ISSN 2288-4866 (Print)
ISSN 2288-4882 (Online)
https://www.jiisonline.org
J Intell Inform Syst 2023 December: 29(4): 287~308 http://dx.doi.org/10.13088/jiis.2023.29.4.287

ChatGPT 및 거대언어모델의 추론 능력 향상을 위한 프롬프트 엔지니어링 방법론 및 연구 현황 분석*

박상언
경기대학교 산업경영정보공학과
(supark@kyonggi.ac.kr)

강주영
아주대학교 e비즈니스학부
(jykang@ajou.ac.kr)

ChatGPT는 2022년 11월에 서비스를 시작한 후 급격하게 사용자 수가 늘어나며 인공지능의 역사에서 큰 전환점을 가져올 정도로 사회 곳곳에 많은 영향을 미치고 있다. 특히 ChatGPT와 같은 거대언어모델의 추론 능력은 프롬프트 엔지니어링 기법을 통해 빠른 속도로 그 성능이 발전하고 있다. 인공지능을 워크플로우에 도입하려고 하는 기업이나 활용하려고 하는 개인에게 이와 같은 추론 능력은 중요한 요소로 고려될 수 있다. 본 논문에서는 거대언어모델에서 추론을 가능하게 한 문맥내 학습에 대한 이해를 시작으로 하여 프롬프트 엔지니어링의 개념과 추론 유형 및 벤치마크 데이터에 대해 설명하고, 이를 기반으로 하여 최근 거대언어모델의 추론 성능을 급격히 향상시킨 프롬프트 엔지니어링 기법들에 대해 조사하고 발전과정과 기법들 간의 연관성에 대해 상세히 알아보고자 한다.

주제어 : 거대언어모델, 프롬프트 엔지니어링, ChatGPT, 추론, GPT

논문접수일 : 2023년 12월 10일 논문수정일 : 2023년 12월 17일 게재확정일 : 2023년 12월 18일
원고유형 : Regular Track 교신저자 : 강주영

1. 서론

ChatGPT는 2022년 11월 30일에 OpenAI에 의해 서비스가 공개되었으며, 불과 5일 만에 사용자 수가 100만 명을 넘었다[1]. 또한 2023년 1월에 1억 명이 넘는 사용자 수를 기록하면서 전세계적인 혁신을 가져왔다. ChatGPT는 서비스를 기준으로 하면 대화형 인공지능 챗봇이라고 할 수 있는데, 기존의 챗봇이 매우 한정된 범위의 지식에서 부자연스럽고 제한적인 대화를 제공했던 것에 비해, 문서 요약, 코드 작성, 광범위한 지식에 대한 질문, 번역, 원하는 글이나 문서의 생성 등 자연어로 할 수 있는 거의 모든 작업을 지원한다는 점에서 큰 차이점이 있다.

이와 같은 다양한 응용 분야로 인해 많은 사람들이 자신에게 필요한 다양한 목적으로 ChatGPT를 사용하게 되었으며, 일상적인 필요 외에 논문 작성, 프로그래밍, 언어 번역 및 교정 그리고 영화 시나리오, 광고 대본, 제안서 등과 같은 업무에서도 활용하고 있는 것으로 조사되었다(김맹근, 2023).

그러나 이 논문에서는 여러 응용 분야 중 추론 분야에 한정하여 ChatGPT와 같은 거대언어모델의 능력을 개선하려는 연구의 발전과정을 조사하고자 한다. 거대언어모델의 다른 활용 영역은

* 이 논문은 2023년 대한민국 교육부와 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2021S1A3A2A02089039)

1) https://marketsplash.com/ko/top-chatgpt-statistics/

대부분 기존의 딥러닝 기반 언어모델에서 연구되었던 부분이기 때문에 최근 성능의 개선은 크게 두드러지지 않았다. 그러나 GPT-3 발표 이후 최근 2년 동안 거대언어모델의 추론 능력은 프롬프트 엔지니어링을 기반으로 급격히 발전했다고 할 수 있다. 무엇보다 추론은 다른 분야에 비해 인공지능의 본래 목표에 더 가까우며, 인간이 가진 사고 능력을 최대한 유사하게 구현하는 분야라고 할 수 있다. 특히 기업의 경우 ChatGPT를 워크플로우에 적용하려는 수요가 많은데, 이와 같이 업무에 인공지능적 요소를 반영하기 위해 ChatGPT를 도입하는 경우에는 다른 어떤 분야보다도 추론 능력이 더 중요한 요소가 될 것이다.

따라서 이 논문에서는 프롬프트 엔지니어링 기반의 추론 능력 향상 연구에 초점을 맞추고, 이전 연구와의 차별성과 발전과정 및 각 기법의 상세 내용과 예제를 정리하고자 한다. 이 논문을 통해 거대언어모델이 얼마나 인간과 유사한 추론을 할 수 있는가에 대한 현황과 향후 발전 가능성에 대해 가늠해볼 수 있기를 기대한다.

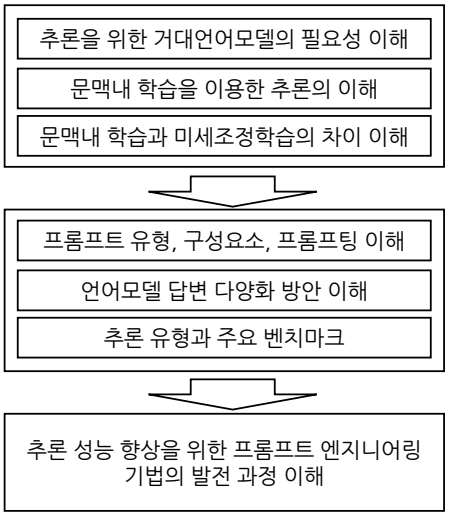
<표 1>은 이 논문에서 다루고자 하는 거대언어모델의 추론 능력 향상을 위한 프롬프트 엔지니어링 기법들의 개요를 보여준다. 표에서 보는 것과 같이 이 기법들은 처음에는 하나의 프롬프트를 어떻게 작성할 것인가에 대한 제안이었으나, 점차 복잡한 단계를 가지는 플랫폼의 형태로 발전하고 있다. 특히 최근의 연구들은 이전 연구들의 장점들을 결합하는 형태로 발전하고 있기 때문에 앞선 연구들에 대한 이해가 필수적이라고 할 수 있다.

본 연구에서는 최근 거대언어모델의 추론 분야에서 중요한 연구들을 중심으로 조사하고자 했으며, 이를 위해 구글 스칼라에서 인용수가 최소 40회 이상으로 높고 다른 중요 논문에서 언급된 논문들만 대상으로 했다.

〈표 1〉 프롬프트 엔지니어링 기법 개요

프롬프팅 기법	내용	문제점	자동 여부
퓨샷	다수 예제를 통해 간단한 추론 가능	복잡한 추론 불가	No
지식 생성	추론을 위해 필요한 지식을 생성하는 단계를 중간에 추가	복잡한 추론 불가	Yes
CoT	추론단계를 답변에 추가함으로써 비교적 복잡한 추론 가능	사람이 추론 단계를 추가해야 함	No
자기 일관성	CoT에서 다양한 답변을 생성하고 일관성이 높은 답변을 선택함으로써 CoT보다 높은 성능을 보임	사람이 추론 단계를 추가해야 함	No
제로샷 CoT	제로샷으로 추론과정 예제를 언어모델이 생성하게 함으로써 추론단계 추가가 불필요	CoT보다 낮은 성능	Yes
Auto-CoT	**퓨샷 + 제로샷 CoT** 문제집합을 군집화하고 주어진 문제에 대해 각 군집에서 하나씩의 문제를 선별해서 예제집합을 구성. 선별 과정에서 제로샷 CoT 적용	답변과 추론 과정이 잘못된 예제가 추가될 수 있음	Yes
액티브	**자기일관성 + 제로샷 CoT** 제로샷으로 추론과정 예제를 다양하게 생성하고, 답변의 불확실성이 높은 예제를 사람이 만든 예제로 교체. 답변 선택에도 자기 일관성을 적용	일부 문제에 대해 사람이 추론단계를 추가해야 함	Mix
ToT	**트리 형태의 단계적 자기 일관성 + 상태 평가를 통한 휴리스틱 탐색** 문제해결 과정을 여러 개의 사고로 분해하고 트리 형태로 사고를 생성하여 평가한 후 문제 해결까지의 다양한 경로를 탐색 가장 높은 수준의 추론 성능을 달성	문제 유형에 따라 사람이 적절한 프레임워크를 설계해야 함	No

