LLM이 리눅스 커널에 기여하기 위해 필요한 것

이동재

2025년 2월 20일

Abstract

언어 모델은 코드 작성에 2가지 방식으로 도움을 준다. 첫째로는 ChatGPT처럼 코드를 자연어 지시 사항에 맞춰 처음부터 끝까지 스스로 작성하는 것이다. 둘째로는 Copilot처럼 기존 코드 저장소 위에서 코드를 작성하는 것이다. 두 가지 접근 방법은 결국 하나로 합쳐질 것이며 그 과정에서 기존 코드를 이해하는 능력이 중요해질 것이라 생각한다. 이번 글에서는 두 방식이 합쳐지기 위해서는 어떠한 문제가 해결되어야 하는지 살펴볼 것이다.

프로그래밍은 AI의 가장 대표적인 활용 방법 중 하나다. 여기에는 크게 2가지 접근법이 있다. 첫째는 ChatGPT처럼 코드를 자연어 지시 사항에 맞춰 처음부터 끝까지 작성하는 것이고, 둘째는 Copliot처럼 기존 코드 저장소 상에서 짧은 코드를 작성하는 것이다.

이 두 가지 방법은 결국 합쳐지게 될 것이다. ChatGPT의 경우 자연어 지시사항만이 주어지기 때문에 대규모 코드 저장소의 맥락을 전혀 알지 못한다. Copliot은 짧은 길이의 코드만을 생성하기 때문에 사실상 현재 작성 중인 코드 혹은 최근에 편집했던 파일 정도만 문맥으로 사용해도 충분히 생성할 수 있다.

문제는 대규모 저장소에서 파일 단위의 코드 생성과 편집이 어렵다는 것이다. 리눅스 커널과 같은 대규모 저장소에서 코드를 작성하려면 데이터 흐름, 기존에 정의된 함수와 API를 전부 파악하고 있어야 한다. 그러나, 저장소의 모든 파일을 프롬프트로 전달하는 것은 사실상 불가능하며, 매우 비효율적이다. 따라서, 저장소에서 생성에 필요한 정보만을 찾아 전달하는 것이 중요하다.

흔히 알려진 해결 방법으로는 검색 증강 생성 (RAG)을 활용하는 방법이 있다. 그러나, 이 방법은 관련 파일을 찾는 과정이 선행되기 때문에 생성 시점에서는 이를 바로잡을 수 없다. 또한, 단순히 API 존재를 아는 것뿐만 아니라, 자료구조의 형태, 입출력 타입, 주석으로 표시된 제약 조건 등 모든 측면을 파악하여 코드를 작성해야 올바른 코드를 생성할 수 있다.

기존 정적 분석기들은 인간과 전통적 도구들을 위해 유용한 정보를 생성했지만, 이제는 LLM을 위한 정적분석기가 등장해야 할 시점이다. 필요한 모든 정보를 안전 (Sound)하게 수집하면, LLM은 인간과 달리 막대한 양의 정보를 병렬적으로 소화하여 올바른 코드를 생성할 수 있을 것이다.