

보안 환경을 위한 RAG 기반 메모리 불량 원인 자동 분석 에이전트 구축

하이브리드 SLM 라우팅 및 RAG 연동 Data Flywheel 파이프라인 설계

정원준 | University of Wisconsin-Madison | Computer Science & Data Science
willjung0726@gmail.com | github

1. Executive Summary

본 프로젝트는 대규모 IT 및 하드웨어 제조 기업의 실제 평가/분석 업무 환경을 고려하여, 시스템 레벨에서 발생하는 방대한 리눅스(Linux) 터미널 로그를 분석하고 하드웨어 불량 원인(Root Cause) 파악을 자동화하는 지능형 AI 에이전트 구축을 목표로 한다. 특히 인텔(Intel) 및 ARM CPU 기반 시스템 환경에서 발생하는 리눅스 로그를 분석하는 것은 곧 장착된 메모리(DRAM 등) 하드웨어의 불량을 역추적하는 핵심 과정이다. 그러나 방대하고 난해한 원시 로그(Raw Log)는 고도로 숙련된 엔지니어가 아니면 즉각적인 해독이 어려우므로, AI 에이전트가 이를 인간이 직관적으로 이해할 수 있는 자연어 문장으로 번역 및 해설하여 제공하는 것을 주요 목표 중 하나로 삼았다.

하드웨어 평가 과정에서 발생하는 로그 데이터와 불량 분석 리포트는 기업의 핵심 기밀이다. 따라서 상용 클라우드 LLM(예: GPT-4o)에 기밀 로그를 직접 전송하는 것은 치명적인 보안 위규 위험을 동반하며, 매일 수십만 건의 로그를 처리하기에는 막대한 API 호출 비용이 발생한다. 또한, 수개월 주기로 신제품이 출시되어 에러 코드 규격이 수시로 변경되는 산업 특성상, 기존의 거대 언어 모델 파인튜닝(Fine-tuning) 방식만으로는 모델의 지식을 최신 상태로 유지하는 데 한계가 존재한다.

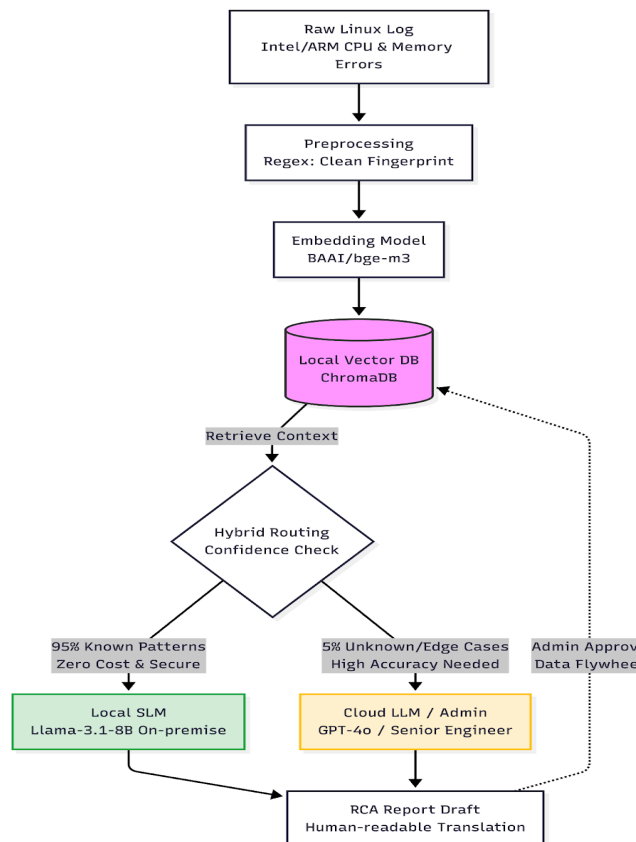
이러한 보안적, 비용적, 기술적 한계를 극복하기 위해 본 프로젝트는 다음과 같은 세 가지 핵심 아키텍처를 도입하여 '데이터 주도권'과 '분석 효율성'을 동시에 확보한 하이브리드 파이프라인을 설계하였다.

- 로컬 기반 RAG(검색 증강 생성) 아키텍처 도입 오픈소스 임베딩 모델(BAAI/bge-m3)과 로컬 벡터 데이터베이스(ChromaDB)를 활용하여 사내망을 벗어나지 않는 100% 폐쇄형 RAG 환경을 구축했다. 로그 데이터의 특성을 반영하여 타임스탬프와 PID 등 동적 노이즈를 정규식(Regex)으로 제거한 'Clean Fingerprint'를 벡터화 기준으로 삼아 검색 정확도를 극대화했다. 이를 통해 신규 에러 발생 시 값비싼 모델 재학습 없이 문서 업데이트만으로 1분 내에

지식을 최신화할 수 있으며, 도출된 답변의 명확한 근거를 제시하여 AI의 환각 현상(Hallucination)을 원천 차단하였다.