<그림 1>은 본 논문의 구성을 보여준다. 본 논문은 독자로 하여금 추론 성능의 향상을 위해 발전해 온 프롬프트 엔지니어링 기법들에 대해 이해하고 이러한 기법들 간의 연관성을 파악하도록 하는 것에 목적이 있다. 이를 위해 먼저 2장에서는 추론에서 거대언어모델이 필요한 이유와 문맥내 학습을 통해 어떻게 추론이 이루어지는지 설명하고자 했다. 또한 프롬프트 엔지니어링과 기존 방식의 차이를 이해하기 위해 미세조정학습과 문맥내 학습과의 차이점에 대해 정리하였다.



〈그림 1〉 논문의 구성

3장에서는 프롬프트 엔지니어링의 기본 사항들에 대해 설명하고, 추론을 위해 언어모델의 답변을 보다 창의적으로 생성하고 다양화하기 위한 방안 그리고 추론 능력을 검증하기 위한 벤치마크에 대해 정리하였다.

4장은 본 논문의 핵심이라고 할 수 있으며, 앞서 설명한 바와 같이 추론을 위한 프롬프트 엔지니어링의 초기 단계부터 가장 최근의 연구까지 그 핵심 원리와 단계 및 예제를 설명하였다. 특히 4장에서는 각 기법들의 단계를 일관성 있고 공통적인 요소에 따라 설명하기 위해 본 논문의 관점으로 다시 도식화하였으며, 각 기법들 간의 연관성과 사용된 기술을 하나의 그림으로 파악할 수 있도록 했다. 또한 예제는 가급적 제시된 기법들 간의 직접적인 비교를 위해 일관성 있는 예제를 사용하고자 했으며, 2023년 11월을 기준으로 GPT-3.5 기반의 ChatGPT에서 실제 실행한 내용으로 작성하였다.

2. 거대언어모델(LLM)과 문맥내 학습

이 장에서는 거대언어모델의 추론에서 중요한 역할을 차지하고 있는 문맥내 학습의 개요 및 기존 방식과의 차이점에 대해 정리하였다.

2.1 추론을 위한 거대언어모델의 필요성

Brown et al. (2020)은 GPT-3를 발표하면서 미세조정학습이 없는 문맥내 학습(In-context Learning)인 퓨샷 러닝, 제로샷, 원샷 등의 개념을 소개했는데, 이것이 가능하기 위해서는 기존의 언어모델보다 충분히 큰 언어모델 즉 거대언어모델이 필요하다는 것을 보였다. 실제로 GPT-2는 15억 개 정도의 매개변수를 사용한 것에 비해 GPT-3는 1,750억 개 정도로 100배가 넘는 수의 매개변수를 사용했으며, 이를 계기로 이후의 언어모델들은 모두 대규모의 딥러닝 모형을 사용하게 된다.

거대언어모델을 사용해야 하는 첫째 이유는 성능 때문이다. Brown et al. (2020)은 모델의 크기가 커질수록 문맥내 학습의 정확도가 급격히 증가하는 것을 보였다. 13억 개의 매개변수를 사용한 모형에서는 퓨샷 프롬프팅의 정확도가 5%를 넘지 못한 반면, 130억 개의 매개변수를 사용한 모형

에서는 25% 정도의 정확도를 보이고, 1,750억 개를 사용한 모형에서는 65%를 넘는 정확도를 보이는 등, 언어모델의 크기는 모형의 성능에 매우 중요한 영향을 미쳤다.

거대언어모델을 사용하는 둘째 이유는 한 번에 입력할 수 있는 텍스트의 크기가 문맥내 학습에서 중요하기 때문이다. 하나의 예제만을 사용하는 원샷 프롬프팅과 다수의 예제를 사용하는 퓨샷 프롬프팅은, 정확도가 각각 45% 정도와 65% 정도로 20%p 가량의 차이가 났다. 즉 예제를 많이 사용함으로써 성능을 향상시킬 수 있다면 입력 텍스트의 크기가 더 큰 거대언어모델이 더 유리할 수 밖에 없다.

2.2 사전학습 언어모델과 미세조정학습

일반적으로 언어모델(Language Model)은 단어의 시퀀스에 대해 다음 단어의 확률을 예측하는 모델을 말하며, 이를 위해 대규모의 말뭉치를 이용해 학습된다. 이를 사전학습이라고 하는데, 사전학습은 라벨을 수동적으로 생성하는 수고가 없이 대용량의 문서에 대해 학습할 수 있다는 장점이 있는 반면, 문서 분류, 번역 등과 같은 목표 작업에 대해 직접적으로 학습하지 않았기 때문에 바로 적용할 수는 없다. 따라서 추가적인 학습이 필요하며 이를 미세조정학습(Fine Tuning)이라고 한다.

그러나, 사전학습과 미세조정학습이 분리된 형태의 언어모델 학습에는 중요한 단점이 존재한다. 사전학습에서 학습을 위해 많은 비용을 사용했음에도 불구하고 미세조정학습을 위해 추가적인 비용을 써야 한다. 사전학습에 비해 상대적으로 적은 데이터로도 학습할 수 있지만, 충분한 학습을 위해서는 여전히 많은 수의 데이터가 필요하고 무엇보다 지도학습이기 때문에 모든 데이터에 라벨이 있어야 한다는 제약이 있다(Brown et al., 2020). 반면, 인간은 언어와 관련한 특정 작업을 하기 위해 미세조정학습이 요구하는 만큼의 많은 데이터를 필요로 하지 않으며, 사람은 단 몇 개의 예제만으로도 새로운 작업에 대해 충분히 배울 수 있다. Brown et al. (2020)은 이에 대한 해답으로 문맥내 학습을 제안했다.

2.3 문맥내 학습과 프롬프트 엔지니어링

Brown et al. (2020)은 언어모델이 하나의 입력 시퀀스를 읽어 들이는 동안 그 안에 있는 서브작업(Subtask)들에 대해서도 학습이 일어나는 것을 발견하고, 이와 같이 입력 시퀀스의 내부 루프에서 발생하는 학습을 문맥내 학습이라고 정의했다. 문맥내 학습의 특징은 입력 시퀀스에서 반복되는 서브작업들에 대해 학습함으로써 유사한 서브작업을 수행할 수 있게 되는 것이다.

문맥내 학습은 사용되는 예제의 수에 따라 퓨샷 러닝, 원샷 러닝, 제로샷 러닝으로 구분되었다(Brown et al., 2020). 퓨샷 러닝은 질문과 대답의 쌍으로 이루어진 여러 개의 예제를 통해 패턴을 학습시키고 마지막에 질문을 하는 형태이며, 원샷 러닝은 한 개의 예제와 자연어로 기술된 작업 설명을 입력으로 사용한다. 제로샷 러닝은 원샷 러닝에서 예제를 빼고 작업에 대한 자연어 설명만을 입력으로 사용한다.

프롬프트는 이상과 같은 퓨샷 러닝, 원샷 러닝, 제로샷 러닝에서 답변을 이끌어 내기 위한 텍스트 입력 전체를 의미한다. 어떤 식으로 프롬프트를 작성하는가에 따라 답변의 정확도가 바뀔 수 있으며, 보다 정확한 답변을 얻기 위해 프롬프트를 설계하는 것을 프롬프트 엔지니어링이라고 한다(Reynolds & McDonell, 2021).

