

Prompt Engineering for GPT

송태영

GPT 모델 개요

- GPT(GPT: Generative Pre-trained Transformer)는 딥러닝 모델 중 하나인 Transformer 모델을 사용하여 생성 모델링에 활용되는 대표적인 모델
- GPT 모델은 OpenAI에서 개발하였으며, 주로 자연어 처리(NLP) 분야에서 활용
- GPT 모델은 크게 GPT-1, GPT-2, GPT-3으로 나눌 수 있으며, 각 모델은 모델의 크기와 학습에 사용된 데이터 양에 따라 구분
 - GPT-1은 117M 개, GPT-3는 175B 개의 파라미터
- GPT 모델은 생성 모델링에서 활용되는데, 이는 주어진 데이터를 기반으로 문장, 단락, 문서 등의 텍스트를 자동으로 생성하는 것을 의미
- GPT 모델은 다양한 자연어 처리(NLP) 작업에 활용
 - 예를 들어, 기계 번역, 챗봇, 요약, 질문 답변(QA), 자동 요약, 키워드 추출 등 다양한 작업에 적용

소스 : [링크](#)

GPT 모델의 단점

- GPT 모델은 학습 데이터에 따라 생성 결과가 크게 달라 짐.
 - 학습 데이터의 질과 양이 중요한 역할, 학습 데이터에 포함되어 있지 않은 문장에 대해서는 정확한 생성이 어렵다.
- GPT 모델은 학습 데이터에서 발견되는 편향성(bias)을 반영할 수 있다.
 - 생성된 결과물이 특정 인종, 성별, 지역 등에 대한 편견을 반영할 수 있다는 것을 의미.
- GPT 프롬프트 엔지니어링은 생성된 결과물이 항상 일관성 있는 의미를 갖지는 않는다.
 - 모델이 문맥을 제대로 이해하지 못하는 경우에 발생할 수 있다.
- GPT 프롬프트 엔지니어링은 모델의 크기와 연산량이 매우 크다.
 - 모델을 학습하고 사용하는 데에 많은 컴퓨팅 자원과 시간이 필요.
- 이러한 단점들은 GPT 프롬프트 엔지니어링의 사용을 제한할 수 있지만, 이러한 문제들에 대한 연구와 개선을 통해 보다 안정적이고 신뢰성 높은 자동완성 시스템을 구축할 수 있을 것으로 예상.