- **가성비와 보안을 모두 충족하는 하이브리드 모델 운용 (SLM + LLM)** 비식별화된 오픈소스 데이터셋(Loghub의 Linux_2k.log)을 바탕으로 Teacher Model(GPT-4o)을 활용해 고품질 합성 데이터(Synthetic Data)를 생성하고, 이를 기반으로 온프레미스(On-premise) 구동이 가능한 소형 언어 모델(Llama-3.1-8B)을 파인튜닝했다. 현업 환경에서는 전체 로그의 95%에 달하는 정형화된 패턴을 로컬 SLM이 무비용('0원')으로 안전하게 처리하며, SLM이 판별하기 어려운 5%의 특이 케이스(Unknown Log)에 대해서만 시니어 엔지니어나 클라우드 LLM이 개입하도록 효율적인 리소스 라우팅 전략을 수립하였다.
- **엔지니어 피드백 기반 Data Flywheel(데이터 선순환) 시스템 구축** Streamlit 기반의 인터랙티브 웹 UI와 관리자(Admin) 대시보드를 구축하였다. 에이전트가 분석한 결과 중 품질이 우수한 Q&A 쌍을 관리자가 승인(Approve)하면, 즉시 로컬 Vector DB에 업데이트되도록 설계하였다. 이는 시스템이 실무에 투입된 이후에도 현업 엔지니어의 노하우를 지속적으로 흡수하며 스스로 성능을 진화시키는 MLOps 기반의 지식 선순환 구조를 완성한 것이다.

결과적으로 본 프로젝트는 단순한 질의응답 챗봇을 넘어, 장애 발생 시 원인 분석 보고서(RCA Report)의 초안을 자동으로 작성하는 실무 밀착형 워크플로우를 구현하였다. 이는 차세대 AI 기술을 어떻게 안전하고 경제적으로 하드웨어 평가 실무에 이식할 수 있는지 실험해본 사례이다.



[Figure 1] Hybrid AI Architecture for Secure Log Analysis and Data Flywheel

2. Introduction

2.1. 하드웨어 불량 분석에서 시스템 로그의 중요성 및 한계

대규모 IT 및 하드웨어 제조 산업에서 신제품(메모리 등) 검증 시, 인텔(Intel) 및 ARM CPU 기반 시스템의 리눅스(Linux) 로그 분석은 하드웨어 결함을 역추적(Root Cause Analysis, RCA)하는 핵심 과정이다. 그러나 시스템에서 쏟아지는 방대한 원시 로그(Raw Log)는 난해한 기계어와 에러 코드로 이루어져 있어, 숙련된 엔지니어도 즉각적인 해독이 어려우며 분석에 막대한 시간과 인력이 소모된다. 따라서 이를 인간이 직관적으로 이해할 수 있는 자연어로 번역하고, 불량 원인 분석 보고서의 초안 작성을 자동화하는 지능형 시스템의 도입이 절실히 요구된다.

2.2. 기존 AI 도입 방식의 한계 (보안, 비용, 그리고 최신성)

최근 대형 언어 모델(LLM)을 텍스트 분석에 도입하려는 시도가 늘고 있으나, 하드웨어 평가 실무에 직접 적용하기에는 세 가지 치명적인 한계가 존재한다. 첫째, **보안(Security) 및 비용(Cost) 문제**이다. 일급 기밀인 하드웨어 로그를 GPT-4o와 같은 상용 클라우드 LLM으로 전송하는 것은 심각한 보안 위규이며, 매일 발생하는 수십만 건의 로그를 API로 처리하는 것은 경제적 타당성이 없다. 둘째, **파인튜닝(Fine-tuning)의 한계**이다. 반도체는 수개월 단위로 신제품이 출시되어 에러 규격이 수시로 변한다. 지식이 추가될 때마다 막대한 자원을 들여 거대 모델을 재학습시키는 기존 방식으로서는 데이터의 최신성을 유지하기 어렵다. 셋째, 환각 현상(Hallucination)이다. 단 1%의 오류도 용납되지 않는 정밀한 엔지니어링 영역에서, 명확한 출처를 제시하지 못하는 AI의 추론 결과는 현장에서 신뢰받을 수 없다.

2.3. 연구 목적 및 제안

본 프로젝트는 이러한 한계를 극복하고자 '폐쇄망(Air-gapped) 환경 기반의 하이브리드 RAG(검색 증강 생성) 로그 분석 에이전트'를 제안한다. 파인튜닝된 로컬 소형 언어 모델(SLM, Llama-3.1-8B)을 통해 무비용으로 안전하게 로그를 분석하고, RAG 아키텍처를 결합하여 모델 재학습 없이도 신규 지식을 1분 내에 업데이트하도록 설계하였다. 궁극적으로 난해한 기계어 로그를 번역하고 명확한 근거에 기반한 RCA 보고서를 자동 작성하는 실무 밀착형 MLOps 파이프라인을 구축하는 것을 목적으로 한다.

3. Proposed Method

3.1. 전처리 파이프라인 및 로컬 벡터 DB 구축

시스템 로그 분석의 첫 단계로 비식별화된 오픈소스 데이터셋(Loghub의 Linux_2k.log)을 활용하여 전처리 파이프라인을 구축하였다. 검색 정확도를 극대화하기 위해 타임스탬프, PID 등 분석에 불필요한 동적 노이즈를 정규식(Regex)으로 제거하고 핵심 패턴만 남긴 'Clean Fingerprint'를 추출하였다. 임베딩 모델로는 코드 문맥 이해에 특화되고 온프레미스 구동이 가능한 오픈소스 모델인 BAAI/bge-m3를 선정하였다. 변환된 다차원 벡터 데이터는 서버 외부로 유출되지 않도록 로컬 벡터 데이터베이스(ChromaDB)에 적재하여, 기밀 로그 유출을 원천 차단하는 보안 환경을 구성하였다.

