



Reasoning Prompt

:언어모델의 수학적 추론

5/5, 컴퓨터학의 이해, 최종선
AI융합학부, 20213093, 정현우

목차

- 1 B a c k g r o u n d
- 2 R e a s o n i n g P r o m p t
- 3 문 제 점
- 4 외 부 정 보 사 용
- 5 실 제 사 용 예 시
- 6 전 망 및 소 감



Part 1

Background

기존 학습 방법: Fine-tuning

기존의 대표적인 방식으로 Fine-tuning이 있다. Fine-tuning은 대량의 데이터를 통해 학습된 pre-trained 모델을 가져와 **더 작은 대상 데이터에서 추가로 훈련하는 프로세스**입니다. Fine-tuning의 목적은 pre-trained 모델의 원래 기능을 유지하면서 보다 전문적인 사용 사례에 맞게 조정하는 것입니다. [1]

이의 장점은 모델을 처음부터 학습 시키지 않아도 된다는 장점이 있습니다. 일반적으로 대규모 코퍼스로 pre-trained 모델은 다양한 작업에 대한 일반적인 지식을 가지고 있지만, 특정 작업에 최적화되어 있지는 않기 때문입니다.

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.

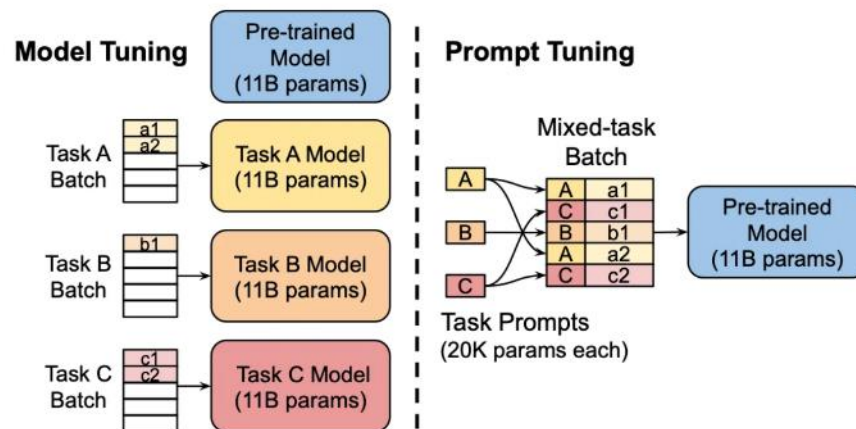


Fine-tuning vs Prompt-tuning

기존의 AI의 학습인 Fine-tuning은 Pre-training된 모델을 각 task A, B, C에 해당하는 데이터셋을 이용하여 학습하는 것으로 모델의 weight가 변경됩니다. 즉, 추가적인 데이터로 학습을 해야 합니다.

반면 프롬프트 튜닝은 Pre-training된 모델의 weight를 변경하지 않고 다양한 작업의 예제와 프롬프트가 하나의 모델을 통해 학습 없이 다양한 task를 수행합니다.[3]

즉, 언어모델에게 입력을 다르게 넣어주어서 다양한 작업을 수행하게 하는 것입니다.



What is prompt engineering?

프롬프트 엔지니어링(prompt engineering)의 주요 업무는 인공지능의 역량을 발휘하도록 지시어를 적합하게 내려주는 것입니다. 프롬프트 엔지니어링은 **대형 언어 모델에게 일처리를 지시**합니다[4]. 또한, 다양한 응용 프로그램 및 연구 주제에 대해 **언어 모델(LM)을 효율적으로 사용하기 위한 프롬프트를 개발하고 최적화하기 위한 비교적 새로운 분야**입니다 [5].

프롬프트 엔지니어링은 기술적으로 어려워 보이지 않을 수 있지만, 이를 잘 활용하기 위해서는 언어모델의 특성을 잘 이해하고 어떤 프롬프트가 잘 동작하는지 다양한 시도가 있어야 합니다.



Prompt 사용 예제

요약, 텍스트 생성, 번역과 같은 작업은 언어모델을 일상에서 많이 사용하는 작업들입니다.

언어모델은 이러한 작업에 강점을 가지고 우리에게 좋은 도구로써 작용합니다.

Prompt Engineering in Large Language and Vision-Language Models ▾



HJ 논문 요약해줘

이 논문은 대규모 언어 모델(Large Language Models, LLMs)과 시각-언어 모델(Vision-Language Models, VLMs)에서 프롬프트 엔지니어링 기술에 대한 체계적인 조사를 제공합니다. 주요 내용은 다음과 같습니다:

1. 프롬프트 엔지니어링은 모델 파라미터를 수정하지 않고 LLM과 VLM의 성능을 향상시키는 데 필수적인 기술로 부상했습니다.
2. 프롬프트는 모델에 컨텍스트를 제공하는 자연어 지시사항이나 관련 지식을 활성화하는 학습된 벡터 표현일 수 있습니다.

HJ 위 내용으로 대본 작성해줘

안녕하세요, 여러분. 오늘은 자연어 처리에서 중요한 역할을 하는 **Fine-tuning**에 대해 알아보도록 하겠습니다.

Fine-tuning은 사전 학습된 언어 모델을 특정 작업에 적용하기 위해 추가 학습을 수행하는 과정입니다. 대규모 코퍼스로 사전 학습된 모델은 일반적인 언어 이해 능력을 갖추고 있지만, 특정 작업에 최적화되어 있지는 않습니다. 이때 **Fine-tuning**을 통해 모델을 작업별 데이터셋으로 재학습시켜, 해당 작업에 특화된 모델을 만들 수 있습니다.

Fine-tuning은 BERT, GPT와 같은 대규모 언어 모델의 성능을 크게 향상시키는 데 기여했으며, 다양한 자연어 처리 작업에 적용되고 있습니다. Devlin et al.의 BERT 논문, Radford et al.의 GPT-2 논문 등에서는 **Fine-tuning**을 통해 언어 모델을 다양한 작업에 적용할 수 있음을 보여주었습니다.

그러나 **Fine-tuning**에는 몇 가지 단점도 존재합니다. 먼저, 계산 비용 측면에서 모델 전체를 재학습해야 하므로 상당한 자원과 시간이 소모됩니다. 또한, 충분한 양질의 작업별 데이터셋이 필요하며, 과적합 문제가 발생할 수 있습니다. **Fine-tuning** 과정에서 모델이 이전에 학습했던 지식을 잊어버리는 캐터스트로픽 포getting 현상도 나타날 수 있습니다.

이러한 단점을 해결하기 위해, **Few-shot learning**, **Meta-learning**, **Adapter** 등의 대안적인 방법들이 연구되고 있습니다. 또한 프롬프트 엔지니어링을 통해 **Fine-tuning** 없이도 언어 모델을 다양한 작업에 적용하려는 시도가 이루어지고 있습니다.

수학 문제를 풀어보면?

하지만, 수학 문제를 넣거나 정밀한 계산이 필요한 문제(물리, 화학 등)를 넣으면 언어모델은 논리적인 접근을 하지만 중간 과정에서 틀리며, 신뢰할 수 없는 답을 만들어낸다.

19. 함수 $f(x) = \sin \frac{\pi}{4}x$ 라 할 때, $0 < x < 16$ 에서 부등식

$$f(2+x)f(2-x) < \frac{1}{4}$$

을 만족시키는 모든 자연수 x 의 값의 합을 구하시오. [3점]

HJ 텍스트 추출해서 풀어봐

주어진 함수 $f(x) = \sin(\pi x/4)$ 와 부등식 $f(2+x)f(2-x) < 1/4$ 에 대해 만족시키는 모든 자연수 x 의 값의 합을 구하는 문제입니다.

