논문 2026-1-1 http://dx.doi.org/10.29056/jsf.2026.03.01

소프트웨어 완성도 감정에서 대규모 언어모델 활용 기법 연구

김유경*†

A Study on the Use of Large Language Models for Software Completeness Appraisal

Yukyong Kim*†

요 약

소프트웨어 완성도 감정은 계약 이행 검증, 분쟁 해결, 공공 조달 평가 등 다양한 법적·산업적 환경에서 요구사항 충족 여부를 실행 결과를 통해 검증하는 동적 분석 절차이다. 그러나 기존 감정은 테스트 설계, 로그 분석, 증거 정리 등을 전문가의 수작업에 의존하여 시간과 비용이 많이 소요되며, 주관적 편차와 재현성 한계를 초래한다. 대규모 언어모델(LLM)은 요구사항 이해와 코드 분석을 지원할 수 있으나, 환각과 비결정성 문제로 인해 최종 판단 도구로 사용하기에는 신뢰성 제약이 존재한다. 본 논문은 소프트웨어 완성도 감정이라는 고신뢰성·증거 중심 절차에 LLM을 통합하기 위한 구조적 설계 원칙을 제시한다. LLM을 판단 주체가 아닌 보조 분석 도구로 한정하는 환각 통제 기반 완성도 감정 방법을 정의한다. LLM은 테스트 시나리오 초안 생성, 실행 로그 요약, 요구사항-증거 간 의미 매칭, 보고서 초안 작성 등을 지원하며, 최종 판정은 실행 증거와 규칙에 기반을 두어 전문가가 수행한다. 본 연구는 LLM을 통제된 방식으로 감정 절차에 통합함으로써 효율성과 신뢰성을 동시에 향상시키는 구조적 설계 방안을 제시한다.

Abstract

Software completeness appraisal plays a critical role in contractual compliance verification, dispute resolution, and public procurement evaluation. It is a dynamic, execution-based assessment process that determines whether contracted requirements are fulfilled through observable system behavior. Large Language Models (LLMs) offer potential support for requirement understanding and code analysis, yet their susceptibility to hallucination and non-determinism limits their suitability as final decision-making tools in high-reliability contexts. This paper proposes a hallucination-controlled, LLM-assisted appraisal framework in which LLMs are restricted to auxiliary analytical roles. Specifically, LLMs support test scenario drafting, requirement-evidence semantic matching, and report drafting, while final completeness judgments are made by experts based on execution evidence and predefined rules. The proposed framework demonstrates how LLMs can be integrated into software completeness appraisal in a controlled manner, improving efficiency while preserving reliability.

한글키워드 : 소프트웨어 완성도 감정, 대규모 언어모델, 동적 분석, 환각 통제

keywords : SW completeness appraisal, Large Language Models, Hallucination Control

* 숙명여자대학교 첨단공학부
† 교신저자: 김유경(email: ykim.be@sookmyung.ac.kr)

접수일자: 2026.02.23. 심사완료: 2026.03.14.
게재확정: 2026.03.20.

1. 서 론

소프트웨어 시스템은 공공 행정, 금융, 의료, 국방 등 사회 전반의 핵심 인프라로 활용되고 있으며, 대규모 정보화 사업과 외주 개발의 확대에 따라 계약된 요구사항의 실제 구현 여부를 객관적으로 검증하는 절차의 중요성이 지속적으로 증가하고 있다. 특히 계약 이행 판단, 분쟁 해결, 유지보수 인수인계, 공공 조달 검수 등 법적·행정적 맥락에서는 시스템이 명세된 기능을 충족함을 입증할 수 있는 기술적 근거 확보가 필수적이다.

소프트웨어 완성도 감정은 문서 검토에 국한되지 않고, 실제 실행 결과를 통해 기능 구현 여부를 확인하는 동적 실행 기반 검증 절차이다[1]. 감정인은 테스트를 설계하고 시스템을 실행하여 로그, 화면 출력, 데이터베이스 상태 변화 등 실행 증거를 수집한 후 이를 요구사항과 대조하여 구현 여부를 판단한다. 그러나 이러한 과정은 요구사항 해석, 테스트 작성, 실행 결과 분석, 증거 정리, 보고서 작성 등 대부분이 전문가의 수작업에 의존하며, 시간·비용 부담과 주관적 편차, 확장성 한계를 초래한다[2].

최근 대규모 언어모델(LLM)은 코드 이해, 문서 요약, 의미 분석 등 다양한 소프트웨어 공학 작업에서 높은 성능을 보이고 있으며, 요구사항 구조화, 실행 로그 요약, 테스트 시나리오 초안 생성, 의미 기반 매칭과 같은 업무를 효과적으로 지원할 수 있다[3]. 이는 반복적이고 노동 집약적인 감정 업무의 생산성을 향상시킬 가능성을 제공한다.