2.4 미세조정학습과 문맥내 학습의 차이

문맥내 학습의 장점은 첫째, 학습을 위해 많은 데이터셋을 필요로 하지 않는다는 점이다. 미세조정학습에 비해 훨씬 적은 데이터로도 좋은 성과를 보인다. 둘째, 기존에 학습되지 않은 새로운 데이터셋을 쉽게 활용할 수 있다. 미세조정학습은 새로운 데이터셋이 확보되면 다시 학습을 해야 한다. 그러나 문맥내 학습은 모형에 대한 재학습을 기다릴 필요가 없다. 셋째, 하나의 거대언어모델에 대해 작업에 맞는 프롬프트를 이용해 모든 작업을 수행할 수 있다.

이상과 같은 장점 외에 단점도 존재하는데, 우선 문맥내 학습은 가중치를 변경하지 않기 때문에 영구적으로 학습이 유지되지 않는다. 둘째, 문맥내 학습을 위해 필요한 예제들을 입력에 함께 포함시켜야 하기 때문에 주어진 입력의 크기를 많이 소모한다. 셋째, 미세조정학습에 비해 특정 자연어 처리 작업에서는 성능이 떨어질 수 있다(Brown et al., 2020). 비록 인간과 유사하게 몇 개의 예제만으로도 비교적 높은 성능을 낼 수 있다고 하더라도 수천에서 수만 개의 지도학습 데이터로 학습된 미세조정학습 만큼의 성능은 내기 어려운 분야가 있게 마련이다.

3. 추론을 위한 프롬프트 엔지니어링

이 장에서는 프롬프트 엔지니어링에 관련된 기본 사항들을 정리하였다. 프롬프트의 유형과 구성 요소를 비롯하여 추론을 위해 언어모델의 답변을 보다 창의적으로 생성하고 다양화하기 위한 방안 그리고 추론 능력을 검증하기 위한 벤치마크에 대해 정리하였다.

3.1 프롬프트의 유형과 구성 요소

Liu 등은 프롬프트를 빈칸 메우기 프롬프트(Cloze Prompt)와 접두 프롬프트(Prefix Prompt)의 두 유형으로 분류했다(Liu et al., 2023). 빈칸 메우기 프롬프트는 글자 그대로 프롬프트 중간에 빈칸을 넣고 언어모델이 빈칸을 채우도록 하는 프롬프트이다. 반면, 접두 프롬프트는 빈칸 없이 프롬프트를 서술하고 언어모델이 프롬프트에 이어서 답변하도록 하는 방식으로 빈칸이 가장 끝에 있는 프롬프트라고 할 수 있다. 프롬프트는 지시(Instruction), 질문(Question), 문맥(Context), 입력(Input), 예제(Examples), 답변 (Answer) 등과 같은 요소로 이루어진다. <표 2>는 이러한 요소들에 대한 간략한 설명과 예제를 보여준다.

〈표 2〉 프롬프트 구성요소

요소	설명	예제
지시	언어모델의 역할 혹은 해야 할 일을 지정	아래 문장을 읽고 질문에 답해줘
질문	응답을 요구하는 문장으로 단순한 프롬프트에서는 지시를 대신하여 사용될 수 있고, 답변과 쌍을 이루어 예제로도 사용될 수 있음	Q: 15, 32, 5, 13, 82, 7, 1에서 홀수만 더해서 짝수가 되는지 알려줘
문맥	언어모델을 이해 혹은 학습시키기 위해 사용되는 예제, 추가 설명 등의 모든 정보	Context: 수원 화성은 1794년 축성을 시작해 1796년에 완성했다.
입력	언어모델의 처리를 필요로 하는 텍스트. 요약할 문장, 번역할 문장, 질문, 수식 등으로, 지시, 문맥과 결합하여 원하는 작업을 수행할 수 있음.	Input: 정말 감동적인 영화였어
예제	언어모델이 입력에 대해 어떤 출력을 생성해야 하는지를 학습하기 위해 사용하는, 입력과 출력의 쌍으로 이루어진 텍스트	Input: 영화 캐스팅이 별로였던 것 같아 A: 부정
답변	예제에서 주어진 입력에 대한 처리결과를 나타내는 텍스트	A: 짝수가 되지 않습니다.

3.2 프롬프트 엔지니어링 연구의 시작

GPT-3가 발표되면서 보다 효과적인 답변 생성을 위해 프롬프트를 설계하는 기법에 대한 연구들이 시작되었다. Reynolds and McDonell (2021)은 프롬프트를 어떻게 쓰는가에 따라서 언어모델의 성능에 많은 차이가 있는 것을 발견하였다. 이를 기반으로 보다 효과적인 자연어 프롬프트를 생성할 수 있는 방안들을 제시했는데, 예를 들어 번역 등을 할 때 번역할 문장을 따옴표를 이용해 다른 문장과 명확하게 분리해주거나, 기표(Signifier) 즉 “translate”와 같이 해야 할 일을 명확하게 지칭하는 공인된 단어를 사용하도록 제안했다. 그 외에도 많은 설명 대신 비유를 이용해 간략한 프롬프트를 사용하는 것이 효과적일 수 있음을 제안했는데, 도덕적 문제에 대해 여러 조건을 나열하기보다 도덕적 상징을 갖고 있는 간디와 같은 인물을 역할로 줌으로써 설명을 간략화할 수 있다.

3.3 언어모델 답변의 다양화 방법

사람은 추론을 할 때 여러 가지 요소를 고려하여 다양한 대안을 만들고, 그러한 대안들을 검토해서 가장 좋은 대안을 선택한다. 따라서 언어모델을 이용해 사람과 유사한 방식으로 추론을 하기 위해서는 다양한 답변 특히 기존과는 다른 창의적인 답변을 생성할 필요가 있다. 언어모델에서는 이와 같은 답변의 다양성을 조절하기 위해 온도 샘플링(Temperature Sampling), Top-k 샘플링 그리고 뉴클리어스 샘플링(Nucleus Sampling) 기법을 사용한다(Fan et al., 2018; Ficler & Goldberg, 2017; Holtzman et al., 2019; Radford et al., 2019).

온도 샘플링은 다음 토큰에 대한 후보 집합의 토큰들에 대해 확률의 비율을 변경하는 스케일링 기법이라고 할 수 있다(Ficler & Goldberg, 2017). 온도는 일반적으로 0에서 1 사이의 값을 사용하는데, 온도가 낮을수록 후보 집합에 있는 토큰들 간의 확률 격차가 더 커진다. 즉 온도가 낮으면 높은 순위에 있는 토큰의 확률이 상대적으로 더 커지고 그로 인해 다음 토큰으로 선택될 가능성이 더 높아진다. 반대로 온도가 높으면 후보 집합에 있는 토큰들 간의 확률 격차가 줄어들고 이로 인해 원래 확률이 낮은 토큰이 선택될 가능성이 높아진다. 이것은 답변이 보다 창의적이 될 수 있다는 것을 의미한다. 온도를 높이고 답변 생성을 반복하면 다양한 답변을 볼 수 있는 반면, 온도를 0에 가깝게 낮추면 답변 생성을 반복해도 같은 답변만 나올 가능성이 높아진다.

Top-k 샘플링은 다음 토큰으로 선택될 후보 집합의 수를 확률이 높은 상위 k개로 제한한다(Fan et al., 2018; Radford et al., 2019). 만일 k를 1로 하면 확률이 가장 높은 한 토큰만 후보 집합에 포함되고 항상 동일한 토큰이 선택된다. 반면 k를 10으로 하면 확률이 높은 10개의 토큰이 후보 집합이 되고 따라서 답변이 다양해질 가능성이 높아진다. 단 Top-k 샘플링은 후보 집합에 있는 토큰 간의 단어 편차가 클 경우, 전혀 어울리지 않는 토큰이 선택될 가능성이 존재한다.