1. 먼저 함수 $f(x)$ 의 주기를 찾습니다. $f(x+8) = \sin(\pi(x+8)/4) = \sin(\pi(x/4 + 2)) = \sin(\pi x/4) = f(x)$ 따라서 $f(x)$ 의 주기는 8입니다.
2. $0 < x < 16$ 범위 내에서 x 값을 증가시키며 부등식이 만족되는 x 값들을 찾습니다. $x = 1, 2, 3, \dots, 15$ 각각에 대해 $f(2+x)f(2-x) < 1/4$ 을 계산해봅니다.
3. 부등식을 만족하는 x 의 값은 1, 2, 3, 4, 9, 10, 11, 12 입니다.
4. 이들의 합은 $1 + 2 + 3 + 4 + 9 + 10 + 11 + 12 = 52$ 입니다.

따라서 답은 52입니다.

옆의 예시에서 24년도 수능 문제를 풀게 하였다. 텍스트를 잘 이해하고 문제에 대해 접근해보려고 합니다. (실제 답: 32)

하지만, 이 문제에서 중요한 것은 삼각함수 변환이지만, 이러한 주요 개념을 사용하지 않고 잘못된 접근을 합니다.

즉, 그럴싸해 보이는 오답을 만들어내 버리고 사용자는 더 혼란스러워집니다.

수학 문제를 풀어보면?

HJ 아닌거 같은데 다시 풀어봐

다시 한번 풀어보겠습니다.

주어진 조건:

$$f(x) = \sin(\pi x/4), 0 < x < 16$$

$$f(2+x)f(2-x) < 1/4$$

1. $f(x)$ 의 주기를 찾습니다. $f(x+8) = \sin(\pi(x+8)/4) = \sin(\pi x/4 + \pi/2) = \cos(\pi x/4) = \sin(\pi(x+4)/4) = f(x+4)$ 따라서 $f(x)$ 의 주기는 4입니다.
2. $0 < x < 16$ 범위에서 부등식을 만족하는 x 값을 찾습니다. $f(2+x)f(2-x) = \sin(\pi(2+x)/4)\sin(\pi(2-x)/4) < 1/4 = (\sin(\pi x/4 + \pi/2)\sin(\pi/2 - \pi x/4)) < 1/4 = (\cos(\pi x/4)\cos(\pi x/4)) < 1/4$

$\cos(\pi x/4)$ 의 값이 $1/\sqrt{2}$ 보다 작으면 위 부등식이 성립합니다.

3. $0 < x < 16$ 에서 $\cos(\pi x/4) < 1/\sqrt{2}$ 를 만족하는 x 의 범위는 $4 < x < 12$ 입니다.
4. 따라서 만족하는 자연수 x 의 값은 5, 6, 7, 8, 9, 10, 11 이고, 이들의 합은 $5 + 6 + 7 + 8 + 9 + 10 + 11 = 56$ 입니다.

언어모델에게 대화 하듯이 요구하면 결국 원하는 답을 얻어내지 못할 가능성이 높습니다.

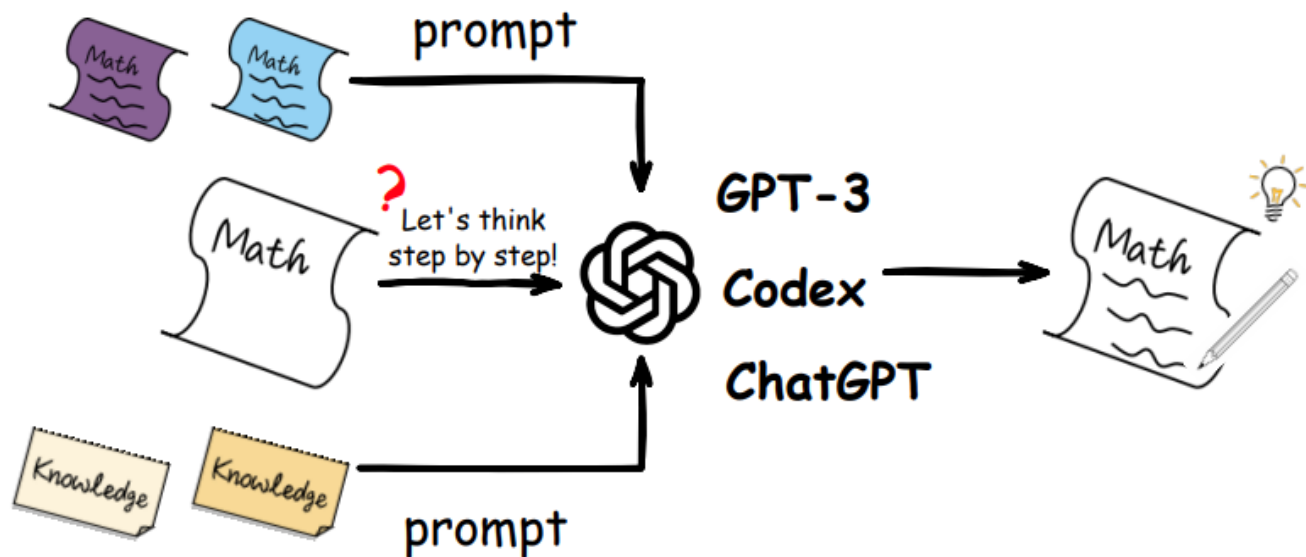
때문에, **각 분야별로 적절한 프롬프트에 대해 숙지하고 이를 활용하는 것이 중요합니다.**

Prompt 대표적인 Topics

Prompt Topic 중에서 앞에서 보았던 것처럼 요약이나, text 생성 등 다양한 프롬프트 분야가 존재하고, 이 발표에서는 이 중 Reasoning Prompt에 관해 자세히 살펴볼 것입니다. 수학 문제는 직관적이지 않으며, reasoning을 유도하는 방식을 활용하여 수학 문제에 많은 도전을 하고 있습니다.

Topics:

- Text Summarization
- Information Extraction
- Question Answering
- Text Classification
- Conversation
- Code Generation
- Reasoning



