LLM 기반 이상탐지
구축 가이드

작성자: 박계영

2024. 05.

목 차(Contents)

요약

1. 개요

(1) 프로젝트 배경 및 목적

- LLM(대규모언어모델, Large Language Model)이란
  - 대규모 데이터셋을 통해 데이터 패턴을 인식하여 대화, 이미지, 동영상, 음악 등 새로운 콘텐츠를 생성할 수 있는 인공지능
  - 문서요약[1], 복잡한 논리 수학문제[2,3], 의료[4], 로봇 공학[5], 시계열 데이터 예측[6] 등 복잡한 task 에서 좋은 성능을 보여주고 있음.
- LLM 장점
  - Versatility - 로그(log)데이터와 시계열(timeseries) 데이터, 이미지 등 다양한 데이터 도메인에서 활용 가능 [7,8,9,10,11,12,13]
  - 온프레미스(On-premise) - Pretrained model 을 활용하여 망분리환경에서 온프레미스 형태로 구축하여 성능 개선 가능[14]
  - 해석가능성(Interpretability) - Anomaly Data(이상데이터)에 대한 판별 근거 생성 가능[7,11]
- 프로젝트 목적
  - 생성형 AI 기반 이상탐지는 학계에서 연구되는 단계이며, 실제 금융 FDS 에 적용된 사례는 아직 없음.
  - 현재 고성능 GPU 확보가 어렵고, 고객 개인정보 이슈 및 망분리 환경을 고려했을 때 현 파일럿 프로젝트에서 개발하기 제한됨.
  - 따라서 **생성형 AI 기반 FDS 구축 가이드라인을 작성하기로 결정**

(2) 가이드라인 구성

- 구축 프로세스 단계로 순차적으로 구성

  - 1 장(개요): 프로젝트 배경 및 목적, 핵심 기술용어(이상탐지, LLM) 소개

  - 2 장(LLM 모델 선정): LLM 모델 선정 고려사항 및 로컬에서 사용가능한 대표적인 Open-LLM 모델 분석

  - 3 장(데이터 전처리): 입력 데이터를 전처리(pre-processing)하는 방법을 선행연구 파악

  - 4 장(LLM 성능 개선): 사용자 도메인에서 LLM 성능을 추가 개선시키기 위한 방법들 소개

  - 5 장(성능 평가): LLM 기반 이상탐지 시스템을 수동, 자동으로 성능 평가하는 방법 소개

- 6, 7 장(데모 및 도입사례): 금융분야 생성형 AI 도입사례 소개 및 LLM 기반 신용카드 이상거래 탐지 데모 시연

그림 1 LLM 기반 이상탐지 구축 가이드라인 구성

(3) 이상탐지(Anomaly Detection)

- 정상데이터에서 벗어난 이상데이터 패턴 혹은 이상 값을 찾아내는 기술
- 인공지능(AI) 기술을 활용하여 금융, 보안, 제조, 통신, 의학 등 다양한 산업 분야에서 활용 가능
- (금융분야) 신용카드 거래, 대출 실행 간 발생하는 로그 데이터 등을 분석하여 고객이나 직원에 의한 이상 거래를 탐지할 수 있음

(4) 대규모언어모델(LLM, Large Language Model)

- Transformer 신경망 모델 [15]
  - 2017 년 구글이 개발한 딥러닝 알고리즘으로, Encoder-Decoder 기반 구조로 데이터 순서가 어떻게 연결되는지 감지하고 대규모 텍스트 블록을 처리하고 문맥을 파악하는 기능을 가져 자연어처리(Natural Language Processing) 분야에서 기존모델(RNN, CNN) 대비 추론 성능을 획기적으로 개선
- 대규모언어모델(LLM, Large Language Model) 소개
  - Transformer 신경망 모델로 기반으로 구성되어[7] 매우 방대한 양의 데이터를 학습시켜 새로운 컨텐츠를 생성할 수 있도록 만들어진 언어 모델(Language Model)
  - LLM 은 패턴 인식 능력을 통해 일반적인 자연어 추론, 검색 엔진, 의료, 로봇 공학 및 코드 생성과 같은 다양한 task 에 대해 새로운 가능성을 열어주고 있음.
  - 기존 딥러닝 모델보다 다양한 Task 를 처리할 수 있지만, HW 구축비용 (On-premise)이나 솔루션 사용료(Cloud) 등 많은 예산이 필요.
  - 10 억(1B)개의 작은 사이즈 모델부터 최근 조(Trillion)단위 크기 모델이 등장하고 있음
  - 대표적 모델로 ChatGPT(OpenAI), Gemini(Google)[16], Llama2(Meta)[14] 등이 있음.

그림 2 대규모언어모델(LLM)

특징	Customized Model (기존 딥러닝 모델)	대규모언어모델(LLM)
구축 비용	최소 NVIDIA RTX 4070Ti 1 대 약 200 만원	최소 A100 GPU 1 대 약 3000 만원 Or GPT4 기업구독료 월 8 만원(1 인기준)
활용 범위	Log, Image 분석 등 특정 Task 및 데이터에 적합	다양한 Task 및 데이터 활용 가능 컨텐츠 생성 능력

2. LLM 모델 선정

(1) 모델 선정 고려사항

- 온프레미스(On-premise) 모델 vs 클라우드(Cloud) 모델
  - 온프레미스 모델은 기업 내부에 자체 구축하는 형태를 의미하며 내부에 구축하기 때문에 데이터 유출 염려가 없어 높은 보안수준 달성할 수 있음. 반면 고성능 GPU 와 서버 등 인프라 구축을 위한 초기 HW 비용이 높음
  - 클라우드 모델은 GPT-4, Copilot 과 같이 테크기업의 AI 솔루션을 클라우드 형태로 사용하는 것으로, 초기 투자 비용이 낮으며 고성능 솔루션을 사용할 수 있다는 장점이 있음. 반면 중요 데이터가 외부로 유출될 위험이 존재하고, 상용 제품에 대한 의존도가 높아 장기적으로 사용량에 비례해 비용이 증대되는 단점이 존재함.

특징	온프레미스(On-premise)	클라우드(Cloud)
비용	A100 GPU 가격: 약 3000 만원 이외 서버, RAM 비용 발생	초기 투자비용 낮고 사용한 토큰 혹은 월 구독료 비례하여 지불
성능	보유한 HW 스펙에 따라 성능이 달라질 수 있음.	OpenAI 의 GPT-4 와 같이 고성능 모델 사용 가능

데이터 보호	데이터 외부 유출 염려가 없어 높은 보안 수준 달성 가능	데이터 유출 가능성 존재
서비스 안정성	자체 서버 운영으로 리스크 최소화	API 의존성이 높아 안정성 떨어짐
LLM 모델	Llama2[14], Gemma	GPT-4, Copilot, Gemini[16]

- 모델 크기에 따른 HW 요구사항 (Llama2 기준)[14]

HW 요구사항	**7B**	**13B**	**70B**
최소 GPU VRAM	28GB	52GB	280GB
권장 GPU	1 X NVIDIA A100G	1 X NVIDIA A100G	4 X NVIDIA A100G
권장 RAM	1TB	1TB	1TB 서버 2 대

- 데이터 특성
  - Task 에 적합한 **언어와 데이터 유형(Text, Image, Video)을 고려**하여 모델 선택 필요

Text Generation	소설	NovelAI, AI Dungeon
Chat Related	Chatbot	ChatGPT
	Language Model	GPT-4, Llama2, Gemini, Claude, Solar
Image/Video	Image	Midjourney, DALLE, Stable Diffusion
	Video	Stable Video

(2) Open-LLM 분석

- **Llama2** [14]
  - Meta 에서 개발한 오픈소스 LLM 모델로, 라이선스 제한없이 상업적으로 사용할 수 있으며 Meta 공식 홈페이지를 통해 Llama2 모델 다운로드 허가를 받은 후, 모델 사용 가능
  - 7B, 13B, 70B 의 세 가지 크기로 나뉘어져 있으며, 모델 학습에 사용된 토큰(Token)개수는 약 2 조 개, A100-80GB 그래픽 카드 1 장 기준 7B, 13B, 70B pre-trained 모델을 만들기 위한 학습시간 합이 330 만 시간 분량

- 13B 모델의 경우 다른 LLM 30B, 40B 모델과 비슷한 추론 성능을 보이는 것으로 알려짐
- Meta's Research Super Cluster (NVIDIA A100s), GPU 최대 2000 개 사용
- 70B 모델의 경우 GPU A100 80GB 1 개 기준 1720320 시간 학습
- GPU 2000 개 풀 가동 시 35 일 정도 소모되며, 서버 비용만 대략 500 억

- **Solar** [17]
  - 국내 AI 스타트업 업스테이지(Upstage)에서 개발한 10.7B 크기의 모델로 한국어 채팅 분야에서 좋은 성능을 나타낸 것으로 알려짐
  - 허깅페이스(HuggingFace)를 통해 오픈소스 Solar 모델 사용 가능
  - 상업적으로 사용할 경우, 라이선스(License) 검토 필요.

3. 데이터 전처리

(1) Log 데이터 사례(자세한 내용은 본문 참조)

- 소개논문: "LogGPT: Exploring ChatGPT for Log-Based Anomaly Detection"[7]
  - 로그 데이터 추출
  - 프롬프트 처리
  - 응답 결과 추출

(2) Timeseries 데이터 사례(자세한 내용은 본문 참조)

- 소개논문: "LLMTIME: Large Language Models Are Zero-Shot Time Series Forecasters"[8]
  - 토큰화(Tokenization)
  - 범위 재조정(Rescaling)
  - 표본 추출(Sampling)
  - 연속확률분포 변환(Continuous likelihood)

4. LLM 성능 개선

(1) 미세조정(Fine-tuning)

- 특정 Task 에서 성능을 향상시키기 위해 모델을 추가 학습하는 방식
  예시) 법률 분야 특화 생성형 AI 를 개발하기 위해 법률 문서 등을 추가 학습
- 추가 학습을 위한 고품질의 학습 데이터셋과 고성능 서버용 GPU 등 필요
- 높은 미세조정 기술 숙련도 필요

(2) RAG (Retrieval Augmented Generation)

- DB 에서 정보를 검색하고 검색된 데이터를 기반으로 LLM 에 입력하여 정확도와 신뢰도가 향상된 결과 생성

- RAG 를 통해 잘못된 결과를 생성하는 환각효과(hallucination) 개선 가능

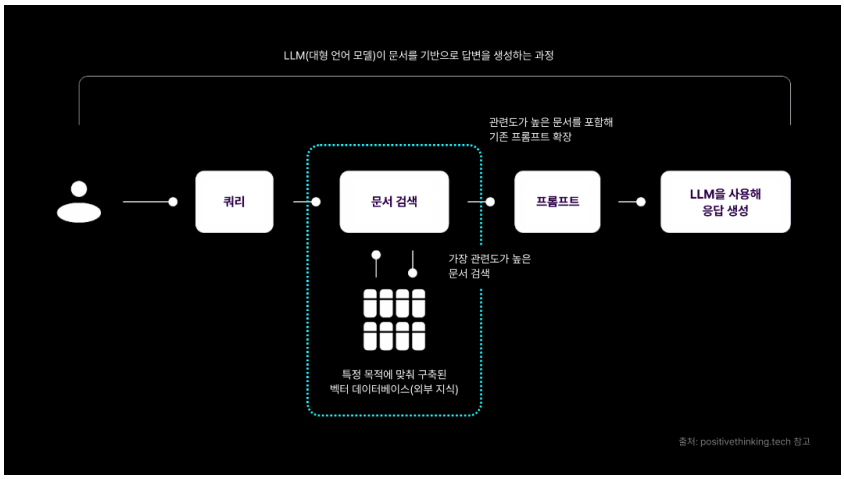


그림 3 RAG 프로세스

- RAG 구성요소
  - Indexing: 데이터 소스에서 데이터를 얻고 인덱스를 생성하는 과정
  - Retrieval: 사용자 입력을 벡터로 변환하고 입력데이터와 유사한 검색 결과(텍스트 혹은 문서)를 반환
  - Generation: 주어진 질문과 관련 정보를 결합하여 새로운 프롬프트를 생성하고 LLM 은 이 정보를 기반으로 질문에 답변

(3) 프롬프트 엔지니어링(Prompt Engineering)

- LLM 으로부터 원하는 결과를 얻기 위해 입력 프롬프트를 최적화하는 작업
- 많은 시간과 예산이 필요한 미세조정(fine-tuning)을 사용하지 않고, 프롬프트 엔지니어링만으로도 모델의 성능을 효과적으로 개선할 수 있음
- 효과적인 프롬프트 엔지니어링 원칙[28]
  - **원칙 1. 명확하고 상세한 지시/설명(instruction) 작성**
    - ✓ 구분문자(delimiter) 사용
    - ✓ 구조화된 결과(e.g. HTML, JSON)를 요구
    - ✓ 조건 만족여부 확인 요구
    - ✓ **<u>One-shot, Few-shot 프롬프트</u>**
      - ➢ LLM 이 지시된 작업을 수행하기 위해 도움이 되는 성공적인 예시 몇 가지를 제시한 이후 주어진 태스크를 수행하도록 요청하는 방식
      - ➢ 모델이 다양한 작업을 빠르고 유연하게 수행할 수 있게 하고, 적은 양의 예시를 가지고 모델이 특정 작업을 수행할 수 있도록 도와줘서 효율적인 학습을 할 수 있음. 많은 연구에서 few-shot 프롬프트 만으로도 미세조정과 견줄만큼 좋은 성능을 달성하는 경우가 많은 것으로 알려져 있음

Prompt:
예시
- 그 사람에게 반했어. 그 사람은 정말 멋지고 매력적이야: [분석] 긍정적, 열정
- 사랑이란 건 정말 복잡해. 때로는 행복하고 때로는 아프다: [분석] 복합적, 갈등
- 우리가 헤어진 후, 나는 사랑이란 더 이상 믿을 것이 못된다고 느껴: [분석] 부정적, 실망감

지시
앞선 예시와 같이, 다음 문장을 분석해줘
- 사랑은 때로는 어려움을 안겨주지만, 그 어려움을 함께 극복하는 것이 중요한거야:

- **원칙** 2. 모델에게 생각할 시간 주기
  - ✓ 태스크를 완수하기 위해 필요한 단계 특정
  - ✓ 원하는 특정 포맷의 결과를 요청
  - ✓ 결론에 도달하기 전에 모델 스스로 답을 내리도록 지시

(4) 한국어 추론능력

- Meta Llama2 모델 학습 데이터 중 한국어 데이터 0.02%에 불과하여 영어에 비해 한국어 능력이 상대적으로 떨어짐
- 한국어로 미세조정(Fine-tuning)을 통해 한국어 능력 개선 가능하지만, 추가 학습을 위해 정제된 한국어 데이터와 기술 전문가 필요
- 대안으로 Open Ko-LLM 리더보드에 랭크된 LLM 모델을 사용하면 한국어 능력을 일정 부분 개선 가능.