Top-p 샘플링이라고도 불리는 뉴클리어스 샘플링은 Top-k 샘플링의 단점을 보완하기 위해 제안되었으며, 후보 집합에 있는 토큰의 누적 확률을 p 이하로 제한한다(Holtzman et al., 2019). 예를 들어 p가 0.5이면 확률이 높은 상위 토큰부터 확률을 누적해서 0.5를 채운다. 만약 상위에 확률이 비교적 낮으면서도 비슷한 단어들이 많다면 이러한 단어들이 많이 포함되고, 상위에 확률이 매우 높은 적은 수의 단어들이 있다면 포함된 단어의 수는 줄어들게 된다. 뉴클리어스 샘플링은 토큰들이 어떤 확률분포를 보이더라도 비교적 안전한 샘플링을 할 수 있다는 장점이 있다.

3.4 추론의 유형과 주요 벤치마크

추론의 성능을 평가하기 위한 벤치마크 데이터셋의 유형은 산술 추론(Arithmetic Reasoning), 상식 추론(Commonsense Reasoning), 기호 추론(Symbolic Reasoning)의 세 개로 분류할 수 있다(Wei et al., 2022). 산술 추론은 자연어로 기술된 수학 문제를 읽고 이해한 후 정답을 생성하는 과정이다. 초기에는 주로 초등학교 수준의 수학 문제들로 벤치마크를 만들었으나, 점차 여러 단계를 거쳐 추론해야 하는 문제와 보다 복잡한 문제들이 추가되고 있다. 주요 벤치마크로는 CGSM8K(Grade School Math 8,000), SVAMP(Simple Variations on Arithmetic Math word Problems), AddSub MultiArith, SingleEq가 있다(Cobbe et al., 2021; Hosseini et al.. 2014; Koncel-Kedziorski et al., 2015; Patel et al., 2021; Roy & Roth, 2015)

상식 추론은 일반적인 상식을 기반으로 질문을 이해하고 대답하는 추론이다. 상식 추론의 수행을 위해서는 추가로 주어지는 정보 없이 자체적인 상식만으로 문제를 이해하고 문제를 해결하기 위한 지식들을 이용해 추론할 수 있어야 한다. 주요 벤치마크로는 CSQA(Commensense QA), StrategyQA, BIG-Bench가 있다(Geva et al., 2021; Srivastava et al., 2022; Talmor et al., 2018).

기호 추론 분야는 Wei et al. (2022)이 CoT(Chain of Thought) 프롬프팅 기법을 제안하면서 거대언어모델의 추론 분야로 함께 제시한 분야이다. 주요 벤치마크로는 마지막 문자 연결(Last Letter Concatenation), 코인 뒤집기(Coin Flip)가 있다.

다음 표는 각 추론 유형 별로 주요 벤치마크와 간단한 설명을 보여준다. 벤치마크는 본 논문에서 다루는 추론 관련 프롬프트 엔지니어링 논문들에서 공통적으로 사용된 벤치마크를 중심으로 정리했다.

〈표 3〉 프롬프팅 추론 성능 평가 벤치마크

분야	벤치마크	설명
산술 추론	GSM8K	언어모델의 산술 추론 능력 평가를 위한 8,500여 개의 초등학교 자연어 수학 문제로 구성
	SVAMP	초등학교 4학년 이하를 대상으로 한 자연어 수학 문제들을 언어모델에 맞게 여러 형태로 변형한 문제들로 구성
	AddSub	자연어 수학 문제에서 변수와 값을 식별하고 식을 생성해서 답을 구할 수 있는지를 평가
	MultiArith	여러 단계로 구성되는 자연어 수학 문제로 구성. 답안은 수식과 답으로 구성되며 수식을 표현 트리로 표현. 언어모델의 단계적 추론 성능을 평가
	SingleEq	변수 한 개로 구성된 방정식을 표현하는 자연어 수학 문제의 방정식 변환 성능을 평가
상식 추론	CSQA	일반적인 상식을 평가하는 12,247개의 객관식 문제로 구성. 언어모델 평가를 주목적으로 개발. 개발 당시 BERT는 56%, 사람은 89%의 정확도를 보임
	StrategyQA	질의에 대해 응답하기 위해 전략을 이용해 자연어 문제에 내재된 추론 단계를 추출할 수 있는 성능을 평가
	BIG-Bench	거대언어모델의 성능 평가를 목적으로 만들어진 다양한 벤치마크 데이터셋들의 집합으로, 구글이 개발 및 소유하고 있음[2]
기호 추론	Last Letter Concat.	두문자어(Acronym)가 단어의 첫 문자로 만든 말인 반면, 단어의 마지막 문자를 연결한 문자열을 반환
	Coin Flip	코인의 초기 상태를 서술하고, 사람의 코인의 뒤집기 여부를 반복한 후 코인의 상태를 추론

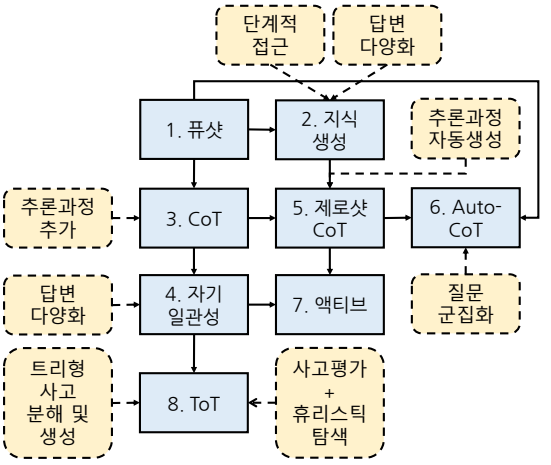
참고로 MAWPS는 산술 추론 평가에서 주로 사용되는 벤치마크인 AddSub, MultiArith, SingleEq

2) https://github.com/google/BIG-bench

를 포함하여 SingleOp, SimulEq-S, SimulEq-L로 로 구성된 온라인 벤치마크 데이터셋을 제공한다 (Koncel-Kedziorski et al., 2016).

4. 추론 성능 향상을 위한 프롬프트 엔지니어링 기법의 발전 과정

이 장에서는 거대언어모델의 추론 기능을 향상시키기 위해 개발된 프롬프트 엔지니어링 기법의 발전 과정과 각 기법의 실행과정 및 예제에 대해 상세히 알아보고자 한다.



〈그림 2〉 추론을 위한 프롬프팅 발전과정

<그림 2>는 이 장에서 설명하는 8개의 프롬프팅 기법들 간의 관계와 사용하는 기술을 요약하여 보여준다. 그림에서 번호가 있는 네모 박스는 이 장에서 설명하는, 추론을 위한 프롬프트 엔지니어링 기법을 나타낸다. 이 박스들을 연결하는 화살표는 각 기법이 어떤 기법을 계승 혹은 사용하고 있는지를 보여준다. 점선으로 된 박스는 기법이 사용하고 있는 도구 혹은 기술을 나타낸다. 점선 화살표는 기술과 그 기술을 사용하는 프롬프팅 기법을 연결한다.

예를 들어 그림에서 4번 자기일관성 프롬프팅 기법은 3번 CoT 기법을 기반으로 하여 답변 다양화 기술을 적용하여 새로운 기법을 제안했다. 또한 ToT(Tree of Thought) 프롬프팅 기법은 자기 일관성을 기반으로 하여 트리형 사고 분해 및 생성 기술과 사고평가, 휴리스틱 탐색 기술을 적용하여 새로운 기법을 제안하였다.

