**생성형 인공지능 모델과 대화하는 Prompt engineering**

**프롬프트 엔지니어링이란?**

프롬프트 엔지니어링은 LLM (Large Language Model), 즉 대규모 언어 모델에서 원하는 결과를 얻기 위해 prompt를 최적화하는 과정으로 최근 ChatGPT의 부상으로 새롭게 급부상하는 분야입니다. 이러한 프롬프트 엔지니어링의 주요 목표는 사용자의 의도와 원하는 결과를 전달하는 가장 적합한 프롬프트를 만들어 모델의 성능, 정확성과 유용성을 최대화하는 데 있습니다.

**프롬프트란 무엇인가?**

프롬프트는 특정 작업을 수행하도록 생성형 인공지능에게 전달하는 자연어 텍스트입니다. 예를 들어, 신문기사의 내용을 요약하고 싶다면 LLM에게 “다음의 기사를 요약해줘” 라는 텍스트를 입력할 수 있습니다. 1980년대 초반부터 MS-DOS라는 마이크로소프트가 개발한 명령 줄 기반 운영체제가 대중에게 알려지기 시작했습니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

그림 1 마이크로소프트사가 개발한 MS-DOS

하지만 인간은 끊임없이 컴퓨터와 더 쉽게 소통하기 위해 노력하고 있으며 그로 인해 현재 대부분의 컴퓨터 시스템에는 GUI (Graphic User Interface)방식의 OS가 설치되어 있어 직관적이고 쉽게 컴퓨터에게 작업을 지시하고 시각적으로 결과를 확인할 수 있습니다. 하지만, 인간에게 인간이 사용하는 자연어만큼 컴퓨터와 소통할 수 있는 수단은 없습니다. 반면 컴퓨터는 자연어만큼 어려운 소통 수단은 없습니다. 따라서, 자연어 고유의 복잡성과 다양성을 유지하면서 컴퓨터가 가장 잘 이해하고 최적의 결과를 생성할 수 있는 프롬프트를 개발하는 것이 인공지능 시대에 중요한 분야로 급부상하고 있습니다. 본문에서는 프롬프트를 최적으로 설계할 수 있는 유용한 기술을 소개합니다.

**프롬프트 엔지니어링 기법**

프롬프트 엔지니어링은 역동적으로 진화하고 있는 분야입니다. 프롬프트를 미세조정하고 생성형 인공지능에서 원하는 결과를 얻기 위해 언어를 다루는 기술과 창의적인 표현이 모두 필요합니다. 아래에서 프롬프트 구성요소와 여러가지 잘 알려진 기법을 소개합니다.

**프롬프트 구성 요소**

프롬프트를 설계하다 보면 몇 가지 공통된 구성요소들이 있다는 것을 알게 됩니다. 다음은 프롬프트에 주로 포함되는 요소들입니다.

* 지시: 생성형 인공지능이 수행할 특정 작업이나 지시사항에 관한 부분
* 문맥: 더 나은 응답을 얻기 위해 생성형 인공지능에 전달하는 내용
* 입력 데이터: 사용자의 질문
* 출력 형태 지시자: 생성형 인공지능이 결과를 출력할 때 형식 지정

예를 들어, 다음과 같이 프롬프트를 구성하여 LLM에 입력을 전달할 수 있습니다. 아래의 프롬프트는 위에서 언급한 구성요소가 모두 포함되어 있습니다.

“아래 문맥을 보고 질문에 답변해줘. 답변은 짧고 간결하게 1,000자 이내로 해줘. 그리고 모르는 질문에 대해서는 “””답변을 찾지 못해 잘 모르겠습니다.”””로 답변해줘

문맥: 페니실린은 1928년 9월 28일 새벽에 영국의 물리학자 알렉산더 플레밍이 우연히 발견했다. 실험을 위해 샬레에 포도상구균을 배양하고 휴가를 갔다 왔는데 하필 샬레의 뚜껑을 제대로 닫지 않았고 다시 연구실에 돌아와보니 어디선가 날아온 괴상한 곰팡이가 포도상구균을 전부 먹어 치워버린 것이다. 따라서 세균을 먹어버린 그 곰팡이의 성질을 연구함으로써 페니실린을 발견하게 된 것이다.