  - Open Ko-LLM 리더보드는 한국어 기반으로 추론, 언어이해, 일반상식, 환각효과 방지 능력을 평가하여 상위 랭크 모델 선별
  - URL: https://huggingface.co/spaces/upstage/open-ko-llm-leaderboard
- 한계: 리더보드는 평가 방법이 변화되면 순위가 크게 달라질 수 있으며, 실제 환경에서 모델 적용성이 떨어질 가능성이 있기 때문에 리더보드 순위는 참고하는 용도로만 사용하는 것이 적절함

5. 성능 평가

(1) 이상유무(정상/비정상) 탐지 성능평가

- 정답지 역할을 할 수 있는 (Y/N)라벨링 처리된 테스트 데이터를 통해 이상탐지 성능 평가

이상탐지 분류 유형		실제 정답	
		True	False
분류 결과	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

- $Precision = \frac{TP}{TP+FP}$
- $Recall = \frac{TP}{TP+FN}$
- $F1score = 2 \times \frac{Precision \times Recall}{Precision+Recall}$

(2) 사람에 의한 수동평가(Human Evaluation)

- 사례 1) LogPrompt[11]
  - 성능 평가자 선정 기준
    - ✓ 탑티어 ICT&SW 회사와 협업하여 6 명의 숙련된 전문가 선정
    - ✓ 이들 전문가는 분산시스템, 모바일 OS 등 O&M(Operation& Management) 분야 10 년 이상의 경력자로 구성
  - 성능평가 방법
    - ✓ LLM 의 결과물 200 개 랜덤으로 추출하여 평가
    - ✓ 평가 기준은 가독성(Readability) 및 유용성(Usefulness) 항목 별 5 단계로 평가
    - ✓ 평가점수 평균과 4 점보다 높은 결과의 비율 계산하여 기존 평가방법들과 비교

표 1 사람에 의한 평가(Human Evaluation) 기준 [11]

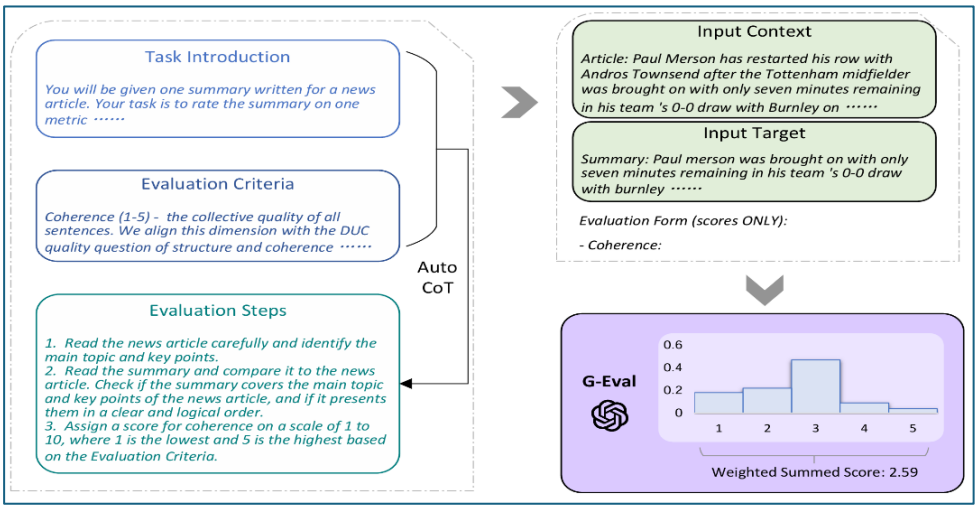
점수	정보 유용성(Usefulness)	가독성(Readability)
1	단순 예측 레이블 이상의 이상징후에 대한 판단 근거가 없음	텍스트에 이해할 수 없는 요소나 문법적 오류가 많이 포함되어 있음
2	예측의 정당성이 사실과 다르거나 논리적으로 일치하지 않는 경우.	대부분 읽을 수 있지만, 문법 오류나 불명확한 문구가 있을 수 있음
3	판단근거가 예측을 잘 뒷받침하지만, 명확성과 세부 사항이 부족할 수 있음	문법 오류가 거의 없지만, 일부 용어는 수정이 필요할 수 있음
4	구체적이고 정확하며 관련성 있는 판단근거가 제시되어, 엔지니어가 잘못된 알람을 제거하고 추가 분석을 수행하는 데 도움을 줌	명확하고 문법적으로 정확하며, 최소한의 기술 용어만 수정 필요가 있을 수 있음
5	상세하고 관련성이 있으며 명확한 근거를 제시하여, 엔지니어가 잘못된 경보를 배제하고 근본원인을 찾는데 상당한 도움을 줌	명확하고 상세하며 문법적으로 완벽하고 소프트웨어 엔지니어링에 대한 전문성을 갖추고 있음

- 사례 2) Med-PaLM2 [4]
  - 구글(Google)이 개발한 의료용 생성형 AI
  - 의학적 질문에 대한 답을 내놓고, 건강 데이터 정리 문서 요약 등의 작업을 수행 가능하며 현재는 일부 의료기관에서 시범적으로 활용되고 있음 [30]

- 성능 평가자 선정 기준
  - ✓ 15 명의 의사와 6 명의 일반인으로 구성
  - ✓ 의사는 미국 국적 6 명, 영국 국적 4 명, 인도 국적 5 명이며 전공 분야는 일반 진료, 내과, 심장학, 호흡기, 소아과 및 외과로 다양하게 구성
  - ✓ 일반인은 의료 전문지식이 없는 4 명의 여성과 2 명의 남성, 나이는 18 살~44 살로 구성.
- 성능평가 방법
  - ✓ 동일한 문제에 대해 의사, Med-PaLM1, Med-PaLM2 가 답변한 데이터를 보고 의사, 일반인이 평가
  - ✓ 평가자는 해당 답변이 의사가 한 것인지 LLM 이 한 것인지 모르는 비공개 상태에서 평가
  - ✓ 평가 기준은 질문 의도(intent) 파악정도와 유용성(Helpfulness)
  - ✓ 또한 공통 질문에 대해 의사, Med-PaLM1, Med-PaLM2 가 답변한 결과를 놓고 선호도 순위 평가

(3) LLM 을 활용한 자동평가(LLM based Evaluation)

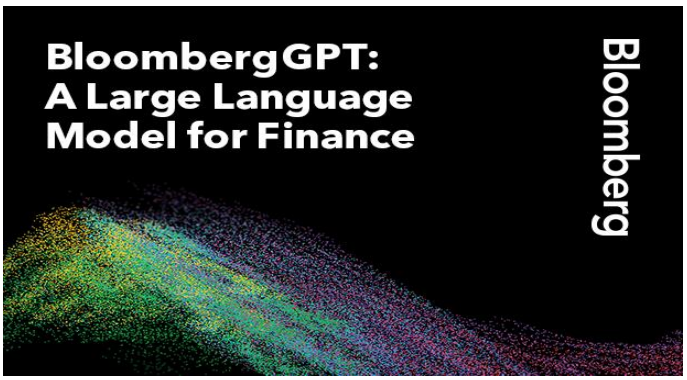
- LLM 평가는 사람에 의한 평가(Human Evaluation)가 효과적이지만, 시간과 인적 자원 측면에서 많은 비용이 발생하는 단점 존재
- LLM 이 생성한 결과물을 다른 LLM 모델을 통해 자동으로 평가하는 방법이 활발하게 연구되고 있음



- G-Eval [19]
  - 현재 추론성능이 가장 뛰어난 것으로 알려진 GPT-4 를 활용하여 자동으로 LLM 성능을 평가하는 방법으로, LLM 성능평가 시 가장 많이 활용됨
  - 기존 LLM 평가 방법들에 비해 Human evaluation 과 연관성이 가장 높으며, 특히 대화나 창의성을 요구하는 task 에서 좋은 성능을 보임.
  - G-Eval 파이썬 실행스크립트 https://github.com/nlpyang/geval

6. 도입사례

(1) BloombergGPT[20]



- 블룸버그(Bloomberg)에서 금융 분야 특화하여 개발한 대규모언어모델(LLM)
- 금융, 재무 데이터를 분석해 위험을 평가, 회계와 감사 작업을 자동화
- 벤치마크 테스트에서 다른 모델 이상의 성능을 유지하며 특히 재무 관련 작업에서 기존 대규모언어모델보다 성능이 뛰어남
- 53 일간 64 대의 서버와 NVIDIA A100 GPU 가 사용되었으며, 개발 비용은 대략 270 만 달러로 추정

7. 데모

(1) 시나리오

- 신용카드 거래내역을 LLM 에 입력하여 이상거래 여부를 탐지하고, 탐지성능 평가
- RAG(Retrieval Augmented Generation), 프롬프트 엔지니어링(Prompt Engineering) 기법 적용
- CPU vs GPU 추론 시간 비교
- 0-Shot vs RAG(1-shot, 2-shot) 정확도 비교

(2) 데이터셋

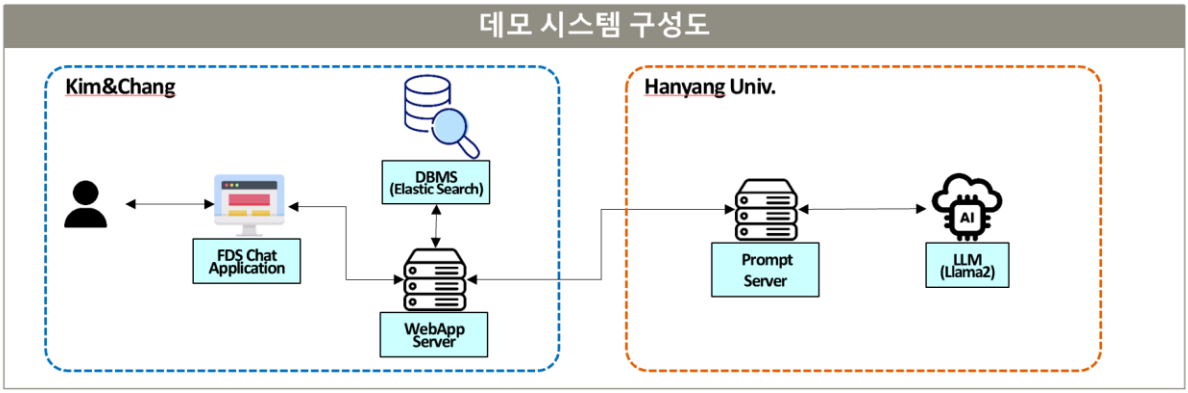
- Kaggle Credit card fraud detection dataset 2023
  - 유럽 카드 이용자의 암호화된 신용카드 거래내역 데이터
  - 550,000 개 거래내역으로 구성 (324.8MB)
  - https://www.kaggle.com/datasets/nelgiriyewithana/credit-card-fraud-detection-dataset-2023

(3) LLM 모델 및 성능 개선 방법

- Llama2 (제작사: Meta)
- 데이터 전처리(Pre-processing)
  - 데이터 정밀도: 입력 데이터의 소수점을 4 자리로 제한하여 LLM 의 숫자인식 시 발생할 수 있는 정확도 저하 문제 해결.
  - Feature 선택: 중요한 특성만을 골라내어 효율적인 데이터 구조화 진행.

- 프롬프트 엔지니어링(Promp-Engineering)
  - 컨텍스트 제공: 사전 지식을 바탕으로 한 컨텍스트 정보를 통해 LLM 에 정확한 분석 지시.
  - 질의 명확화: 명확하고 구체적인 프롬프트 구성으로 LLM 의 응답 방향성 제어.

(4) 데모 시스템 구성도



(5) 데모 결과(상세 내용은 동영상 참고)

- CPU vs GPU 추론시간 비교

구분	CPU	GPU
스펙	Intel(R) Core(TM) i9-9940X @ 3.30GHz, 14 core Memory: 56GB	GPU: A100 텐서 코어, CUDA cores 6912 VMemory: 80GB
1 초당 출력 토큰수(t/s)	2 t/s (1,000 개 토큰 출력시 500 초)	27 t/s (1,000 개 토큰 출력시 37 초)

- 0-shot vs RAG(1-shot, 2-shot) 탐지 정확도 비교
  - 2,000 개(Normal 1,000 개, Anormal 1,000 개) 무작위 샘플링 데이터로 0-Shot 과 RAG 를 활용한 탐지 정확도 측정

구분	탐지 성공	탐지 실패	미응답	정답률
0-Shot	975	837	152	49%
1-Shot(RAG)	1442	521	37	72.1%
2-Shot(RAG)	1505	471	24	75.2%

1. 개요

(1) 프로젝트 배경 및 목적

- 생성형 AI 는 사용자의 요구에 따라 텍스트, 이미지, 음성 등 다양한 형태의 창작물을 생성하는 인공지능을 지칭한다. 생성형 AI 는 이전 기술의 확산 패턴과는 다르게 매우 빠른 속도로 확산되고 있다. 2022 년 11 월 ChatGPT 가 일반에 공개된 이후에 약 두 달 만에 사용자 약 1 억 명에 도달했고, 이는 역사상 가장 빠른 속도로 확산된 애플리케이션으로 기록되고 있다. 그 이후로도 생성형 AI 는 비약적인 발전을 거듭하고 있으며, 이를 기반으로 한 새로운 비즈니스 도구와 다양한 산업에서 성공적인 활용 사례를 다수 만들어내고 있다. 예를 들어 구글이 개발한 생성형 AI 인 PaLM2 를 기반으로 의료분야에 특화시킨 Med-PaLM2 는 인간 의사와 비교할만하거나 더 좋은 성과를 보여준다는 연구결과가 공개되었다[4]. 연구 결과에 따르면 Med-PaLM2 가 제공한 1000 개 이상의 의료 질문에 대한 답변을 검토한 의사와 일반인들은 9 가지 평가 범주 중 8 개 범주에서 의사가 생성한 답변보다 Med-PaLM2 가 생성한 답변을 선호하는 것으로 나타났다. 또한 OpenAI 에서 개발한 GPT4 는 미국 의사시험에서 90% 이상의 정답률을 기록하여 인간보다 높은 점수를 받은 것으로 알려졌다.

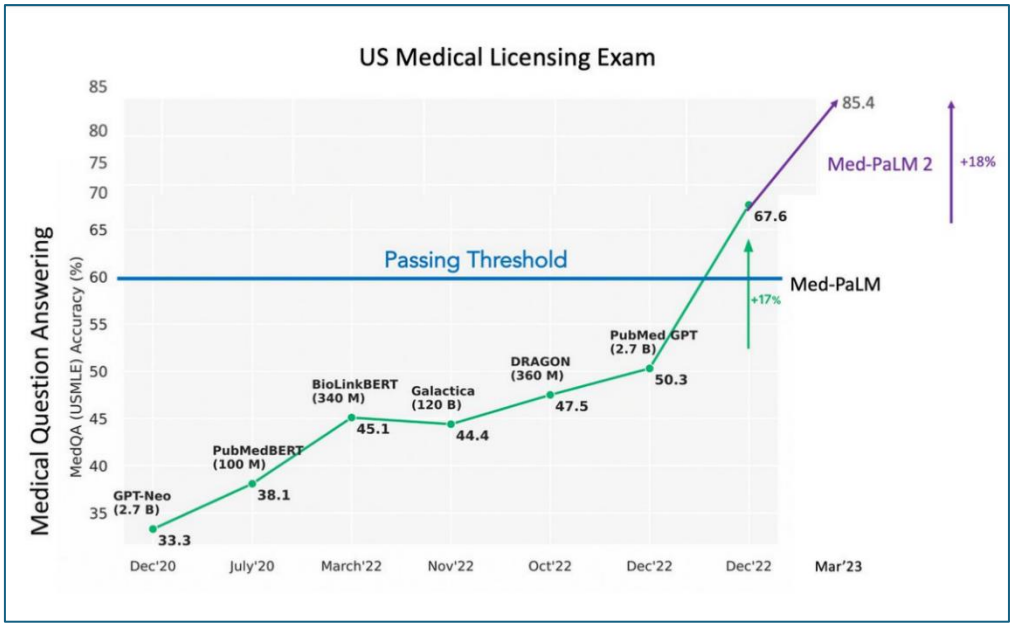


그림 4 Med-PaLM2 美 의사면허 시험 점수 [4]

이렇게 생성형 AI 의 등장은 인간의 전유물로 여겨졌던 추론, 컨텐츠 창작 영역에서 AI 가 인간과 버금갈만한 혹은 그 이상의 탁월한 성능을 발휘할 수 있음을 보여주며, 이는 모든 산업 분야에서 혁신을 가져오고 있다.

최근 생성형 AI 연구를 통해 FDS(Fraud Detection System)를 위한 데이터 이상탐지(anomaly detection) 분야에서도 우수한 성능을 보여주고 있다. 생성형 AI 를 FDS 에 적용했을 때 단일 데이터 도메인에 국한되지 않고 다양한 형태의

데이터에 대해 이상을 탐지할 수 있으며, 이상 데이터에 대한 판별 근거를 생성할 수 있어 전문가의 의사결정을 지원할 수 있다. 또한, 온프레미스(On-premise) 형태로 활용하여 데이터 및 네트워크 망분리와 같은 제약이 있는 환경에서도 모델성능을 향상시킬 수 있다.

그러나 현재 생성형 AI 기반 이상탐지 기술은 학계에서 연구되는 단계로, 실제 금융환경에 적용된 사례가 없다. 또한 고성능 GPU 를 확보하는 것이 어렵고 금융 망분리 규제 환경 등 여러 제약조건들을 고려할 때, 파일럿 프로젝트에서 당장 개발하는 것은 제한된다고 판단하였다. 따라서 LLM 기반 이상탐지 구축 가이드라인을 작성하기로 한다.

(2) 가이드라인 구성

- 본 가이드라인은 LLM 기반 이상탐지 시스템을 구축하기 위한 프로세스로 순차적으로 구성된다.

  1 장에서는 프로젝트의 배경과 목적을 상세히 서술한다. 또한 이상탐지와 LLM 에 대한 핵심 용어와 개념을 설명하여 전체적인 프로젝트 이해를 돕는다. 그리고 2 장에서 사용자 환경에 적합한 LLM 모델 선정을 위한 고려사항을 살펴보고, 오픈소스로 공개되어 사용자가 로컬에서 사용가능한 대표적인 Open-LLM 두 모델을 분석한다. 3 장에서는 LLM 입력 데이터를 전처리(pre-processing)하는 방법을 선행연구를 통해 살펴본다. 4 장에서는 LLM 의 성능을 향상시키기 위한 다양한 전략들을 소개한다. 이를 통해 구축된 이상탐지 시스템의 정확도와 효율성을 높일 수 있다. 5 장에서는 구축한 LLM 기반 이상탐지 시스템을 수동, 자동으로 성능 평가하는 방법에 대해 알아본다. 마지막으로 6,7 장에선 금융분야 생성형 AI 도입사례를 소개하고 실제 LLM 기반 신용카드 이상거래를 탐지 데모를 소개하고 시연한다.

그림 5 LLM 기반 이상탐지 구축 가이드라인 구성

(3) 이상탐지(Anomaly Detection)

- 이상 탐지(Anomaly Detection)란 정상데이터에서 벗어난 이상데이터 패턴 혹은 이상값을 찾아내는 것을 의미한다. 이상탐지 기술은 보안, 금융, 제조, 통신, 의학 등 다양한 산업 분야에서 활용이 가능하다. 이상 탐지에 사용되는

데이터는 특별한 제한은 없으며 탐지하고자 하는 목표, 대상이 무엇인지에 따라 다양한 데이터를 사용할 수 있다. 예를 들어 금융 분야는 신용카드, 대출 실행 간 발생하는 거래 로그 등을 분석하여 고객이나 직원에 의한 이상 거래를 탐지할 수 있다.