4.1 추론의 시작: 퓨샷 러닝

퓨샷 러닝이 여러 개의 예제를 통해 추론 과정에서 이루어지는 문맥내 학습이라면 퓨샷 프롬프팅은 문맥내 학습이 이루어지도록 퓨샷 프롬프트를 작성하는 작업이라고 할 수 있다. 퓨샷 프롬프팅의 목적은 거대언어모델이 여러 예제를 통해 패턴을 파악함으로써 프롬프트의 의도를 이해하고 올바른 답을 주도록 하는 것이다. 그러나 이 과정에서 거대언어모델이 반복되는 예제를 통해 질문의 의도를 이해하는 것 외에 추론 능력이 잠재되어 있음을 알게 되었다.

<표 4>는 퓨샷 프롬프트 만으로 연산의 기능을 추론하는 예제이다. 세 개의 인수를 갖는 연산에 대해 네 개의 예제를 주고 연산(5, 5, 5)의 값을 물어보면 <표 4>와 같이 추론을 통해 답을 구하는 것을 볼 수 있다. 이와 같이 퓨샷 러닝을 이용한 추론의 가능성이 발견되었으나, 퓨샷 프롬프팅이 단순한 질의응답 문제에서 놀랄만한 성능을 보여준 것에 비해 추론 문제에서는 큰 성과를 보이지 못했다(Brown et al., 2020). 특히 모형의 전체적인 성능이 매개변수의 수가 증가함에 따라 크게 증가한 것에 비해 추론의 성능은

별로 향상되지 못했다. <표 5>는 추론에 실패하는 예제를 보여준다(Wei et al., 2022).

〈표 4〉 퓨샷 프롬프트를 이용한 추론 예제

프롬프트	연산(1, 1, 1): 3 연산(2, 2, 2): 12 연산(3, 3, 3): 27 연산(4, 4, 4): 48 연산(5, 5, 5):
답변	연산(5, 5, 5)의 결과를 구하기 위해서는 주어진 연산 패턴을 이해해야 합니다. 주어진 연산 패턴은 각 숫자를 자기 자신의 제곱으로 만들어서 더하는 것입니다. 따라서, 연산(5, 5, 5)의 결과는 다음과 같이 계산됩니다: (5^2) + (5^2) + (5^2) = 25 + 25 + 25 = 75 따라서, 연산(5, 5, 5)의 결과는 75입니다

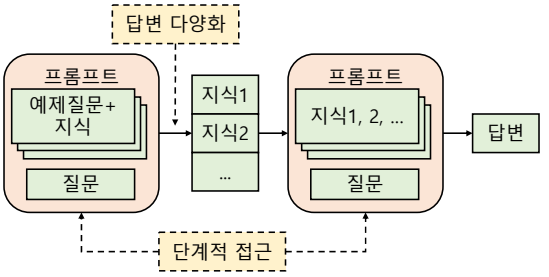
〈표 5〉 퓨샷 러닝 추론의 실패 사례

프롬프트	Q: 로저는 5개의 테니스 공을 갖고 있다. 그는 2캔의 테니스 공을 더 샀다. 각 캔에는 3개의 테니스 공이 들어 있다. 그는 이제 몇 개의 공을 가지고 있는가? A: 답은 11이다. Q: 카페에 23개의 사과가 있다. 점심을 만들기 위해 20개를 사용하고 6개를 더 샀다면 몇 개의 사과가 남아있는가? A:
답변	답은 3이다.

<표 5>에서 퓨샷 프롬프트에 대해 3개라고 답하고 있지만, 실제 답은 9개가 맞다. 퓨샷 러닝에 추론 능력이 있는 것은 맞지만, 여러 단계의 추론 과정을 거치는 경우에는 추론에 한계가 있는 것이 나타났다. 따라서 추론 성능을 개선하기 위한 연구가 이후 진행되기 시작한다.

4.2 지식 생성(Generated Knowledge) 프롬프팅

Liu et al. (2021)은 거대언어모델에서 일반적인 상식추론의 성능을 높이기 위한 지식 생성 프롬프팅 방법을 제안했다. 이 방법은 퓨샷 러닝을 이용해 먼저 문제를 해결하기 위한 지식을 생성하고, 생성된 지식과 문제를 다시 거대언어모델에 입력하여 올바른 답을 생성했다. <그림 3>은 이와 같은 과정을 보여준다.



〈그림 3〉 지식 생성 프롬프팅 과정

예를 들어 거대언어모델에 “골프는 다른 사람보다 더 높은 점수를 얻고자 하는 운동이다”의 진위 여부를 질문하면 “예”라고 대답하는데, 이것은 대부분의 운동이 더 높은 점수를 얻으려고 하기 때문에 발생하는 오류이다. Liu et al. (2021)은 이를 해결하기 위해 먼저 퓨샷러닝을 이용해 골프와 관련한 지식을 생성하는 단계를 거치고, 생성된 지식을 배경지식으로 제공하면서 질문을 함으로써 “아니요”라는 대답을 얻을 수 있었다.

<표 6>은 논문에서 사용된 예제를 GPT 3.5에서 실제로 실행시킨 예제이다. 다만, ChatGPT는 사람의 피드백을 기반으로 강화학습(RLHF, Reinforcement Learning from Human Feedback)을 수행하는 과정에서 지식 생성 프롬프팅이 제안되기 이전에 있었던 문제점들이 개선되어 굳이 지식 생성 프롬프팅을 하지 않더라도 올바른 답을 제공한다(Ouyang et al., 2022). 그러나 지식 생성 프롬프팅은 1단계 프롬프트에서 나온 결과를 다시 2단계 프롬프트

에서 사용하는 단계적 방법을 제시함으로써 이후 프롬프트 엔지니어링 연구에 영향을 미치게 된다.

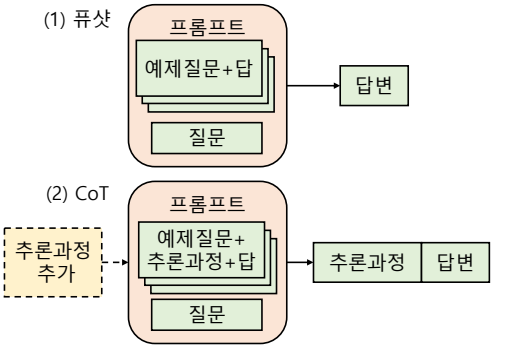
Liu et al. (2021)의 연구에서 제안된 방법을 사용한 결과, 상식추론 문제 벤치마크 데이터셋인 QASC-dev에서 지식을 사용하지 않은 경우의 정확도 72%보다 9%p 높은 81%의 정확도를 보였다.

〈표 6〉 지식 생성 프롬프팅 예제

프롬프트 (1)	입력에 대해 지식을 생성해줘. 입력: 구름은 어떤 종류의 물로 형성되어 있나요? 지식: 구름은 수증기로 이루어져 있습니다 입력: 유전자가 전달되는 과정은? 지식: 유전자는 부모로부터 자손에게 전달됩니다 입력: 골프는 다른 사람보다 더 높은 점수를 얻으려는 운동이다 지식:
답변 (1)	골프에서는 더 낮은 점수가 더 높은 성적을 나타냅니다
프롬프트 (2)	골프에서는 더 낮은 점수가 더 높은 성적을 나타냅니다 아래 문장이 맞는지 알려줘 "골프는 다른 사람보다 더 높은 점수를 얻으려는 운동이다"
답변 (2)	아니요, 골프에서는 낮은 점수가 더 높은 성적을 나타냅니다. 따라서 정확한 문장은 다음과 같습니다: "골프는 다른 사람보다 더 낮은 점수를 얻으려는 운동이다."