질문: 페니실린은 어떻게 발견되었나?”

프롬프트가 반드시 위의 모든 구성을 포함할 필요는 없으며 어떻게 구성되어야 할 지는 어떤 목적을 가진 작업인지에 따라 달라질 수 있습니다. 이어지는 가이드에서 좀 더 구체적인 예시를 다루어 보겠습니다. 본 글을 읽고 있는 독자는 이제 프롬프트를 개선하여 다양한 작업에서 더 나은 결과를 얻는 것이 중요하다는 것이 분명해졌을 것입니다. 앞으로 더 복잡하고 다양한 작업을 수행할 수 있는 고급 프롬프트 기법을 소개하겠습니다.

**제로-샷 프롬프트 (zero-shot prompting)**

인공지능은 이미 대량의 데이터로 학습되어 있어 아래와 같이 다양한 질문에 답변을 할 수 있습니다.

아래 텍스트를 중립, 부정 또는 긍정으로 분류하시오.

텍스트: 오늘 날씨가 아주 맑고 좋다.

위이 프롬프트에서는 인공지능 모델에 어떠한 예제도 전달하지 않고 있으며 이러한 프롬프트 방식을 제로-샷 프롬프트라고 합니다. 하지만 프롬프트에 몇 가지 질문과 함께 예제를 전달하여 모델이 사람이 원하는 결과에 더 적합하게 응답할 수 있도록 프롬프트를 튜닝할 수 있습니다.

**퓨-샷 프롬프트 (few-shot prompting)**

LLM은 위에서 언급한 제로-샷 프롬프트만으로도 뛰어난 결과를 출력할 수 있지만 좀 더 복잡하고 다양한 작업에서는 몇 가지 예제를 전달하여 원하는 결과를 얻을 수 있습니다. 예를 들어, 고객의 피드백이 긍정적인 부정적인지 구분하는 지시를 내리기 위해 몇 가지 예제를 제공하고 아직 구분되지 않은 피드백을 보여줍니다.

정말 맛없어: 부정

이거 대박인데: 긍정

가격이 너무 비싸: 부정

음식이 따끈해: 긍정

음식이 뜨거워:

위의 프롬프트를 ChatGPT에 입력하면 아래와 같이 주어진 몇 가지 사례를 바탕으로 답변을 생성하는 것을 볼 수 있습니다.

“음식이 뜨거워”는 주어진 문맥에서 긍정적인 특성을 나타내고 있습니다. 따라서 음식이 뜨거워는 긍정입니다.

일반적인 작업에는 퓨-샷 프롬프트 기법이 많은 효과를 가져오지만 복잡한 추론 작업을 처리할 때는 유용하지 못할 때가 있습니다.

**사고의 사슬 프롬프트 (chain-of-thought prompting)**

본 기법은 좀 더 복잡한 추론작업에 유용하게 사용할 수 있는 기법입니다. 예를 들어 ChatGPT에 아래와 같은 질문을 하고 어떻게 답변을 하는 지 보면 LLM이 추론 문제에 약하다는 것을 알 수 있습니다.

나는 시장에 가서 사과 10개를 샀어. 사과 2개를 이웃에게 주고, 2개를 수리공에게 주었어. 그리고 사과 5개를 더 사서 1개는 내가 먹었어. 나는 몇 개의 사과를 가지고 있니?

이 때 “단계별로 생각해 보자”라는 문구를 프롬프트에 추가하여 다시 입력해 보겠습니다.

나는 시장에 가서 사과 10개를 샀어. 사과 2개를 이웃에게 주고, 2개를 수리공에게 주었어. 그리고 사과 5개를 더 사서 1개는 내가 먹었어. 나는 몇 개의 사과를 가지고 있니? 단계별로 생각해 보자.