3.2. RAG(검색 증강 생성) 아키텍처 원리 및 수학적 임계값(Threshold) 기반 환각 통제

RAG(Retrieval-Augmented Generation)는 대형 언어 모델이 답변을 생성하기 전에, 외부의 신뢰할 수 있는 데이터베이스에서 관련 정보를 먼저 검색(Retrieval)하여 문맥(Context)으로 제공하는 기술이다. 사용자가 불량 로그 분석을 요청하면, 시스템은 입력된 로그를 다차원 벡터로 변환(Embedding)하여 로컬 벡터 DB에서 수학적 거리(Distance)가 가장 가까운 상위 3개(Top-3)의 과거 사례를 검색한다.

이 과정에서 발생할 수 있는 '검색 오류에 의한 환각 현상을 원천 차단하기 위해, 본 프로젝트는 **수학적 임계값(Distance Threshold) 기반의 엄격한 필터링 알고리즘**을 도입하였다. 임베딩 모델이 산출하는 코사인/L2 거리(Distance)는 0에 가까울수록 문장의 의미가 완벽히 일치함을 뜻하며, 거리가 멀어질수록 무관한 구문임을 나타낸다. 사용자가 DB에 존재하지 않는 특이 에러나 전혀 무관한 질의를 입력할 경우, 시스템이 억지로 상위 3개 문서를 참조하여 잘못된 답변을 지어내는 것을 방지하기 위해 검색된 문서의 거리값이 기준을 초과하면 "직접적으로 관련된 로그를 찾을 수 없습니다"로 예외 처리하여 시스템의 신뢰성을 보장한다.

특히, 이 임계값(0.95)은 단순한 휴리스틱이나 매직 넘버가 아니다. 별도의 벤치마킹 스크립트(check_threshold.py)를 자체 구축하여 '관련 있는 쿼리' 그룹과 '관련 없는 쿼리' 그룹 간의 벡터 거리 간극을 계산하고, 이를 바탕으로 도출된 **수학적 캘리브레이션(Calibration)의 결과물**이다. 이는 머신러닝의 유사도 검색 결과를 맹신하지 않고 수학적 기준을 통해 검색 정밀도(Precision)를 직접 통제하는 엔지니어링적 접근 방식이다.

이러한 고도화된 RAG 아키텍처는 모델 내부에 지식을 정적으로 고정시키는 파인튜닝 방식과 대비된다. RAG는 외부 DB에 텍스트를 추가하는 것만으로 지식의 최신화가 가능하며, 도출된 답변에 대한 과거의 정확한 리포트를 근거로 제시하므로 산업 현장에서 요구되는 무결성을 충족한다. 요약하자면, RAG는 데이터의 최신성을 보장하고, 파인튜닝은 도메인 특화를 보장하는 상호 보완적 관계에 있다.

3.3. 보안 및 비용 최적화를 위한 합성 데이터 기반 SLM 파인튜닝

클라우드 LLM(GPT-4o)의 직접 사용은 데이터 외부 전송에 따른 심각한 보안 위규 위험과, 일일 수십만 건의 로그 처리에 따른 막대한 추론 비용을 발생시킨다. 이를 해결하기 위해, 인터넷이 차단된 폐쇄망 환경에서 단독 구동이 가능한 소형 언어 모델(SLM, Llama-3.1-8B)을 도입하고 다음과 같은 4단계 프로세스를 통해 도메인 특화 파인튜닝을 수행하였다.

- **Step 1. 로그 추출 및 데이터 증강 (Data Augmentation):** 원본 데이터셋(Linux_2k.log)에서 에러가 발생한 주요 로그 1,000라인을 무작위로 추출한 뒤, 유사한 에러 패턴을 추가 생성하도록 지시하여 총 3,000여 개의 다양한 변형 데이터를 구축하였다. 이를 통해 특정 로그에 대한 과적합(Overfitting)을 방지하고 데이터의 다양성을 확보하였다.
- **Step 2. 교사 모델(Teacher Model)의 RAG 기반 레이블링:** 정답지(분석 결과)가 부재한 원시 데이터의 한계를 극복하기 위해, GPT-4o를 교사 모델로 활용하는 합성 데이터(Synthetic Data) 생성 기법을 적용했다. 단순히 로그만 제공하는 것이 아니라 RAG 시스템이 검색한 과거 유사

사례를 프롬프트에 포함시켰다. 이를 통해 "이 로그는 유사 문서 내용에 비추어 볼 때 과거 특정 하드웨어 이슈와 일치한다"는 고도의 맥락이 반영된 모범 답안(JSON 형식의 원인 및 해결책)을 도출하였다.

- **Step 3. 데이터셋 구축:** 위 과정을 통해 생성된 (로그-분석 결과) 쌍 3,061건을 수합하여 최종 파인튜닝용 데이터셋(synthetic_dataset_rag.jsonl)을 완성하였다.
- **Step 4. 학생 모델(SLM) 학습 및 일반화(Generalization):** 구축된 데이터셋을 바탕으로 매개변수가 적은 오픈소스 모델(Llama-3.1)을 학습시켜 교사 모델의 논리 구조와 해결 방식을 모방하도록 하였다.

이러한 파인튜닝을 통해 SLM은 단순 암기가 아닌 '로그를 해석하는 패턴'을 습득하게 된다. 따라서 'CPU 0번 과열'을 학습한 모델이 'CPU 3번 과열'이라는 미지의 로그를 마주하더라도 정확히 일반화하여 분석할 수 있다. 단, 완전히 새로운 도메인의 로그에 대해서는 환각 현상이 발생할 수 있다. 이를 방지하기 위해 전체 로그의 95%에 달하는 정형화된 패턴은 파인튜닝된 로컬 SLM이 통신 비용 없이('0원') 안전하게 처리하고, SLM이 판독하지 못하는 5%의 희귀 로그(Edge Case)에 대해서만 시니어 엔지니어가 직접 개입하는 **하이브리드 라우팅 전략**을 수립하여 리소스 운영 효율을 극대화하였다.