4.3 CoT(Chain of Thought) 프롬프팅

Wei et al. (2022)은 4.1절에서 지적한 퓨샷 프롬프팅의 추론 문제점을 개선하기 위해 CoT 프롬프팅 기법을 제안했다. CoT 프롬프팅의 기본적인 아이디어는, 일반적인 퓨샷 프롬프팅이 질문과 답의 쌍을 예제로 사용한 것에 비해 답을 도출하는 추론 과정을 예제에 추가하는 것이다. 이를 통해 거대언어모델이 답을 만들어내는 추론 과정을 학습하도록 했다. <그림 4>는 퓨샷 프롬프팅의 프롬프트 구성과 CoT 프롬프팅의 구성을 비교해서 보여준다. 이와 같이 추론과정을 프롬프트에 추가하는 아이디어는 이후 거의 모든 프롬프트 엔지니어링 연구에서 활용된다.



〈그림 4〉 CoT 프롬프팅 과정

〈표 7〉 CoT 프롬프팅 예제

프롬프트	Q: 로저는 5개의 테니스 공을 갖고 있다. 그는 2캔의 테니스 공을 더 샀다. 각 캔에는 3개의 테니스 공이 들어 있다. 그는 이제 몇 개의 공을 가지고 있는가? *A: 로저는 5개의 공으로 시작했다. 각기 3개의 공이 들어 있는 2캔은 6개의 공이다. 5 + 6 = 11.* 답은 11이다. Q: 카페에 23개의 사과가 있다. 점심을 만들기 위해 20개를 사용하고 6개를 더 샀다면 몇 개의 사과가 남아있는가? A:
답변	카페에 처음에는 23개의 사과가 있었습니다. 이 중 20개를 사용했으므로 23 - 20 = 3개의 사과가 사용된 후 남았습니다. 그리고 추가로 6개를 더 샀습니다. 따라서 총 남아있는 사과의 수는 3 + 6 = 9개입니다.

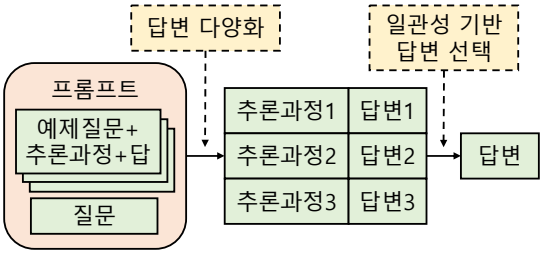
<표 7>의 예제는 연구에서 사용된 산술추론을 위한 CoT 프롬프트의 예제이다(Wei et al., 2022). 예제에서 첫 Q:와 A: 쌍은 문제를 푸는 추론 단계를 보여줌으로써 거대언어모델이 과정을 학습하게 한 후 두 번째 문제를 제시하고 있다. 첫 A:에서 이탤릭체로 되어있는 부분은 일반적인 프롬

프트에는 없고 CoT에만 있는 추론과정에 대한 설명이다. 이 부분을 통해 거대언어모델이 문제 해결방법을 학습하게 된다. 그 결과 퓨샷에서는 3개라고 답하면서 틀렸지만, 위 예와 같이 올바른 추론과정과 답을 생성한다.

이 연구에서는 CoT 프롬프팅의 효과를 검증하기 위해 GPT-3, LaMDA, PaLM 등 총 5개 모형을 대상으로 일반 프롬프트와 CoT 프롬프트의 성능을 비교했다(Chowdhery et al., 2023; Thoppilan et al., 2022). 실험에 사용된 GPT-3는 1,750억 개의 매개변수를 사용하고, LaMDA는 1,370억 개의 매개변수를 사용한 반면, PaLM은 5,400억 개의 매개변수를 사용했다. 실험은 산술 추론, 상식 추론, 기호 추론 세 분야로 나누어 총 12개의 벤치마크 데이터셋을 대상으로 진행되었다. 실험을 수행한 결과, 대부분의 경우에서 CoT 프롬프트가 일반적인 프롬프트에 비해 좋은 성능을 보였으며, PaLM 540B과 같이 매개변수가 큰 모형에서 그 차이가 더 두드러지는 경향이 있었다.

4.4 자기 일관성(Self-Consistency) 프롬프팅

자기 일관성은 퓨샷 프롬프팅이 그리디 방식을 이용해 가장 확률이 높은 하나의 답만을 사용하는 것을 보완하기 위해 제안되었다(Wang et al., 2022). 아이디어는 비교적 간단한데, 하나의 답만을 구하는 대신 다양한 추론과정을 가지는 여러 개의 답을 거대언어모델로부터 샘플링하고 그 중에서 .가장 많은 답(다수 투표 방식) 혹은 일관성이 높은 답을 선택한다. 여러 개의 답을 생성하기 위한 샘플링은 온도샘플링, Top-k 샘플링과 같은 방법을 사용했다(Ficler et al., 2017; Radford et al., 2019). <그림 5>는 자기 일관성 프롬프팅의 실행 단계를 보여준다. 답변의 선택은 앞서 설명한 것과 같이 다수 투표에 의해 가장 많은 답을 선택하거나, 일관성을 계산해서 일관성이 높은 답을 선택했다. 이 때 일관성은 언어모델이 생성한 추론과정과 답변에 있는 모든 토큰의 생성 확률을 여러 가지 방식으로 합산해서 사용했다.



〈그림 5〉 자기 일관성 프롬프팅 과정

〈표 8〉 자기 일관성 프롬프팅 예제

프롬프트 (1)	Q: 로저는 5개의 테니스 공을 갖고 있다. 그는 2캔의 테니스 공을 더 샀다. 각 캔에는 3개의 테니스 공이 들어 있다. 그는 이제 몇 개의 공을 가지고 있는가? A: 로저는 5개의 공으로 시작했다. 각기 3개의 공이 들어 있는 2캔은 6개의 공이다. 5 + 6 = 11. 답은 11이다. Q: 자넷의 오리는 하루에 16개의 알을 낳는다. 그녀는 매일 아침 세 개를 먹고 머핀을 만들기 위해 4개를 사용한다. 그녀는 하나에 2달러를 받고 남은 알을 판다. 자넷은 하루에 얼마의 돈을 벌 수 있는가? A:
답변 (1)	자넷의 오리는 하루에 16개의 알을 낳습니다. 그녀는 매일 아침 3개를 먹고, 4개를 머핀을 만들기 위해 사용합니다. 따라서 그녀가 판매할 수 있는 알의 수는 16 - 3 - 4 = 9개입니다. 각 알에 대해 2달러를 받으므로, 하루에 벌 수 있는 돈은 9 * 2 = 18달러입니다.
답변 (2)	자넷의 오리는 하루에 16개의 알을 낳기 때문에, 그녀는 하루에 16개의 알을 가지고 있다. ...그녀는 하루에 9개의 알을 판다. 하나의 알에 2달러를 받으므로, 그녀는 하루에 9 x 2 = 18달러를 벌 수 있다.
답변 (3)	자넷의 오리는 하루에 16개의 알을 낳습니다. 그러므로 그녀는 하루에 16 - 4 = 12개의 알을 팔 수 있습니다. 한 개의 알에 2달러를 받기 때문에, 자넷은 하루에 12 * 2 = 24달러를 벌 수 있습니다.

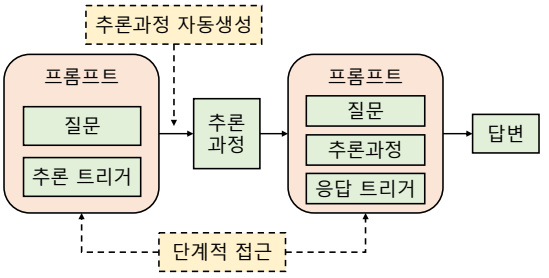
<표 8>은 자기 일관성 프로프팅 기법의 예제를 보여준다. 프롬프트는 CoT 프롬프팅과 동일한 프롬프트를 사용한다. 다만 추론을 실행할 때 거대언어모델의 온도를 0.5에서 1.5 사이로 조절하면서 세 개의 답변을 샘플링했다. 답변을 보면, 답변 1, 답변 2는 18달러라고 답했으나, 답변 3은 24달러라고 답했다. 따라서 다수 투표의 기준 혹은 일관성 기준에 따라 18달러를 답으로 선택한다.

