

Prompt Engineering

김준재

ML Engineer



Prompt Engineering Lifecycle

프롬프트 엔지니어링의 전 과정에 대하여 소개합니다.

iupstage Edu stage

개념 소개

프롬프트 엔지니어링이란?

- LLM(Large Language Model)이 특정 작업에 대하여 최상의 결과를 생성하도록 유도하는 프롬프트 개발 과정.
- 프롬프트의 체계적인 설계, 개발, 최적화를 통해 단순히 모델에 질문하는 것을 넘어선 기술적 접근 방식.

프롬프트 엔지니어링의 중요성

- 잘 설계된 프롬프트는 LLM의 응답 품질, 관련성, 정확성, 일관성을 극대화 할 수 있음.
- 또한, 환각 현상(Hallucination)으로 대표되는 LLM의 잠재적 오류나 편향을 최소화 할 수 있음.
- 결과적으로 프롬프트 엔지니어링은 LLM 활용의 성공 여부를 결정짓는 핵심 요소.

iupstage Edustage

라이프사이클 개요

1. 목표 설정

프롬프트를 통해 달성하고자 하는 구체적인 목표와 성공 지표를 정의. (예: 고객 문의 유형 분류 정확도 95% 달성)

2. 프롬프트 설계

• 정의된 목표에 맞춰 초기 프롬프트를 작성, 다양한 설계 기법을 고려하여 초안을 구축.

3. 테스트 및 평가

설계된 프롬프트를 사용하여 LLM의 응답을 생성하고, 미리 정의된 지표에 따라 성능을 평가.

4. 반복 및 개선

● 평가 결과를 바탕으로 프롬프트의 문제점을 분석하고, 개선 방안을 적용하여 프롬프트를 수정. 목표 성능에 도달할 때까지 반복.



목표 설정

문제 정의

• LLM을 통해 해결하고자 하는 구체적인 문제가 무엇인지 명확하게 정의. (예: "사용자의 감정을 분석한다" 보다는 "고객 리뷰 텍스트에서 긍정/부정/중립 감성을 분류한다")

성공 기준 설정

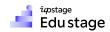
어떤 결과가 '좋은' 결과인지 구체적이고 측정 가능한 기준으로 설정.
 (예: "응답이 유용해야 한다" 보다는 "생성된 요약문이 원문의 핵심 내용을 80% 이상 포함해야 한다")

제약 조건 파악

• 결과물의 길이(예: 100단어 이내), 특정 스타일(예: 격식 있는 어조), 사용 금지 어휘 등 필요한 제약 조건을 명확화.

사용자의 요구사항 이해

● 생성된 결과를 최종적으로 사용할 사용자는 누구이며, 그들이 무엇을 기대하는지 파악. 사용자의 기술 수준, 배경지식 등을 고려.



프롬프트 설계의 핵심 요소

명확한 지시사항

- LLM이 수행해야 할 작업을 모호함 없이 구체적이고 직접적으로 지시.
- 동사 사용을 명확히 하고, 단계적으로 지시하는 것이 효과적. (예: "[주제]에 대한 글을 써줄래?" 보다는 "다음 주제에 대해 500자 내외의 블로그 게시물을 작성해줘: [주제]")

컨텍스트 제공

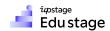
• LLM이 작업을 수행하는 데 필요한 배경 정보, 관련 지식, 이전 대화 내용 등을 충분히 제공.

출력 형식 지정

● 응답이 특정 구조(예: 목록, 표, JSON)나 형식(예: 마크다운)을 따르도록 명시적으로 요구하여 후속 처리 과정을 용이하게 만듦.

예시 활용 (Few-shot)

• 원하는 입력과 출력의 예시를 1개 이상 제공하여 LLM이 패턴을 학습하고 유사한 형식과 내용의 응답을 생성하도록 유도.



프롬프트 설계의 주요 설계 기법

Zero-shot

- 예시를 제공하지 않고, 오직 작업 지시만으로 LLM이 응답하도록 하는 기법.
- 간단한 작업이나 LLM이 이미 잘 학습한 작업에 유용.

Few-shot

• 몇 개의 입력-출력 예시를 프롬프트에 포함시켜 LLM이 해당 패턴을 학습하고 작업을 수행하도록 유도하는 기법.

Chain-of-Thought (CoT)

• 복잡한 추론이 필요한 문제에 대해 LLM이 정답에 도달하기까지의 단계별 사고 과정을 명시적으로 생성하도록 유도하는 기법.