3.4. MLOps 기반 Data Flywheel(데이터 선순환) 시스템 구축 및 평가 파이프라인의 내재화 설계

AI 에이전트가 실무에 투입된 이후에도 현업의 데이터를 바탕으로 스스로 진화할 수 있도록 MLOps 기반의 데이터 선순환(Data Flywheel) 구조를 구축하였다. Streamlit 기반의 웹 UI에서 사용자가 로그 분석을 요청하면, 자동 평가 로직이 답변의 품질을 점수화한다. 일정 기준을 통과한 고품질 Q&A 쌍은 대기열(Pending Review)에 등록되며, 시니어 엔지니어가 관리자 대시보드에서 이를 검토 후 승인하면 즉시 로컬 벡터 DB에 새로운 지식으로 업데이트된다. 이는 엔지니어의 암묵적인 분석 노하우가 시스템의 영구적인 지식 자산으로 지속 축적되는 구조이다.

본 프로젝트의 PoC(Proof of Concept) 단계에서는 물리적인 GPU 리소스의 한계로 인해, 해당 데이터 수집 및 자동 평가 파이프라인의 동작성을 검증하기 위한 목적으로 경량형 클라우드 API(gpt-4o-mini)를 임시 활용하였다. 그러나 실제 완벽한 보안이 요구되는 반도체 평가 현장에 상용화할 경우, 이 평가자 모델 역시 사내망에 구축된 온프레미스 소형 언어 모델(예: 프롬프트 채점 전용으로 파인튜닝된 Llama-3-8B 등)로 유연하게 교체할 수 있도록 파이프라인을 모듈화하여 설계하였다. 이를 통해 데이터 유출의 위험을 원천 차단하면서도, 100% 내재화된 환경에서 지식 기반을 지속적으로 확장할 수 있다.

4. System Implementation & Validation

본 프로젝트는 단순한 언어 모델 테스트를 넘어, 실제 하드웨어 평가 현장에서 즉시 활용 가능한 수준의 End-to-End 파이프라인(웹 UI, 벡터 DB, 모델 라우팅 등)을 구축하였다. 개발된 지능형 에이전트의 실효성을 증명하기 위해 3가지 핵심 실무 시나리오에 대한 실증(Validation)을 수행하였다.

4.1. End-to-End 로그 분석 및 2단계 환각(Hallucination) 방어 검증

사용자가 Streamlit 기반의 인터랙티브 웹 UI에 원시 로그(Raw Log)를 입력했을 때, 시스템이 노이즈를 필터링하고 정확한 문맥을 검색하여 원인 분석 리포트를 생성하는 과정을 검증하였다. 현실적인 테스트를 위해 시간 및 서버명 노이즈(Feb 25 08:30:12 mem-eval-node03 kernel: [1.234567])가 포함된 ACPI 에러 로그를 입력하였다. 그 결과, 임베딩 모델이 표면적인 텍스트가 아닌 '의미(Semantic)'를 정확히 파악하여 로컬 DB에서 가장 적합한 과거 사례를 1순위로 검색해 왔으며, 파인튜닝된 SLM(Local Llama)이 이를 바탕으로 완벽하게 구조화된 해결책을 제시함을 확인하였다.

Result

Analyzed Log Data (Click to expand)

Feb 25 08:30:12 mem-eval-node03 kernel: [1.234567] PCI: Invalid ACPI-PCI IRQ routing table

Retrieved Context

Context

[Log]
Jul 27 14:41:59 combo kernel: PCI: Invalid ACPI-PCI IRQ routing table

[Solution]

1. **Root Cause**: The log message indicates that there is an issue with the ACPI (Advanced Configuration and Power Interface) configuration related to PCI (Peripheral Component Interconnect) IRQ (Interrupt Request) routing. This typically occurs due to a BIOS or firmware issue where the system is unable to correctly interpret the IRQ routing table.

2. **Solution/Command**:

- Update BIOS/Firmware**: Check for and apply any available updates for your system's BIOS or firmware, as manufacturers often release updates to fix such compatibility issues.
- Kernel Parameter**: If updating the BIOS is not possible or does not resolve the issue, try adding the `pci=noacpi` kernel parameter to your boot configuration to bypass ACPI for PCI IRQ routing. You can do this by editing the GRUB configuration file (usually located at `/etc/default/grub`) and adding `pci=noacpi` to the `GRUB_CMDLINE_LINUX` line. After editing, update GRUB with:

```
sudo update-grub
```

AI Analysis

Root Cause Analysis

- Log Entry 1**: The log message indicates that there is an issue with the ACPI configuration related to PCI IRQ routing. This typically occurs due to a BIOS or firmware issue where the system is unable to correctly interpret the IRQ routing table.
- Log Entry 2**: The log message shows that the ACPI tables do not contain any PCI IRQ routing entries, which means the system's BIOS/firmware is not providing the necessary information for the kernel to configure PCI devices properly.
- Log Entry 3**: This log entry indicates that the system is using the PIIX/ICH chipset for PCI IRQ routing, which is a normal informational message unless there are issues with PCI devices not functioning correctly due to IRQ conflicts or misconfigurations.