그러나 LLM은 환각(hallucination), 비결정성, 내부 추론의 불투명성 등 신뢰성 한계를 내포하고 있어 법적 효력을 갖는 감정 절차에서 최종 판단 도구로 직접 활용되기 어렵다. 따라서 LLM은 판단 주체가 아닌 보조 분석 도구로 제한적으로 활용되어야 하며, 최종 판단은 실행 증거와 명시적 기준에 기반하여 이루어져야 한다.

이에 본 연구는 동적 실행 기반 감정 구조를 유지하면서 LLM을 보조 분석 계층으로 통합하는 환각 통제 기반 방법론을 제안한다. LLM은 테스트 시나리오 초안 생성, 실행 로그 요약 및 구조화, 요구사항 - 증거 간 의미 매칭 지원, 보고서 작성 보조 등을 수행하고, 최종 판정은 규칙 기반 평가와 전문가 검증을 통해 결정된다. 또한 프롬프트 고정, 결정론적 설정, 근거 인용 강제, 로그 기록을 통해 재현성과 감사 가능성을 확보한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 고찰하고, 3장에서는 LLM을 활용한 감정 사례를 살펴보고, 소프트웨어 완성도 감정에서 LLM 활용 문제를 정의해 본다. 4장에서는 제안하는 LLM 기반의 감정 방법론을 상세히 설명한다. 5장에서는 결론과 향후 연구 방향을 제시한다.

2. 관련 연구

2.1 소프트웨어 완성도 감정

한국저작권위원회의 컴퓨터 프로그램 감정 매뉴얼에서는 “프로그램의 완성도는 프로그램이 질적으로 완성된 정도, 혹은 프로그램이 사용자의 요구를 만족한 정도”라고 정의하고 있다[4]. 소프트웨어 개발의 완성도 또는 진행률을 정량적으로 평가하기 위한 연구는 소프트웨어 공학, 프로젝트 관리, 비용 추정, 그리고 요구사항 공학 등 다양한 분야에서 지속적으로 수행되어 왔다. 대부분의 연구에서 소프트웨어 완성도는 기능 기반으로 산출되어야 함을 제시하고 있다[2]. 기능 기반 접근은 소프트웨어의 실질적 가치를 사용자 관점의 기능 단위로 정의하고, 구현된 기능의 양을 정량화함으로써 개발 규모와 진행률을 추정하는 방법론이다[5]. 이러한 접근은 개발 활동이나 문

서의 존재 여부보다는 실제 제공되는 기능적 서비스에 초점을 둔다는 점에서 비교적 결과 중심적인 평가 관점을 제공한다. 소프트웨어 계약 분쟁에서는 법원이나 중재기관이 전문가 감정을 통해 완성도와 기성고를 판단한다. 이러한 평가는 코드 검토, 테스트 실행, 인터뷰, 문서 분석 등 다양한 정성적 방법을 포함한다. 그럼에도 불구하고 실제 감정 실무는 평가자의 경험과 전문성에 크게 의존하며 표준화된 절차나 정량적 신뢰도 표현이 부족하다. 이로 인해 동일 사안에 대해 상이한 결론이 도출될 가능성이 존재하며, 법적 방어 가능성과 투명성이 저하되는 문제가 발생한다. 이러한 한계는 보다 구조화되고 체계적인 완성도 평가 프레임워크의 필요성을 시사한다.

2.2 소프트웨어 완성도 감정에서 LLM 활용

최근 LLM의 발전은 소프트웨어 공학 전반에 걸쳐 자동화 및 지능화 가능성을 확대하고 있으며, 이러한 흐름은 소프트웨어 완성도 감정 영역에도 점진적으로 영향을 미치고 있다. 비록 완성도 감정 자체를 직접적으로 다루는 연구는 아직 제한적이나, 요구사항 공학, 테스트 자동화, 실행 로그 분석, 추적성 관리 등 감정 절차의 구성 요소에 해당하는 영역에서 LLM 활용 연구가 활발히 이루어지고 있다.

Brown 등[6]과 OpenAI[7]는 LLM이 자연어 이해 및 코드 추론에서 높은 성능을 보임을 보고하였으며, 이를 기반으로 자연어 요구사항을 구조화하거나 의미적 관계를 분석하는 연구가 확장되고 있다. 요구사항-코드 간 의미 매칭 연구는 [8]의 LSI 기반 추적성 복원 연구 이후 정보검색 및 임베딩 기법으로 발전해 왔으며, 최근에는 LLM을 활용한 의미 기반 추적성 분석으로 확장되고 있다.