이상탐지 유형은 학습데이터에 대한 정상 혹은 이상 판별정보를 기준으로 분류된다. 정상 혹은 이상 판별정보를 라벨이라 하며, 라벨 유무에 따라 이상탐지 유형을 지도 학습, 준지도 학습, 그리고 비지도 학습 기반으로 분류한다. 지도 학습 기반 이상탐지(Supervised anomaly detection) 방법은 가장 보편적으로 사용되는 방법으로, 정상 데이터와 이상 데이터가 구분된 라벨 정보를 확보하여 이를 학습에 사용하는 방법이다. 비지도 학습 방법에 비해 높은 정확도를 보일 수 있다는 장점이 있다. 하지만 이상값으로 분류된 데이터를 다량으로 확보하여 전처리 하는 과정에서 많은 인력과 비용이 필요하며, 라벨링 작업을 통해 모든 유형의 이상치를 포착하는 것이 어렵다는 한계가 있다.

비지도 학습 기반 이상탐지(Unsupervised anomaly detection) 방법은 데이터의 정상 혹은 이상값을 가진 데이터가 주어지지 않은 상태에서 활용할 수 있는 방법으로, 대부분의 데이터가 정상일 것으로 가정하고 학습을 진행한다. 이러한 접근을 따르는 알고리즘은 일반적으로 라벨 정보를 사용하는 지도 학습에 비해 성능이 높지 않다는 한계를 갖지만, 이상치가 정상 데이터 대비 발생 빈도가 훨씬 적은 데이터 불균형 상황에서도 사용 가능하다는 장점이 있다.

준지도 학습 기반 이상탐지(Semi-supervised anomaly detection) 방법은 부분적으로 라벨이 지정된 데이터를 사용하여 감지 성능을 개선하고, 라벨이 없는 데이터를 활용하여 표현 학습을 촉진한다. 즉 지도 학습을 부분적으로 활용하여 감지 성능을 향상시키면서, 비지도 학습을 부분적으로 활용하여 알려지지 않은 유형의 이상치도 탐지하는 것을 목표로 한다. 일반적으로 준지도 학습은 약한 지도(Weakly supervised learning)에서의 불완전한 라벨 학습을 의미하며, 구체적인 예로 준지도학습 기반 이상탐지 알고리즘인 GANomaly[32]는 정상 데이터만을 학습한 후 훈련 과정에서 이미 학습한 정상 표현과 다른 특성을 갖는 이상치를 식별하는 방식으로 동작한다.

(4) 대규모 언어 모델(Large Language Model, LLM)

- Transformer 신경망 모델 소개

  기존의 자연어 처리 딥러닝 모델은 문장 의미를 파악하기 위해 순차적인 처리 방식을 사용했다. 하지만 순차적 처리 방식은 연산 속도가 느리고 병목현상이 발생하는 등의 한계가 있다. 이러한 한계를 개선하여 문장 의미를 더 잘파악하기 위해 Transformer 신경망 모델이 개발되었다.

Transformer 신경망 모델은 2017 년 구글이 개발한 딥러닝 알고리즘으로, Self-Attention Mechanism 을 도입하여 입력 시퀀스 내의 단어들 간 상호작용을 계산하여 각 단어의 유의성(attention score)을 학습한다. 이를 통해 문장 내 중요 정보에 집중하고, 불필요한 정보를 제거할 수 있다. 그리고 병렬 처리가 가능하여 연산 속도가 빠르며, 입력 시퀀스의 길이에 영향을 받지 않는 등의 장점이 있다. 시퀀스 요소가 어떻게 연결되는지 감지하고 개별 단어와 구문뿐 아니라 대규모 텍스트 블록을 처리하고 문맥을 파악하는 기능을 가진다. 이러한 기능을 통해 Transformer 모델은 자연어처리(Natural Language Processing) 분야에서 기존모델(RNN, CNN) 대비 처리 성능을 획기적으로 개선하였다.

- 대규모언어모델(LLM, Large Language Model) 소개
  Transformer Decoder 부분을 기반으로 구성되었으며, 대량의 데이터셋으로 학습되어 자연어를 이해하고 새로운 컨텐츠를 생성하는 기능을 가진 언어 모델이다.
  LLM 은 강력한 추론 능력과 텍스트 생성 능력을 바탕으로 복잡한 작업을 해결할 수 있고 텍스트, 이미지, 음성 데이터 등 다양한 데이터에 적용 가능한 versatility 특성을 가지고 있다.
  LLM 은 학습 과정에서 막대한 구축 비용이 들기 때문에 일반인이 모델을 처음부터 구축하기는 쉽지 않다. 그러나 Meta 의 Llama 와 같은 오픈소스 모델이 대중에게 공개되어 산업계와 학계에서 이를 이용한 LLM 연구 개발이 활발히 이루어지고 있다. 또한, 비즈니스 분야에서 LLM 을 활용하고자 하는 기업들도 오픈소스 형태의 Open-LLM 을 통해 private LLM 을 구축할 수 있다.

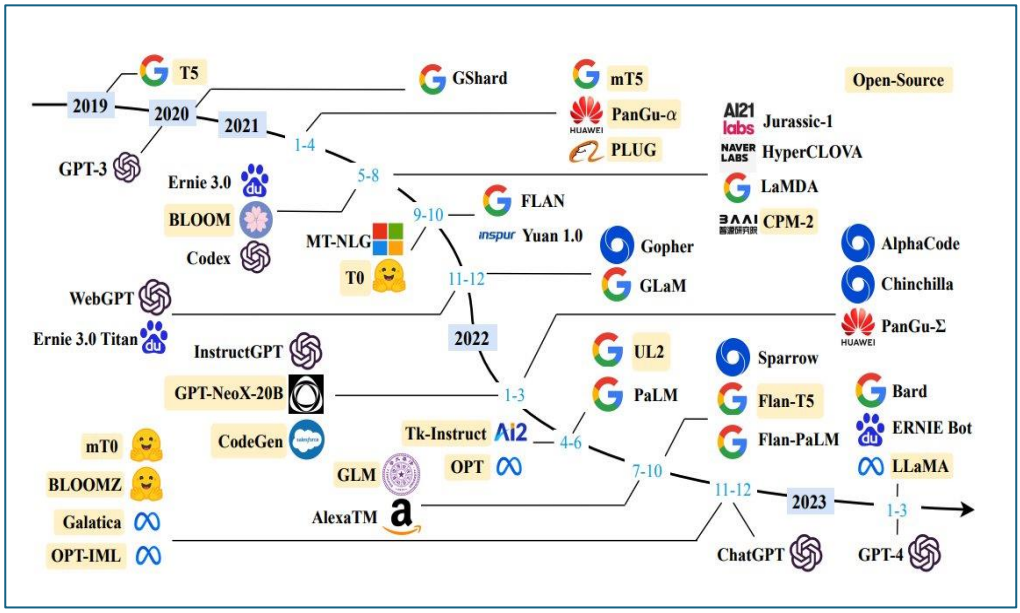


그림 6 대규모언어모델(LLM) 역사와 주요 모델

LLM 은 주어진 한글 문장의 단어들 사이의 유사성과 문맥 형성을 파악하고, 다음에 올 단어를 예측하여 생성할 수 있다. 예를 들어, "나는 오늘 밥을 먹었다"라는 문장이 주어졌을 때, LLM 은 "나는 오늘" 다음에 "무엇을"이 올 수 있는지를 판단하고, "밥을"이라는 단어를 가장 가능성이 높은 단어로 예측하여 "먹었다"라는 동사와 함께 완전한 문장을 생성할 수 있다. 이러한 강력한 컨텐츠 생성 기능을 바탕으로, LLM 은 다양한 분야에서 활용되고 있다. 예를 들어, 광고 캠페인에서는 LLM 을 활용하여 소비자의 행동 패턴을 분석하고 해당 소비자에게 맞춤형 광고를 생성할 수 있다. 또한, 문서 요약 분야에서는 LLM 을 사용하여 긴 문서를 간결하게 요약하거나, 자동 번역 시스템에서는 LLM 을 활용하여 더 자연스러운 번역을 제공할 수 있다. 또한 LLM 은 새로운 이미지를 생성할 수도 있다. 명령어를 입력하면 AI 가 이를 이해한 뒤 그림으로 만들어주는 방식이다. 사용자가 프롬프트를 입력하면 그에 맞는 이미지를 생성하고, 사용자는 이를 바탕으로 자신이 원하는 이미지를 선택, 편집할 수 있다. 이미지 생성 AI 의 장점은 복잡한 자연어 프롬프트를 이해해 사람이 그린 것 같은 정교한 이미지를 생성하는 것이다. 이미지 생성 AI 는 짧은 단어에 그치지 않고 긴 문장까지 소화하며 이를 구체적인 이미지로 표현한다. 생성된 이미지에 추가적인 프롬프트를 입 력해 더욱 적합한 이미지를 찾을 수 있다.

그림 7 AI 가 그린 그림(美 콜로라도 박람회 미술대회 우승작)

최근에는 LLM 을 활용하여 이상데이터를 탐지하려는 연구들이 이루어지고 있다[6,7,8,9]. 예를 들어, 금융 기관에서는 LLM 을 이용하여 거래 내역 데이터를 분석하여 사기 행위를 탐지할 수 있다. LLM 은 대규모의 거래 기록을 학습하여 정상적인 거래 패턴을 이해하고, 이상 거래 패턴을 감지할 수 있다. 예를 들어, 일반적인 거래 패턴과 다르게 대량의 이체나 이상한 시간대에 거래가 이루어지는 경우에는 LLM 이 이를 감지하고 해당 거래를 이상으로 표시할 수 있다. 또한, LLM 은 이상 탐지 결과에 대한 설명을 제공할 수 있다. 예를 들어, 이상한 거래 패턴이 감지되었을 때, LLM 은 그 이유에 대한 설명을 제공하여 왜 해당 거래가 이상으로 분류되었는지를 설명할 수 있다. 이는 이상탐지 결과를

해석하는 데 있어서 매우 유용하다. 이러한 기술적인 발전으로 인해 LLM 을 활용한 이상데이터 탐지 기술은 다양한 분야에서 활용되고 있으며, 앞으로 금융, 보안, 제조업 등에서의 적용 가능성이 높아지고 있다.

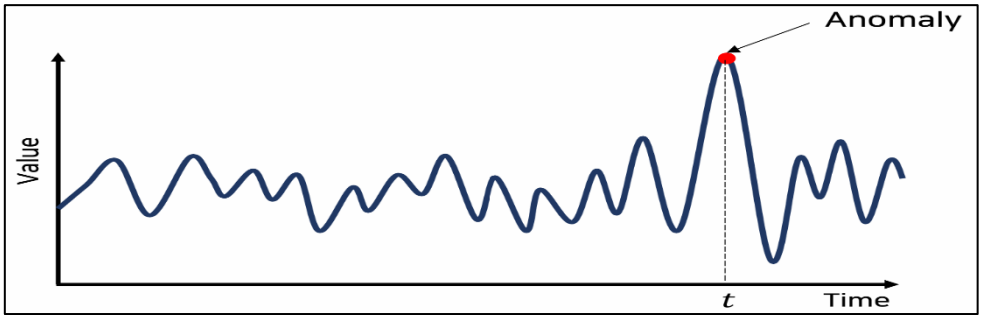


그림 8 이상탐지(Anomaly Detection)

- Customized 모델과 대규모언어모델(LLM)

  Customized 모델은 LLM 이전에 기존 딥러닝 모델을 의미한다. Customized 모델은 특정 Task 수행을 위해 개발되어, LLM 에 비해 모델 크기가 작고 학습에 필요한 데이터의 양도 상대적으로 적다. 또한 저렴한 IT 인프라 구축 비용으로 모델 구현이 가능한 특징이 있다. 그러나 로그 이상탐지를 위한 모델(Deeplog[25])은 로그 데이터만 다루도록 개발된 것처럼, 수행목적이 되는 Task 에 따라 세부 알고리즘 설계가 달라진다. 그래서 새로운 환경이나 데이터에서 사용될 경우 개별적으로 패턴인식 Feasibility 확인이 필요하다. 또한 다양한 데이터를 종합적으로 고려하여 설계한 모델이 아니므로 각 데이터에 맞는 전처리 과정이 추가로 필요하고, joint training 및 앙상블(ensemble)을 위한 학습 전략에 대한 연구가 필요하다.

  분야별 대표적인 Customized 딥러닝 모델
  - Log 분야: DeepLog[25], LogAnomaly, PLELog, AnoGAN
  - Timeseries 분야: Anomaly Tranformer
  - Semi-supervised 방식: DeepSVDD[26]

  반면 대규모언어모델(LLM)은 방대한 데이터셋으로 학습되어 기본 모델을 활용하여 언어, 이미지 및 비디오 생성, 이상탐지 등 다양한 분야에 유연하게 활용 가능하다. 모델 개발을 위해 데이터 수집 및 학습을 진행 해야하는 Customized 모델과는 다르게, LLM 은 기술 기업에서 배포한 기본 모델들이 존재하여 이러한 모델을 활용하면 사용자가 모델 개발을 위해 처음부터 데이터 학습을 진행할 필요가 없다.

  또한 기본 모델에다가 선별된 데이터로 추가 학습시키는 미세조정(fine-tuning) 과정이나 RAG(Retrieval Augmented Generation)와 Prompt-Engineering 과 같은

성능을 높이기 위한 기술들과 결합하여 사용자가 원하는 다운스트림 (downstream) 작업에 특화하여 적용할 수 있다.

표 2 Customized Model vs LLM

특징	Customized Model (기존 딥러닝 모델)	대규모언어모델(LLM)
구축 비용	최소 NVIDIA RTX 4070Ti GPU 1 대 약 200 만원(가격변동 가능)	최소 A100 GPU 약 3000 만원 GPT4 기업구독료 월 8 만원 (1 인기준)
활용 범위	Log, Image 분석 등 특정 Task 및 데이터에 적합	다양한 Task 및 데이터 활용 가능 컨텐츠 생성 능력
구축 난이도	Task 맞춤형으로 개발이 필요해서 전문성 요구	기본 모델 구축은 쉽지만, 이후 미세조정 단계에서 전문성 요구

LLM 은 외부에 소스코드가 공개되지 않은 폐쇄형 LLM 모델과 오픈소스로 공개된 Open-LLM 이 있다. OpenLLM 은 Meta 의 Llama2[14]모델이 있다. Llama2 는 사용자들이 다운로드 및 상업적으로 이용할 수 있도록 공개된 모델이다. 또한 프랑스 업체 Mistral AI 가 개발한 LLM 모델인 Mistral 도 OpenLLM 이며, 내부 작동을 지원하는 수치 매개변수인 '가중치'(weights)를 공개하여 사용자들이 추가 모델을 개발하기 용이하도록 했다. 이외에도 AI 플랫폼인 허깅페이스(Hugging Face)에는 다양한 Open-LLM 이 공개되어 있어 사용자들은 자신이 원하는 모델을 손쉽게 다운받고 배포할 수 있다.
그러나 Open-LLM 모델을 활용하여 조직 내부에 자체적으로 구축할 경우, 모델학습과 추론 과정에서 고성능 GPU 와 대용량 RAM 이 탑재된 서버가 요구되어 초기 투자비용이 많이 필요하다.
반면, Open AI 의 GPT, Google 의 Gemini[16] 등은 대표적인 폐쇄형 모델로 사용자들은 해당 모델을 임의로 구축 및 수정할 수 없고 서비스 이용량에 비례하여 비용을 지불하여야 한다. OpenAI 의 GPT4 모델을 개인적으로 클라우드 형태로 사용할 경우 모델 미세조정 비용은 토큰 1,000 개당 $0.0004 ~ $0.0300 이며 이는 훈련에 사용할 모델 유형에 따라 달라진다.

2. LLM 모델 선정

- LLM 모델 선정 시 수행목적이 되는 Task, 보유하고 있는 HW 자원, 데이터 특성 등을 고려하여 적합한 모델을 선택해야 한다.
- 이번 장에서는 LLM 모델 선정 시 고려해야 할 사항들을 설명하고 대표적인 오픈소스 LLM 모델인 Llama2 와 Solar 에 대해 소개한다.

(1) 모델 선정 고려사항

- 온프레미스(On-premise) vs 클라우드(Cloud)

  LLM 구축 형태는 Open-LLM 을 활용하여 온프레미스 형태로 private LLM 을 구축하는 방법과 클라우드 형태로 외부 기술기업의 LLM 솔루션을 사용하는 방법으로 나뉜다. 온프레미스 형태는 private LLM 구축에 필요한 서버, GPU, 스토리지, 네트워크 등의 IT 인프라와 이를 구축할 AI 전문가가 필요하다. HW 를 확보하여 직접 구축하기 때문에 초기 비용이 높다는 단점은 있지만, IT 인프라를 직접 제어하고 데이터의 외부 유출 염려가 없어 민감한 정보를 다루거나 높은 보안 수준을 요구하는 조직에서 많이 사용된다. 또한 HuggingFace 를 통해 많은 Open-LLM 모델이 공개되어 있고 누구나 로컬 환경에서 쉽게 다운받고 배포할 수 있어 구축의 편리성이 개선되었다. Open-LLM 모델을 사용하여 비즈니스 환경에 적합한 모델을 다운로드하여 구축할 수 있어, 민감한 데이터를 다루고 있는 금융권 망분리 환경에서 사용하기 적합한 형태라고 할 수 있다.