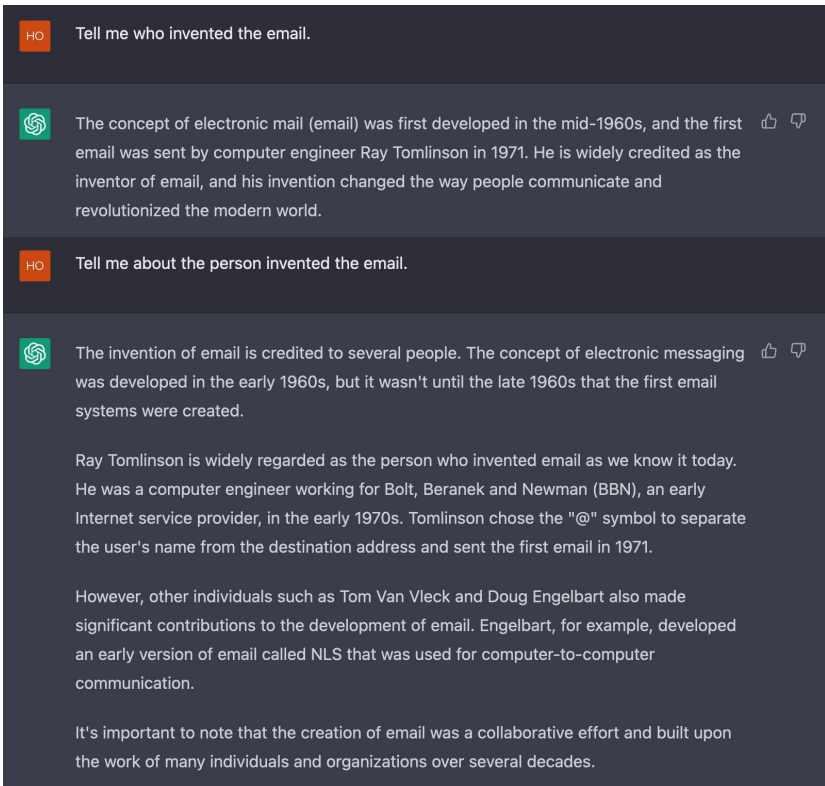
GPT 모델의 개선 방향

- 데이터의 품질과 양을 높여야 한다.
 - GPT 모델의 자연어 생성 능력을 높이기 위해서는 대규모의 고품질 데이터셋이 필요. 데이터셋 구축 및 데이터 전처리 기술의 발전이 필요.
- 편향성 문제를 해결할 수 있는 방법을 모색해야
 - 특정 인종, 성별, 지역 등에 대한 편견이 반영되지 않도록, 학습 데이터의 다양성을 고려하는 것이 중요. 또한, 생성된 결과물을 사람이 검수 하는 등의 방법을 통해 편향성을 줄일 수 있는 방법도 고려.
- 문맥을 이해하고 일관성 있는 의미를 갖도록 모델을 개선해야
 - GPT 모델의 다양한 구조 개선 및 **pre-training** 방법론 등을 연구하여, 모델이 문맥을 이해하고 일관성 있는 의미를 갖도록 하는 방법을 모색해야
- 모델의 크기와 연산량을 줄이는 방법을 연구해야
 - 모델의 크기와 연산량이 클수록 학습과 사용에 필요한 컴퓨팅 자원이 많이 필요, 모델의 크기와 연산량을 줄일 수 있는 경량화(**lightweight**) 기술 등을 연구하여, 모델의 효율성을 개선해야.

GPT의 프롬프트 엔지니어링(Prompt Engineering)

- 생성 인공지능(**Generative AI**) 분야에서 프롬프트(Prompt)는 거대 언어 모델(**Large Language Model; LLM**)로부터 응답을 생성하기 위한 입력값을 의미
- 프롬프트 엔지니어링(**Prompt Engineering**)은 거대 언어 모델로부터 높은 품질의 응답을 얻어낼 수 있는 프롬프트 입력값들의 조합을 찾는 작업

왜 프롬프트 엔지니어링이 필요할까?



HO Tell me who invented the email.

The concept of electronic mail (email) was first developed in the mid-1960s, and the first email was sent by computer engineer Ray Tomlinson in 1971. He is widely credited as the inventor of email, and his invention changed the way people communicate and revolutionized the modern world.

HO Tell me about the person invented the email.

The invention of email is credited to several people. The concept of electronic messaging was developed in the early 1960s, but it wasn't until the late 1960s that the first email systems were created.

Ray Tomlinson is widely regarded as the person who invented email as we know it today. He was a computer engineer working for Bolt, Beranek and Newman (BBN), an early Internet service provider, in the early 1970s. Tomlinson chose the "@" symbol to separate the user's name from the destination address and sent the first email in 1971.

However, other individuals such as Tom Van Vleck and Doug Engelbart also made significant contributions to the development of email. Engelbart, for example, developed an early version of email called NLS that was used for computer-to-computer communication.

It's important to note that the creation of email was a collaborative effort and built upon the work of many individuals and organizations over several decades.

- **who**를 **about the person**으로 바꿨을 뿐인데 결과물의 양과 질이 판이하다. 약간의 표현 차이 만으로도 짧고 간결한 답이 풍부한 맥락으로 구성된 에세이 수준의 결과물로 바뀌었다. 이처럼 내게 필요한 더 높은 품질의 응답을 얻으려면 해당 언어 모델이 잘 이해할 수 있는, 정제된 언어로 구조화된 프롬프트를 구성하는 것이 중요하다.
- 생성 인공지능이 기반을 둔 언어 모델의 특성에 따라 적합도 높은 결과물을 얻어내는 프롬프트의 유형도 달라진다. 특정 모델로부터 특정 답안을 얻어내는 데에 완전히 최적화 된 프롬프트 해킹 수준의 기술은 이 글이 다루려는 범위를 넘어선다. 언어 모델에 대한 기술적 원리에 대해 상세히 파악하기 어려운 일반 사용자 입장에서, 프롬프트 엔지니어링이란 결국 수많은 실험의 연속일 수밖에 없다.