아래의 ChatGPT 답변을 보면 위와 같은 프롬프트 방식이 복잡한 추론 문제에 매우 유용하다는 것을 알 수 있습니다.

단계별로 생각해 보겠습니다.

1. 시작할 때 사과를 10개 샀습니다.
2. 이웃에게 2개를 주었으므로 10-2=8개의 사과가 남습니다.
3. 수리공에게도 2개를 주었으므로 8-2=6개의 사과가 남았습니다.
4. 추가로 5개의 사과를 더 샀습니다.
5. 이전에 남은 6개의 사과에 5개를 더해서 총 6+5=1개의 사과가 있습니다.
6. 마지막으로 1개의 사과를 먹었으므로 11-1=10개의 사과를 가지고 있습니다.

따라서 당신은 10개의 사과를 가지고 있습니다.

이제 좀 더 복잡한 지식을 기반으로 답변을 생성하는 모델을 구축하기 위한 프롬프트 기법을 소개하겠습니다.

**Retrieval Augmented Generation (RAG)**

LLM은 이미 방대한 데이터를 기반으로 사용자의 질문에 꽤 효과적인 응답을 생성할 수 있습니다. 그러나, 만약 학습하지 못한 데이터에 기반한 질문이 입력된다면 정확한 답변을 하지 못합니다. 현재 시점 기준으로 ChatGPT-3.5는 2022년 1월까지의 정보만 학습되어 있는 상태이며 최근의 지식은 알고 있지 않습니다. 예를 들어, 2022년 올림픽에 관한 기사 내용을 기반으로 아래와 같은 질문을 해보겠습니다.

2022년 동계 올림픽에서 어떤 선수가 금메일을 땄어?

ChatGPT는 아래와 같이 답변을 제대로 생성하지 못하는 것을 볼 수 있다. 최근 지식을 학습하고 있지 않아 정확한 답변을 제시하지 못하고 있는 것입니다.

죄송합니다만, 제가 마지막으로 업데이트된 시점은 2022년 1월이라 동계 올림픽의 최신 결과를 알 수 없습니다. 최신 정보를 확인하려면 올림픽 공식 웹사이트나 신뢰할 수 있는 뉴스 소스를 참고하시기 바랍니다.

그러나 만약 ChatGPT에게 2022년 동계올림픽 정보를 질문과 함께 전달하면 해당 정보를 바탕으로 원하는 답변을 얻을 수 있습니다. 예를 들어 위키피디아에서 2022년 동계올림픽 정보를 복사하여 아래와 같이 질문과 함께 추가하여 프롬프트를 작성한다면 해당 정보를 바탕으로 적절한 답변을 생성하며 이러한 프롬프트 기법을 RAG라고 합니다.

지시: 아래에 주어진 2022년 동계 올림픽에 관한 기사를 바탕으로 질문에 답하시오. 만약 적합한 답변을 찾지 못하면 “나는 모른다” 라고 답변을 하시오.

기사: {2022년 동계 올림픽 위키피디아 내용}

질문: 2022년 동계 올림픽 컬링 대회에서 누가 금메달을 수여했나요?

RAG는 외부 소스 정보를 Retrieve해서 Augment하고 답변을 Generation한다는 의미로, 내부 사내 데이터, 공개되지 않은 데이터 혹은 전문적인 데이터를 기반으로 사용자 질의에 답변하는 프롬프트 기법입니다. 이것은 마치 오픈 북으로 시험을 치르는 방식과 유사하며 LLM의 Hallucination 문제를 줄일 수 있는 효과가 있습니다.

**프롬프트 엔지니어링 가이드**

위에서 몇 가지 프롬프트를 작성하는 기술에 대해 설명하였습니다. 이제 프롬프트를 설계할 때 알면 좋은 몇 가지 유용한 팁을 제시하겠습니다.

