LLM(Large Language Model)을 사용한 AI 챗봇 연구

팀 명: **ForPaw**

부산대학교 **정보컴퓨터공학부**

201924475 박재홍
202155595 이한홍
202155590 이종일

지도교수: 김호원

목차

1. 과제 배경 및 목적

1.1 과제 배경

최근 대형 언어 모델(LLM)은 자연어 처리(NLP) 분야에서 혁신적인 발전을 이루어 다양한 애플리케이션에서 활용되고 있다.

LLM 은 대규모의 텍스트 데이터를 학습하여 언어의 의미와 맥락을 이해할 수 있으며, 이를 바탕으로 고도로 정교한 텍스트 분석 및 생성 기능을 제공한다. 이러한 LLM 의 발전은 로그 데이터 분석 및 운영 모니터링 분야에서도 많은 가능성을 열어주고 있다.

전통적으로 로그 데이터 분석은 주로 패턴 매칭 및 정해진 규칙에 기반하여 수행되었다.

이러한 방법은 특정 패턴이나 규칙을 사전에 정의해야 하기 때문에 새로운 유형의 공격이나 비정상적인 활동을 탐지하는 데 한계가 있다.

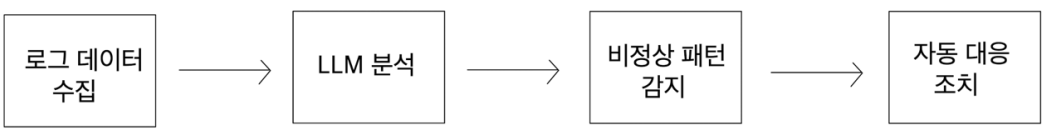
[그림 1] 기존 로그 데이터 분석 시스템

특히, 클라우드 환경에서는 대규모의 분산 시스템에서 생성되는 다양한 형태의 로그 데이터를 실시간으로 분석하고 대응하는 것이 매우 중요한데, 기존 방법으로는 이처럼 복잡하고 방대한 로그 데이터를 효과적으로 처리하기 어렵다.

이와 같은 문제를 해결하기 위해, LLM 을 도입해서 개선할 수 있다.

LLM 은 비정형 로그 데이터를 이해하고, 데이터 간의 복잡한 상관 관계를 인식하는 데 뛰어난 성능을 보이는데, 기존의 패턴 매칭 방식보다 훨씬 더 유연하고 강력한 방법으로 로그 데이터를 분석할 수 있다.

이를 통해 로그 데이터를 자동으로 가공하고, 비정상적인 활동이나 오류를 실시간으로 탐지하여 운영 효율성을 크게 향상시킬 수 있다.



[그림 2] LLM 기반 로그 데이터 분석 시스템

이러한 특징을 활용하여 LLM 은 다양한 로그 분석 시나리오에 적용될 수 있다.

예를 들어, 비정상적인 로그인 시도를 탐지하거나, 서비스 성능 저하를 예측하고, 시스템 오류를 자동으로 진단하는 등 다양한 용도로 활용할 수 있다.

1.2 과제 목표

첫 번째 목표는 LLM 을 활용하여 로그 데이터를 실시간으로 분석하는 시스템을 개발하는 것이다.

이 시스템은 클라우드 환경에서 생성되는 방대한 로그 데이터를 자동으로 수집하고 전처리하여, LLM 을 통해 비정상적인 패턴과 이상 징후를 탐지할 수 있어야 한다.

두 번째 목표는 LLM 의 분석 결과를 기반으로 자동 대응 시스템을 구축하는 것이다.

이 시스템은 비정상적인 활동이 탐지되었을 때, 사전 정의된 규칙에 따라 자동으로 조치를 취할 수 있어야 한다.

2. 요구사항 분석

2.1 기능적 요구사항

2.1.1 LLM 기반 로그 데이터 분석 시스템 개발

(1) 로그 데이터 수집 및 통합

다양한 클라우드 서비스(AWS EC2, AWS Lambda, Logstash 등)에서 생성되는 로그 데이터를 자동으로 수집하고, AWS CloudWatch Logs 와 S3 를 활용하여 로그 데이터를 통합 저장하며, 로그 데이터의 형식과 구조를 표준화하여 일관된 분석이 가능하도록 전처리한다.

(2) LLM 모델 학습 및 최적화

다양한 비정상적인 활동 패턴(비정상적인 로그인 시도, 데이터 유출 시도 등)을 포함한 로그 데이터를 사용하여 LLM 을 학습시키고, 학습된 모델을 클라우드 환경과 운영 환경에 맞게 최적화하여 높은 정확도로 비정상적인 패턴을 탐지할 수 있도록 개선한다.

(3) 실시간 분석 시스템 구현

실시간으로 로그 데이터를 분석하여 비정상적인 활동을 탐지하는 시스템을 구축하고, LLM 의 예측 결과를 기반으로 실시간 경고 및 알림을 생성하여 관리자에게 전달한다.