LLM을 활용하여 테스트 케이스를 자동 생성하거나 테스트 스크립트를 작성하는 접근이 제안되고 있으며[6][7], 이는 실행 기반 검증 절차에서 반복적인 테스트 설계 부담을 완화하는 수단으로 활용될 수 있다. IEEE 1012 표준[9]이 명시하는 실행 기반 검증 절차와 결합할 경우, LLM은 테스트 준비 및 문서화 단계의 생산성을 향상시키는 보조 도구로 기능할 수 있다.

또한 실행 로그 분석 및 증거 요약 보조 분야에서도 LLM의 적용 가능성이 논의되고 있다. 대량의 실행 로그와 출력 데이터를 요약하거나 오류 메시지를 해석하는 작업은 LLM의 자연어 처리 능력을 효과적으로 활용할 수 있는 영역이다. 특히 RAG(Retrieval-Augmented Generation) 기법[10]은 외부 근거를 참조하여 생성 신뢰성을 높이는 방식으로 제안되었으며, 이는 증거 기반 분석이 요구되는 감정 업무에 적용 가능한 기술적 기반을 제공한다.

한편, LLM 활용 연구는 동시에 신뢰성 및 환각 문제에 대한 우려를 제기하고 있다. LLM은 환각과 비결정성 특성을 가지며[7], 동일 입력에 대해서도 상이한 출력을 생성할 수 있다. 이러한 특성은 법적 효력을 갖는 감정 절차에서 최종 판단 도구로 활용되기에는 한계가 있음을 시사한다. 이에 따라 최근 연구 경향은 LLM을 단독 의사결정 주체로 사용하는 것이 아니라, 규칙 기반 체계 또는 인간 전문가 검증과 결합하는 하이브리드 접근을 채택하는 방향으로 전개되고 있다[10]. 이러한 연구 경향은 LLM이 반복적·노동 집약적 분석 업무를 보조함으로써 완성도 감정과 같은 고신뢰성 소프트웨어 검증 과제에서 생산성 향상과 품질 개선에 기여할 수 있음을 시사한다.

2.3 환각통제 기법의 주요 기술

최근 LLM의 발전은 소프트웨어 공학 전반에 LLM에서 발생하는 환각 현상은 유창하고 일관성 있어 보이지만 사실과 다르거나 논리적으로 모순되거나 완전히 조작된 출력을 의미한다.

LLM이 교육, 의료, 법률 및 과학 연구 분야에서 점점 더 많이 활용됨에 따라 환각 현상을 이해하고 완화하는 것이 중요해지고 있다.[11]의 연구에서는 LLM에서 발생하는 환각 현상의 원인을 규명하기 위한 포괄적인 조사와 실증분석을 제시하였다. 프롬프트 민감도와 모델 변동성 지표를 통해 환각 발생 특성을 정량화하고, GPT-4, LLaMA 2, DeepSeek 등을 다양한 조건에서 실험·비교하였다.

RAG는 내부 코드베이스, 기술 문서, 요구사항 명세서 등의 외부 지식 기반에서 관련 정보를 실시간으로 검색하여 LLM에 제공한다. LLM은 이 컨텍스트를 기반으로 답변하므로 논리적 비약을 줄이고 사실에 기반한 품질 평가를 수행할 수 있다. [12]에서는 RAG 기반 대규모 언어 모델에서 발생하는 환각의 원인을 검색 단계와 생성 단계로 나누어 체계적으로 분석하였다. 각 단계에서 발생하는 문제를 완화하기 위한 다양한 환각 저감 기법과 탐지·수정 전략을 정리하고, 이를 통합한 종합 대응 프레임워크를 제시하였다.

[13]에서는 LLM과 비전-언어 모델(VLM, Vision Language Model)의 성능을 확장하기 위한 핵심기법으로, '역할 정의(Role)', '가이드라인(Guidelines)', '출력 형식(Output Format)'을 명확히 설정하는 것과 같은 프롬프트 엔지니어링의 구조적 체계를 제시한다. 예를 들면, "코드의 취약점을 찾으라"는 일반적 명령 대신 "코드의 SQL 인젝션 취약점을 Owasp Top 10 기준으로 찾아 JSON 형식으로 출력하라"와 같이 제약 설정을 하도록 제안한다. 기존의 리소스 집약적인 파인튜닝 대신 프롬프트 엔지니어링이 주목받는 가운데, 수동 프롬프트 엔지니어링의 확장성, 적응성, 그리고 모달 간 정렬의 한계를 극복하기 위해 FM(Foundation Model) 기반 최적화, 진화적 방법, 경사 기반 최적화, 강화 학습 등의 자동화 방법이 제시되고 있다. [14]에서는 이러한 다양한 방법들을 통합된 최적화 이론적 관점에서 체계적으로 정리하여, 이산형, 연속형, 하이브리드 프롬프트 공간에서의 최대화 문제로 프롬프트 최적화를 공식화하고 있다.