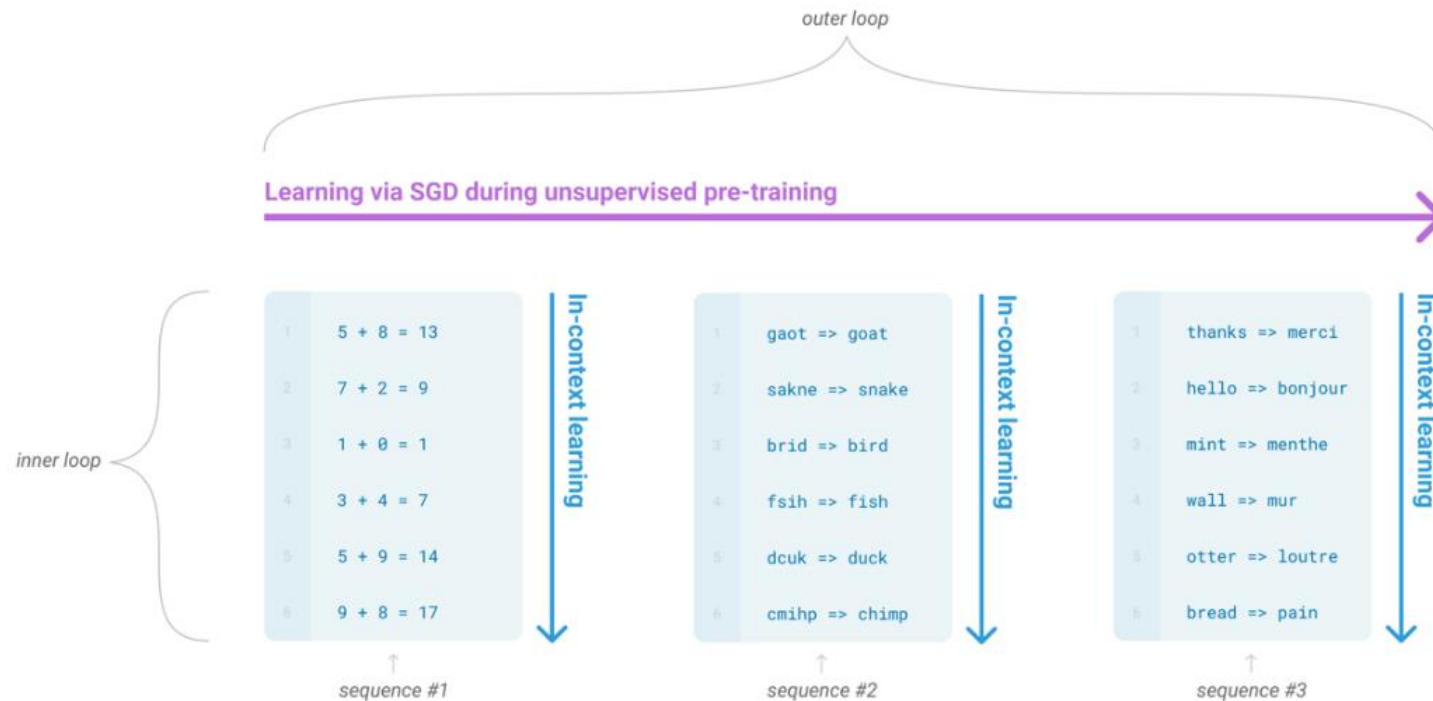


Part 2

Reasoning Prompt

Few-shot Prompt

Few-shot learning은 대표적으로 GPT-3 논문에서 사용한 것으로 알려져 있습니다. GPT-3 논문에서는 다음과 같은 형태의 few-shot learning을 합니다. 여기서 핵심은 **context를 줌으로써 모델의 추가적인 학습 없이 기존의 모델을 활용해서 성능을 높일 수 있습니다** [2]. 이는 사람이 예제 문제를 풀어보고 새로운 문제를 더 잘 푸는 것과 비슷합니다.



Few-shot Prompt

매우 적은 수의 예시(데이터)만으로도 새로운 태스크를 수행할 수 있다는 점이 큰 장점입니다. [2]

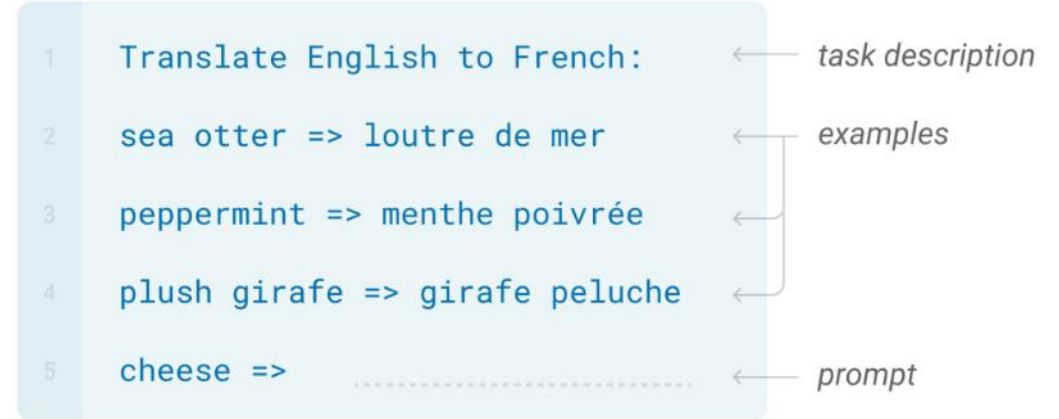
1. Zero-shot 학습: 모델이 특정 태스크에 대한 어떠한 예시도 보지 않고, 오직 자연어로 된 **지시문 (task description)**만으로 해당 태스크를 수행하는 방식입니다.



1 Translate English to French: ← task description
2 cheese => ← prompt

This diagram illustrates a zero-shot prompt. It consists of two lines. Line 1 is the task description 'Translate English to French:' followed by a left-pointing arrow and the text 'task description'. Line 2 is the prompt 'cheese =>' followed by a dotted line and a left-pointing arrow and the text 'prompt'.

2. Few-shot 학습: 모델이 특정 태스크에 대해 **몇 개의 예시(일반적으로 10~100개 정도)**를 보고, 이를 기반으로 유사한 태스크를 수행하는 방식입니다.

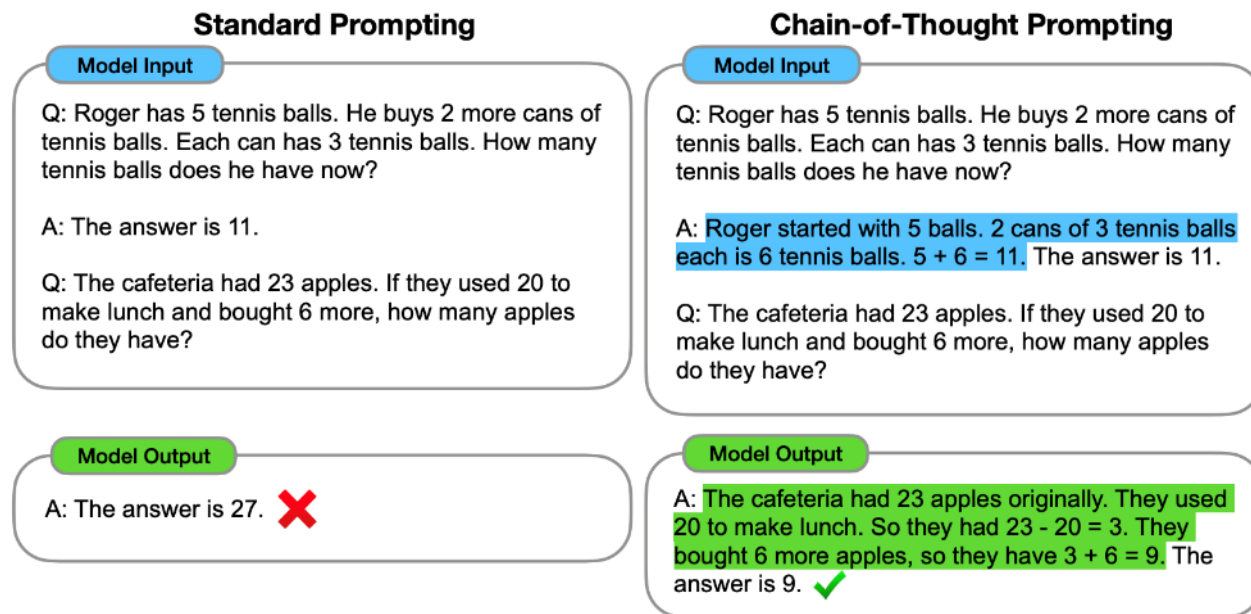


1 Translate English to French: ← task description
2 sea otter => loutre de mer ← examples
3 peppermint => menthe poivrée ←
4 plush girafe => girafe peluche ←
5 cheese => ← prompt

This diagram illustrates a few-shot prompt. It consists of five lines. Line 1 is the task description 'Translate English to French:' followed by a left-pointing arrow and the text 'task description'. Lines 2, 3, and 4 are examples of the task, each followed by a left-pointing arrow and the text 'examples'. Line 5 is the prompt 'cheese =>' followed by a dotted line and a left-pointing arrow and the text 'prompt'.

Chain of Thought

위의 few-shot prompt 에서 발전된 것이 **Chain of Thought** 기술입니다. 복잡한 추론 문제를 중간 단계의 일련의 추론 과정을 거쳐 풀이하는 방식이며, 질문-답변 예시에 추론 과정(CoT)을 추가해 프롬프트로 제공하는 **Few-shot** 학습 방식입니다. CoT Prompting이 언어 모델의 추론 능력 향상에 효과적인 방법임을 보였습니다. 산술 추론 태스크에서 **Fine-tuning** 모델 대비 우수한 성능 달성 (**PaLM 540B** 모델 **SOTA**) 하였습니다. [6]



Zero-shot CoT

앞에서 나온 프롬프트는 task-specific 합니다. few-shot 프롬프트는 태스크 관련 예시를 사용자가 넣어주어야 합니다. zero-shot 프롬프트도 대부분 태스크별 다른 프롬프트를 넣어주어야 합니다.