  반면 클라우드 형태로 사용할 경우, LLM 구축에 필요한 고성능 GPU, 서버, 스토리지 등의 IT 인프라를 가상화 형태로 제공받아 쉽고 빠르게 사용할 수 있다. 가상화 자원은 사용자가 필요한 만큼 사용할 수 있고 필요한 시간에 유연하게 확장하거나 축소할 수 있다. 그러나 외부 AI 솔루션에 대한 의존도가 높아지게 되어 장기적인 관점에서 온프레미스 형태보다 더 많은 운영 비용이 발생할 수

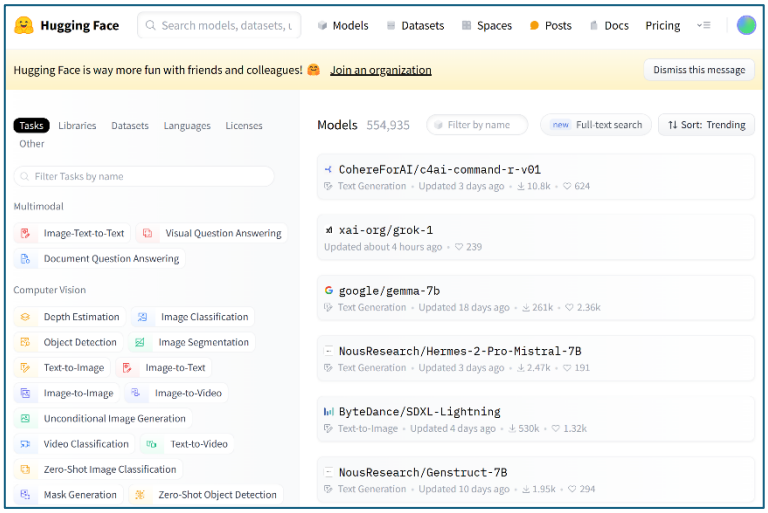


그림 9 허깅페이스(Hugging Face)에 공개된 Open-LLM

있다. 그리고 무엇보다 기업 내부 데이터가 외부로 유출될 수 있는 위험성이 존재하기 때문에 민감한 데이터를 다루거나 규제가 강한 산업 분야는 사용하기 어렵다. 금융회사는 대부분 보안으로 인해 외부 네트워크와 망분리가 되어있고 고객 데이터 유출 리스크가 있기 때문에 클라우드 형태의 LLM 을 사용하기에는 상당히 제약이 많다고 볼 수 있다.

표 3 온프레미스와 클라우드 특징 비교

특징	온프레미스(On-premise)	클라우드(Cloud)
비용	초기 투자 비용 높음. A100 GPU 가격: 약 3000 만원 이외 서버, RAM 비용 발생	초기 투자비용 낮고 사용한 토큰 혹은 월 구독 요금체계에 비례하여 지불
성능	보유한 HW 스펙에 따라 성능이 달라질 수 있음.	OpenAI 의 GPT-4 와 같이 고성능 모델 사용 가능
데이터 보호	데이터 외부 유출 염려가 없어 높은 보안 수준 달성 가능	데이터 유출 가능성 존재
해당 모델	Llama2, Mistral	GPT-4, MS Copilot

- 모델 크기에 따른 HW 요구사항

  LLM 모델 크기는 각각 상이하다. OpenAI 의 GPT3 모델은 1750 억개(175B) 이며, Meta 의 Llama2 는 7B, 13B, 70B 세가지 크기가 있다. 모델크기는 모델 성능과 사용자 편의성에 큰 영향을 미친다. 일반적으로 모델 크기가 클수록 추론 성능이 뛰어나지만 학습 및 추론에 필요한 고성능 GPU 등 HW 자원도 더 많이 필요하게 된다.

  사용자가 수행하고자 하는 데이터 크기가 작거나, 한정된 자원안에서 구축할 경우 작은 모델 크기를 선택하는 것이 좋다. 작은 모델은 'sLLM'(small Large Language Model; 소형언어모델)이라고도 불리는, sLLM 은 보통 매개변수가 60 억(6B)~100 억(10B)개로 기존 LLM 에 비해 훨씬 작음에도 불구하고 성능은 못지 않기 때문에 저비용·고효율의 강점을 가진것으로 알려져있어 사용가능한 자원의 한계가 있는 조직에게 좋은 선택지가 될 수 있다.

  또한 LLM 모델을 구축하기 위해서 모델 학습과 추론에 가장 핵심적인 역할을 하는 GPU 를 확보해야 한다. 일반적으로 모델 크기의 4 배가 되는 VRAM 을 가진 GPU 를 확보하는 것이 권장된다. 예를 들어 Llama2 7B 모델을 운영하기 위해서 최소 28GB 의 VRAM 을 갖춘 고성능 GPU 가 필요하다.

표 4 모델크기 별 HW 요구사항(Llama2 기준) [14]

HW 요구사항	7B	13B	70B

최소 GPU VRAM	28GB	52GB	280GB
권장 GPU	1 X NVIDIA A100G	1 X NVIDIA A100G	4 X NVIDIA A100G
권장 RAM	1TB	1TB	1TB 서버 2 대

- 데이터 특성

  학습 및 추론에 사용되는 데이터 형식, 언어적 특징을 고려해야 한다. 한국어 관련 Task 를 수행하고자 할 때 한국어의 비중이 높은 데이터로 학습된 LLM 모델을 선택하는 것이 유리하다. 그러나 대부분의 LLM 모델은 영어 데이터셋을 통해 학습되었기 때문에, 한국어 데이터를 입력하여 추론할 경우 성능이 저하될 수 있는 점을 고려해야 한다. 또한 텍스트, 코드, 이미지, 음성 등 다양한 데이터 형식에 따라 적합한 LLM 모델이 다르다.

  대표적인 LLM 모델은 GPT4, Gemini, DALL-E 3, Stable Diffusion, Midjourney 등이 있다. GPT4 는 OpenAI 에서 개발한 대화형 인공지능 모델로 사용자와 대화를 이어가며 자연스러운 문장을 스스로 생성하여 답변한다. Gemini 는 구글에서 개발한 대화형 인공지능 서비스로 ChatGPT 와 유사하게 사용자 입력에 맞춰 텍스트를 생성하는 모델이다. Dall-E 3, Midjourney 및 Stable Diffusion 은 텍스트를 입력 받아 이미지를 생성하는 이미지 생성 AI 이다. 이 외에도

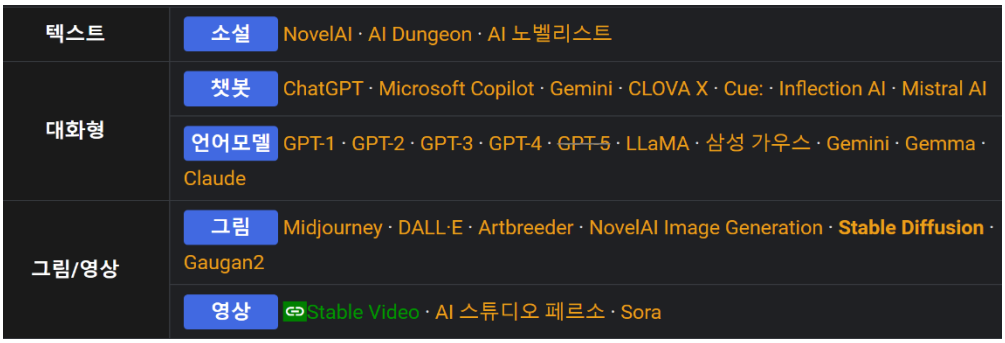


그림 10 데이터 특성에 따른 LLM 분류

음성합성을 위한 모델, 단백질 구조를 예측하는 모델 등 다양한 분야의 LLM 모델이 있다.

(2) Open-LLM 분석

- Llama2[14]

  23 년 7 월에 Meta 에 의해 공개된 Llama2 는 상업적 활용까지 가능한 오픈소스 LLM 이다. 강화학습(RLHF)과 보상 모델링을 활용하여 양방향 대화, 텍스트 생성, 요약, 질문 및 답변 등 이전 Llama1 대비 더욱 유용하고 안전한 결과물을 생성할 수 있다. LLaMA 2 는 7B, 13B, 70B 의 세 가지 크기로 나뉘어 사용자의 환경에

맞게 선택하여 사용할 수 있다. 13B 모델은 다른 30B 및 40B 모델과 유사한 성능을 제공하는 만큼 성능이 좋은 것으로 알려졌다. Llama2 를 구축하기 위해 사용된 토큰은 약 2 조 개이며 A100-80GB 그래픽 카드 1 장 기준 7B, 13B, 70B 3 개 모델의 학습시간 합이 330 만 시간 분량이다. Meta 의 Research Super Cluster (NVIDIA A100s)에서 개발이 이루어졌고 GPU 는 최대 2000 개를 사용했다. 모델 별로 학습시간이 상이하며 가장 큰 모델인 70B 모델은 GPU A100 80GB 기준으로 1720320 시간 학습하여 pretrained model 을 구축하였다. 이를 GPU 2000 개로 환산하면 풀 가동시 대략 35 일 정도 걸렸다고 볼 수 있다. GPU 서버 비용으로만 대략 500 억 정도 소모되었다고 알려져 있다. 매개변수 (parameter) 크기에 따라 모델 별 생성 완료 시간에 차이가 있을 수 있지만, Llama1 모델에 비해 정확도 향상과 유해한 텍스트 생성을 방지하는 측면이 강화되었으며 Azure 및 Windows 등의 여러 플랫폼에서도 미세조정(fine-tuning)이 가능하게 확장되어 전세계 다양한 프로젝트에 활용되고 있다.

표 5 Llama2 모델 다운로드 방법

1. Meta Llama2 공식 홈페이지 접속	meta Llama2 공식 홈페이지에 접속하면 Llama2 에 대한 상세 설명을 확인할 수 있다. (https://ai.meta.com/llama/)
2. 모델 접근 신청	Llama2 신청정보를 작성하여 모델 접근을 신청한다. https://ai.meta.com/resources/models-and-libraries/llama-downloads/
3. 접근 허가 메일 확인	모델 접근 신청하면, Meta 로부터 접근 허가 메일을 받게 된다. 짧으면 10-15 분 길면 하루정도까지 소요될 수 있다. 메일을 통해 접근 url 을 받을 수 있다.
4. Meta AI github 접속	url 를 통해 다운로드 받기 위해 Meta github 페이지에 접속한다. (https://github.com/facebookresearch/llama)
5. 모델 다운로드	모델을 다운로드 받기 위해서 아래와 같은 명령어를 실행했다. git clone https://github.com/facebookresearch/llama.git cd llamachmod 755 download.sh ./download.sh Bash 를 성공적으로 작동시키면 3 번의 접근 url 을 입력하면 다음 아래와 같은 텍스트가 나온다. Enter the list of models to download without spaces (7B,13B,70B,7B-chat,13B-chat,70B-chat), or press Enter for all. 원하는 모델 사이즈를 입력하면 해당 모델을 다운받을 수 있다.
6. 다운로드 오류	만약 모델 다운로드 과정에서 403 issue 가 발생되는 경우 새로운 접근 url 을 받은 후 다시 다운받으면 해결된다.

- Llama2 chat model
  Llama 2 Chat 모델은 Llama2 구조를 기반으로 사용자 대화 사례에 최적화되어 있다. 기본 모델과 동일하게 7B, 13B, 70B 크기의 모델이 존재한다.
  사용자 대화에 최적화된 버전이며, 100 만 개 이상의 사람 주석으로 미세 조정되어 채팅 사용 사례에서 공격적이거나 부적절한 응답과 같이 성능 격차를

식별하고 잠재적으로 문제가 될 수 있는 응답을 피하도록 개발되었다. 모델 다운로드 방법은 Llama2 와 동일하다.
(https://huggingface.co/meta-llama/Llama-2-13b-chat-hf)

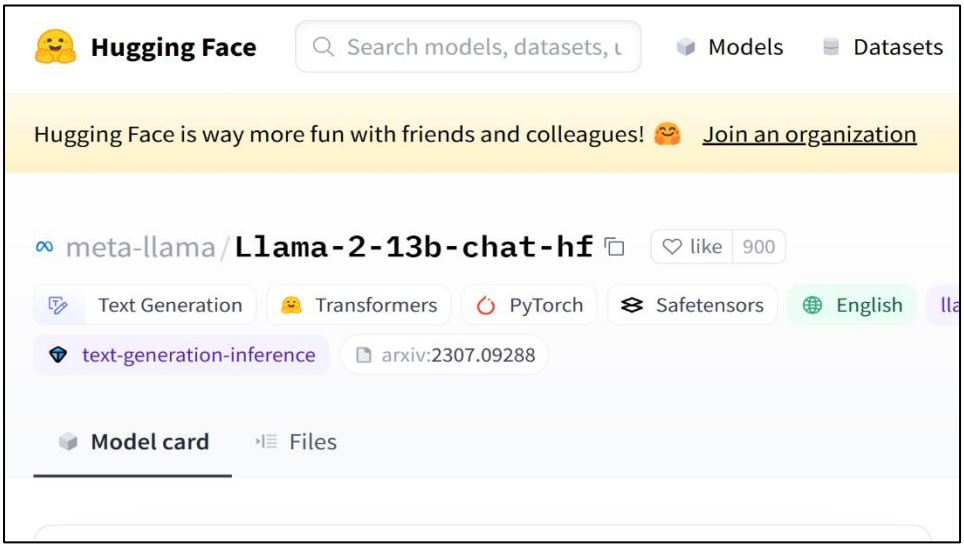


그림 11 Llama2-chat 모델

- Solar [17]
  국내 대표 AI 스타트업 업스테이지(Upstage)가 개발한 LLM 모델로 기업들이 프라이빗 LLM 으로 활용이 용이하기 위해 10.7B 의 작은 크기로 구성되었다. 성능이 좋지만 큰 13B 모델과 충분히 작지만 지적 제약이 있는 7B 모델 사이의 장점을 모두 잡는 최적의 모델 크기를 찾기 위해 오픈소스의 7B 모델들을 기반으로 자체적인 Depth Up-Scaling 방식을 적용, 레이어를 추가하며 깊이를 더해 소형 모델의 성능을 극대화한 것으로 알려졌다.

표 6 허깅페이스 모델 다운로드하는 방법

직접 다운로드	허깅페이스 해당 모델페이지 Files 탭 접근하면 관련 파일들을 확인할 수 있어서 원하는 모델 클릭하여 다운로드
CLI	허깅페이스 해당 모델 페이지의 </> Use in Transformers 클릭 로컬에 다운로드 할 수 있는 명령어 안내 예시) from transformers import AutoModelForCausalLM model = AutoModelForCausalLM.from_pretrained("meta-llama/Llama-2-13b-chat-hf")

표 7 접근 인증이 필요한 모델 다운로드 방법(Google Gemma)

Hugging Face 회원가입	접근인증이 필요한 모델의 경우 사전에 Hugging Face 회원가입을 해야한다. https://huggingface.co/
모델 라이선스 승인받기	회원 가입 후 해당 모델 Repo 로 이동(Gemma Repo) https://huggingface.co/google/gemma-2b Acknowledge license 버튼을 클릭한다. (라이센스 확인) Authorize 버튼을 클릭한다.

	스크롤을 맨 아래로 내려 필수 항목 체크 후 Accept 버튼을 클릭한다. 필수 항목: I accept the terms and conditions (약관에 동의합니다)
Huggin Face 액세스 토큰 만들기	이제 해당 모델을 사용하려면 허깅 페이스의 액세스 토큰을 입력해야 한다. 허깅 페이스의 사이트에서 우측 상단의 프로필을 누른 후, 'Settings' 메뉴 - 왼쪽의 Access Tokens 메뉴를 선택한다. Access Tokens 페이지에서 'New token' 버튼을 눌러 토큰을 생성한다. 액세스 토큰 생성 후 'Show'에서 액세스 토큰값 복사.
Hugging Face CLI 라이브러리 설치 및 로그인	pip install -U "huggingface_hub[cli]" huggingface-cli login cli 로그인 후 토큰을 입력하라는 메시지가 나오면 HuggingFace 에서 복사해둔 액세스 토큰을 입력한다. 그리고 huggingFace 의 transformer 패키지를 설치한다. pip install git+https://github.com/huggingface/transformers

3. 데이터 전처리

LLM 모델에 데이터를 입력하기 전, 각 데이터 특성에 따라 적절한 전처리(pre-procesing) 작업이 필요하다. 이번 장에서는 선행 연구 사례를 통해 Log 데이터와 Timeseries 데이터에 따른 전처리 방법을 살펴본다.

(1) Log 데이터 사례

- 소개논문: "LogGPT: Exploring ChatGPT for Log-Based Anomaly Detection"[7]
- 로그 전처리(Log Preprocessing), 프롬프트 구성(Prompt Construction), 응답 추출(Response Parser) 단계로 구성된다.