기존 연구들은 주로 정적 요구사항 추적성 중심 접근 또는 LLM 기반 자동화 접근 방식에 집중되어 있으며, 동적 실행 기반 감정 절차에 LLM을 보조 분석 도구로 안전하게 통합하는 체계적 방법론은 부족하다. 이에 따라 본 논문에서는 “실행 기반 증거 중심 감정 체계를 유지하면서 LLM을 어떻게 보조적으로 활용하여 생산성과 일관성을 동시에 향상시킬 수 있는가?”라는 문제를 논의해 보고자 한다.

3. 감정사례를 통한 문제 정의

소프트웨어 완성도 감정에서 LLM의 활용 가능성과 한계를 구체적으로 살펴보기 위하여, 먼저 실제 감정 사례를 기반으로 문제를 정의한다.

3.1 감정사례 분석

대규모 소프트웨어 시스템의 완성도 감정 과정에서 LLM을 보조 도구로 활용한 사례를 분석하였다. 첫 번째 사례는 교육기관을 위한 이러닝 통합 운영관리시스템에 대한 완성도 감정으로, 표 1과 같이 해당 시스템은 수강생 관리, 학습관리, 콘텐츠 운영, 학원 관리, 메시징, 인증, 배치처리 등 다수의 업무 모듈을 포함하는 대규모 엔터프라이즈 시스템이다.

이와 같은 규모에서 기능-코드 매핑 및 관련 파일 탐색은 감정 절차의 상당한 비중을 차지하게 된다. 이에 LLM을 활용하여 자연어 요구 기능에 기반한 관련 파일 후보 자동 탐색, 기능-코드 매핑 지원을 수행하였다. LLM은 자연어 질의 기반 탐색과 코드 구조 요약을 통해 분석 범위를

표 1. 이러닝통합관리시스템 규모
Table 1. Scale of the Integrated e-Learning Management System

항목	크기
전체 파일 수	8,293개
분석 대상 소스 파일	7,159개
총 코드 라인 수	약 139만 줄
요구 기능 수	1,374개

신속히 축소하는 데 기여하였으며, 부분 구현 기능에 대한 이해를 보조함으로써 초기 코드 분석 시간을 단축하는 효과를 보였다.

또 다른 감정 사례로는 P사의 커머스 플랫폼 구축 사업에 대한 완성도 감정이다. 표 2와 같이, P사의 커머스 플랫폼은 PC 사용자용, 모바일 사용자용, 인플루언서, 브랜드 판매자 및 관리자 모듈을 포함하는 엔터프라이즈 시스템으로 상품 구매, 상품 판매 및 관리, 사용자 및 판매자, 전체 관리 기능 등 총 2,245개 기능을 포함한다.

표 2. 커머스 플랫폼 시스템 규모
Table 2. Scale of the Commerce Platform System

항목	크기
전체 파일 수	10,303개
분석 대상 소스 파일	9,018개
총 코드 라인 수	약 297만 줄
요구 기능 수	2,245개

이 완성도 감정에서도 LLM을 적용하여 함수 매핑, 연관 파일 추천, 코드 요약 등을 수행하였다. 이는 중대 하자 및 경미 하자 판단을 위한 사전 분석 단계에서 생산성 향상에 기여하였다.

그러나 LLM을 감정 과정에 적용하는 과정에서 신뢰성 문제가 확인되었다. 존재하지 않는 함수 또는 UI 구현을 실제 구현된 것으로 설명하거나, 구현되지 않은 기능에 대해 코드 매핑을 제시하는 사례가 발견되었다. 이러한 오류는 감정인의 육안 검증을 통해 식별되었으며, LLM의 의미 기반 추론이 실제 실행 결과 또는 코드 구현과 항상 일치하지 않음을 보여준다. 특히 법적 효력을 가지는 감정 절차에서는 이러한 환각 및 비결정성 문제는 구조적 위험 요소로 작용한다.

이상의 사례는 두 가지 점을 시사한다. 첫째, LLM은 대규모 코드 탐색, 요약, 기능-코드 대응 후보 제시 등 반복적이고 탐색 중심적인 분석 단계에서 실질적인 보조 가치를 가진다. 둘째, LLM은 실행 결과에 대한 최종 판단 주체로 활용되기에는 신뢰성 한계를 가지며, 판단 과정과 구조적으로 분리되어야 한다. 따라서 완성도 감정에서 LLM을 활용하기 위해서는 실행 기반 증거와 규칙 중심의 판정 체계를 유지하면서, LLM을 보조 분석 도구로 통제적으로 통합하는 설계가 요구된다. 이와 같은 문제 인식을 바탕으로, 다음 절에서는 동적 실행 기반 완성도 감정 문제를 형식적으로 정의하고, LLM 보조 활용을 포함하는 감정 모델을 제시한다.