GPT 프롬프트 엔지니어링의 예시

- 검색 엔진이나 메신저, 이메일 등에서 자동완성 기능으로 사용
 - 예를 들어, 구글 검색창에서 "오늘 뭐하지?"라는 질문을 입력하면 **GPT** 모델이 이어서 "영화 볼까요?"라는 제안
- 대화형 챗봇
 - 고객상담, 의료상담, 교육 등 다양한 분야에서 활용. **GPT** 모델이 자연스러운 대화를 생성하여 사용자와 상호작용할 수 있으며, 상대방의 발언을 이해하고 그에 적절한 답변 생성
- 문장 생성, 소설이나 시 등의 창작 작품에 활용
 - **GPT** 모델이 이전 문장을 이해하고 이어서 적절한 문장을 생성할 수 있기 때문에, 작가는 보다 쉽고 빠르게 창작
- 번역기
 - **GPT** 모델이 이전 문장을 이해하고 이어서 번역 결과를 생성할 수 있기 때문에, 번역 속도와 정확도가 개선.

GPT 프롬프트 엔지니어링의 명령문 형식

- **Generate [숫자] [텍스트]**
 - 여기서 [숫자]는 생성할 문장의 개수를 의미하며, [텍스트]는 문장을 생성하기 위한 시작 문장으로 사용되는 텍스트
 - 예를 들어 "Generate 5 I like to"와 같은 명령은 "I like to swim.", "I like to read books.", "I like to travel." 등 5개의 문장을 생성
- **"Complete [텍스트]"**
 - 주어진 텍스트를 자동으로 완성하는 기능
 - 예를 들어 "Complete I like to sw"와 같은 명령은 "I like to swim."이나 "I like to swing." 등의 완성된 문장을 생성
- **"Answer [질문]"**
 - 주어진 질문에 대한 답변을 자동으로 생성하는 기능
 - 예를 들어 "Answer What is the capital of France?"와 같은 명령은 "The capital of France is Paris."와 같은 답변을 생성.

대표적인 10개의 예시 - 1/2

- **Generate** [숫자] [텍스트]
 - "Generate 5 I love to" : "I love to dance.", "I love to cook.", "I love to travel."
- **Complete** [텍스트]
 - "Complete I enjoy listening to" : "I enjoy listening to music.", "I enjoy listening to podcasts.", "I enjoy listening to audiobooks."
- **Answer** [질문]
 - "Answer What is the capital of South Korea?" : "The capital of South Korea is Seoul."
- **Translate** [텍스트] to [언어]
 - "Translate Good morning to Spanish" : "Buenos dias"
- **Explain** [개념]
 - "Explain What is artificial intelligence?"

대표적인 10개의 예시 - 2/2

- Paraphrase [텍스트]
 - "Paraphrase This book is very interesting." : "This book is quite fascinating."
- Summarize [텍스트]
 - "Summarize the article about climate change" :
- Compare [개념1] and [개념2]
 - "Compare democracy and dictatorship"
- Convert [숫자1] [단위1] to [단위2]
 - "Convert 100 kilometers to miles"
- Predict [데이터]
 - "Predict the stock price of Apple for the next month"

쉽고 간결한 표현을 사용하자.

- 두 프롬프트는 완전히 똑같은 지시 내용을 담고 있다. 불필요한 미사여구가 많은 첫 번째보다는 오히려 두 번째 프롬프트가 원하는 응답을 얻어낼 가능성이 더 높다. 거대 언어 모델에서는 작은 입력값의 차이가 큰 변화로 이어질 수 있다. 가급적 쉽고 간결한 표현을 쓰도록 하자.
 - 혹시 괜찮다면 제임스 웹 우주망원경이 무엇인지를 가지고 이제 갓 9살이 된 어린 아이도 알아들을 수 있을 만큼 쉽게 설명해주겠니?
 - 제임스 웹 우주망원경을 9살 어린이에게 설명해줘.