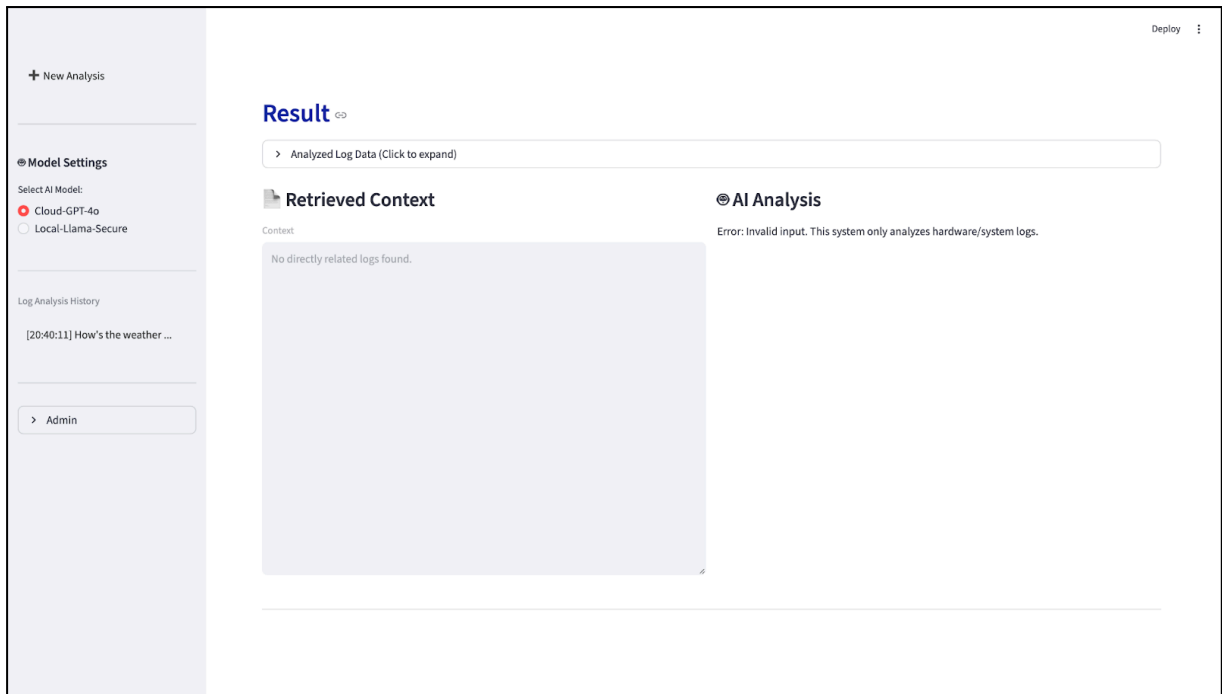
Solution

- For all three log entries:
 - Update BIOS/Firmware**: Ensure that the system's BIOS/firmware is up to date as updates may include fixes for ACPI table and IRQ routing issues.
 - If updating the BIOS does not resolve the issue, try booting the system with the kernel parameter `pci=noacpi` to bypass ACPI for PCI configuration.
 - Command**: Add the `pci=noacpi` kernel parameter to your boot configuration by editing the GRUB configuration file (usually located at `/etc/default/grub`) and adding `pci=noacpi` to the `GRUB_CMDLINE_LINUX` line. After editing, update GRUB with:

```
sudo update-grub
```

[Figure 2] End-to-End Root Cause Analysis for a Valid Hardware Log using Local-Llama-Secure

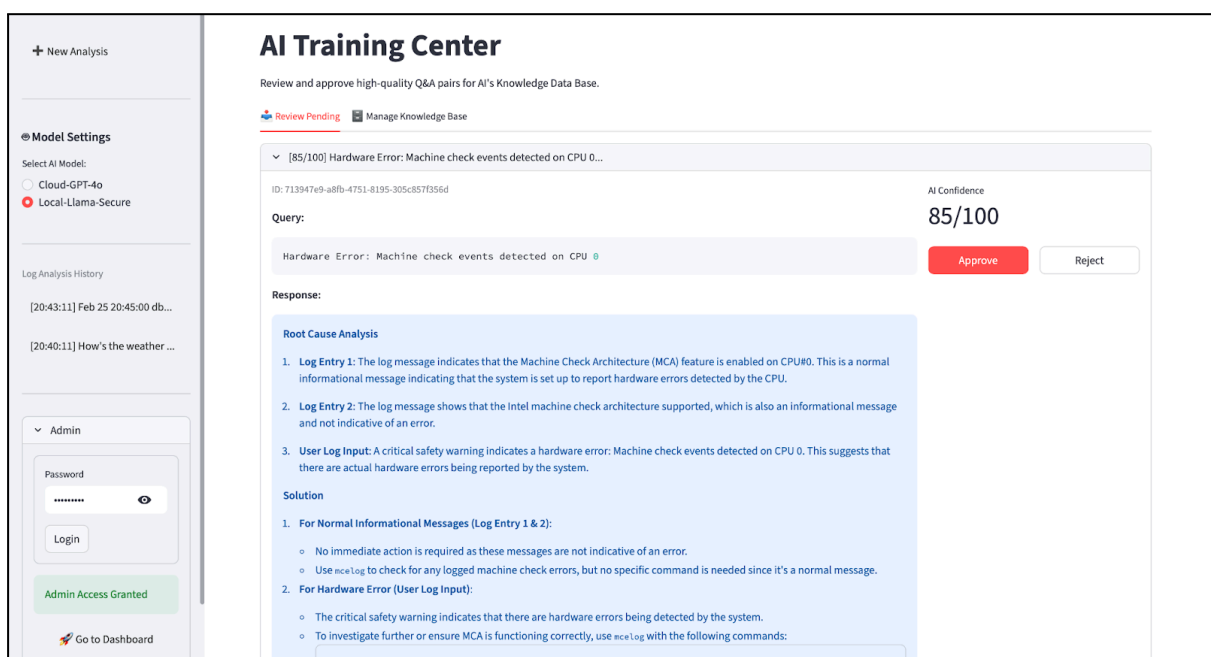
또한, RAG 시스템의 가장 큰 취약점인 환각 현상을 차단하기 위해 '임계값(Distance Threshold) 필터링'과 '프롬프트 내 조건부 제어'를 결합한 2중 방어 체계를 구현하였다. 하드웨어 도메인과 무관한 질의("How's the weather today?")를 입력한 결과, 벡터 검색 단계에서 1차로 관련 로그를 차단(No directly related logs found)하고, 언어 모델이 억지로 답변을 생성하는 대신 "Invalid input" 에러 메시지를 반환하며 시스템의 안전성을 완벽하게 방어해 내었다.