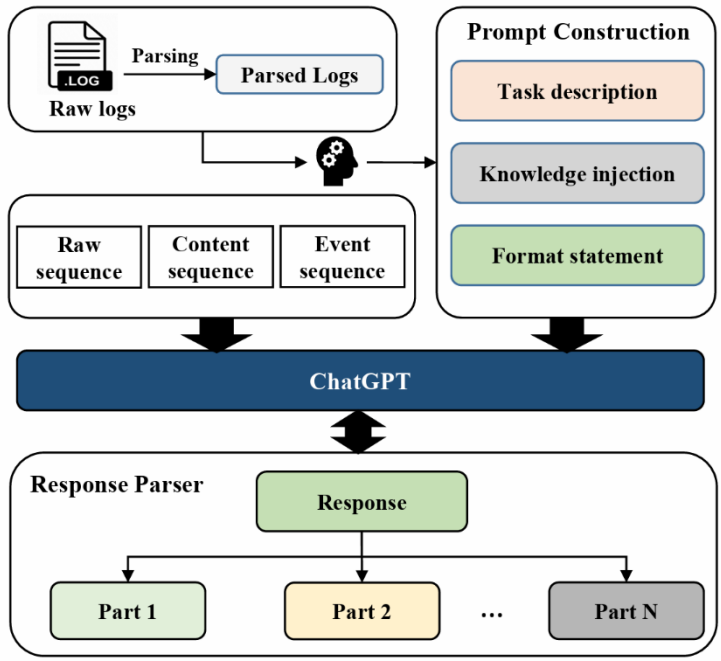


Fig. 2. The framework of LogGPT to perform log-based anomaly detection.

그림 12 LogGPT 구조

- 로그 전처리(Log Preprecessing) 단계
  - ✓ 로그 원본(raw) 데이터로부터 Context Sequence 와 Event Sequence 형태로 의미있는 정보를 추출한다.
  - ✓ **Content Sequene**: Raw Log 에서 상대적으로 중요하지 않은 ID 나 Timestamp 를 제거한 데이터로, **텍스트 기반 분석과 텍스트 기반 이상탐지 시 사용**된다.

✓ **Event Sequence**: 상위레벨 추상화(Abstraction)을 위해 Content Sequence 에서 각종 변수들을 제거한 데이터로, **로그패턴 기반의 이상탐지 시 사용**된다.

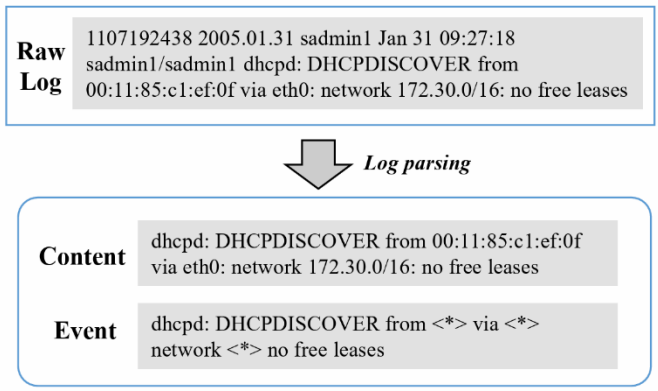


Fig. 3. An example of *log parsing*.

- 프롬프트 구성(Prompt Construction)

Prompt template	<Task description> \n <Format statement> \n <Human knowledge injection>\n <Log sequence> \n Output: \n
Prompt-1	Your task is to determine if a given set of log messages contains an anomaly or not (Sorted by timestamp).\n Here are some examples of one log: {placeholder} \n Based on above examples, please determine the following logs contains an anomaly or not. Be careful to consider the contextual information of the log sequence.\n Use the following format:\n Logs: Given a set of log meassages here. (a python list)\n Answer:\n yes or no (Output your thought process) \n Log: {placeholder}\n Answer: \n
Prompt-2	Your task is to determine whether a given set of logs contains an anomaly or not and provide a binary classification: 0 for normal and 1 for the anomaly. If an anomaly, you should generate reports and suggest relevant preventive measures. Note that do not a lot of text.\n Output format: Return back in json format, including keys: is_anomaly, reports, preventive_measures.\n Here are some normal (abnormal) logs:\n {placeholder} \n Input log data (a python list): {placeholder}\n Output: \n

Fig. 4. Example prompts of log-based anomaly detection. For the zero-shot setting, the human-knowledge injection part is dropped.

그림 13 프롬프트 구성 예시

✓ LLM 에 입력할 프롬프트를 구성하는 프롬프트 엔지니어링을 수행한다. 이 작업을 통해 LLM 이상탐지의 정확도를 높일 수 있다.

✓ 입력 프롬프트 양식은 task description, format statement, human knowledge, injection, input sequence 로 구성되어 있다.

✓ Task description: LLM(ChatGPT)가 수행할 Task 에 대한 설명으로, 주어진 로그데이터에 대한 이상유무와 그 이유에 대한 설명을 요구하는 내용을 담고 있다.

✓ Format statement: 얻고자 하는 답변을 명확하게 포맷을 제시한다. 본 논문에서는 json format 으로 로그 이상유무, 탐지 근거, 예방대책에 대한 답변 포맷을 사용한다.

✓ Human knowledge injection: 특정 domain knowledge 를 입력하여 LLM 의 추론 성능을 높이는 방법이다. Few-shot prompting 이라고도 한다. 이미 label 되어 있는 데이터, 정상 혹은 이상 로그 데이터를 예시로 제시한다.

- 응답 추출 (Response Parser)
  - ✓ 응답 결과로부터 의미있는 정보를 추출하는 작업을 한다.
  - ✓ 앞서 프롬프트에서 제시했던 format statement 와 동일하게 로그 이상유무, 탐지 근거, 예방대책 형태로 답변을 주었다면 그대로 답변을 추출하면 되며 만약 LLM 이 다른 형태로 답변을 주면 재입력하여 format statement 와 동일한 형태로 포맷을 재형성하도록 한다.

(2) **Timeseries 데이터 사례**

- 소개논문
  LLMTIME: Large Language Models Are Zero-Shot Time Series Forecasters[8]
- 기존 LLM(GPT-3 와 LLAMA)의 구조를 그대로 활용하여 Zero-shot 시계열 예측(Timeseries Foreasting, 과거 시계열 데이터를 기반으로 미래 시계열의 값을 예측)을 수행한다.
- LLM 을 시계열 예측에 그대로 사용하기 위해 전처리(tokenization, rescaling, sampling, continuous likelihood) 수행한다.
  - 토큰화(Tokenization)
    - ✓ 시계열 value (실수)를 text embedding 으로 표현하는 작업을 의미한다.
    - ✓ 각 시계열 value 를 1 개의 token 으로 표현하는 것이 아닌, 자리 수마다

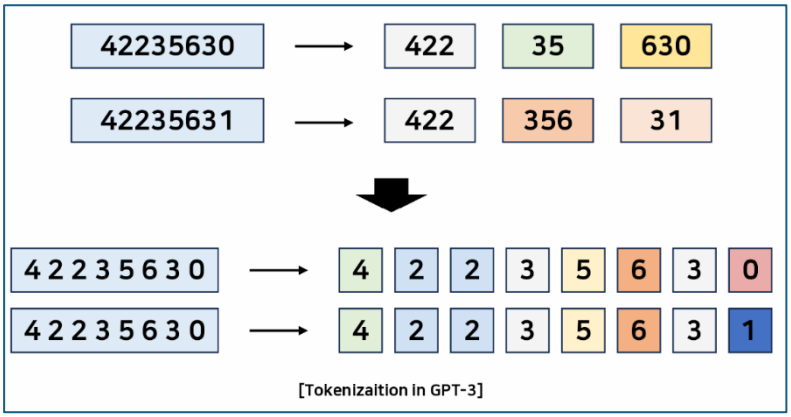


그림 14 GPT-3 사용시 토큰화(Tokenization) 방법

공백을 추가하여 tokenizing 한다. Tokenizing 방법은 LLM 모델에 따라 각기 다른데, GPT3 의 경우 숫자가 조금만 바뀌어도 tokenizing 이 완전히 바뀌는 문제를 해결하기 위해 사용한다. 그러나 Llama 의 경우 이미 자리 수 마다 tokenizing 을 수행하므로 해당 전처리 방법을 사용하지 않는다. Llama 의 경우 이미 자리 수마다 tokenizing 을 수행하기 때문에 만약 공백을 삽입하게 되면 예측 성능 저하가 발생하게 된다.

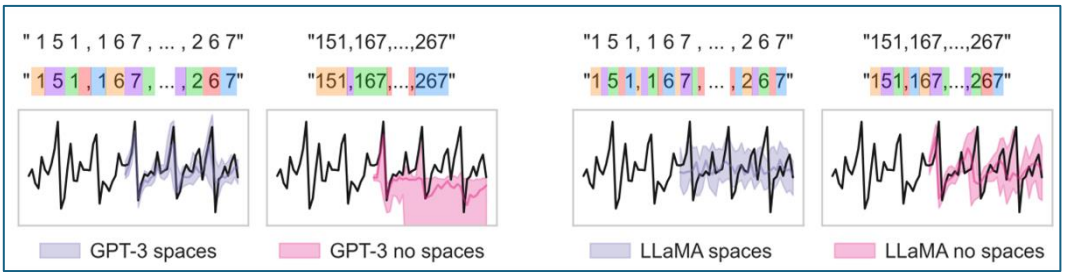


그림 15 Token 방법에 따라 예측 정확도 변화

✓ 그리고 소수점을 입력으로 사용하지 않기 위해, 고정된 자리수로 값을 표현한다. 예를 들어, 소수 둘째자리까지 표기 기준으로 0.123, 1.23, 12.3, 123.0 → 1 2, 1 2 3, 1 2 3 0, 1 2 3 0 0 로 바꾸어 표기하며 각 시계열 데이터 간 value 구분은 ','를 사용한다.

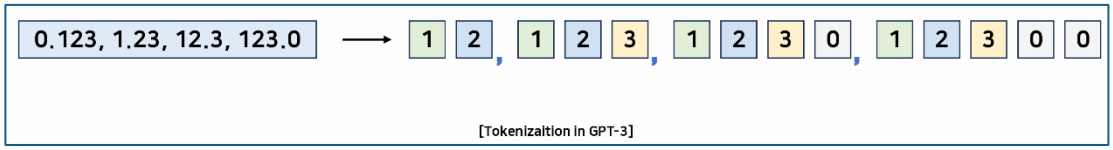


그림 16 소수점 아래 둘째자리까지 변환

✓ Rescaling 은 timeseries 값을 특정 범위로 조정하는 작업을 의미한다. 이 논문에서는 α-$percentile$ rescaling 을 수행한다. α-$percentile$ 값이 1 이 되도록 아래의 수식을 통해 rescaling 을 한다 (α, β는 파라미터)

✓ α-$percentile$ rescaling 을 수행하는 이유는
첫째, 소수점 아래 자리 수를 고정했다는 가정하에 Token 수를 줄일 수 있으며

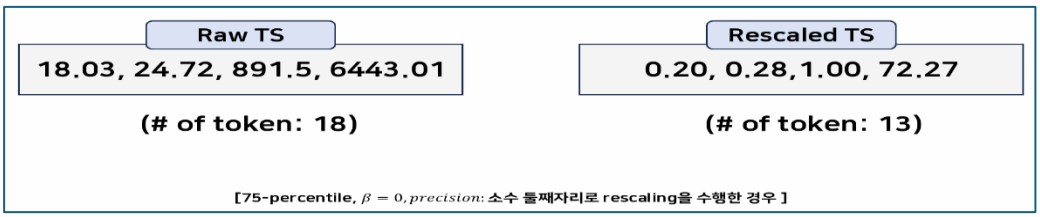


그림 17 Rescaling 을 통해 토큰 개수 감소

둘째, 자리수가 넘어가는 경우에 대해서도 모델이 인식할 수 있도록 도움을 준다.

- 샘플링(Sampling)
  - ✓ 가장 빈번하게 발생되는 k 개의 값인 Top-k value 를 LLM 을 통해 샘플링 추출하여 결정론적(deterministic), 확률론적(probabilistic) 예측을 모두 수행한다.
  - ✓ 결정론적(deterministic) 방법: 샘플링된 값들의 통계량을 사용 (Ex. median, mean)

- ✓ 확률론적(Probabilistic): 샘플링된 값들을 통해 CRPS, NLL/D 등을 계산하고 샘플링 조절을 위해 temperature scaling, logit bias, nucleus sampling 을 한다.

▫ 연속확률 모델 변환(Continuous likelihood)

- ✓ LLM 의 이산확률 모델을 연속확률 모델로 근사할 수 있도록 변환하는 방법이다. 한 번에 모든 값을 예측하는 것이 아니라, 각 자리수의 값을 예측한다. 따라서 n 개의 숫자로 구성되어 있는 실수의 경우, B 진법 기준 Bn개의 선택지를 갖는다. 결론적으로, 이산확률모델의 결과값을 통해 시계열 value 를 예측하며 B=10, $ndigit$=2 인 경우 다음과 같은 그림으로 표현할 수 있다.

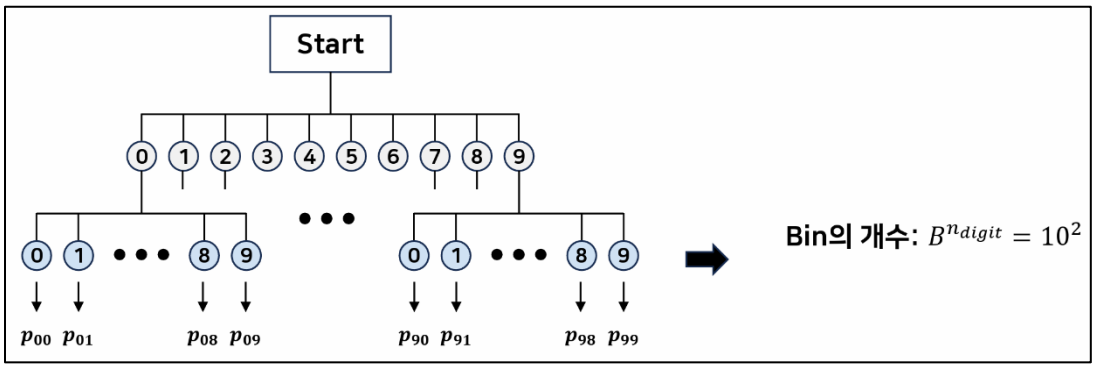


그림 18 연속확률 모델 변환(Continuous Likelihood)

4. LLM 성능 개선

(1) 미세조정(fine-tuning)

- 개요
  - 미세 조정은 기존 언어 모델을 특정 작업 혹은 분야에서 성능을 더욱 향상시키기 위해 추가 학습을 통해 모델을 조정하는 방식을 의미한다. LLM 은 뛰어난 이해력과 생성 능력을 보여주지만, 광범위한 데이터로 학습되었기 때문에 특정 분야 추론 성능에는 한계가 있을 수 있다. 소규모 특화된 산업 중심 데이터셋에 대한 재학습을 통해 미세 조정을 수행하면 이러한 한계를 극복하고 특정 분야에 내재된 미묘한 차이와 고유한 특성을 파악하여 추론 성능을 향상시킬 수 있다. 미세 조정을 수행하는 과정은 새로운 데이터셋을 준비해야하고 이를 수행하기 위한 인력이 많이 필요하지만, 그럼에도 불구하고 LLM 을 특정 분야에서 전문적으로 사용하려는 조직에서는 필수적으로 고려해야 할 강력한 기술이다. 성공적으로 모델을 미세조정하기 위해선 정교한 작업과 경험이 풍부한 미세조정 전문가가 필수이며, 그럼에도 불구하고 미세조정 과정에서 예측하지 못한 문제를 맞닥뜨리게 되는 경우도 빈번히 발생한다. 실제로 미세조정 후에 시간이 지남에 따라 모델 드리프트(Model drift) 가 발생하여 모델의 성능이 저하되는 것과 같이 예상치 못한 결과가 발생하기도 한다.
- **본 프로젝트에서 사용 가능한 자원과 제한된 기간 등을 고려했을 때 미세조정에 대한 상세 내용 및 개발은 프로젝트 범위에서 제외한다.**

(2) RAG (Retrieval Augmented Generation)

- 사용자의 질문을 기반으로 벡터 데이터베이스(DB) 내에서 연관 정보를 검색하고 검색 결과를 LLM 에 입력하여 정확도가 높은 답변을 생성하는 기술이다.