'열린' 질문보단 '닫힌' 지시문이 좋다.

- 프롬프트의 내용은 응답 결과의 형식에도 영향을 미친다. 작업 목적에 맞는 일정한 형식의 응답을 원한다면 가급적 '닫힌' 지시문의 형태로 프롬프트를 작성하는 것이 좋다.
 - 로봇이 의사의 역할을 대체할 수 있을까?
 - 로봇이 의사의 역할을 대체할 수 있는가에 대해 에세이를 써줘.
- 후자가 좀 더 체계적으로 정리된 응답을 얻어낼 수 있다.

수행할 작업의 조건을 구체적으로 명시하자.

- ChatGPT에게 에너지 드링크에 대한 카피라이팅을 시킨다고 가정해보자.
 - Write some messages for an energy drink
 - Develop 5 key messages and 5 slogans for an energy drink targeting young adults aged 18-30.
- 첫 번째 프롬프트 만으로도 쓸 만한 문구들을 얻는 데엔 문제가 없을 것이다. 그러나 두 번째 프롬프트처럼 작업 수행에 필요한 몇 가지 가이드라인을 추가하는 것만으로 훨씬 더 구조화 된 응답을 얻어낼 수 있다.
- 가이드라인은 구체적이고 명료해야 한다. **GPT** 같은 언어 모델이 알아서 판단해야 할 여지를 줄일수록 원하는 양식의 응답을 얻을 가능성이 높아진다.
 - ‘API를 설명해줘’
 - ‘나는 코딩 교육 스타트업의 마케터야. 중학생을 대상으로 **API**의 개념을 설명하는 자료를 만들어줘. 5단락 정도 되면 좋겠고, 구체적인 예시를 2개 이상 들어줘’

챗GPT에게 역할을 부여

- 챗GPT에게 역할을 부여한 다음 대화
- '코딩 교육 스타트업의 마케팅 팀장처럼 행동해 줘. 네가 질문을 하면 내가 대답을 할게. 첫 번째 질문은 알아서 생성해 줘.'
- 관련 프롬프트 : [링크](#)
 - Act as a Linux Terminal
 - Act as an English Translator and Improver
 - Act as a JavaScript Console
 - Act as position Interviewer
 - Act as an Excel Sheet
 - Act as a English Pronunciation Helper
 - Act as a Spoken English Teacher and Improver
 - Act as a Travel Guide

지시의 맥락을 함께 제공하자.

"Let's write a JavaScript function 'create_post()'.
Fetch the form inputs to url '/create'."

Let's write a JavaScript function 'create_post()' with given conditions:

- Get the input data from '#create-post-input'.
- Use `fetch` method to send the data to url '/create' with POST method without `await` calls.
- Parse the response to JSON format if the response status is `ok`.
- If the parsed response data has a 'message' or 'error' value, log it to the console.
- Add comments to your code.

원하는 결과물 형식의 예시를 함께 입력하자.

Suggest three names for a horse that is a superhero.

Suggest three names for an animal that is a superhero.

Animal: Cat

Names: Captain Sharpclaw, Agent Fluffball, The Incredible Feline

Animal: Dog

Names: Ruff the Protector, Wonder Canine, Sir Barks-a-Lot

Animal: Horse

Names:

충분히 실험하자.

- 사용자가 필요로 하는 작업의 속성에 따라 프롬프트 엔지니어링의 디테일은 얼마든지 달라질 수 있다.
- **GPT**와 같은 언어 모델로부터 필요한 답안을 원하는 양식과 분량으로 적절하게 얻어내려면, 그러한 결과값에 이를 수 있는 자신만의 프롬프트 작성법을 계속해서 실험해야 한다.
- 어디서부터 시작해야 할지 막막하다면, 아래 링크를 참고해보자. 전세계의 많은 사용자들이 **ChatGPT**를 활용하면서 작성한 각종 프롬프트 예시들을 레시피처럼 활용할 수 있다.
- <https://github.com/f/awesome-chatgpt-prompts>