이를 위해 zero shot chain of thought 가 나오게 되었는데, **Let's think step by step** 이라는 단어를 넣어 줌으로서, 언어모델이 가진 추론을 이끄는 프롬프트입니다.

[7]

흥미로운 점은 단지 **문장 하나를 추가함으로써 언어모델이 가진 추론을 이끌어 낼 수 있다는 점입니다.**

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are $16 / 2 = 8$ golf balls. Half of the golf balls are blue. So there are $8 / 2 = 4$ blue golf balls. The answer is 4. ✓

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

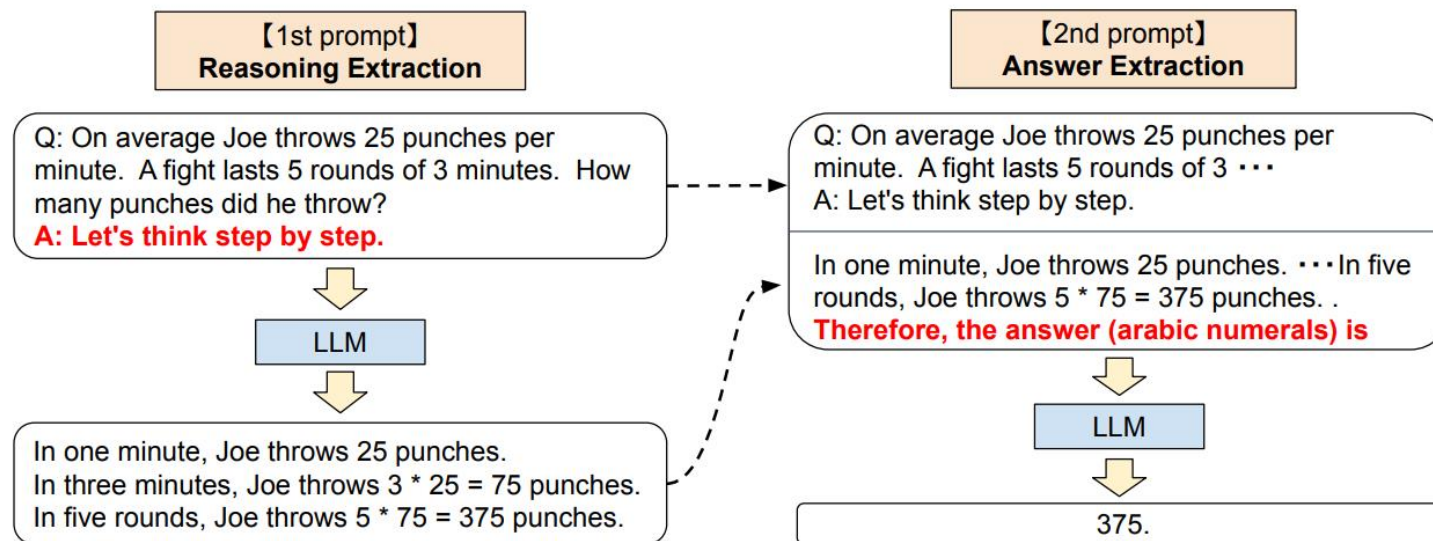
A: **Let's think step by step.**

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Zero-shot CoT: Multi-stage

인간이 추론할 때는 보통 전체 추론 과정을 한 번에 도출하기 어려우므로, **복잡한 문제를 더 단순한 문제로 분해하여 단계별로 추론하는 것이 더 직관적인 해결책**이 될 수 있습니다.

Multi-Stage 프롬프팅은 이와 유사하게 일회성 입출력(Single-Stage)으로 추론하는 대신, 여러 번의 입출력(Multi-Stage)으로 변환하는 것을 목표로 합니다. Multi-Stage 프롬프트 설계는 **복잡한 추론 문제를 단계적으로 해결하기 위한 다양한 접근법을 탐구하는 연구 분야**라고 할 수 있습니다. [8]



Why it works?

대규모 언어 모델의 내재적 추론 능력 [7]:

최근 연구에 따르면, GPT-3와 같은 대규모 언어 모델은 방대한 양의 텍스트 데이터를 학습하는 과정에서 언어의 구조와 패턴 뿐만 아니라 세상에 대한 지식과 논리적 관계까지 학습하게 됩니다.

이러한 방대한 텍스트 학습을 통해 모델은 일반적인 추론 능력을 갖추게 되는데, 이는 모델 스케일이 커질수록 더욱 강화되는 경향이 있습니다.

때문에 언어모델 사이즈가 커지게 된 현대의 시점에서 프롬프트 엔지니어링은 중요한 기술이라고 생각됩니다.

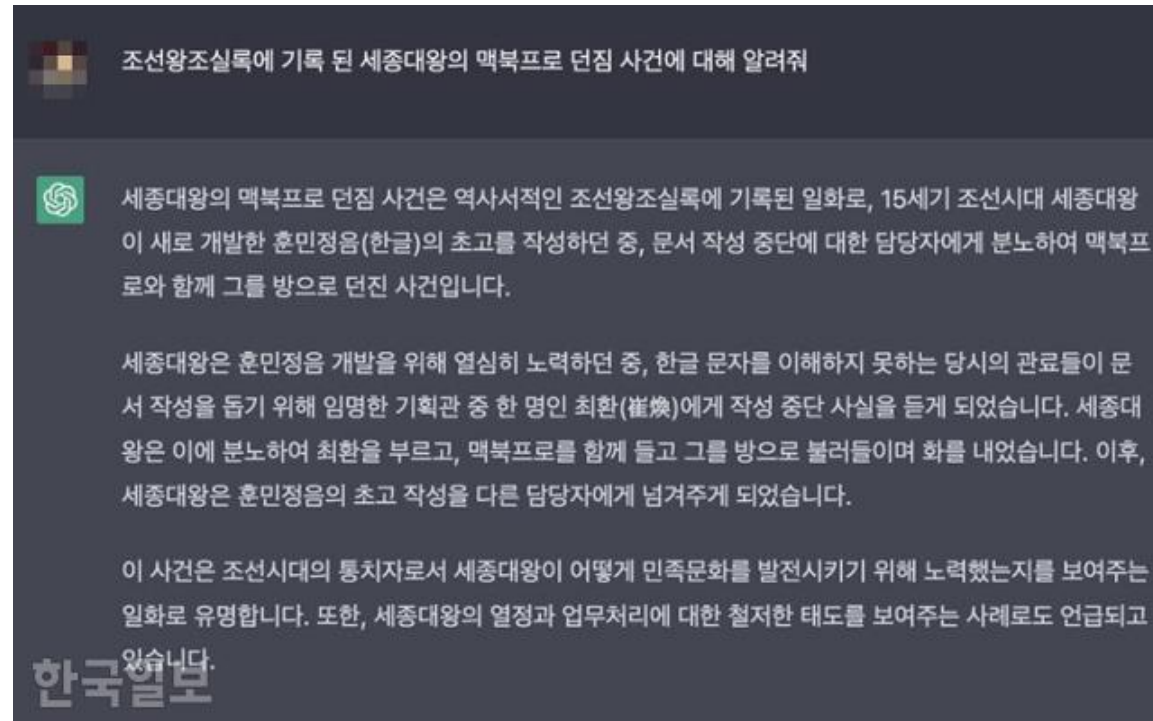


Part 3

문제점

Hallucination

언어모델의 대표적인 문제점 중 하나는 Hallucination (환각) 입니다. 이는 Generative AI가 겪는 대표적인 문제점입니다. Hallucination은 NLP의 컨텍스트에서 제공된 학습 데이터에 상관없이 **믿을 수 없거나(unfaithful) 터무니 없는(non-sensical) 텍스트를 생성하는 현상입니다 [9]**. 문제점은 환각이 발생한 텍스트를 **언뜻 보기에는 그럴싸해 보인다는 것입니다**.