3.2 문제 정의

완성도 감정은 단순한 기능 존재 여부 확인이 아니라, 계약된 요구사항이 실제 실행 결과를 통해 충족되었음을 객관적 증거로 입증하는 절차이다. 특히 대규모 시스템 환경에서는 요구사항-코드-실행증거 간의 관계가 복잡하게 얽혀 있으며, 감정의 난이도는 시스템 규모에 비례하여 증가한다. 앞서 제시한 LMS 및 커머스 플랫폼 사례는 감정 대상이 수천 개의 파일과 백만 줄 이상의 코드로 구성될 경우, 기능-코드 대응 탐색 자체가 감정의 핵심 병목이 됨을 보여준다. 동시에 LLM은 이러한 탐색 및 요약 단계에서 상당한 생산성 향상을 제공할 수 있으나, 환각 문제로 인해 최종 판단 주체로는 사용될 수 없다는 한계를 드러냈다. 이에 본 논문에서는 완성도 감정

문제를 구조적으로 재정의하고, LLM을 보조 분석 계층으로 포함하는 개념적 감정 모델을 제시한다.

완성도 감정의 대상은 요구사항, 구현, 실행 증거의 세 가지 계층으로 구성된다.

- 요구사항 계층: 계약 또는 명세에 의해 정의된 기능적 요구사항
- 구현 계층: 요구사항을 구현하는 코드, 모듈, 데이터 구조
- 실행 증거 계층: 실제 시스템 실행 시 관찰 가능한 동적 결과

이 세 층위는 선형적 관계가 아니라 다대다 관계를 가진다. 하나의 요구사항은 여러 파일에 분산 구현될 수 있으며, 하나의 코드 모듈은 여러 요구사항에 기여할 수 있다. 또한 실행 증거는 단일 코드 경로의 결과가 아니라 다수의 모듈 상호작용의 산물일 수 있다.

완성도 감정은 본질적으로 이 세 층위 간의 대응 관계를 입증하는 문제로 정의된다. 즉, 특정 요구사항이 실행 결과를 통해 재현가능한 방식으로 구현되었음을 확인하는 것이다.

또한 완성도 판정은 단일 단계의 판단이 아니라, 탐색-해석-판정의 인지적 과정을 포함한다. 탐색 단계에서는 요구사항과 관련된 코드 및 실행 경로 식별하고, 해석 단계에서는 실행 결과 및 로그의 의미 분석한다. 그리고 판정 단계에서 명시적 기준에 따른 구현 여부를 결정하게 된다. 기존 감정 방식에서는 이 단계가 대부분 전문가의 수작업에 의해 수행된다. 특히 탐색 단계는 대규모 시스템에서 시간 소모가 가장 큰 단계이며, 다양한 도구들을 활용하기는 하지만 감정인의 경험과 직관에 크게 의존한다. 사례 분석에서 확인된 바와 같이, LLM은 탐색 및 해석 단계에서 정보 축약과 후보 제시를 통해 인지 부담을 경감시킬 수 있다. 그러나 판정 단계는 여전히 명시적 기준과 실행 증거에 근거하여 이루어져야 한다.

LLM의 환각 문제는 의미 기반 추론이 실제 구현 상태와 불일치할 수 있다는 점에서 발생한다. 이는 언어적 유사성과 기능적 구현의 불일치에서 비롯된다. 즉, 코드 상의 표현이 요구사항과 의미적으로 유사하더라도, 실제 실행 흐름에서 해당 기능이 완전하게 구현되지 않았을 수 있다. 따라서 LLM이 제시하는 정보는 “가능성 있는 후보”로 간주되어야 하며, 실행 증거와의 교차 검증을 통해만 감정 판단에 반영되어야 한다.

이 점에서 본 연구의 감정 모델은 다음의 원칙을 따른다.

- 의미 기반 분석(LLM)과 실행 기반 검증을 구조적으로 분리
- 탐색·요약 기능과 판정 기능의 역할 구분
- 모든 LLM 출력에 대한 검증 절차 포함

이는 LLM을 지식 생성 도구가 아닌 분석 보조 도구로 위치시키는 이론적 근거가 된다. 이와 같이 LLM을 보조 분석 계층으로 포함하는 계층형 감정 모델은 LLM 활용을 통한 생산성 향상, 실행 증거 중심 판정으로 신뢰성 확보, 그리고 역할 분리 및 검증을 통한 환각 통제가 가능해질 것이다.

4. LLM 기반 완성도 감정

제안 방법론은 동적 실행 기반 검증 구조를 유지하면서, 대규모 코드 탐색과 증거 구조화 단계에 LLM을 보조적으로 통합하는 계층형 접근을 따른다. 핵심 원칙은 다음과 같다.