ReAct (Reasoning and Acting)

• LLM이 추론(Reasoning) 과정을 생성하고, 이를 바탕으로 필요한 행동(Acting, 예: 정보 검색, 계산)을 결정하며, 그 결과를 다시 관찰(Observation)하여 최종 답변을 생성하는 과정을 반복하도록 유도하는 기법.

iupstage Edu stage

프롬프트 평가 방법

정성적 평가

- 생성된 응답을 사람이 직접 읽고, 품질, 창의성, 논리성, 스타일 등 주관적인 기준에 따라 평가.
- 초기 탐색 단계나 미묘한 품질 차이를 평가할 때 유용.

정량적 평가

- 미리 정의된 지표(metric)를 사용하여 응답의 성능을 수치적으로 측정. (예: 분류 정확도, ROUGE 점수, BLEU 점수)
- 자동화가 가능하며, 대규모 테스트에 적합.

A/B 테스팅

• 두 가지 이상의 프롬프트 후보군을 동일한 입력 데이터셋에 적용하고, 그 결과를 비교하여 어떤 프롬프트가 더 나은 성능을 보이는지 평가.



프롬프트 평가 지표

정확도 (Accuracy)

• 생성된 정보가 사실과 얼마나 일치하는지, 또는 분류 작업 등에서 정답률이 얼마나 높은지를 측정.

관련성 (Relevance)

생성된 응답이 사용자의 질문이나 주어진 작업과 얼마나 밀접하게 관련되어 있는지를 평가.

완전성 (Completeness)

• 응답이 사용자의 요구사항이나 질문의 모든 측면을 충분히 다루고 있는지를 평가.

유용성 (Utility/Helpfulness)

• 생성된 응답이 실제로 사용자의 문제를 해결하거나 목표 달성에 얼마나 도움이 되는지를 평가.

iupstage Edu stage

프롬프트 개선 전략

질의 구체화/명확화

• 모호하거나 해석의 여지가 있는 지시사항을 더 명확하고 구체적인 표현으로 수정. (예: "자세히 설명해줘" -> "주요 특징 3가지를 설명하고 각각의 장단점을 명시해줘")

구조 최적화

● 프롬프트 내의 정보 순서(예: 지시사항 혹은 컨텍스트 먼저), 역할 부여(예: "당신은 ~ 전문가입니다"), 구분자 사용 등을 변경하여 LLM의 이해도를 높임.

제약조건 추가/수정

• 원치 않는 결과(예: 너무 길거나 짧은 응답, 특정 단어 사용)를 방지하기 위해 제약 조건을 추가하거나 기존 제약 조건을 조정.

예시 추가/수정

• 성능 개선을 위해 더 적절하거나 다양한 예시를 추가하거나 기존 예시를 수정.

01 Prompt Engineering Lifecycle



문제 해결

환각 현상(Hallucination)

- LLM이 사실이 아닌 정보를 그럴듯하게 생성하는 문제.
- 컨텍스트에 관련 정보를 명시적으로 제공하거나,
 생성된 내용을 외부 지식 소스와 비교하여 검증하는 단계를 추가하는 방식으로 완화 가능.

지시 무시

- LLM이 프롬프트의 특정 지시를 따르지 않는 문제.
- 지시를 더 명확하게 하거나, 복잡한 지시는 단계별로 나누어 제시하거나,
 중요한 지시를 프롬프트 앞부분이나 뒷부분에 강조하여 배치하는 것이 도움이 될 수 있음.

응답 형식 불일치

- 요구한 형식(예: JSON, 목록)과 다른 형식으로 응답하는 문제.
- 원하는 출력 형식을 명확히 명시하고, 예시를 통해 형식을 보여주거나, 출력 형식 템플릿을 프롬프트에 포함시키는 방법을 사용할 수 있음.



정리

프롬프트 엔지니어링의 반복성

• 목표 설정부터 평가까지 이어지는 라이프사이클은 한번에 끝나지 않고, 평가와 피드백을 통해 지속적으로 개선되는 순환적 과정.

효과적인 프롬프트의 핵심 요소

명확한 지시, 충분한 컨텍스트, 적절한 예시, 명시적인 출력 형식 지정 등이 고품질 응답을 위한 필수 요소.

평가 & 지속적 개선의 중요성

● LLM 모델 자체의 변화, 사용자 요구의 변화, 새로운 문제 발생 등에 대응하기 위해 프롬프트를 꾸준히 테스트하고 개선하는 노력이 필수적.



Building intelligence for the future of work