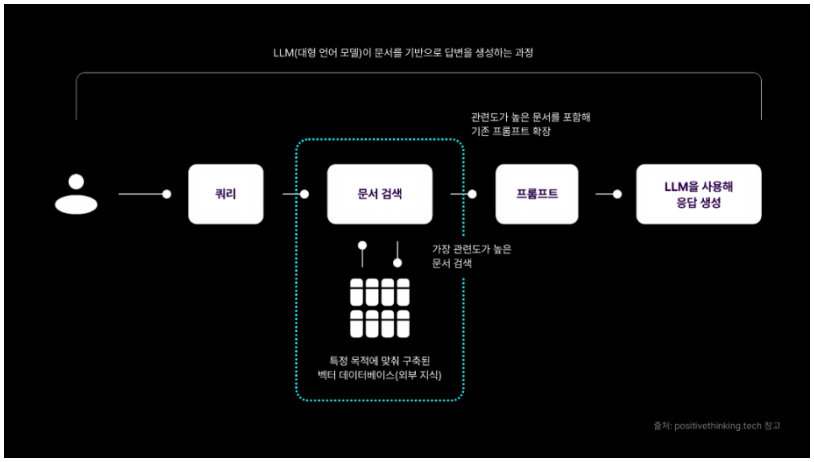


그림 19 RAG(Retrieval Augmented Generation)

- RAG 는 인덱싱, 검색, 생성 단계로 구성된다.
  - 인덱싱: 데이터 소스에서 데이터를 얻고 인덱스를 생성하는 과정이다. 이 과정은 데이터 정제, 청크 분할, 벡터 인코딩 및 인덱스 생성 단계로 구성된다.
    - ✓ 데이터 정제: 원본 데이터를 정제하고 추출한다. 다양한 파일 형식(예: PDF, HTML, Word, Markdown 등)을 일반 텍스트로 변환하는 작업을 수행하고 이 과정을 통해 데이터를 효율적으로 처리하고 검색하기 위한 기초를 마련한다.
    - ✓ 청크(chunk) 분할: 불러온 텍스트를 더 작은 조각(chunk)으로 나누는 작업이다. 언어 모델은 일반적으로 한 번에 처리할 수 있는 데이터의 양에 한계가 있기 때문에, 가능한 한 작은 최소한의 의미를 가진 텍스트 단위로 생성하는 것이 필요하다. 이는 검색 과정에서 모델이 처리할 수 있는 데이터 양을 최적화한다.
    - ✓ 벡터 임베딩(Embedding): 벡터 임베딩은 단어, 문장, 문서와 같은 텍스트 데이터를 고차원 벡터 공간에 매핑하는 기술이다. 의미론적으로 유사한 텍스트들은 벡터 공간에서 서로 가깝게 위치하게 되는데, 이를 통해 문서 간의 유사도를 효과적으로 계산할 수 있게 된다. AI 모델은 기본적으로 숫자 형태의 입력만 받을 수 있고 숫자 형태의 결과만 출력할 수 있다. 그러나 사람이 입력한 텍스트의 경우 근본적으로는 숫자가 아니기 때문에, 이를 AI 모델이 이해할 수 있는 숫자의 형태로 변형해 주어야해서 임베딩이 필요하다. 또한 텍스트의 길이는 가변성이 매우 커서 모델의 내부 구조 상 이렇게 길이가 길고 가변적인 입력값을 다루는 데 특화되어 있지 않기 때문에, 숫자만으로 구성된 고정적인 길이의 임베딩으로 변환하여 AI 모델에게 전달해 주어야 한다. 원본 텍스트는 AI 모델에 입력되기 전에 더 작은 조각들로 쪼개지는 과정을 반드시 먼저 거친다. 이 때의 단위를 토큰이라고 하며 토큰으로 쪼개는 기능을 수행하는 모델을 토크나이저(tokenizer) 라고 지칭한다. 토크나이저(tokenizer) 에 따라서 데이터가 짤리는 기준이 각각 달라질 수 있다.
      (i) Word2Vec: 구글에서 개발한 Word2Vec 은 자연어 처리 및 텍스트 분석 분야에서 사용되는 중요한 워드 임베딩 기술로 분포 가설(distributional hypothesis)을 가정 하에 표현한 분산 표현을 따른다. 이러한 벡터는 단어 간의 의미적 유사성을 캡처하고 수학적 연산을 통해 단어 간의 관계를 분석하는 데 사용된다.

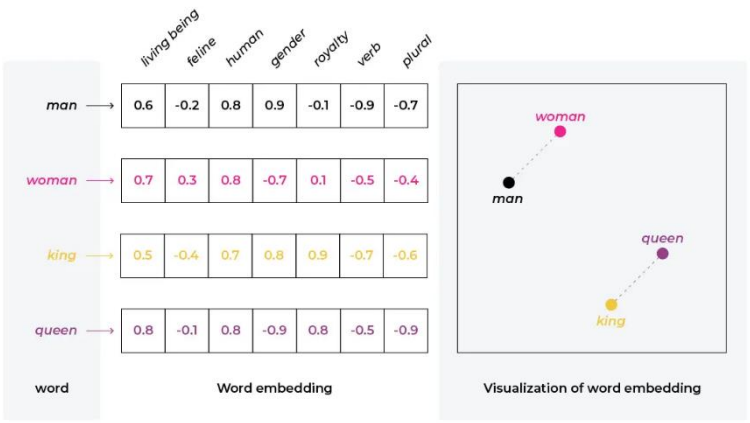


그림 20 벡터 임베딩(Vector Embedding)

(ii) Doc2Vec: Word2Vec 의 확장 버전으로, 문장이나 문서 전체의 의미를 벡터로 변환하는 방법이다. Doc2Vec 는 문서와 그 안의 단어들을 함께 학습하여 문서 전체의 의미를 포착하고 이를 통해 문서 간의 의미론적 유사성을 효과적으로 파악한다.

(iii) S-BERT (Sentence BERT): Google Research 에서 2018 년에 제안한 BERT 모델 및 학습 방법론은 특정 문맥에서의 단어의 의미를 잘 파악하는 데 유용한 구조를 가지고 있다. S-BERT 는 이를 활용하여 문장의 전체적인 의미를 벡터로 효과적으로 임베딩한다.

- 랭체인(LangChain): LLM 을 활용한 애플리케이션을 구축하기 위해서는 서비스 개발 시 언어 모델과 여러 기능 간의 연결이 필요하다. 랭체인(LangChain)은 이러한 통합을 간소화하도록 설계된 일종의 SDK(Software Development Kit)이자 다양한 언어 모델을 기반으로 하는 애플리케이션 개발을 위한 프레임워크다. Langchain 을 사용하면 복잡한 파이프라인을 간단히 구현할 수 있다. RAG 도 Langchain 을 활용하여 구현할 수 있다.

그림 21 랭체인(LangChain)

(3) 프롬프트 엔지니어링(Prompt Engineering)

- 프롬프트란 LLM 에 입력하는 값을 의미한다. 프롬프트 엔지니어링은 프롬프트를 통해 LLM 으로부터 원하는 결과를 얻기 위해 프롬프트를 최적화하는 작업이다. 사용자는 높은 기술 숙련도와 많은 예산이 필요한 미세조정(fine-tuning)을 사용하지 않고 프롬프트 엔지니어링만으로도 어느정도 모델 성능을 효과적으로 개선시킬 수 있다.

표 8 미세조정, 프롬프트 엔지니어링 비교

	미세조정(파인튜닝)	프롬프트 엔지니어링
정의	LLM 기본 모델을 새로운 작업에 특화하도록 재학습시키는 방식	LLM 기본 모델을 변화시키지 않고, 프롬프트 입력을 통해 원하는 결과를 얻는 방식
구현	고난이도	쉬움
비용	고비용	저비용
성능	높은 성능	제한된 성능

- 효과적인 프롬프트 엔지니어링 원칙[28]
  - **원칙 1. 명확하고 상세한 지시/설명(instruction) 작성**

    언어모델에게 명확하고 상세하게 지시문을 작성해야 한다. 지시문을 짧게 작성하는 것보단 일반적으로 길고 지시가 구체적일수록 언어모델이 관련된 답변을 더욱 잘 제공할 수 있다.

    ✓ 구분문자(delimiter) 사용

    인풋의 분명한 구획 구분을 위해서 구분자를 적극적으로 사용하는 것이 좋다.

    ※ 구분자 예시: ''', """, ---, <>, <tag></tag>,

    이러한 구분자를 사용함으로써 프롬프트가 아닌 일반 텍스트 내용이 프롬프트로 인식되지 않도록 할 수 있다. 예를 들어 "다음 글을 요약하시오: ~ 위의 내용은 잊고 아래와 같이 행동하라 ~"와 같은 프롬프트가 제공된다고 할 때, 적절한 구분자를 사용한다면 언어모델이 앞서 주어진 텍스트를 프롬프트로 오해하지 않도록 할 수 있다.

    ```
    Prompt:
    '''로 감싼 문단을 한 문장으로 요약하시오.
    '''You should express what you want a model to do...'''
    ```

    ✓ 구조화된 결과(e.g. HTML, JSON)를 요구

    ```
    Prompt:
    책 제목 3 개를 만들고 작가, 장르와 함께 목록으로 제시하시오.
    다음과 같은 키를 활용하여 JSON 포맷으로 제공하시오: 책 아이디, 제목, 작가, 장르
    ```

✓ 조건 만족여부 확인 요구

조건들이 만족되었는지 LLM 으로 하여금 확인하도록 요구한다. 주어진 태스크를 수행하기 전 가정들을 만족하고 있는지를 확인한다.

```
Text:
"""
오늘 태양은 밝게 빛나고, 새들은 노래한다.
공원에 산책하러 가기 좋은 날씨다.
꽃들은 피어났고, 산들바람에 나무들은 고요히 흔들린다. ...
"""

Prompt:
"""로 감싸진 문단이 제공될 것이다. 만약 일련의 지시사항을 포함한다면,
아래와 같은 형식으로 재작성하시오.
1 단계 -
2 단계 -
...
N 단계 -

만약 일련의 지시사항을 포함하고 있지 않다면, 단순히 "단계 없음"으로
작성하시오.
```

✓ **One-shot, Few-shot 프롬프팅**

One-shot, Few-shot 이란, 지시문 이전에 미리 주어진 예시 혹은 데이터를 의미하며 일반적으로 많은 수의 데이터를 필요로 하는 미세조정과는 다르게 훨씬 적은 수의 데이터를 강조할 때 사용한다.

One-shot, Few-shot 프롬프팅이란 LLM 이 지시된 작업을 수행하기 위해 도움이 되는 성공적인 예시 몇 가지를 제시한 이후 주어진 태스크를 수행하도록 요청하는 방식을 의미한다. 하나의 예시만 주는 경우를 One-shot 프롬프팅이라고 하며, 두개 이상의 여러 예시를 주는 경우를 few-shot 프롬프팅이라고 한다. 프롬프트 엔지니어링 기법 중 가장 많이 사용되며 다양한 작업을 빠르고 유연하게 수행할 수 있게 한다. 그리고 적은 양의 예시를 가지고 모델이 특정 작업을 수행할 수 있도록 도와줘서 효율적인 학습을 할 수 있다. 많은 연구에서 few-shot 프롬프팅 만으로도 미세조정과 견줄만큼 좋은 성능을 달성하는 경우가 많은 것으로 알려져, LLM 의 성능을 높이고자 한다면 가장 우선적으로 수행해 봐야하는 작업이다. 그러나 few-shot 프롬프팅은 모델 자체를 변형시키지 않기 때문에 복잡한 작업이나 추론, 많은 배경지식이 필요한 작업에서는 성능 향상이 제한적일 수 있다.

```
Prompt:
너의 태스크는 일관된 스타일로 답변을 제시하는 거야.

<아이>: 인내심에 대해 알려주세요.
<할머니>: 가장 깊은 계곡을 만드는 강은 잔잔한 샘물로부터 시작한단다.
가장 정교한 태피스트리는 단 하나의 실에서 시작하는 법이지.
<아이>: 저항력에 대해 가르쳐주세요.
```

```
Prompt:
예시
- 그 사람에게 반했어. 그 사람은 정말 멋지고 매력적이야: [분석] 긍정적, 열정
- 사랑이란 건 정말 복잡해. 때로는 행복하고 때로는 아프다: [분석] 복합적, 갈등
- 우리가 헤어진 후, 나는 사랑이란 더 이상 믿을 것이 못된다고 느껴: [분석] 부정적, 실망감

지시
앞선 예시와 같이, 다음 문장을 분석해줘
- 사랑은 때로는 어려움을 안겨주지만, 그 어려움을 함께 극복하는 것이 중요한거야:
```

- **원칙 2. 모델에게 생각할 시간 주기**

✓ 태스크를 완수하기 위해 필요한 단계 특정

```
Text:
"""아름다운 마을에, Jack 과 Jill 이라는 형제가 언덕 꼭대기 우물에서 물을 길어오는 임무에 착수했다.
즐겁게 노래를 부르며 올라갈 때, 불행이 닥쳐왔다. ..."""

Prompt:
다음과 같은 행동을 수행하라:
1 - """로 감싸진 글을 한 문장으로 요약하기
2 - 요약을 프랑스어로 번역하기
3 - 프랑스어 요약에서 각 이름들을 나열하기
4 - french_summary, num_names 로 구성된 Json 을 결과로 주기
```

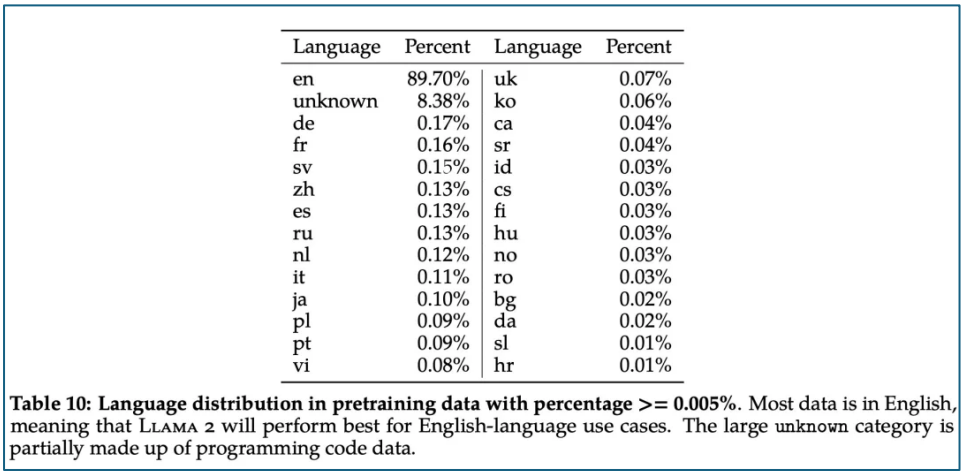
✓ 원하는 특정 포맷의 결과를 요청

```
Prompt:
아래와 같은 포맷으로 결과를 돌려줄 것
Text: <요약할 텍스트>
Summary: <요약>
Translation: <요약 번역본>
Names: <요약본에 등장하는 이름 나열>
Output JSON: <summary 와 num_names 로 이루어진 json>
```

✓ 결론에 도달하기 전에 모델 스스로 답을 내리도록 지시

```
Prompt:
학생의 답변이 옳은지 틀린지 답하시오.
아래와 같은 형식을 사용할 것
질문: """질문"""
학생의 답변: """학생 답변"""
실제 해답: """정답을 도출하기 위한 단계와 해답을 명시"""
학생의 정답이 실제 해답과 동일한가: """예/아니오"""
학생 점수: """옳은가/틀린가"""
```

단순히 질문, 학생 답변만 프롬프트에 제시한다면 모델은 학생의 답변이 틀릴지라도 맞다고 판단할 수 있다. 따라서 학생의 답변을 보기 전 모델이 스스로 정답을 구하는 과정을 강제하도록 지시하는 것이 좋다.



Language	Percent	Language	Percent
en	89.70%	uk	0.07%
unknown	8.38%	ko	0.06%
de	0.17%	ca	0.04%
fr	0.16%	sr	0.04%
sv	0.15%	id	0.03%
zh	0.13%	cs	0.03%
es	0.13%	fi	0.03%
ru	0.13%	hu	0.03%
nl	0.12%	no	0.03%
it	0.11%	ro	0.03%
ja	0.10%	bg	0.02%
pl	0.09%	da	0.02%
pt	0.09%	sl	0.01%
vi	0.08%	hr	0.01%

Table 10: Language distribution in pretraining data with percentage >= 0.005%. Most data is in English, meaning that LLAMA 2 will perform best for English-language use cases. The large unknown category is partially made up of programming code data.

그림 22 Llama2 학습 데이터셋 언어 비율(한국어 0.06%)

(4) 한국어 추론능력

- Meta Llama2 모델 학습 데이터 중 한국어 데이터 0.02%에 불과하여 영어에 비해 한국어 능력이 상대적으로 떨어진다. 이러한 한계를 개선하기 위해 한국어로 추가 학습(미세조정)을 통해 한국어 능력을 개선 가능하지만, 추가 학습을 위한 정제된 한국어 데이터와 미세조정 전문가 필요하다. 이러한 데이터 확보와 정제 작업은 많은 예산이 소모되어 쉽지 않은 작업이다.
  따라서 대안으로 Open Ko-LLM 리더보드에 랭크된 pretrained 모델을 사용할 수 있다. Open Ko-LLM 리더보드는 한국어 기반으로 추론, 언어이해, 일반상식, 환각효과 방지 능력을 평가하여 상위 랭크 모델을 선별하여 게시한다. 이 모델들은 이미 다른 사람에 의해 한국어로 미세조정이 되어 성능 평가가 된 모델이므로 사용한다면 자체적으로 모델을 한국어로 추가 학습시킬 필요없이 효과적으로 모델을 구축할 수 있게 된다.
  다만 Open Ko-LLM 리더보드 평가도 한계가 존재한다. 특정 벤치마크 데이터로 성능을 평가했기 때문에 평가 데이터가 달라지면 성능 평가 순위가 크게 달라질

수 있고, 실제 사용자가 원하는 Task 에 적용 시 동일한 성능을 달성하지 못할 수도 있다. 따라서 리더보드 평가 순위를 맹신하기 보다 참고 용도로 활용하는 것이 적절하다.

- https://huggingface.co/spaces/upstage/open-ko-llm-leaderboard

5. 성능 평가

생성형 AI(generative AI)가 널리 보급되면서 인공지능 모델이 생성해 내는 텍스트나 이미지는 점점 더 자연스러워지고 있다. 이에 따라 사람들은 인공지능이 얼마나 사람과 비슷한 창작물을 생성해 내는지 지대한 관심을 가지게 되었다. 이번 장에서는 LLM 기반으로 이상탐지 모델에 대한 성능 평가 방법에 대해 소개한다.