- 실행 기반 검증은 감정의 중심 절차로 유지한다.
- LLM은 탐색·요약·구조화 단계에 한정하여 활용한다.
- 최종 판정은 실행 증거와 명시적 기준에 따라 전문가가 수행한다.

4.1 방법론 개요

제안 방법론은 그림 1과 같이 총 6단계로 구성된다. 요구사항 정제 및 기능 단위 확정, LLM 기반 구현 후보 탐색, 테스트 시나리오 설계 및 실행, 실행 증거 수집 및 정합성 검증, LLM 기반 증거 구조화 및 매핑 지원, 그리고 규칙 기반 판정 및 감정 결과 도출 단계를 거치게 된다.

감정의 출발점은 계약서 또는 요구사항 명세서에 정의된 기능이다. 그러나 실제 명세는 추상적이거나 복합 기능 형태로 기술되는 경우가 많다. 따라서 감정인은 첫 번째 단계인 요구사항 정제 및 기능 단위 확정 과정에서 4가지 작업을 수행한다. 요구사항을 개별 기능 단위로 분해하고, 입력 조건과 기대 결과를 명확하게 하고, 기능 수행 경계를 정의하고, 판정 기준을 사전 정의한다. 이 단계는 감정의 객관성을 확보하는 핵심 단계이며, LLM은 보조적으로 요구사항 구조화를 지원할 수 있다.

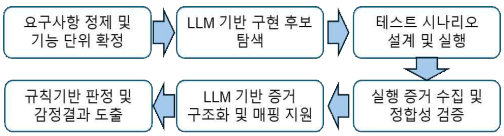

그림 1. LLM 보조 완성도 감정절차
Fig. 1. LLM based Appraisal process

두 번째 단계는 LLM 기반 구현 후보 탐색 과정이다. 대규모 시스템에서는 특정 기능과 관련된 코드 영역을 식별하는 것이 가장 큰 병목이다. 이 단계에서는 LLM을 활용하여 4가지 작업을 수행한다.

- 자연어 요구사항을 기반으로 코드 검색 질의 생성
- 관련 가능성이 높은 파일·함수·모듈 후보제시
- 코드 블록 요약 및 핵심 로직 설명
- 유사 기능 구현 위치 추천

이 단계의 목적은 “구현 여부 판단”이 아니라 “분석 범위 축소”이다. LLM이 제시한 후보는 감정인이 직접 확인하며, 실제 코드 존재 여부와 논리 흐름을 검증한다. 이 과정은 기존의 전면 수작업 탐색 대비 분석 시간을 현저히 단축하는 효과를 보인다.

다음 3단계는 테스트 시나리오 설계 및 실행이다. 완성도 감정은 동적 실행 기반 절차이므로, 실제 동작 검증이 필수적이다. 이 단계에서 수행되는 작업은 (1) 기능별 테스트 시나리오 정의, (2) 입력 데이터 및 경계 조건 설정, (3) 예상 출력 명시, (4) 테스트 자동화 스크립트 작성, 그리고 (5) 실제 시스템 실행 과정을 거친다. LLM은 테스트 시나리오 초안 작성이나 경계 조건 제안에 활용될 수 있으나, 실행 자체는 통제된 환경에서 결정론적으로 수행된다.

4단계 실행 증거 수집 및 정합성 검증은 테스트 실행 후 동적 증거를 수집한다. 수집되는 동적 증거는 실행 로그, 오류 메시지, 화면 출력 캡처, API 응답 데이터, 데이터베이스 상태 변화 등이다. 이 단계에서는 수집된 증거가 실제 실행 결과를 정확히 반영하는지 확인한다. 즉, 증거의 무결성과 일관성을 검증한다.

다음 5단계 LLM 기반 증거 구조화 및 매핑 지원이다. 대규모 로그 데이터와 실행 결과를 해석하는 과정에서 LLM의 역할은 크게 4가지이다. 로그 요약 및 핵심 이벤트 추출, 실행 흐름 설명, 요구사항과 실행 결과 간 의미적 연결 후보 제시, 그리고 보고서 초안 문장 생성이다. 이 단계에서 LLM은 정보 정리 및 요약 도구로 활용되며, 판단을 수행하지 않는다. 모든 요약 결과는 실제 로그 및 출력과 교차 검증된다.

마지막 단계는 규칙 기반 최종 판정이다. 최종 구현 여부는 사전에 정의된 판정 기준에 따라 결정된다. 예를 들면 판정 기준은 예상 출력과 완전 일치한다면, “완전 구현”, 일부 조건 누락 또는 부분 동작한다면, “부분 구현”, 그리고 실행

실패 또는 기능 부재라면, “미구현”이 될 수 있다. 판정 단계에서는 LLM을 사용하지 않으며, 실행 증거와 판정 기준에 기반하여 감정인이 결론을 도출한다.