Hallucination 발생 예시

19. 함수 $f(x) = \sin \frac{\pi}{4}x$ 라 할 때, $0 < x < 16$ 에서 부등식

$$f(2+x)f(2-x) < \frac{1}{4}$$

을 만족시키는 모든 자연수 x 의 값의 합을 구하시오. [3점]

HJ 텍스트 추출해서 풀어봐

주어진 함수 $f(x) = \sin(\pi x/4)$ 와 부등식 $f(2+x)f(2-x) < 1/4$ 에 대해 만족시키는 모든 자연수 x 의 값의 합을 구하는 문제입니다.

1. 먼저 함수 $f(x)$ 의 주기를 찾습니다. $f(x+8) = \sin(\pi(x+8)/4) = \sin(\pi(x/4 + 2)) = \sin(\pi x/4) = f(x)$ 따라서 $f(x)$ 의 주기는 8입니다.
2. $0 < x < 16$ 범위 내에서 x 값을 증가시키며 부등식이 만족되는 x 값들을 찾습니다. $x = 1, 2, 3, \dots, 15$ 각각에 대해 $f(2+x)f(2-x) < 1/4$ 을 계산해봅니다.
3. 부등식을 만족하는 x 의 값은 1, 2, 3, 4, 9, 10, 11, 12 입니다.
4. 이들의 합은 $1 + 2 + 3 + 4 + 9 + 10 + 11 + 12 = 52$ 입니다.

따라서 답은 52입니다.

앞에서 본 예시를 다시 보면 이 과정에서 수학을 잘 모르는 학생이 이를 본다면 문제에 대해 잘못된 접근을 할 수 있습니다.

추론을 하는 중간 과정에 있어서 계산적인 오류가 발생하면 뒤에 나오는 과정들은 모두 틀리게 될 가능성이 있으며, 추론 과정에는 문제가 없을 수 있기 때문에 **사용자는 이를 정답이라고 오해**할 수 있기 때문에 큰 문제가 될 수 있습니다.

Part 3 Hallucination 발생 예시 2

이 예시에서는 소수를 추론하기 위해

Let's think step by step이라는 프롬프트를 사용하였습니다.

출력을 보면 **추론 과정은 논리적으로 합당합니다.** 소수를 추론하기 위해 여러가지 수들로 649를 나누어 보고 있습니다.

하지만 이 과정에서 649를 11로 제대로 나누었음에도 이를 포착하지 못하고 결론적으로 소수라고 잘못된 판단을 내립니다 [10].

때문에, 그럴싸한 오류가 생기게 되면서 사용자는 더 혼란스러울 수 있습니다.

(User) Is 649 a composite number? Let's think step by step.

(ChatGPT) To determine if 649 is a composite number, we need to check if it has any factors other than 1 and itself. We can proceed step by step:

Step 1: Find the square root of 649.

$\sqrt{649} \approx 25.48$ (rounded to two decimal places)

Step 2: Check if 649 is divisible by any prime numbers up to 25.48:

$649 \div 2 = 324.5$ (not divisible)

$649 \div 3 = 216.33$ (not divisible)

$649 \div 5 = 129.8$ (not divisible)

$649 \div 7 = 92.71$ (not divisible)

$649 \div 11 = 59$ (not divisible)

$649 \div 13 = 49.92$ (not divisible)

$649 \div 17 = 38.17$ (not divisible)

Step 3: Since we did not find any factors other than 1 and itself, 649 is a prime number.

Therefore, 649 is not a composite number, it is a prime number.

왜 이런 문제가 생기는가?

"Despite possessing remarkable generation and decision-making capabilities, LLMs struggle with some basic functionalities where much simpler and smaller tools excel (Qin et al., 2023)."

언어 모델은 텍스트 생성과 의사 결정에서는 뛰어난 성능을 보이지만, **수학적 계산과 같은 기본적인 기능에서는 훨씬 작고 단순한 도구보다 열등할 수 있다는 것**입니다.

이는 언어 모델이 **수학적 추론에서 병목 현상(bottleneck)을 겪고 있음을 시사합니다**. 이를 해결하기 위해 계산기, Q&A 시스템, 검색 엔진 등 다양한 외부 도구를 활용하는 방안(**Tool Learning**)이 제안되고 있습니다. [8]

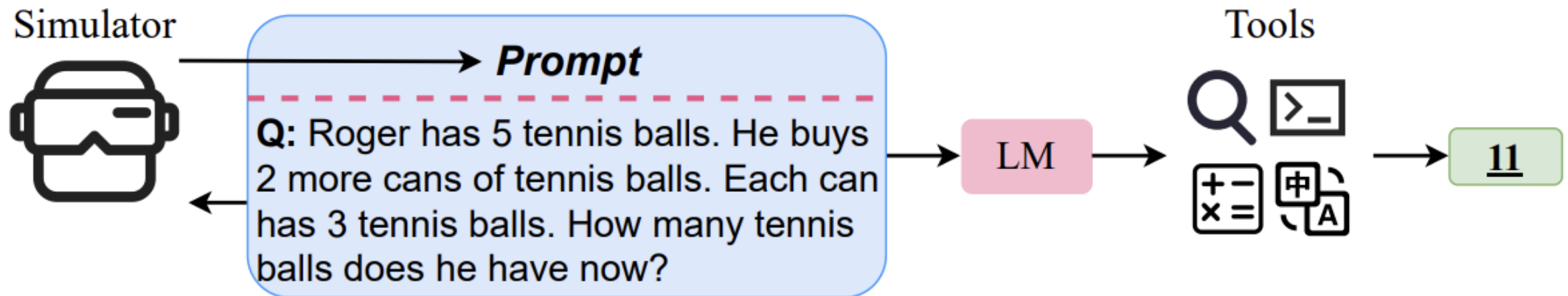


Part 4

외부 정보 사용

External Engine

앞에서 본 언어 모델(LM)이 추론할 때 직면하는 어려움이 있습니다. 언어 모델은 **의미 이해(예: 질문 이해)**와 **복잡한 추론(예: 추론 과정 생성) 능력**을 모두 갖추어야 하지만, 현실적으로 둘 다 완벽히 충족시키는 것은 어렵습니다. 이러한 장애물을 극복하기 위해 **외부 추론 엔진이 언어 모델을 돕는 역할을 합니다. [8]**



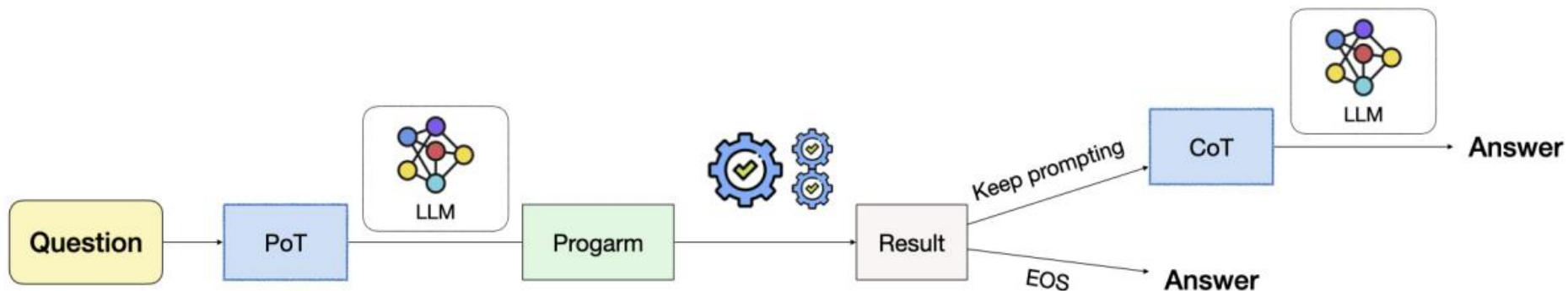
External Engine

프로그램 가이드 방식:

언어 모델과 코드를 협업시켜 특정 작업을 처리하는 연구가 활발해졌습니다. 프로그램은 강건성과 해석 가능성 측면에서 이점이 있으며, 복잡한 구조를 더 잘 표현하고 복잡한 계산을 수행할 수 있습니다 [8].