[Figure 3] Two-Stage Hallucination Defense Triggered by Irrelevant Conversational Query

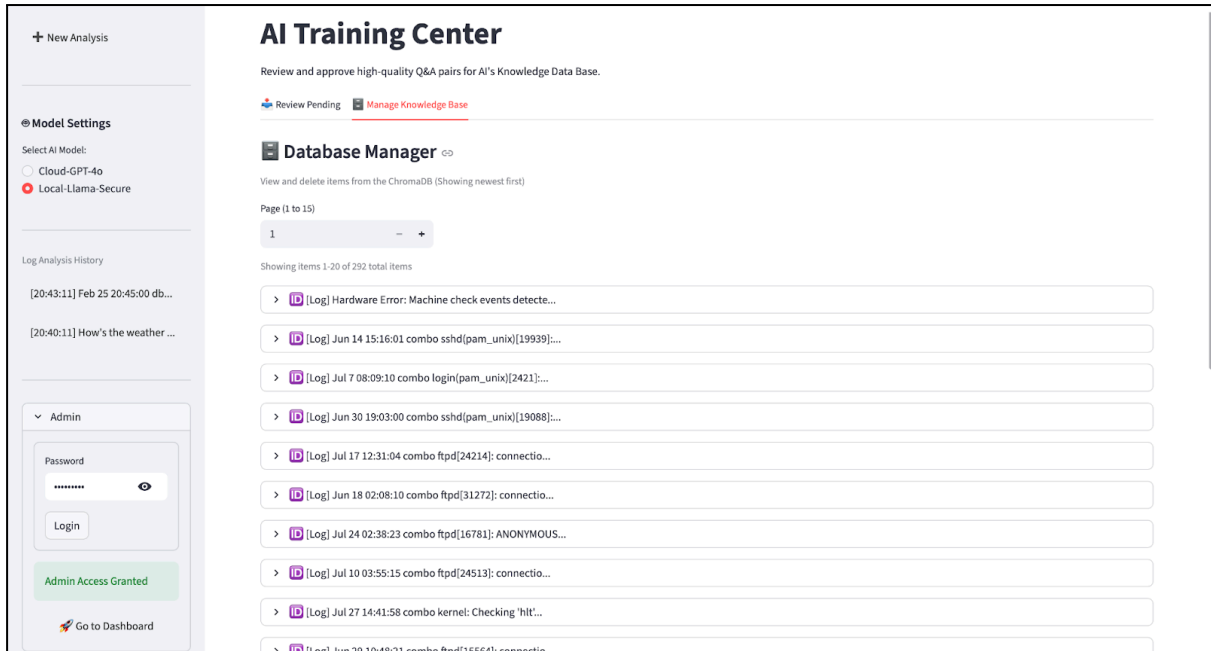
4.2. MLOps 기반 Data Flywheel(데이터 선순환) 동작 검증

에이전트가 현업에 투입된 후 스스로 지식을 확장해 나가는 '데이터 플라이휠(Data Flywheel)' 파이프라인의 실시간 동작을 검증하였다. AI가 생성한 분석 결과 중 AI Confidence 점수(예: 85/100)를 충족하는 고품질 데이터는 관리자 대시보드의 대기열(Review Pending)로 자동 이관된다.



[Figure 4] Human-in-the-Loop MLOps: Admin Dashboard for Pending Q&A Review

시니어 엔지니어 권한(Admin)으로 접속하여 해당 Q&A 쌍을 검토하고 '승인(Approve)' 처리를 수행한 결과, 시스템 재부팅이나 파운데이션 모델의 추가 학습(Fine-tuning) 없이 즉각적으로 해당 내용이 벡터화되어 로컬 ChromaDB의 신규 지식 자산으로 업데이트되는 것을 확인하였다. 이를 통해 시니어 엔지니어의 내재적 지식이 시스템의 영구적인 자산으로 축적되는 실무형 MLOps 사이클을 성공적으로 구현하였다.



[Figure 5] Real-time Knowledge Base Update in Local Vector Database (ChromaDB)

4.3. 비용 및 보안성 비교: 하이브리드 라우팅 정당성 증명

전체 에러 로그의 95% 이상을 로컬 소형 언어 모델(Llama-3.1-8B)로 처리하는 하이브리드 라우팅 전략의 타당성을 증명하기 위해, 동일한 로그에 대한 클라우드 거대 언어 모델(GPT-4o)과 로컬 파인튜닝 SLM의 출력 결과를 정성적으로 비교하였다.