4.2 환각통제 및 신뢰성 확보 전략

사례 분석에서 확인된 환각 문제를 방지하기 위해 다음 통제 전략을 적용한다.

- LLM 출력은 반드시 실제 코드 및 실행 로그와 대조 검증한다.
- 존재하지 않는 파일·함수·UI에 대한 설명은 즉시 배제한다.
- 프롬프트를 고정하고 응답 형식을 표준화한다.
- 모든 LLM 질의 및 응답 기록을 보존한다.
- 동일 입력에 대해 반복 실행하여 결과 일관성을 점검한다.

이를 통해 LLM은 분석 보조 도구로 한정되며, 감정 결과의 책임은 전문가에게 귀속된다. 기존 방식은 대부분 수작업 중심이었으며, 특히 코드 탐색과 로그 해석에 과도한 시간이 소요되었다. 제안 방법은 탐색 단계 및 로그 요약에 자동화를 지원하며, 판단 단계는 감정인의 활동으로 유지하면서 환각 통제 절차를 명시 했다는 점에서 구조적 차이를 가진다고 할 수 있다. 즉, 판단 구조를 변경하지 않으면서 생산성만을 향상시키는 접근이다. 이를 통해 분석 시간을 단축하고, 대규모 시스템에 대한 확장성 확보 및 판단의 일관성 향상이 가능할 것으로 기대한다. 이는 LLM을 자동 판정 시스템이 아닌, 통제된 감정 지원 시스템으로 위치시키는 설계 철학에 기반한다.

4.3 환각 통제 설계의 의미

본 연구는 소프트웨어 완성도 감정이라는 고신뢰성·증거 중심 절차에 LLM을 통합하기 위한 구조적 설계 원칙을 제시하였다. 핵심 기여는 LLM을 판단 주체로 사용하지 않고, 탐색·요약·구조화 단계에 한정된 보조 분석 계층으로 분리한 점에 있다.

완성도 감정은 실행 결과에 대한 입증을 통해 요구사항 충족 여부를 판단하는 동적 검증 절차이며, 법적 효력을 가질 수 있다는 점에서 자동화 시스템과는 다른 수준의 신뢰성을 요구한다. 본 연구는 이러한 전제를 유지하면서, 의미 기반 분석과 실행 기반 판정을 구조적으로 분리하는 계층형 모델을 제안하였다.

이는 기존 요구사항 추적성 연구나 LLM 기반 테스트 자동화 연구와 달리, 실행 기반 입증 구조를 중심에 두었다는 점에서 차별성을 가진다. LLM의 환각과 비결정성은 고신뢰성 환경에서 구조적 위험 요소가 된다. 본 연구는 환각을 제거하는 접근이 아니라, 환각이 감정 결과에 영향을 미치지 않도록 판단 구조와 분리하는 통제 전략을 채택하였다. 이러한 설계는 LLM을 지식 생성 주체가 아니라 분석 보조 도구로 위치시키며, 감정 결과의 책임을 전문가에게 유지하도록 한다. 이는 법적 방어 가능성과 설명 가능성을 동시에 고려한 설계 접근으로 평가할 수 있다.

5. 결 론

본 연구는 동적 실행 기반 소프트웨어 완성도 감정 절차를 유지하면서 LLM을 보조 분석 도구로 통합하는 환각 통제 기반 방법론을 제안하였다. LLM을 판단 주체가 아닌 보조 계층으로 명확히 분리하고, 실행 증거와 명시적 판정 기준에 따라 최종 판단이 이루어지도록 하는 계층형 통합 구조를 설계하였다. 이를 통해 생산성 향상과 신뢰성 유지라는 상충 가능성이 있는 두 목표를 동시에 고려하는 균형 모델을 제시하였다.

본 연구는 LLM을 자동 판정 시스템이 아니라

통제된 분석 지원 체계로 활용하는 설계 원칙을 구체화하였다. 이는 향후 고신뢰성 소프트웨어 검증 및 감정 영역에서 LLM을 안전하게 통합하기 위한 기초 설계 방향을 제시한다는 점에서 의의를 가진다고 할 수 있다.