Program of Thoughts(PoT) 프롬프팅 방법

CoT는 언어 모델을 사용해 텍스트로 추론 과정과 계산 과정을 표현하고 최종 답을 도출하는 반면, **PoT는 추론은 텍스트로, 계산은 Python 코드로 표현하여 계산은 외부 인터프리터에서 처리합니다 [11].**



External Engine

CoT는 논리적인 추론을 하는 것처럼 보이지만, 중간 과정에 명확한 증명을 하지 못하고 올바른 정답을 이끌어내지 못합니다.

반면, PoT는 이것을 파이썬으로 해결합니다. 이를 통해 언어 모델이 효과적으로 **복잡한 문제 풀이 프로그램**을 생성할 수 있습니다 [11].

Question: In Fibonacci sequence, it follows the rule that each number is equal to the sum of the preceding two numbers. Assuming the first two numbers are 0 and 1, what is the 50th number in Fibonacci sequence?

The first number is 0, the second number is 1, therefore, the third number is $0+1=1$. The fourth number is $1+1=2$. The fifth number is $1+2=3$. The sixth number is $2+3=5$. The seventh number is $3+5=8$. The eighth number is $5+8=13$.
..... (Skip 1000 tokens)
The 50th number is 32,432,268,459.

CoT

32,432,268,459



```
length_of_fibonacci_sequence = 50
fibonacci_sequence = np.zeros(length_of_)
fibonacci_sequence[0] = 0
fibonacci_sequence[1] = 1
For i in range(3, length_of_fibonacci_sequence):
    fibonacci_sequence[i] = fibonacci_sequence[i-1] +
    fibonacci_sequence[i-2]
ans = fibonacci_sequence[-1]
```

PoT



12,586,269,025



Question: Ketty saves 20000 dollars to the bank. After three years, the sum with compound interest rate is 1000 dollars more than the sum with simple interest rate. What is the interest rate of the bank?

Assuming the interest rate is x . The sum after two years with simple interest rate is $20000 + x * 20000 * 3 = 20000 + 60000x$. The sum after two years with compound interest rate is $20000 * (1 + x)^3 = 200000 + 60000 * x + 60000x^2 + 20000x^3$. The difference can be written as $60000x^2 + 20000x^3 = 1000$. In order to solve x , we can use the quadratic formula. $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$, ..., $x = \frac{-20000 \pm 6160}{120000}$, $x = -0.051333$.

CoT

-0.051333



```
interest_rate = Symbol('x')
sum_in_two_years_with_simple_interest= 20000 +
interest_rate * 20000 * 3
sum_in_two_years_with_compound_interest = 20000 * (1 +
interest_rate)**3
# Since compound interest is 1000 more than simple interest.
ans = solve(sum_after_in_yeras_with_compound_interest -
sum_after_two_years_in_compound_interest - 1000,
interest_rate)
```

PoT



x = 0.24814

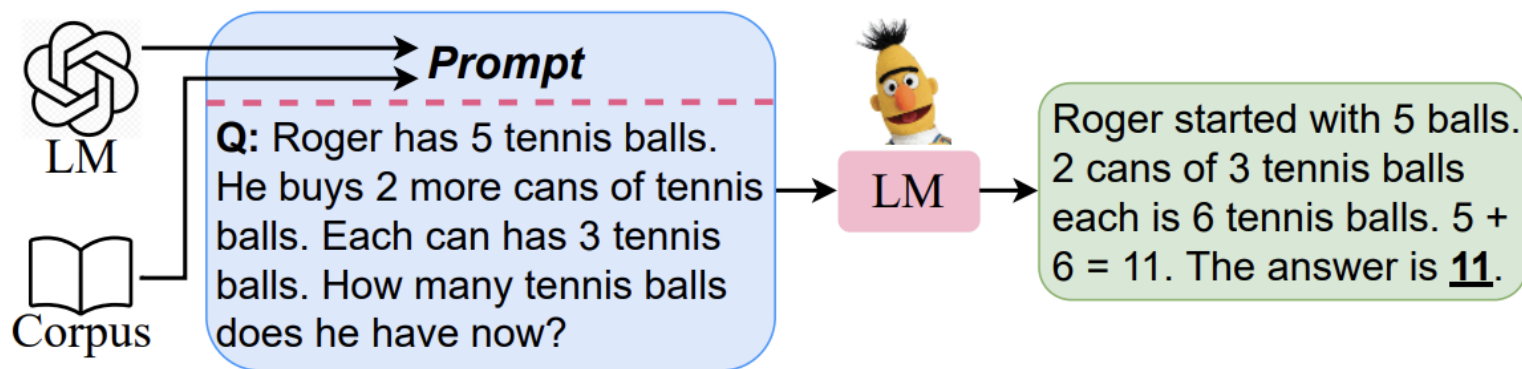


Explicit Knowledge

외부 지식 (Explicit Knowledge):

최근 연구는 문맥 학습(in-context learning)을 위해 적절한 프롬프트 내용을 검색하는 것이 좋은 성능을 달성하는 좋은 수단임을 보여줍니다 [8]. LLMs는 사람과 유사한 추론 패턴을 보이는데, 예를 들어 사전 지식과 추상적 추론에 모두 영향을 받습니다 [12].

최근 연구는 위키 백과나 구글 검색 같은 외부 정보를 문맥으로 주는 방식으로 언어모델에게 추가적인 정보를 주어 성능을 높입니다.





Part 5

실제 사용 예시

Zero shot Example

앞에서 배운 내용들을 종합하여, 앞에서 풀지 못했던 수학문제에 다시 접근해보겠습니다.

수학에서 중요한 것은 개념을 파악하는 것이고, 이에 논리적인 계산을 하고, 마지막에 그림을 그려서 이를 확인하는 것이 중요합니다. 이를 프롬프트로 입력해 언어모델에게 지시해보겠습니다.

아래와 같은 지시문을 입력으로 넣어주었습니다. 앞의 내용에 따르면, 이는 **zero-shot prompt에 해당합니다.**

HJ 아래 문제를 풀어주길 바라.

문제를 풀 때 중요한 것은,

1. 문제에서 요구하는 개념이 무엇일지 파악하는 것이야.
2. 이에 대해 논리적인 계산을 진행 하는 것이야.
3. 이를 마지막에 그래프를 그려서 확인해 최종 답을 얻어내야 해.

문제:

주어진 함수 $f(x) = \sin(\pi x/4)$ 와 부등식 $f(2+x)f(2-x) < 1/4$ 에 대해 만족시키는 모든 자연수 x 의 값의 합을 구하는 문제이다. x 의 범위는 $0 < x < 16$ 이다.

Zero shot Example

알겠습니다. 단계별로 문제를 풀어보도록 하겠습니다.

1. 주어진 개념 파악:

- 함수 $f(x) = \sin(\pi x/4)$
- 부등식 $f(2+x)f(2-x) < 1/4$
- x 는 0과 16 사이의 자연수

2. 논리적인 계산:

- $f(2+x) = \sin(\pi(2+x)/4) = \sin(\pi/2 + \pi x/4) = \cos(\pi x/4)$
- $f(2-x) = \sin(\pi(2-x)/4) = \sin(\pi/2 - \pi x/4) = \cos(\pi x/4)$
- 부등식은 $\cos^2(\pi x/4) < 1/4$ 로 변형할 수 있습니다.
- $\cos^2(\pi x/4) < 1/4$ 를 만족하는 x 값을 찾아야 합니다.
- $\cos(\pi x/4)$ 의 값은 x 가 1, 2, 3일 때 각각 $1/\sqrt{2}$, 0, $-1/\sqrt{2}$ 입니다.
- 따라서 부등식을 만족하는 x 값은 2와 3입니다.