Result

> Analyzed Log Data (Click to expand)

Retrieved Context

Context

[Log]
Jul 27 14:41:59 combo kernel: PCI: Invalid ACPI-PCI IRQ routing table

[Solution]
1. **Root Cause**: The log message indicates that there is an issue with the ACPI (Advanced Configuration and Power Interface) configuration related to PCI (Peripheral Component Interconnect) IRQ (Interrupt Request) routing. This typically occurs due to a BIOS or firmware issue where the system is unable to correctly interpret the IRQ routing table.

2. **Solution/Command**:
- **Update BIOS/Firmware**: Check for and apply any available updates for your system's BIOS or firmware, as manufacturers often release updates to fix such compatibility issues.
- **Kernel Parameter**: If updating the BIOS is not possible or does not resolve the issue, try adding the `pci=noacpi` kernel parameter to your boot configuration to bypass ACPI for PCI IRQ routing. You can do this by editing the GRUB configuration file (usually located at `/etc/default/grub`) and adding `pci=noacpi` to the `GRUB_CMDLINE_LINUX` line. After editing, update GRUB with:
```bash  
sudo update-grub

**AI Analysis**

**Root Cause:**  
The log message from your system indicates an issue with the ACPI (Advanced Configuration and Power Interface) configuration related to PCI (Peripheral Component Interconnect) IRQ (Interrupt Request) routing. This typically occurs due to a BIOS or firmware issue where the system is unable to correctly interpret the IRQ routing table.

**Solution/Command:**

- Update BIOS/Firmware:**
  - Check for and apply any available updates for your system's BIOS or firmware. Manufacturers often release updates to fix compatibility issues, including those related to ACPI and PCI IRQ routing.
- Kernel Parameter:**
  - If updating the BIOS is not possible or does not resolve the issue, you can try adding the `pci=noacpi` kernel parameter to your boot configuration to bypass ACPI for PCI IRQ routing. This can be done by editing the GRUB configuration file (usually located at `/etc/default/grub`) and adding `pci=noacpi` to the `GRUB_CMDLINE_LINUX` line. After editing, update GRUB with:  
```bash  
sudo update-grub
 - Reboot:** Reboot the system to apply the changes.
- Consult Documentation:**
 - Refer to the motherboard or system documentation for any specific settings related to ACPI and PCI configuration.

Note: If the context of the system operation appears normal and there are no apparent issues with PCI devices, the log message might not indicate a critical problem. However, using the

Result

> Analyzed Log Data (Click to expand)

Retrieved Context

Context

[Log]
Jul 27 14:41:59 combo kernel: PCI: Invalid ACPI-PCI IRQ routing table

[Solution]
1. **Root Cause**: The log message indicates that there is an issue with the ACPI (Advanced Configuration and Power Interface) configuration related to PCI (Peripheral Component Interconnect) IRQ (Interrupt Request) routing. This typically occurs due to a BIOS or firmware issue where the system is unable to correctly interpret the IRQ routing table.

2. **Solution/Command**:
- **Update BIOS/Firmware**: Check for and apply any available updates for your system's BIOS or firmware, as manufacturers often release updates to fix such compatibility issues.
- **Kernel Parameter**: If updating the BIOS is not possible or does not resolve the issue, try adding the `pci=noacpi` kernel parameter to your boot configuration to bypass ACPI for PCI IRQ routing. You can do this by editing the GRUB configuration file (usually located at `/etc/default/grub`) and adding `pci=noacpi` to the `GRUB_CMDLINE_LINUX` line. After editing, update GRUB with:
```bash  
sudo update-grub

**AI Analysis**

The provided logs indicate issues related to ACPI-PCI IRQ routing tables in a system's kernel logs.

**Log 1 Analysis**

- Root Cause:** The log message indicates that there is an issue with the ACPI configuration related to PCI IRQ routing. This typically occurs due to a BIOS or firmware issue where the system is unable to correctly interpret the IRQ routing table.
- Solution/Command:**
  - Update BIOS/Firmware:** Check for and apply any available updates for your system's BIOS or firmware, as manufacturers often release updates to fix such compatibility issues.
  - Kernel Parameter:** If updating the BIOS is not possible or does not resolve the issue, try adding the `pci=noacpi` kernel parameter to bypass ACPI for PCI IRQ routing.

**Log 2 Analysis**

**Root Cause:** The log message indicates that the ACPI tables do not contain any PCI IRQ routing information.

... (Skipped) ...

Here's an example code snippet in Bash:

```
#!/bin/bash

Check for updates of BIOS or firmware
if [-f "/path/to/bios/update.sh"]; then
 echo "Updating BIOS..."
 /path/to/bios/update.sh
fi