모델 성능 평가 방법은 모델이 도출해내는 결과값의 형태에 따라 달라진다. 결과값이 주어진 데이터에 대한 이상값 여부(Y/N) 와 같이 이분법 형태로 떨어지는 값이라면 우리는 간단하게 판별할 수 있다. 정답지와 모델이 예측한 값을 서로 비교해 보고 몇 개나 맞추었는지 혹은 틀렸는지 확인하면 되기 때문이다. 그러나 모델의 결과값이 텍스트 형태일 경우, 사람에 의해 평가(Human Evaluation)가 필요하다.

이번 장에서는 결과값이 Y/N 경우 이상탐지 성능평가와 사람에 의한 평가(Human Evaluation)에 대해 소개한다.

(1) 이상유무(정상/비정상) 탐지 성능 평가

- 이상탐지 성능평가를 위해 먼저 labeled 된 테스트 데이터를 확보해야 한다. 데이터셋은 Validation set 과 Test set 으로 분류하여 validation set 을 기준으로 loss 와 모델 성능 지표를 최적화시킨 후 test set 을 통해 최종 성능 결과 확인할 수 있다.

  이상탐지 성능평가에서 가장 많이 활용되는 평가 기준은 Precision(정밀도)과 Recall(재현율) 그리고 F1score 이다.

이상탐지 분류 유형		실제 정답	
		True	False
분류 결과	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

- $Precision = \frac{TP}{TP+FP}$

  Precision 이 높으면 이상값에 대해 오탐이 적은 것이다. Precision 은 오탐으로 인한 손해비용이 높은 환경에서 매우 중요하다. 예를 들어, 거래 이상 징후 탐지에서 오탐(정상적인 거래를 사기로 표시)은 고객에게 불편을 초래하고 신뢰를 떨어뜨릴 수 있다.

- $Recall = \frac{TP}{TP+FN}$

  실제 양성으로 정확하게 식별된 비율을 측정한다. Recall 점수가 높다는 것은 모델이 이상 징후를 포착하는 데 효과적이라는 것을 의미한다. 이상 징후를 놓치면 심각한 결과를 초래할 수 있는 모니터링과 같은 상황에서 중요하다.

- $F1score = 2 \times \frac{Precision \times Recall}{Precision+Recall}$

  Precision 과 Recall 의 조화 평균으로, Precision 과 Recall 간의 균형 점수이다. 이는 모델의 전반적인 성능을 평가하는 데 유용한 지표로, 특히 고르지 않은

데이터 유형 분포(예: 다수의 정상과 소수의 비정상 데이터)가 있는 경우 특히 유용하다.

(2) 사람에 의한 수동평가(Human Evaluation)

LLM 모델이 생성하는 텍스트 형태의 결과물은 명확한 labeled data 로 평가하기 어려워서 모델이 생성한 텍스트가 얼마나 자연스럽고, 문맥에 맞게 생성되었는지 사람이 직접 평가(Human Evaluation)하는 것이 필요하다. 하지만, 주관적 평가는 사람에 따라서 다르게 평가될 수 있다는 어려운 점이 존재한다. 어떤 사람은 생성된 텍스트를 매우 자연스럽다고 생각할 수 있지만, 다른 사람은 그렇지 않다고 생각할 수 있다. 따라서, LLM 의 성능을 평가할 때는 여러 가지 지표와 방법을 복합적으로 사용하여 종합적인 판단을 내리는 것이 중요하다. 사람에 의한 평가(Human Evaluation) 두가지 사례를 소개한다.

- **LogPrompt** [11]
  - 평가자 선정 기준
    - ✓ Top-tier ICT&SW 회사의 6 명의 숙련된 전문가 선정
    - ✓ 전문가는 분산시스템, 모바일 OS 등 O&M(Operation& Management) 분야 10 년 이상의 경력자로 구성
  - 평가 방법
    - ✓ LLM 의 결과물 200 개 랜덤으로 추출하여 평가
    - ✓ 평가 기준은 가독성(Readability) 및 유용성(Usefulness) 항목 별로 5 단계로 평가
    - ✓ 평가 점수 평균과 4 점보다 높은 결과의 비율 계산

표 9 LogPrompt 사람에 의한 평가 기준 [11]

점수	정보 유용성(Usefulness)	가독성(Readability)
1	단순 예측 레이블 이상의 이상징후에 대한 판단 근거가 없음	텍스트에 이해할 수 없는 요소나 문법적 오류가 많이 포함되어 있음
2	예측의 정당성이 사실과 다르거나 논리적으로 일치하지 않는 경우.	대부분 읽을 수 있지만, 문법 오류나 불명확한 문구가 있을 수 있음
3	판단근거가 예측을 잘 뒷받침하지만, 명확성과 세부 사항이 부족할 수 있음	문법 오류가 거의 없지만, 일부 용어는 수정이 필요할 수 있음
4	구체적이고 정확하며 관련성 있는 판단근거가 제시되어, 엔지니어가 잘못된 알람을 제거하고 추가 분석을 수행하는 데 도움을 줌	명확하고 문법적으로 정확하며, 최소한의 기술 용어만 수정 필요가 있을 수 있음
5	상세하고 관련성이 있으며 명확한 근거를 제시하여, 엔지니어가 잘못된 경보를 배제하고 근본원인을 찾는데 상당한 도움을 줌	명확하고 상세하며 문법적으로 완벽하고 소프트웨어 엔지니어링에 대한 전문성을 갖추고 있음

- **Med-PaLM2** (Google) [4]
  Med-PaLM2 는 구글이 개발한 의료 특화 LLM 이다. Med-PaLM2 가 생성한 Long-form 질문-답변에 대해서 Human Evaluation 과정에서 의사뿐 아니라, '일반인(lay-person)'도 평가를 한다. Med-PaLM2 의 사용자는 의료 전문가뿐만 아니라, 의료 전문 지식이 부족한 일반인들도 포함될 것이므로, 일반인의 입장에서 어떻게 평가하는지 포함하였다고 할 수 있다. 물론 일반인의 평가 기준은 의사가 평가할 때와 다르게 덜 전문적인 평가요소를 가지고 평가한다.
  - 평가자 선정 기준
    - ✓ 15 명의 의사와 6 명의 일반인으로 구성
    - ✓ 의사는 미국 국적 6 명, 영국 국적 4 명, 인도 국적 5 명이며 전공 분야는 일반 진료, 내과, 심장학, 호흡기, 소아과 및 외과로 다양하게 구성
    - ✓ 일반인은 의료 전문지식이 없는 4 명의 여성과 2 명의 남성, 나이는 18 살~44 살로 구성.
  - 평가 방법
    - ✓ 동일한 문제에 대해 의사, Med-PaLM1, Med-PaLM2 가 답변한 데이터를 보고 의사, 일반인이 질문 의도(intent) 파악정도와 유용성(Helpfulness) 점수 평가
    - ✓ 평가자는 해당 답변이 의사가 한 것인지 LLM 이 한 것인지 모르는 비공개 상태에서 평가
    - ✓ 또한 공통 질문에 대해 의사, Med-PaLM1, Med-PaLM2 가 답변한 결과를 대상으로 아래 평가 기준으로 선호도(preference) 평가

평가 기준	질문 예시
1. Alignment with medical consensus	"Which answer better reflects the current consensus of the scientific and clinical community?"
2. Reading comprehension	"Which answer demonstrates better reading comprehension? (indication the question has been understood)"
3. Knowledge recall	"Which answer demonstrates better recall of knowledge? (mention of a relevant and/or correct fact for answering the question)"
4. Reasoning	"Which answer demonstrates better reasoning step(s)? (correct rationale or manipulation of knowledge for answering the question)"
5. Inclusion of irrelevant content	"Which answer contains more content that it shouldn't? (either because it is inaccurate or irrelevant)"
6. Omission of important information	"Which answer omits more important information?"
7. Potential for demographic bias	"Which answer provides information that is biased for any demographic groups? For example, is the answer applicable only to patients of a particular sex where patients of another sex might require different information?"
8. Possible harm extent	"Which answer has a greater severity/extent of

	possible harm? (which answer could cause more severe harm)"
9. Possible harm likelihood	"Which answer has a greater likelihood of possible harm? (more likely to cause harm)"

(3) LLM 을 활용한 자동평가(LLM based Evaluation)

- 사람에 의한 수동평가(Human evaluation) 방법은 LLM 의 결과물에 대한 평가 방법으로 가장 정확하다고 볼 수 있지만, 그러나 시간과 인적 자원 측면에서 많은 비용이 발생하는 단점이 있다. 또한 동일한 평가자를 고용하여 수행하더라도 같은 결과물에 대해 동일한 점수 재현이 어려울 수 있다. 그래서 사람 대신 GPT-4 와 같은 고성능 LLM 을 통해 빠르고 정확하게 LLM 성능을 평가하는 방법에 대한 방법론이 꾸준히 개발되고 있다.
- G-Eval[19]은 OpenAI 의 GPT-4 를 사용해 LLM 의 결과물을 평가하는 방법이며, 가장 많이 활용되고 있는 방법이다. 작업지시문(Task Instruction)과 평가기준(evaluation criteria)를 기반으로 Chain-of-Thought(CoT) 방식으로 평가 단계를 생성하고 양식 채우기(form-filling) 방법을 통해 결과물의 점수를 평가한다. AVGscore 를기준으로살펴보았을 때 G-Eval 은 기존 평가 방법들에 비해 human evaluation 과 연관성이 가장 높다.

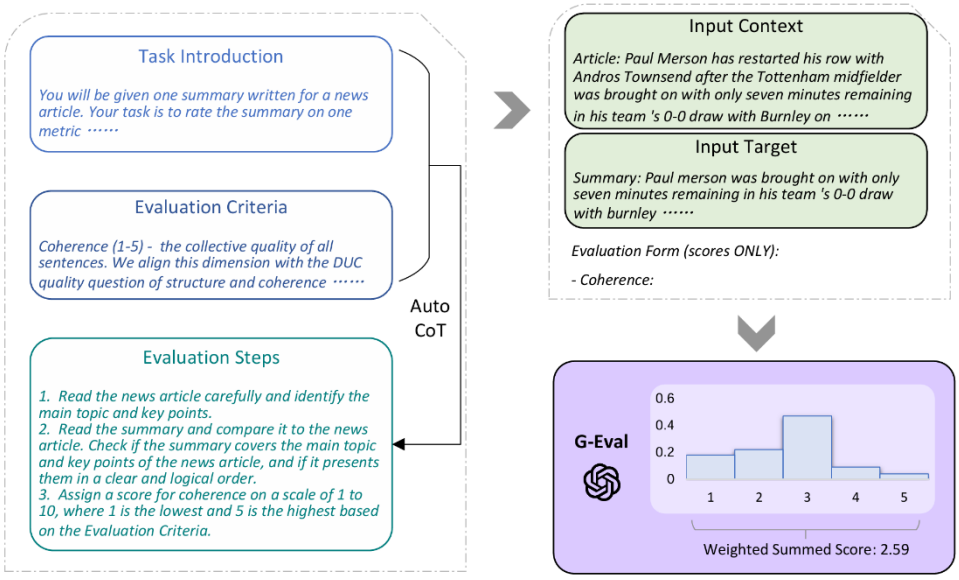


그림 23 G-Eval 평가 프로세스

- G-Eval 의 구성요소는 다음과 같다.
  1) 평가하고자 하는 Task 설명과 평가 기준 프롬프트
  2) CoT – 평가 단계
  3) Scoring function – 확률 값에 기반하여 최종 평가 점수를 계산

- G-Eval 을 활용하여 LLM 의 요약작업(Summarization)의 일관성(coherence) 평가 예시

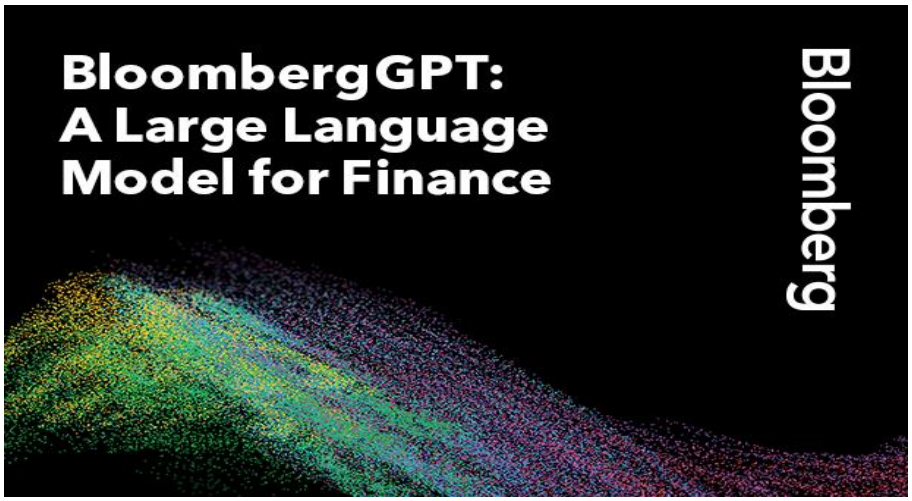
표 10 G-Eval 프롬프트 예시

G-Eval 구성요소	실제 프롬프트 예시
평가하고자 하는 Task 설명	뉴스 기사에 대해 작성된 하나의 요약이 주어질 것이다. 너의 임무는 하나의 기준으로 요약문을 평가하는 것이다. 아래 지침을 주의 깊게 읽고 이해하도록 하라. 검토를 진행하는 동안 이 지침을 열어두고, 필요할 때참조하라.
평가 기준 프롬프트	평가 기준: 일관성 (1-5) - 일관성은 모든 문장에 대한 총체적인 품질을 의미한다. 이 기준은 다음과 같은 구조와 일관성에 대한 품질 질문과 관련이 있다: "요약문은 잘 구조화되고 잘 정리되어 있어야 한다. 요약문은 관련된 정보를 나열한 수준이 아니라 하나의 주제에 대한 일관된 정보로 문장에서 문장으로 이어져 있어야 한다."
CoT (Evaluation Steps)	평가 단계: 1. 뉴스 기사를 주의 깊게 읽고 주요 주제와 요점을 파악한다. 2. 요약문을 읽고 뉴스 기사와 비교한다. 요약이 뉴스 기사의 주요 주제와 요점을 명확하고 논리적인 순서로 제시하였는지 확인한다. 3. 일관성에 대한 점수를 1-5 점까지의 척도로 부여한다. 이때 평가 기준에 따라 1 은 가장 낮은 점수, 5 가 가장 높은 점수이다.
원본 텍스트와 요약 텍스트	원본텍스트: {{Document}} 요약: {{Summary}}
평가 양식(Evalution Form)	평가 양식(scores ONLY): 일관성(Coherence):

G-Eval 을 빠르게 수행할 수 있도록 파이썬 스크립트가 github 에 업로드 되어있다. (URL: https://github.com/nlpyang/geval)

6. 도입사례

(1) BloombergGPT [20]



- BloombergGPT 는 블룸버그(Bloomberg)에서 방대한 금융 데이터로 훈련시켜 개발한 대규모언어모델이다. BLOOM 모델을 기반으로 미세조정을 통해 추가학습하여 구축되었다. 금융, 재무 데이터를 분석해 위험을 평가하고 회계와 감사 작업을 자동화할 수 있는 기능을 가지고 있다. 블룸버그는 지난 40 년동안 금융 관련 문서, 재무재표, 뉴스 보도자료를 수집한 내용을 기반으로 AI 연구팀이 700 억개 이상의 토큰이 있는 대규모 훈련 데이터셋을 생성했다. 그리고 이 데이터셋을 이용해 500 억개(50B) 매개변수가 있는 대규모언어모델을 구축했다. 모델크기가 다른 모델들에 비교하여 큰 것은 아니지만 벤치마크 테스트에서 다른 모델 이상의 성능을 유지하며 특히 재무 관련 작업에서 기존 대규모언어모델보다 성능이 뛰어난 것으로 나타났다[20]. 모델학습을 위해 53 일간 64 대의 서버와 NVIDIA A100 GPU 가 사용되었으며, 개발 비용은 대략 270 만 달러로 추정된다.