이 연구에서는 총 12개의 벤치마크 데이터셋을 사용하여 성능을 평가했으며, 모든 벤치마크에서 CoT 프롬프트의 성능을 개선했고, 그 중 10개 벤치마크에서 기록을 경신했다.

4.5 제로샷 CoT 프롬프팅

CoT 프롬프팅은 프롬프트에 포함된 추론 과정을 통해 LLM의 추론 능력을 향상시킨다는 장점이 있지만, 문제를 해결하는 방법을 사용자가 설명해줘야 한다는 단점이 있다. 다시 말해서 사용자가 풀 수 있는 문제들을 예제로 주면서 새로운 문제를 해결하기 때문에 사용자가 풀 수 없는 유형의 문제는 답을 구할 수 없다. 제로샷 CoT 프롬프팅은 문제의 추론 과정 혹은 해결 과정을 사용자가 입력하지 않고 단계적으로 문제를 해결하는 방식이다(Kojima et al., 2022). 기본적인 아이디어는 Liu et al. (2021)이 제안한 지식 생성 프롬프팅과 유사하다. 지식 생성 프롬프팅이 문제 해결에 필요한 지식을 생성하는 1단계와 생성된 지식을 이용해 문제를 해결하는 2단계로 구성된 것과 같이, 제로샷 CoT 프롬프팅도 추론을 유도하는 1단계와 답을 구하는 2단계로 구성된다.

<그림 6>은 이와 같은 제로샷 CoT 프롬프팅의 두 단계를 보여준다. 1단계에서는 질문과 추론을 유도하는 트리거 문장(Trigger Sentence)을 이용해 주어진 질문에 대한 추론 과정을 생성한다. 2단계에서는 질문과 함께 1단계에서 생성한 추론과정을 포함하고 응답을 유도하는 트리거 문장을 추가하여 답변을 생성한다.



〈그림 6〉 제로샷 CoT 프롬프팅 과정

〈표 9〉 제로샷 CoT 프롬프팅 예제

프롬프트 (1)	Q: 카페에 23개의 사과가 있다. 점심을 만들기 위해 20개를 사용하고 6개를 더 샀다면 몇 개의 사과가 남아있는가? A: 단계적으로 생각해보자.
답변 (1)	카페에 23개의 사과가 있고, 점심을 만들기 위해 20개를 사용했으므로 23 - 20 = 3개의 사과가 남게 됩니다. 그 후 6개를 더 구매했으므로 3 + 6 = 9개의 사과가 남게 됩니다. 따라서, 9개의 사과가 남아있습니다.
프롬프트 (2)	Q: 카페에 23개의 사과가 있다. 점심을 만들기 위해 20개를 사용하고 6개를 더 샀다면 몇 개의 사과가 남아있는가? 카페에 23개의 사과가 있고, 점심을 만들기 위해 20개를 사용했으므로 23 - 20 = 3개의 사과가 남게 됩니다. 그 후 6개를 더 구매했으므로 3 + 6 = 9개의 사과가 남게 됩니다. 따라서, 9개의 사과가 남아있습니다. 그래서 답(아라비아 숫자)은?
답변 (2)	9

<표 9>는 제로샷 CoT 프롬프팅의 예제를 보여준다. 1단계 프롬프트는 “Q:[X]. A: [T]”로 구성되며, [X]는 제로샷 프롬프팅과 동일하게 프롬

프트의 질문을 사용한다. 그러나 일반적인 제로샷은 A: 뒤의 [T] 위치를 비워둠으로써 거대언어모델이 답을 채우도록 유도하는 반면, 제로샷 CoT에서는 추론을 유도하는 트리거 문장을 사용함으로써 거대언어모델이 답을 채우는 대신 문제해결을 위한 추론과정을 유추하도록 만든다. <표 9>에서는 트리거 문장으로 "단계적으로 생각해보자"를 사용했는데, 이는 원문의 "Let's think step by step."를 한국어로 번역한 것이다. CoT 프롬프팅 예제와의 중요한 차이점은, CoT 프롬프팅 예제의 앞 Q:, A: 쌍을 제로샷 CoT에서는 사용하지 않는다는 것이다.

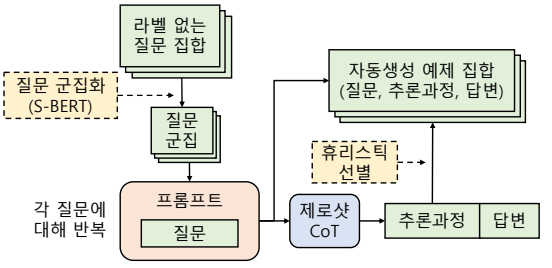
2단계 프롬프트는 "[X] [Z] [A]"로 구성된다. [X]는 1단계에서 사용한 프롬프트이고, [Z]는 1단계의 답변 내용이며, [A]는 응답 트리거이다. <표 9> 프롬프트2의 마지막 문장 "그래서 답(아라비아 숫자)은?"은 답을 유도하기 위해 사용한 응답 트리거 [A]이고, 그 위의 문단은 1단계의 답변 내용인 [Z]이다.

Liu 등은 총 6개의 벤치마크 데이터셋을 사용하여 GPT-3와 PaLM 540B 모델에 대해 제로샷, 제로샷 CoT, 퓨샷, 퓨샷 CoT의 성능을 비교했다. 그 결과, 단순한 계산이 아닌 복잡한 추론을 요구하는 MultiArith, GSM8K와 같은 데이터셋에서 제로샷 CoT가 제로샷과 퓨샷보다 월등한 성능을 보였으며, 그 차이는 모델의 크기가 커질수록 더 커졌다. 다만 퓨샷 CoT보다는 성능이 떨어졌는데, 퓨샷 CoT와 제로샷 CoT를 결합한 제로플러스퓨샷(Zero-Plus-Few-Shot) CoT에서는 퓨샷 CoT보다 더 나은 성능을 보였다.

제로플러스퓨샷 CoT는 퓨샷 CoT에서 사용하는 각각의 예제에 대해 추론과정 앞에 추론 트리거 문장을 추가하여 학습한 방식으로, 하나의 예제를 "질문 + 추론과정 + 답"으로 구성하는 대신 "질문 + 추론 트리거 + 추론과정 + 답"의 형태로 구성한 것이다. 이렇게 함으로써 퓨샷 CoT의 답변에 의도적인 추론과정 생성을 추가하고자 했다.

4.6 Auto-CoT(Automatic CoT) 프롬프팅

제로샷 CoT 프롬프팅이 사용자가 직접 추론과정을 작성해서 예제로 입력해야 하는 수작업을 없애기는 했으나, 성능은 여전히 퓨샷 CoT에 비해 떨어진다. 위에서 설명한 제로플러스퓨샷은 퓨샷 CoT보다 좋은 성능을 보였으나, 예제를 직접 작성해야 하는 문제점이 아직 남아있다. Auto-CoT 프롬프팅은 이러한 문제점을 해결하기 위해 답안의 추론과정을 자동으로 생성하는 방안을 제시했다(Zhang et al., 2022).



〈그림 7〉 Auto-CoT 프롬프팅 과정

답안의 추론과정을 자동으로 생성하기 위해 제로샷 CoT의 1단계를 이용할 수 있으나, 문제는 이 때 생성되는 추론과정이 항상 정답은 아니라는 것이다. 이 추론과정을 그대로 입력으로 이용할 경우, 잘못된 예제를 제시함으로써 틀린 답을 유도하게 될 수 있다.