Add pci=noacpi kernel parameter if updating is not possible
echo "Adding pci=noacpi kernel parameter..."
sudo sed -i 's/GRUB_CMDLINE_LINUX_DEFAULT=".*"/GRUB_CMDLINE_LINUX_DEFAULT="pci=noacpi\nGRUB_CMDLINE_LINUX_DEFAULT="/' /etc/default/grub
sudo update-grub
```

This script checks for updates of the BIOS or firmware and applies them if available. If updating is not possible, it adds the `pci=noacpi` kernel parameter to bypass ACPI for PCI IRQ routing.

[Figure 6] Side-by-Side Qualitative Comparison: Cloud-GPT-4o vs. Fine-tuned Local-Llama-Secure

비교 결과, 두 모델 모두 `pci=noacpi` 파라미터 적용 및 `sudo update-grub` 실행이라는 완벽하게 동일한 엔지니어링 결론을 도출해 내었다. 오히려 도메인 특화 데이터로 파인튜닝된 로컬 SLM은 말로만 설명하는 범용 모델(GPT-4o)과 달리, 현업 엔지니어가 즉시 복사하여 사용할 수 있는 자동화 Bash 스크립트 로직까지 선제적으로 생성해 내는 뛰어난 실무 적용성을 보여주었다. 결과적으로, 일상적인 로그 분석 업무를 성능이 검증된 오픈프레이스 SLM으로 라우팅함으로써 **일일 막대한 클라우드 API 통신 비용을**

절감하였으며, 신규 칩셋 및 서버 아키텍처 정보 등 **일급 기밀 데이터의 외부 유출 위험을 원천 차단하는 데 성공하였다.**

## 5. Conclusion & Future Work

### 5.1. 결론 및 프로젝트 의의

본 프로젝트는 대규모 하드웨어 평가 환경에서 발생하는 난해한 시스템 에러 로그를 분석하기 위해, 폐쇄망 환경에서 작동하는 '하이브리드 RAG 기반 로컬 SLM 에이전트'를 성공적으로 구축하였다.

도메인 특화 합성 데이터로 파인튜닝된 소형 언어 모델(Llama-3.1-8B)은 RAG 아키텍처와 결합하여, 클라우드 거대 모델(GPT-4o)에 필적하는 정확도와 구조화된 분석 리포트(RCA) 생성 능력을 입증하였다. 특히, 사용자 질의와 벡터 DB 간의 임계값(Distance Threshold) 필터링 및 프롬프트 제어를 통한 '2단계 환각 방어'를 구현하여 AI의 신뢰성을 극대화하였다.

비즈니스 측면에서는 전체 에러 로그의 95% 이상을 차지하는 기지(Known) 패턴을 무비용의 온프레미스 로컬 모델로 라우팅함으로써, 막대한 API 통신 비용을 절감함과 동시에 차세대 하드웨어 스펙 등 **일급 기밀 데이터의 외부 유출 위험을 원천 차단하는 실질적인 임팩트를 증명하였다.** 더불어, 모델의 재학습(Fine-tuning) 없이 관리자의 승인(Approve)만으로 시스템의 지식이 영구적으로 업데이트되는 '**Data Flywheel**' MLOps 파이프라인을 성공적으로 실증하였다.

### 5.2. 한계점 및 향후 발전 방향

본 시스템은 뛰어난 성능을 보였으나, 개발 및 테스트 과정에서 벡터 임베딩 모델(bge-m3)의 구조적 한계인 '**오탐지(False Positive)**' 현상을 확인하였다. 임베딩 벡터 간의 거리에만 의존할 경우, 의미가 완전히 다른 로그(예: 물리적인 '하드웨어 부품 장치 불량' 에러와 단순한 '소프트웨어 파일 읽기' 에러)임에도 불구하고 리눅스 커널 로그 특유의 문법적 유사성 때문에 관련 문맥으로 잘못 검색해 오는 엣지 케이스(Edge Case)가 존재한다. 향후 이를 고도화하기 위해 다음과 같은 파이프라인 확장을 고려해보아야 한다.

- '**리랭커(Re-ranker)**'를 통한 **2단계 정밀 검색 도입**: 현재 적용된 1차 검색(벡터 임베딩) 방식은 검색 속도가 빠르다는 장점이 있으나, 표면적인 문법 구조가 유사한 다른 도메인의 로그를 오인할 위험성이 존재한다. 이를 해결하기 위해, 1차로 추출된 후보 문서들을 AI가 다시 한번 정밀하게 분석하여 사용자 질의와의 실질적인 '문맥적 연관성'을 재평가(Re-ranking)하는 2단계 검증 시스템(2-Stage Retrieval)을 도입함으로써 오탐지율을 0%에 가깝게 최소화한다.
- **이미지와 그래프까지 분석하는 멀티모달(Multi-modal) AI로 확장**: 현재 구축된 시스템은 '텍스트' 형태의 에러 로그 분석에 국한되어 있다. 그러나 실제 하드웨어 불량 분석 실무에서는 텍스트뿐만 아니라 시각적 데이터의 종합적인 판단이 필수적이다. 향후에는 시스템 장애 시 발생하는 온도 및 전압 변화 차트(시계열 데이터)나 커널 패닉 시의 메모리 덤프 화면(이미지) 등

다양한 형태의 데이터를 텍스트 로그와 교차 분석할 수 있는 멀티모달(Multi-modal) SLM 기반의 에이전트로 고도화해야 한다.

### 5.3. 대규모 하드웨어 평가 인프라 실무 적용 비전

신규 반도체 및 시스템의 품질 검증과 대규모 서버 인프라 평가 과정에서는 매일 수백만 건의 난해한 기계어 로그가 발생하며, 이를 해석하고 트러블슈팅하는 데 시니어 엔지니어들의 막대한 리소스가 소모된다.

본 프로젝트에서 구축한 지능형 에이전트를 실제 기업의 하드웨어 평가 인프라에 도입한다면, 주니어 엔지니어도 시스템에 축적된 시니어 엔지니어의 노하우(Data Flywheel)를 활용하여 즉각적이고 정확한 불량 원인 분석(RCA)을 수행할 수 있을 것으로 예상된다. 이는 단순 반복적인 에러 로그 해석에 소요되는 시간을 획기적으로 단축함으로써, 엔지니어링 조직 전체가 차세대 하드웨어 개발, 시스템 아키텍처 최적화 및 근본적인 수율 개선 등 더 높은 부가가치를 창출하는 핵심 업무에 집중할 수 있는 실질적인 업무 혁신을 이끌어 낼 것이다.