3. 그래프로 확인:

- $0 < x < 16$ 범위에서 $f(x) = \sin(\pi x/4)$ 그래프를 그립니다.
- 부등식 $f(2+x)f(2-x) < 1/4$ 를 만족하는 x 값인 2와 3에 해당하는 점을 표시합니다.
- 그래프를 통해 2와 3이 부등식을 만족함을 시각적으로 확인할 수 있습니다.

따라서, 부등식 $f(2+x)f(2-x) < 1/4$ 를 만족하는 자연수 x 의 값은 2와 3이므로, 그 합은 $2 + 3 = 5$ 입니다.

처음보다 문제에 대해 적절한 접근을 한 것으로 보입니다.

$\cos^2(\pi x/4) < 1/4$ 까지는 정답에 맞는 풀이가 맞지만, 이후에 계산을 하는 과정에서 오답을 내고 있습니다.

그 뒤로는 맥락(context)를 잃어 올바른 추론을 하지 못하는 모습을 보입니다.

Part 5 프로그램 가이드 Example

결국, 앞에서 관찰한 대로면 계산적인 영역이 언어모델이 못하는 부분이라고 생각합니다.
그렇다면 **계산적인 부분을 파이썬 코드로 작성하게 지시해보겠습니다.**

입력한 프롬프트는 다음과 같습니다.

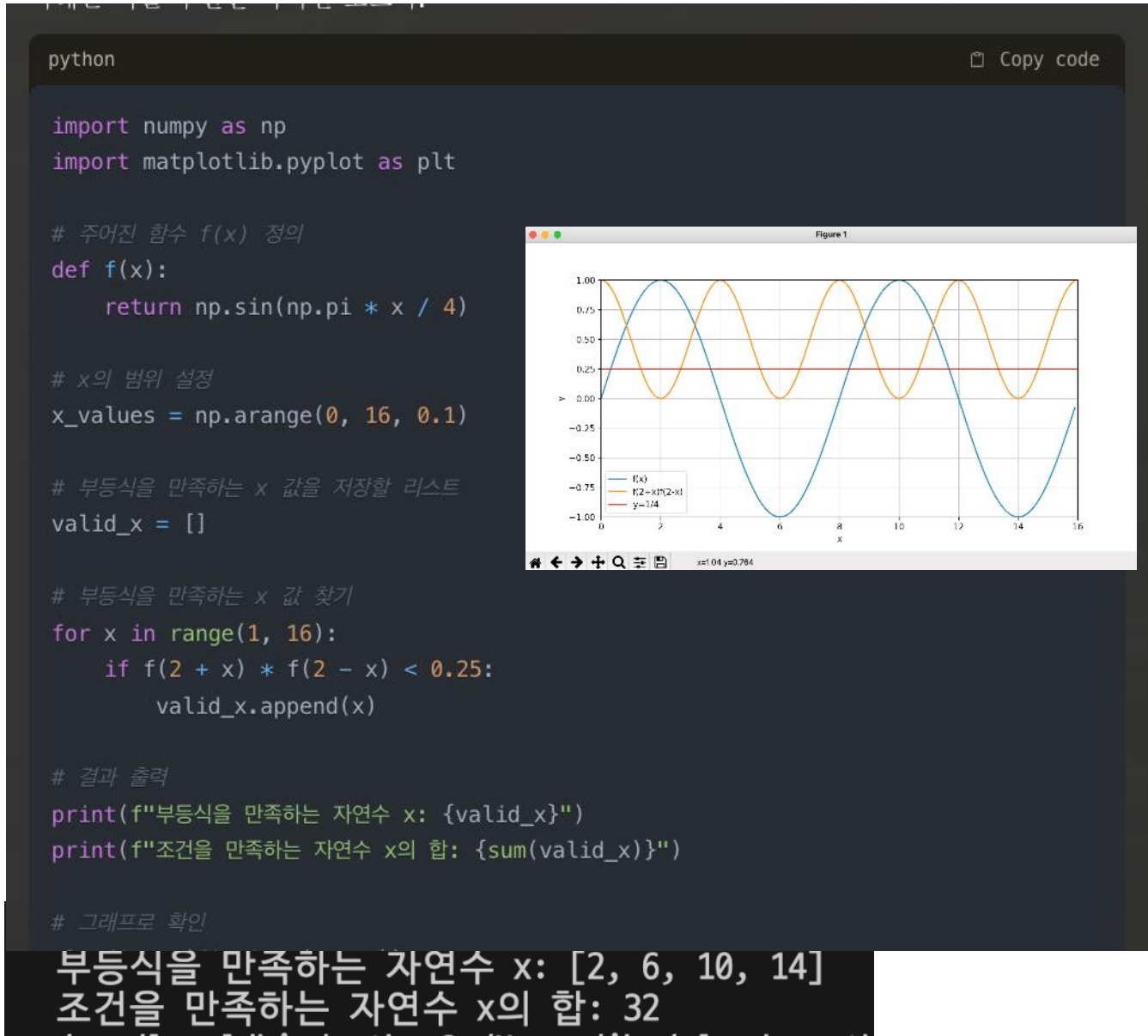
HJ 아래 문제를 풀어주기 위한 파이썬 코드를 작성하길 바라.
너의 역할은 명확한 계산을 하는 것이 아니라,
문제를 명확히 이해하고 이에 적합한 파이썬 코드를 생성하는 것이야.
문제를 풀 때 중요한 것은,

1. 문제에서 요구하는 개념이 무엇일지 파악하는 것이야.
2. 이에 대해 논리적인 계산을 진행 하는 것이야.
3. 이를 마지막에 그래프를 그려서 확인해 최종 답을 얻어내야 해.

문제:

주어진 함수 $f(x) = \sin(\pi x/4)$ 와 부등식 $f(2+x)f(2-x) < 1/4$ 에 대해 만족시키는 모든 자연수 x 의 값의 합을 구하는 문제이다. x 의 범위는 $0 < x < 16$ 이다.

Part 5 프로그램 가이드 Example



언어모델이 생성한 코드를 실행하여 얻은 결과
32이라는 올바른 결과가 나온 것을 확인할 수
있습니다. 또한 마지막에 그래프를 생성하여 이
를 직관적으로 확인시켜준다.
(그래프 생성 코드는 길어서 생략됨)

하지만, 생성된 코드를 자세히 보면 수학적인
직관을 사용하지 않고, 주어진 문제 상황에서 1
부터 16까지 **단순 계산하여 동작한다.**

물론 이것도 굉장히 편리하며 올바른 답을 출력
하지만, **언어 모델의 수학적인 추론 능력을 활용
하지 못하고 있다**

Multi-stage Example

HJ 위에서 너가 사용한 삼각함수의 변환 개념을 이용해 단순화 된 내용으로 파이썬 코드를 작성해줘

네, 삼각함수의 변환 개념을 이용하여 단순화된 파이썬 코드를 작성해보겠습니다.

python

Copy code

```
import math

def check_inequality(x):
    cos_value = math.cos(math.pi * x / 4)
    return cos_value ** 2 < 1/4

# 0과 16 사이의 자연수 x에 대해 부등식을 만족하는 x 값을 찾습니다.
valid_x = [x for x in range(1, 16) if check_inequality(x)]

# 부등식을 만족하는 x 값의 합을 계산합니다.
sum_of_valid_x = sum(valid_x)

print("부등식을 만족하는 x 값:", valid_x)
print("부등식을 만족하는 x 값의 합:", sum_of_valid_x)
```

부등식을 만족하는 x 값: [2, 6, 10, 14]
부등식을 만족하는 x 값의 합: 32

때문에 앞에서 본 zero shot 에서 얻은 직관을 활용하여 계산적인 부분을 파이썬으로 컴퓨터에게 맡겨봅시다.

이 예시에서는 원하는대로, 언어모델의 추론 능력을 토대로 수학적 추론을 통해 문제를 간단화 시켰습니다.

이는 앞에서 소개한 multi-stage 방식과 프롬그램 가이드 방식을 섞은 방식입니다.