(4) 데이터 시각화 및 보고

분석 결과를 시각화하여 관리자에게 제공하고, 대시보드를 통해 실시간으로 시스템 상태를 모니터링할 수 있도록 한다..

2.1.2 자동 대응 시스템 구축

(1) 자동화된 경고 시스템

비정상적인 활동이 탐지되었을 때, 즉시 시스템 관리자에게 경고를 전송하는 기능을 구현하며, 이메일, SMS, Slack 등 다양한 알림 채널을 지원하여 관리자에게 신속하게 정보를 전달한다.

(2) 자동 대응

다양한 비정상적인 활동 유형에 대한 대응 규칙을 정의하고, 규칙 기반의 자동화된 대응 조치를 구현한다.

예를 들어, 다수의 로그인 실패 시 해당 IP 를 일시적으로 차단하거나, 데이터 유출이 의심될 경우 해당 계정을 잠금 처리하며, 규칙 기반의 자동화된 대응 조치를 구현함.

(3) AWS Lambda 와의 통합

AWS Lambda 를 사용하여 비정상적인 활동이 탐지되었을 때 자동으로 스크립트를 실행한다.

예를 들어, 특정 조건이 충족되면 Lambda 함수를 호출하여 해당 IP 를 AWS WAF 에서 차단하거나, 인스턴스를 중지하는 등의 조치를 수행함.

2.2 비기능적 요구사항

(1) 직관적인 UI 를 통해 분석한 결과값을 쉽게 볼 수 있어야 한다.

(2) 로그 분석은 실시간성이 중요하므로, 짧은 주기로 로그들을 배치로 LLM 에 전달해야한다.

(3) 모든 로그 데이터를 분석해서 보여줄 필요는 없고, 개발 혹 운영에 필요한 데이터만 클라이언트에 제공한다.

3.개발 환경 및 사용 기술

3.1 개발 환경

(1) 개발 도구: VScode, IntelliJ ultimate

(2) 사용 언어: Python, Java 17

3.2 사용 기술

(1) 서버 프레임워크: Spring Boot, FastAPI

(2) 데이터베이스: MySQL(관계형 저장소), Milvus (백터 저장소)

(3) 캐시: Redis

(4) AI : PyTorch, LangChain, Transformers

(5) 클라우드: AWS EC2, AWS S3, AWS CloudWatch, AWS Lambda

(6) 배포: Docker

(7) 로그 수집 도구: Logstash

(8) 화면(UI): React, NextJS

3.3 사용 LLM 모델

로깅 시스템에서 고사양 LLM 을 활용하는 것은 매우 중요하다.
로그 데이터 분석은 실시간으로 비정상적인 패턴을 탐지하고, 신속한 대응을 필요로 하기 때문에, 높은 성능의 LLM 을 사용하여 시스템의 효율성을 극대화해야 한다.

LLaMA 3 8B 모델은 HuggingFace 라이브러리를 통해 쉽게 파인튜닝하고 실행할 수 있기 때문에, 우선 Llama3 를 사용한다. HuggingFace 의 트랜스포머 라이브러리를 사용하여 로컬 환경에서 모델을 파인튜닝하고, 클라우드 환경에 배포한다.

만약 테스트를 해보고 로깅을 분석하기에 성능이 충분치 않다면, 파인튜닝이 가능한 다른 HuggingFace 모델을 사용한다.

4. 개발 일정 및 역할 분담

4.1 개발 일정

<table>
<tr><th colspan="5">6 월</th><th colspan="4">7 월</th><th colspan="5">8 월</th><th colspan="4">9 월</th></tr>
<tr><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>1</th><th>2</th><th>3</th><th>4</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td colspan="5" rowspan="2">요구 사항 분석 후
초기 기획
및 아키텍쳐 설계</td><td colspan="4" rowspan="2"></td><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td></tr>
<tr><td colspan="4" rowspan="2">파인튜닝에 사용할
데이터 수집</td><td colspan="2" rowspan="2"></td><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="2" rowspan="2">중간
보고서
작성</td><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="5" rowspan="2">LLM 프롬프트 및
시스템
개발</td><td colspan="4" rowspan="2"></td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="4" rowspan="2">사용자 인터페이스(
개발</td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="3" rowspan="2">테스트 및
수정</td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="4" rowspan="2"></td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="2" rowspan="2"></td><td colspan="2" rowspan="2">배포</td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="4" rowspan="2"></td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="2" rowspan="2"></td><td colspan="2" rowspan="2">최종
보고서
및
발표 준비</td></tr>
<tr><td colspan="5" rowspan="2"></td><td colspan="4" rowspan="2"></td><td colspan="4" rowspan="2"></td></tr>
</table>

4.2 역할 분담

이종일	- 어플리케이션 및 UI 개발 - AI 학습을 위한 데이터 수집
이한홍	- 서비스 아키텍쳐 설계 - 로그 수집/가공/활용을 위한 Spring Boot 서버 개발 - 랭체인을 사용한 FastAPI 서버 개발
박재홍	- AI 학습을 위한 데이터 수집 - 수집한 데이터를 바탕으로 파인튜닝