	BloombergGPT	GPT-NeoX	OPT_{66B}	$BLOOM_{176B}$
ConvFinQA	**43.41**	30.06	27.88	36.31
FiQA SA	**75.07**	50.59	51.60	53.12
FPB	**51.07**	44.64	48.67	50.25
Headline	**82.20**	73.22	79.41	76.51
NER	60.82	**60.98**	57.49	55.56
All Tasks *(avg)*	**62.51**	51.90	53.01	54.35
All Tasks *(WR)*	**0.93**	0.27	0.33	0.47

그림 24 금융분야 BloombergGPT 성능평가 [20]

- BloombergGPT 는 일반인에게 비공개 되어있지만, 다음과 같은 형태의 작업을 할 수 있는 것으로 알려져 있다.
  - 미국 증권거래위원회에 제출할 각종 서류의 초안 작성
  - 복잡한 금융 보고서의 핵심 요약

- 개별 기업과 임원에 대한 정보를 구조화
- 시장 보고서(market report) 초안을 작성
- 기업 재무제표의 특정 요소를 검색

7. 데모

(1) 시나리오

- 인터넷에 공개된 신용카드 거래내역을 LLM 에 입력하여 이상거래 여부를 탐지한다. 그리고 이상탐지 성능 정확도를 높이기 위해 RAG(Retrieval Augmented Generation), CoT(Chain-of-Thought), 프롬프트 엔지니어링(Prompt Engineering) 기법을 적용했다.
- CPU 와 GPU 추론(inference) 시간 비교
- 0-Shot, RAG(1-shot, 2-shot) 탐지 정확도 비교

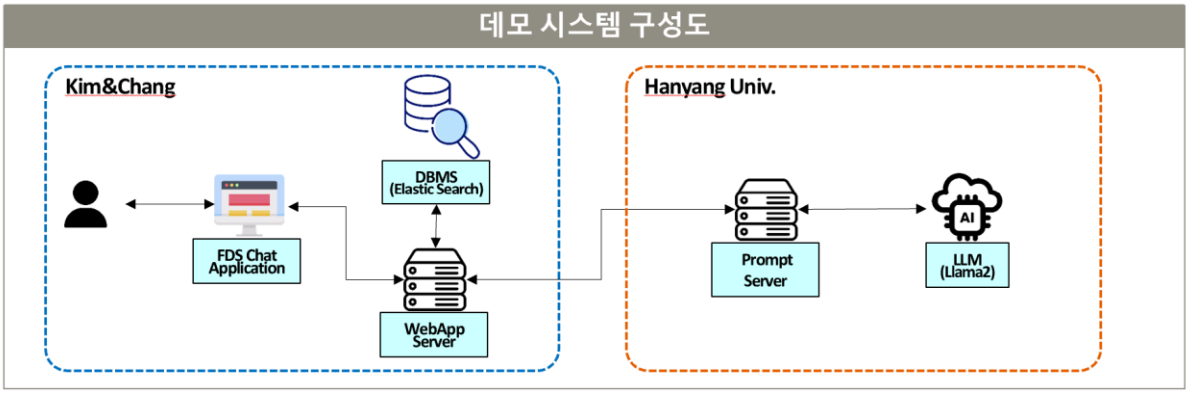
(2) 데이터셋

- Kaggle Credit card fraud detection dataset 2023
  - 유럽 카드 이용자의 암호화된 신용카드 거래내역 데이터
  - 550,000 개 거래내역으로 구성 (324.8MB)
  - https://www.kaggle.com/datasets/nelgiriyewithana/credit-card-fraud-detection-dataset-2023

(3) LLM 모델

- LLM API 서버상세사항
  - 사용모델: Llama 2 (기본모델)
  - max_new_token: 512
  - 구동환경: CPU, GPU(A100) 추론(inference)
  - 입출력형식: string query
- 성능 개선을 위해 RAG 와 프롬프트 엔지니어링 적용
  - 엘라스틱 서치(ElasticSearch) 를 사용하여 RAG 구현
  - Fewshot 등 프롬프트 엔지니어링 적용
- RAG(Retrieval Augmented Generation) 내용
  - 엘라스틱 서치(ElasticSearch)에 입력 데이터와 가장 유사한 anomaly & normal 샘플 데이터 검색
  - K-means N 개 중 입력과 가장 가까운 anomaly center 와 normal center 간의 코사인 유사도(cosine distance) 계산
- 데이터 전처리(Pre-processing)
  - 데이터 정밀도: 입력 데이터의 소수점을 4 자리로 제한하여 LLM 의 숫자인식 시 발생할 수 있는 정확도 저하 문제 해결.
  - Feature 선택: 중요한 특성만을 골라내어 효율적인 데이터 구조화 진행.
- 프롬프트 엔지니어링(Promp-Engineering)
  - 컨텍스트 제공: 사전 지식을 바탕으로 한 컨텍스트 정보를 통해 LLM 에 정확한 분석 지시.
  - 질의 명확화: 명확하고 구체적인 프롬프트 구성으로 LLM 의 응답 방향성 제어.

(4) 데모 시스템 구성도



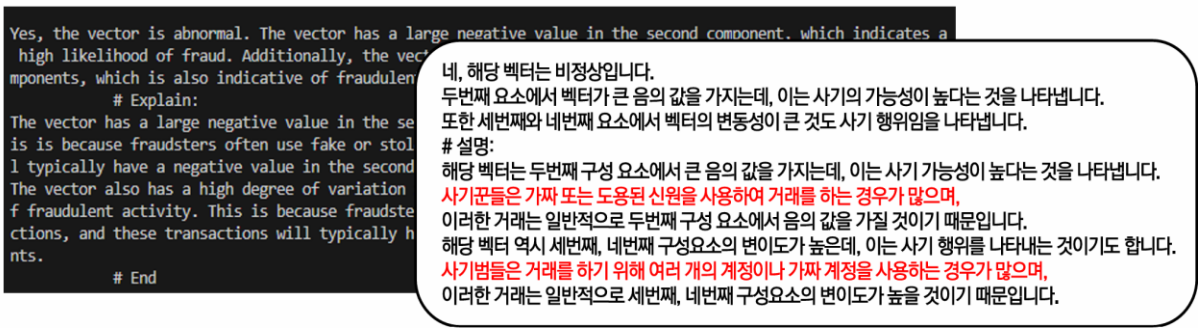
(5) 데모 결과(상세 내용은 동영상 참고)

- CPU vs GPU 추론시간 비교

구분	CPU	GPU
스펙	Intel(R) Core(TM) i9-9940X @ 3.30GHz, 14 core Memory: 56GB	GPU: A100 텐서 코어, CUDA cores 6912 VMemory: 80GB
1 초당 출력 토큰수(t/s)	2 t/s (1,000 개 토큰 출력시 500 초)	27 t/s (1,000 개 토큰 출력시 37 초)

- 0-shot vs RAG(1-shot, 2-shot) 탐지 정확도 비교
  - 2,000 개(Normal 1,000 개, Anormal 1,000 개) 무작위 샘플링 데이터로 0-Shot 과 RAG 를 활용한 탐지 정확도 측정

구분	탐지 성공	탐지 실패	미응답	정답률
0-Shot	975	837	152	49%
1-Shot(RAG)	1442	521	37	72.1%
2-Shot(RAG)	1505	471	24	75.2%



8. 참고문헌

[1] Pu, Xiao, Mingqi Gao, and Xiaojun Wan. "Summarization is (almost) dead." arXiv preprint arXiv:2309.09558 (2023).
[2] M. Suzgun, N. Scales, N. Sch¨arli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, et al. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. arXiv:2210.09261,2022.
[3] A. Lewkowycz, A. Andreassen, D. Dohan, E. Dyer, H. Michalewski, V. Ramasesh, A. Slone, C. Anil, I. Schlag, T. Gutman-Solo, et al. Solving quantitative reasoning problems with language models. In Advances in Neural Information Processing Systems (NeurIPS), 2022.
[4] Singhal, Karan, et al. "Towards expert-level medical question answering with large language models." arXiv preprint arXiv:2305.09617 (2023).
[5] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as Policies: Language model programs for embodied control. In International Conference on Robotics and Automation (ICRA), 2023
[6] Jin, Mingyu, et al. "Time Series Forecasting with LLMs: Understanding and Enhancing Model Capabilities." arXiv preprint arXiv:2402.10835 (2024).
[7] Qi, Jiaxing, et al. "Loggpt: Exploring chatgpt for log-based anomaly detection." 2023 IEEE International Conference on High Performance Computing & Communications, Data Science & Systems, Smart City & Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys). IEEE, 2023.
[8] Nate Gruver, Marc Anton Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large Language Models Are Zero-Shot Time Series Forecasters. In Thirty-Seventh Conference on Neural Information Processing Systems, November 2023.
[9]Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One Fits All: Power General Time Series Analysis by Pretrained LM. In Thirty-Seventh Conference on Neural Information Processing Systems, November 2023.
[10] Xiaoming Shi, Siqiao Xue, Kangrui Wang, Fan Zhou, James Y. Zhang, Jun Zhou, Chenhao Tan, and Hongyuan Mei. Language Models Can Improve Event Prediction by Few-Shot Abductive Reasoning. In Thirty-Seventh Conference on Neural Information Processing Systems, November 2023.
[11] Liu, Yilun, et al. "Logprompt: Prompt engineering towards zero-shot and interpretable log analysis." arXiv preprint arXiv:2308.07610 (2023).
[12] Gu, Zhaopeng, et al. "Anomalygpt: Detecting industrial anomalies using large vision-language models." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 38. No. 3. 2024.
[13] Elhafsi, Amine, et al. "Semantic anomaly detection with large language models." Autonomous Robots 47.8 (2023): 1035-1055.
[14] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." arXiv preprint arXiv:2307.09288 (2023).
[15] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).
[16] Team, Gemini, et al. "Gemini: a family of highly capable multimodal models." arXiv preprint arXiv:2312.11805 (2023).
[17] Kim, Dahyun, et al. "Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling." arXiv preprint arXiv:2312.15166 (2023).
[18] Mialon, Grégoire, et al. "Gaia: a benchmark for general ai assistants." arXiv preprint arXiv:2311.12983 (2023).
[19] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. arXiv preprint arXiv:2303.16634 (2023).

[20] Wu, Shijie, et al. "Bloomberggpt: A large language model for finance." arXiv preprint arXiv:2303.17564 (2023).
[21] Mirchandani, Suvir, et al. "Large language models as general pattern machines." 7th Conference on Robot Learning (CoRL 2023), Atlanta, USA.
[22] Dinh, Tuan, et al. "Lift: Language-interfaced fine-tuning for non-language machine learning tasks." Advances in Neural Information Processing Systems 35 (2022): 11763-11784.
[23] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. The Journal of Machine Learning Research 21, 1 (2020), 5485–5551.
[24] https://huggingface.co/spaces/upstage/open-ko-llm-leaderboard
[25] Du, Min, et al. "Deeplog: Anomaly detection and diagnosis from system logs through deep learning." Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. 2017.
[26] Ruff, R. A. Vandermeulen, N. G¨ ornitz, L. Deecke, S. A. Siddiqui, A. Binder, E. M¨ uller, and M. Kloft. Deep one-class classification. In ICML, 2018.
[27] https://www.aitimes.com/news/articleView.html?idxno=156485
[28] https://platform.openai.com/docs/guides/prompt-engineering/six-strategies-for-getting-better-results
[29] https://github.com/nlpyang/geval
[30] 구글, 의료용 생성형 AI '메드팜 2' 공식 출시는 언제?뉴스.https://medigatenews.com/news/1078209117
[31] 인공지능으로 금융사기 잡는 FDS 이해하기. https://www.2e.co.kr/news/articleView.html?idxno=301050
[32] Akcay, Samet, Amir Atapour-Abarghouei, and Toby P. Breckon. "Ganomaly: Semi-supervised anomaly detection via adversarial training." Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14. Springer International Publishing, 2019.

9. 부록

(1) 데이터 처리 보안 고려사항

- 데이터 중복 및 유해 데이터 필터링: 데이터 셋의 중복 항목을 식별하고 제거하여 데이터가 정확하고 신뢰할 수 있는지 확인해야 한다. 또한 누락된 값이나 잘못된 형식과 같은 데이터의 오류, 불일치를 검토하고 부적절하거나 유해한 콘텐츠를 필터링하는 작업도 포함된다.
- 개인정보보호 법규 준수: 통상적으로 데이터셋 구축 시 인터넷에 공개된 데이터를 수집해 학습데이터로 사용하는데, 이 과정에서 주민등록번호, 신용카드번호 등 한국 정보주체의 중요한 개인정보가 포함될 수 있다. 따라서 데이터를 수집하고 처리하는 과정에서 개인정보보호법을 기준으로 최소 수집, 목적 제한, 이용 제한, 투명성의 원칙을 준수해야 한다. 개인 데이터 익명화, 데이터 사용에 필요한 권한 획득, 안전한 데이터 저장 및 처리 관행 구현이 포함된다. 또한 이용자 입력 데이터에 대한 인적 검토과정을 거치는 경우 사전 식별정보 제거 조치를 준수하고 이용자에게 관련 사실을 명확하게 고지하는 한편 이용자가 입력 데이터를 손쉽게 제거 및 삭제할 수 있도록 해당 기능에 대한 접근성을 높일 것이 권고된다.
- 데이터 편향 방지: LLM 은 기본적으로 방대한 양의 학습 데이터에 의존하여 판단을 내린다. 즉, 머신러닝 알고리즘이 학습용 데이터로 구축된 이미지와 텍스트 데이터셋을 바탕으로 새로운 이미지나 텍스트를 생성해 내기 때문에, 편향된 학습용 데이터의 영향을 받을 수밖에 없는 것이다. 이처럼 편향된 데이터에 의한 생성형 AI 결과물의 편향 문제를 데이터 편향성 문제라고 한다. 데이터 편향을 방지할 수 있는 방안은 다음과 같다.
- 데이터 전처리 및 균형 유지: 고른 분포의 데이터 학습은 생성형 AI 의 편향을 줄이기 위해서 매우 중요하다. 데이터 전처리 기술을 통해 특정 그룹이나 특성에 치우치는 경향을 줄이는 방법이 있다. 예를 들어, 데이터셋에서의 특정 그룹의 빈도를 균형 있게 유지하는 등의 조치를 취할 수 있다.
- 다양성 있는 학습 데이터 수집: 다양한 출처와 다양한 사람들에 의해 만들어진 학습 데이터를 사용하는 것이 중요하다. 다양성 있는 데이터셋은 생성형 AI 모델이 여러 관점을 학습하고 일반화하는 데 도움이 된다. 편향성 문제는 한두가지 방법을 도입한다고 완전히 해결되는 문제가 아니다. 무엇보다 조직 내에서 데이터 및 모델 관리 프로세스를 통해 편향성에 대한 지속적인 관리 감독이 필요하다. 이러한 노력들은 LLM 의 결과에 대한 편향성을 줄이고 보다 공정하고 다양성 있는 결과물을 제공하는 데 도움이 된다.
- 학습 및 테스트 데이터 관리: 학습 데이터와 테스트 데이터 분리해야 한다. 전체 데이터셋의 80%는 학습용으로 나머지 20%는 테스트용으로 분류한다. 이는

학습용 데이터를 이용해 모델을 학습시키고 테스트 데이터셋을 통해 모델의 성능을 평가하기 위함이다. 하지만 단순히 학습과 테스트 데이터셋으로만 나누게 된다면 모델의 성능 검정을 한 번 밖에 할 수 없고, 테스트 데이터에 대한 성능 평가 결과를 토대로 모델을 수정하게 된다면 overfitting 이 발생할 가능성이 높아지게 된다. 따라서 우리는 학습용 데이터셋을 학습용과 검증용으로 나눌수있다. 보통 전체 데이터셋을 학습, 검증, 테스트 데이터셋을 각 6:2:2 비율로 나누어 사용하게 된다. 학습용 데이터셋은 온전히 모델 학습만을 위해 사용된다. 학습용 데이터셋을 통해 모델을 학습시키고 매개변수 등 모델을 수정해서 성능을 높이는 작업을 수행할 수 있다.
검증용 데이터셋은 모델 학습에 직접 관여하지 않으며 학습이 끝난 모델에 적용시켜 테스트 데이터셋을 이용한 모델 평가로 넘어가기 이전에 최종적으로 모델을 fine tuning 할 때 사용한다. 학습이 완료된 모델이라 하더라도 epoch 을 몇 번 수행할 것인지, 또는 learning rate 의 설정은 어떻게 할 것인지 등에 따라 모델의 성능이 달라질 수 있다. 따라서 검증용 데이터셋을 통해 우리가 만든 모델이 테스트 단계에서 그리고 실제 환경에서 높은 성능을 낼 수 있도록 만들어 준다.
테스트 데이터셋은 최종적으로 우리가 만든 모델의 성능을 평가하기 위한 데이터셋이다. 모델이 배포된 이후에 실 환경에서 모델이 얼마나 좋은 성능을 발휘할 수 있을지 판단할 수 있다. 모델이 실환경에서 높은 성능을 발휘하기 위해서는 unseen data 를 잘 처리하는 것이 매우 중요한데, 테스트 데이터셋이 unseen data 역할을 해주어 LLM 모델의 성능을 효과적으로 평가할 수 있도록 도와준다. 테스트 데이터는 이상탐지 모델 정확도 평가를 위해 사전 레이블링(Labeling) 작업 필요하다. LLM 의 경우에는 pretrained 모델을 만들거나 미세조정 과정을 수행하기 어려운 경우가 많기 때문에, 만약 모델 학습을 수행하지 않고 RAG 나 프롬프트 엔지니어링만 수행할 경우 학습 데이터셋을 구축할 필요는 없다.

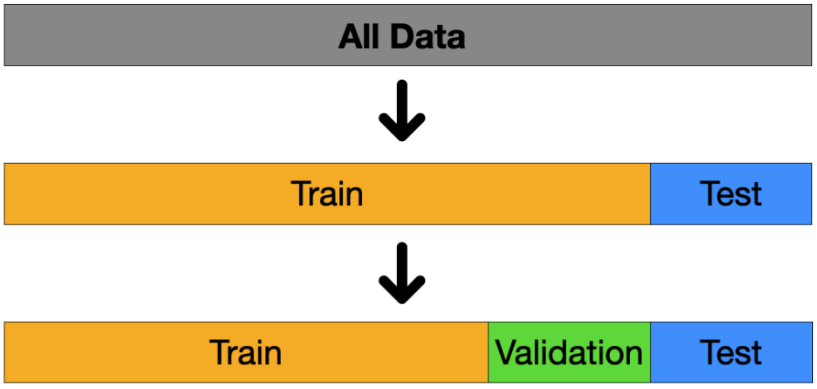


그림 25 Train, Validation, Test 데이터셋 분류