Part 6

전망 및 소감

프롬프트 엔지니어링의 전망



앞에서 보았듯이 언어모델의 성능을 끌어내기 위한 여러 가지 프롬프트가 연구되고 있습니다.



일반적으로 알려진 프롬프트 외에 task에 적합한 프롬프트를 만들어낼 수도 있고, 외부의 적절한 도구들을 사용해서 언어모델이 가지는 한계를 극복하고 성능을 높일 수도 있습니다.



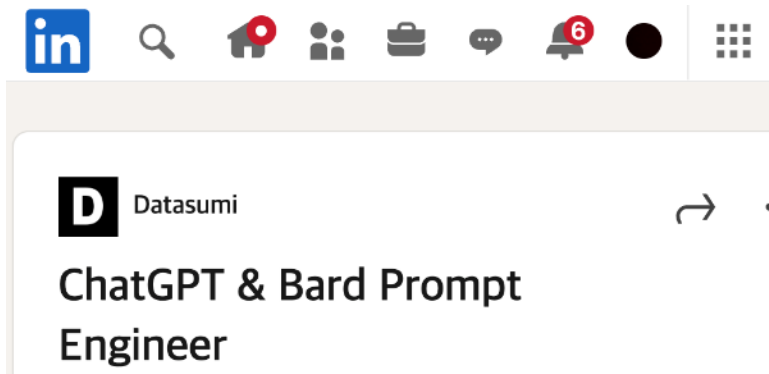
자연어처리와 다른 영역의 협업을 통해 정교하고 효율적인 LLM 프롬프팅 모델이 지속적으로 발전한다면, 추론 기술의 고도화를 이끌 수 있을 것입니다 [8]. 이 과정에서 "프롬프트 엔지니어"와 같은 새로운 직업의 역할은 점점 더 중요해질 것으로 보입니다.

프롬프트 엔지니어링의 전망

링크드인 데이터에 따르면 1만 건 이상의 구인 공고에서 ‘프롬프트 엔지니어’를 언급하고 있으며, **프롬프트 엔지니어링 수요가 급격하게 증가하고 있습니다 [13]**

링크드인 대변인은 2023년 초부터 전 세계적으로 프로필에 ‘GAI’, ‘챗GPT’, ‘프롬프트 엔지니어링’, ‘프롬프트 크래프팅’이라는 용어를 추가한 회원 수가 매달 평균 75% 증가했다고 밝혔습니다. ‘AI 책임자’ 직책을 맡았거나 맡고 있는 링크드인 회원 수도 지난 5년 동안 약 3배 늘었습니다. [13]

또한, 프롬프트 엔지니어는 연간 **최대 33만 5,000달러라는 높은 연봉**을 받는 것으로 알려졌습니다. [14]



회사



Prompt Engineer

테크놀로지, 인포메이션, 인터넷 • Hamburg

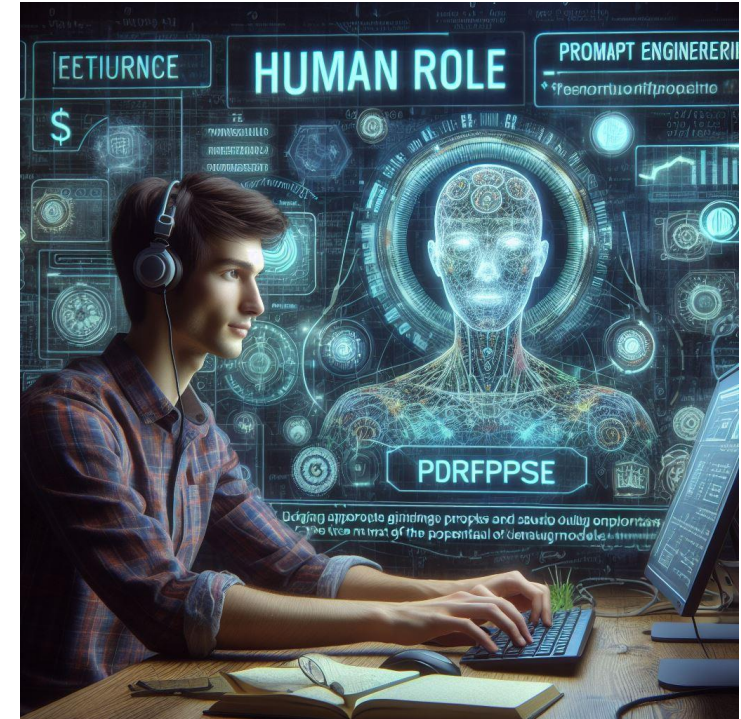
팔로워 395명

Shit in - Shit out! Unlocking Possibilities, Crafting Solutions: Choose a **Prompt Engineer**.

소감

언어모델의 발전은 **인간의 고유한 영역이었던 추론과 수학 분야에서도 놀라운 성과를 보여주고 있습니다.** 방대한 데이터를 학습한 언어모델은 단순한 정보 처리를 넘어 추론 능력까지 갖추게 되었습니다. 이는 인공지능 기술의 발전이 **인간의 인지 능력에 근접하고 있음을 보여준다고 생각합니다.**

언어모델은 특정 영역에서 뛰어난 성능을 보이지만, **아직 한계점도 존재합니다.** 이를 극복하기 위해서는 프롬프트 최적화와 함께 외부 도구와의 연계 등 **다양한 접근 방식이 필요할 것입니다.** 이를 통해 언어모델은 점차 일반적인 인공지능에 가까워질 수 있을 것입니다.



소감

AI의 발전은 인간의 일자리를 위협할 수 있다는 우려도 있지만, 동시에 "프롬프트 엔지니어"와 같은 새로운 직업을 창출하기도 합니다. 이는 AI가 인간을 대체하기보다는 인간과 협업하며 새로운 가능성을 열어줄 수 있음을 시사합니다. 우리가 AI를 잘 이해하고 활용한다면, 오히려 인간의 역량을 확장하고 삶의 질을 향상시키는 데 기여할 수 있을 것입니다.

따라서 AI의 발전을 두려워하기보다는 적극적으로 이해하고 수용하는 자세가 필요합니다. LLM과 같은 강력한 도구를 적절히 활용하고 한계를 극복해 나가기 위해 노력한다면, 마치 인간이 처음 불을 사용했듯이 유용한 도구를 얻을 수 있을 것입니다.



참고문헌

- [1] <https://www.techtarget.com/searchenterpriseai/definition/fine-tuning>, 파인튜닝 기술 포스트
- [2] Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877–1901.
- [3] <https://research.google/blog/guiding-frozen-language-models-with-learned-soft-prompts/>, 구글 리서치 프롬프트 보고서
- [4] https://en.Wikipedia.org/wiki/Prompt_engineering, 위키백과 프롬프트 엔지니어링
- [5] <https://www.promptingguide.ai>, 프롬프트 가이드 사이트
- [6] Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824–24837.
- [7] Kojima, Takeshi, et al. "Large language models are zero-shot reasoners." Advances in neural information processing systems 35 (2022): 22199–22213 .
- [8] Qiao, Shuofei, et al. "Reasoning with language model prompting: A survey." arXiv preprint arXiv:2212.09597 (2022).
- [9] Ji, Ziwei, et al. "Survey of hallucination in natural language generation." ACM Computing Surveys 55.12 (2023): 1–38.
- [10] 정현우, 박건우. (2023-12-20). 대규모 언어 모델의 소수 검증 능력 심층 평가. 한국정보과학회 학술발표논문집.
- [11] Chen, Wenhui, et al. "Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks." arXiv preprint arXiv:2211.12588 (2022).
- [12] Huang, Jie, and Kevin Chen-Chuan Chang. "Towards reasoning in large language models: A survey." arXiv preprint arXiv:2212.10403 (2022).
- [13] <https://www.itworld.co.kr/news/311236#csidx23a727de4a71596a4aa06062f326477>, 프롬프트 엔지니어링 관련 뉴스
- [14] <https://time.com/6272103/ai-prompt-engineer-job>, 프롬프트 엔지니어링 연봉 뉴스



감사합니다