**더욱 정확한 답변을 얻기 위해 매우 상세하게 요청하라**

미팅회의록을 요약해달라는 프롬프트를 작성할 때 단순히 “미팅회의록 요약해줘” 라는 프롬프트보다 더욱 목적적합한 결과를 얻기 위해서는 매우 상세하게 요청하는 것이 바람직합니다. 예를 들어, “화자마다 각각 논의 내용을 키포인트와 함께 목록으로 뽑아주고 action item을 따로 목록으로 만들어줘” 라고 프롬프트를 작성하는 것이 유용할 수 있습니다.

**모델에게 페르소나(persona)를 적용하라**

프롬프트를 작성할 때 모델에게 적합한 역할을 부여함으로써 좀 더 유용한 결과를 얻을 수 있습니다. 예를 들어, “당신은 유명한 국제부 기자입니다. 따라서 기자를 작성하듯 질문에 답변하시오” 라는 메시지를 추가할 수 있습니다.

**문맥은 구분자를 이용하여 명확하게 구분하라**

문맥과 같이 질문에 답변하기 위해 참조할 내용은 구분자를 활용하여 명확하게 구분하는 것이 좋습니다. 예를 들어. 다음과 같이 프롬프트를 작성할 수 있습니다.

아래의 따옴표로 구분된 내용을 서론, 본론, 결론의 형식으로 500자 이내로 요약해줘.

“””{문맥}”””

**필요하다면 단계적으로 작업을 지시하라**

복잡한 작업을 지시할 경우에는 단계적으로 구체적인 지시를 전달하는 것이 좋습니다. 예를 들어, 다음과 같이 프롬프트를 작성할 수 있습니다.

사용자의 입력에 응답하기 위해 다음의 과정을 단계적으로 수행하시오.

1단계: 아래 따옴표로 구분되는 문장을 한 문장으로 간략히 요약하시오.

“””문자”””

2단계: 1단계에서 요약한 문장을 “번역:”이라는 prefix를 첨부하여 영어로 번역하시오.

**원하는 결과를 얻기 위해 몇 가지 예시 문장을 제공하라**

앞서 설명한 퓨-샷 기법을 활용하여 몇 가지 문장을 함께 전달하면서 작업을 지시할 수 있습니다. 예를 들어, 아래와 같이 원하는 말투로 답변을 생성하도록 요청할 수 있습니다.

아래의 주어진 예시를 참고하여 일관된 말투로 “오늘 점심 메뉴 정말 맛있었다”에 대한 댓글 작성해줘.

“””예시: 와~ 우리 강아지 너무 이쁘네~, ㅎㅎ 잘했어”””

**모델에게도 생각할 시간을 줘라**

이것은 앞서 기술한 chain-of-thought 기법과 유사한 방식으로 모델에게 바로 답변을 요구하는 것이 아니라 추론할 수 있는 방법을 함께 제시하여 좀 더 정확한 답변을 생성할 수 있습니다.

**맺음말**

본 기사에서는 프롬프트 엔지니어링의 정의와 기법 그리고 유용한 몇 가지 팁을 제공했습니다. 이 글을 읽는 독자가 언어모델을 활용하여 애플리케이션을 작성하는 개발자 입장이라면 무엇보다 비용과 보안 측면에서 프롬프트 엔지니어링 기법을 중요하게 고려할 필요가 있습니다. 언어모델은 입력과 출력 문장의 길이를 토큰으로 표현하고, 토큰의 길이에 비례하여 비용이 발생하므로 cost-sensitive한 서비스의 경우 토큰 수 조절을 위해 적합한 프롬프트 기법을 활용하고 연구해야 합니다. 아울러, 적대적 프롬프트를 사용하여 의도하지 않은 기밀 사항 등이 유출될 수 있습니다. 따라서, 사용자의 입력과 언어 모델의 출력을 모니터링하고 필터링 할 수 있는 기법을 개발하고 적용하여 예기치 못한 사고에 대비해야 할 것입니다.

참고사이트:

[**https://platform.openai.com/overview**](https://platform.openai.com/overview)

[**https://www.promptingguide.ai/**](https://www.promptingguide.ai/)

[**https://python.langchain.com/docs/modules/model\_io/prompts/**](https://python.langchain.com/docs/modules/model_io/prompts/)