생성한 UT와 manual하게 생성한 UT를 모두 활용하여 coverage와 mutation score를 높이고자 함

[4] Evaluating the Effectiveness and Cost-Efficiency of Large Language Models in Automated Unit Test Generation

- Gemini와 ChatGPT의 서로 다른 두 가지 버전씩을 사용하여 총 4가지의 LLM 사용
- HumanEval dataset에서 164개의 problem을 사용하여 prompt template을 활용한 UT 생성 진행 (role 부여, goal 정의, task 분해, few-shot prompting)
- Coverage, time, price 평가

[5] On the Evaluation of Large Language Models in Unit Test Generation

- Defects4J 2.0에서 17개의 java project 선정
- CodeLlama와 DeepSeek-Coder 기반의 LLM 5종류 활용
- Syntactically correct & effective & maintainable test를 위해서는 prompt 설계와 RAG가 중요
- Coverage 평가

[6] Using Large Language Models to Generate JUnit Tests: An Empirical Study

- Codex, GPT-3.5-Turbo, StarCoder를 활용
- HumanEval dataset과 SF110에서 선정한 47개의 open-source project를 사용하여 Junit5 test 생성
- Compilation rate, correctness rate, coverage 평가

기존 연구와의 차이점 비교분석

- LLM 기반으로 UT를 생성하는 기존 연구들 중 여러 LLM을 한 도구에 통합하는 시도가 없었음
- Python project를 대상으로 LLM을 활용한 연구 발견하지 못함
- 단순 prompt engineering 연구가 대부분
- 체계적인 unit test 평가 기준을 적용한 연구가 많지 않음 (대체로 coverage 위주로 평가)
- LLM에게 정해진 iteration만큼 수정을 요구하여 더 나은 결과를 얻고자 한 연구의 수가 많지 않음

테스트에 사용할 수 있는 open source python project github link

<https://github.com/TheAlgorithms/Python>

<https://github.com/Python-World/python-mini-projects>

<https://github.com/Efeckc17/simple-example-projects-in-Python>

<https://github.com/Mrinank-Bhowmick/python-beginner-projects>