향후 연구는 다음과 같은 방향으로 체계화·고도화될 필요가 있다. 첫째, LLM 적용 전후의 감정 수행 과정에 대해 소요 시간, 판정 일관성, 오류율, 재현성 등 핵심 성과지표를 설정하고, 이를 기반으로 한 정량적 비교 실험 설계를 통해 LLM 도입 효과를 실증적으로 검증할 필요가 있다. 둘째, 현재 기능 요구사항 중심의 실행 기반 검증 구조를 확장하여 성능, 보안, 안정성, 사용성 등 비기능 요구사항까지 포함하는 통합 실행-분석 모델로 확장함으로써 완성도 감정의 평가 범위를 이론적으로 정교화할 필요가 있다. 셋째, 공공·사법 감정 환경의 특수성을 고려하여 외부 의존성을 최소화한 온프레미스 LLM 기반의 통제 환경을 설계하고, 로그 추적·버전 고정·프롬프트 관리 체계를 포함하는 재현 가능 감정 인프라 모델을 구축할 필요가 있다. 넷째, LLM 출력 결과에 대한 신뢰도 점수화 메커니즘을 설계하고, 환각 가능성·증거 불충분성·해석 불확실성 등을 반영한 위험 기반 보조 판단 모델을 정립함으로써 인간 감정인의 최종 판단을 체계적으로 지원하는 의사결정 프레임워크를 제안할 필요가 있다. 특히, 실행 증거와 의미 기반 분석 결과 간의 정합성을 자동으로 교차 검증하는 메타 분석 체계의 설계는 본 연구의 핵심 확장 과제로서, 감정 판단의 객관성과 설명 가능성을 동시에 강화할 수 있는 중요한 연구 주제로 발전될 수 있다.

참 고 문 헌

[1] Y. Kim, “A Study on the Completeness Measurement of Requirements Specifications for Software Completion Appraisal”, Journal of Software Assessment and Valuation, vol. 19, no.1, pp.11-18, 2023. DOI: https://dx.doi.org/10.29056/jsav.2023.3.02.

[2] G. Woo, “Solving the Problems in the Appraisal on the Completeness of Software”, Journal of Software Assessment and Valuation, vol. 19, no.4, pp.23-31, 2023. DOI: https://dx.doi.org/10.29056/jsav.2023.12.03

[3] J. Wang, Y. Huang, C. Chen, Z. Liu, S. Wang, Q. Wang, “Software Testing With Large Language Models: Survey, Landscape, and Vision”, IEEE Transactions on Software Engineering, Vol. 50, Issue 4, pp. 911-936, 2024. DOI: https://doi.org/10.1109/TSE.2024.3368208

[4] D. Kim, “A Study on the Need for Separation of Software Completeness Appraisal and Software Ready-made Appraisal”, Journal of Software Assessment and Valuation, vol. 17, no. 2, pp.11-17, 2021. DOI: https://doi.org/10.29056/jsav.2021.12.02

[5] Y. S. Yun, “Meaning and Computation of Completeness and Payment in SW Appraisal”, Journal of Software Forensics, vol.15, no.2, pp. 35-42, 2019. DOI : https://doi.org/10.29056/jsav.2019.12.05

[6] T. Brown, et al., “Language Models are Few-Shot Learners”, Proceedings of the International Conference on Neural Information Processing Systems, article no 159, pp. 1877- 1901, 2020.

[7] OpenAI, “GPT-4 Technical Report”, 2023. [online] https://cdn.openai.com/papers/gpt-4.pdf

[8] A. Marcus, and J. I. Maletic, “Recovering documentation-to-source-code traceability links using latent semantic indexing”, Proceedings of the International Conference on Software Engineering, pp.

125-135, 2003. DOI: https://doir.org/10.1109/ICSE.2003.1201194.
[9] IEEE, “IEEE Standard for System and Software Verification and Validation”, IEEE Std 1012-2024.
[10] P. Lewis, et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”, 2020. [online] https://doi.org/10.48550/arXiv.2005.11401
[11] D. Hoang, V. Tran, L. Nguyen, “Survey and analysis of hallucinations in large language models: attribution to prompting strategies or model behavior”, Frontiers in Artificial Intelligence, vol. 8, pp.1-21, 2025. DOI: https://doi.org/10.3389/frai.2025.1622292
[12] W. Zhang and J. Zhang, “Hallucination Mitigation for Retrieval-Augmented Large Language Models: A Review”, Mathematics, vol. 13, no. 5, ID 856, 2025. DOI: https://doi.org/10.3390/math13050856
[13] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, “A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications”, 2025. [Online] https://doi.org/10.48550/arXiv.2402.07927
[14] W. Li, X. Wang, W. Li, and B. Jin, “A Survey of Automatic Prompt Engineering: An Optimization Perspective”, 2025. [Online] https://doi.org/10.48550/arXiv.2502.11560

저 자 소 개

김유경(Yukyong Kim)

2005.9-2006.8 UC Davis, Post-doc.
2006.9-2013.9 한양대학교 컴퓨터공학과 연구교수
2018.3-현재 숙명여자대학교 첨단공학부 교수
<주관심분야> 웹서비스 QoS 평가, SOA 기반 IoT 신뢰 평가, 소프트웨어 품질 평가