Zhang et al. (2022)은 이러한 문제점이 미치는 영향을 최소화하기 위해 입력에 포함되는 예제들을 최대한 주어진 문제와 유사한 문제로 구성

하는 아이디어를 생각해냈다. 그러나 유사 문제에 대해 제로샷 CoT를 적용해 자동으로 추론과정을 생성하고 이를 예제로 사용해 퓨샷 CoT를 적용한 결과, 제로샷 CoT로 생성되는 잘못된 추론과정이 한꺼번에 몰려서 더 나쁜 결과를 가져오는 경우가 발생한다는 사실을 알았다. 이는 결과적으로 평균 성능의 하락을 가져왔다. 따라서 Auto-CoT는 이와 같은 문제점을 극복하기 위해 군집 분석을 활용해 퓨샷 CoT에서 사용할 예제 집합을 분산하여 생성하는 방안을 제시했다.

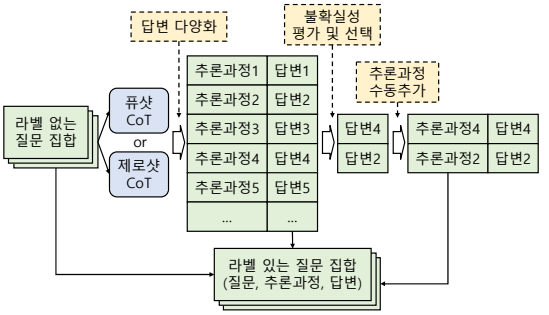
<그림 7>은 Auto-CoT의 프롬프팅 단계를 보여준다. Auto-CoT는 두 단계로 이루어지는데, 먼저 1단계에서는 라벨 즉 추론과정과 답이 없는 문제 집합에 군집화를 적용해 의미적으로 유사한 군집들로 분리한다. 2단계에서는 제로샷 CoT를 이용해 군집에 있는 모든 문제들에 대해 추론과정을 생성하고, 휴리스틱을 적용해 각 군집을 대표하는 하나씩의 문제, 추론과정, 답변을 선별한다. 휴리스틱은 추론과정이 60개의 토큰을 넘지 않고 5개의 추론 단계를 넘지 않는 기준을 사용했다. 선별된 결과는 예제 집합에 추가되고, 주어진 질문에 대해 제로플러스퓨샷 CoT 프롬프트의 형태로 사용된다.

실험은 산술 추론, 상식 추론, 기호 추론 세 분야로 나누어 총 10개의 벤치마크 데이터셋을 대상으로 진행되었다. 실험 결과 상식 추론을 제외한 8개의 벤치마크에서 가장 우수한 성능을 보였으며, 상식 추론에서는 퓨샷 CoT에 근접한 성능을 보였다.

4.7 액티브(Active) 프롬프팅

Auto-CoT 프롬프팅이 비록 퓨샷 CoT에 근접하거나 더 나은 성능을 보인다고 하더라도, 제로샷 CoT에 의해 생성되는 추론과정이 사람이 만든 추론과정보다 정확하지 않을 가능성은 항상 있다. 그럼에도 불구하고 Auto-CoT 프롬프팅과 같은 기법을 사용하는 이유는 사람이 추론과정을 생성하는 것이 높은 비용을 요구하기 때문이다. 액티브 프롬프팅은 Auto-CoT 프롬프팅과 사람이 예제를 작성하는 퓨샷 프롬프팅의 절충안과 같은 방법이라고 할 수 있다(Diao et al., 2023).

액티브 프롬프팅은 학습을 위한 문제 집합과 평가를 위한 문제 집합이 나누어져 있는 환경에서 학습용 문제 집합을 이용해, 평가용 문제 집합에서 사용할 예제들을 개발하는 환경을 가정하고 있다.



〈그림 8〉 액티브 프롬프팅 과정

<그림 8>은 액티브 프롬프팅의 추론 과정을 보여준다. 액티브 프롬프팅은 네 단계로 구성되어 있다. 1단계에서는 학습용 문제 집합에 있는 문제들에 대해 기존에 존재하는 예제들을 사용하는 퓨샷 CoT 프롬프트를 적용하거나 혹은 예제가 없는 경우에는 제로샷 CoT 프롬프트를 이용해 각 문제 별로 k개의 다양한 추론과정 예제들을 생성한다. 2단계에서는 생성된 예제들에 대해 다른 예제들과의 불일치 정도를 기반으로 불확실성(Uncertainty)을 수치로 추정하고, 불확실성이 높은 상위 n개의 예제들을 추출한다. 3단계

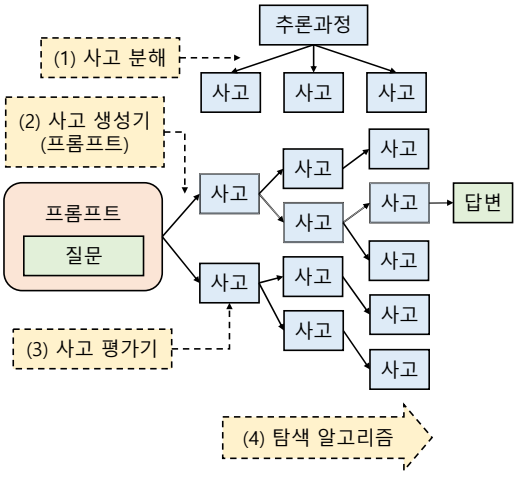
에서는 추출된 예제들에 대해 사람이 직접 정확한 예제를 생성해서 교체한다. 이렇게 하면 평가용 문제 집합에 대해 사용할 예제 집합이 완성된다. 마지막 4단계에서는 완성된 예제 집합을 사용하여 평가용 문제 집합에 대해 추론을 한다. 이 때 자기 일관성 연구를 적용하여, 하나의 답만 구하는 대신 온도를 변경하면서 m개의 답을 구하고 그 중에서 일관성이 가장 높은 답을 선택하도록 했다.

8개의 벤치마크 데이터셋을 대상으로 한 실험 결과, 제안된 액티브 프롬프팅의 결과가 모든 벤치마크에서 가장 우수한 것으로 나타났다. 그러나 기존 연구는 하나의 질문에 대한 프롬프트 생성 방식을 제안한 반면, 액티브 프롬프팅은 학습 데이터셋과 평가 데이터셋이 나누어져 있는 환경에서 학습 데이터셋을 이용해 평가 데이터셋에서 사용할 예제 집합을 생성하는 프레임워크를 제안했다는 점에서 직접적인 비교는 어렵다고 할 수 있다.

4.8 ToT(Tree of Thought) 프롬프팅

ToT 프롬프팅은 앞서 소개된 프롬프팅 기법들과는 달리, 보다 복잡한 추론 문제를 해결하기 위해 제안되었다(Yao et al., 2023). 사람이 의사결정을 하는 방식은 크게 두 가지로 분류할 수 있는데, 첫째 방식은 빠르고 자동적이고 무의식적인 반면, 둘째 방식은 느리고, 심사숙고하며 의식적이다. 둘째 방식의 특징을 더 살펴보면 현재의 선택에 대한 다양한 대안을 탐색한다는 점, 현재의 상황을 주의 깊게 평가하고 전체적인 관점에서 보다 나은 의사결정을 하기 위해 미리 다음 상태를 들여다보거나 다시 이전으로 돌아가서 생각한다는 점 등이 있다. Yao et al. (2023)은 복잡한 문제를 해결하기 위해 심사숙고를 하는 둘째 방식을 반영한 프롬프팅 기법을 제안하였다.

<그림 9>는 입력으로부터 다양한 사고를 생성하고, 트리 형태로 다음 단계의 사고들을 생성하면서 그 중 가장 적합한 사고를 따라가서 출력을 만들어내는 ToT 프롬프팅의 추론 단계를 보여준다.



〈그림 9〉 ToT 프롬프팅 과정

Yao et al. (2023)은 ToT 프롬프팅의 구현을 위해 4개의 요소를 제안했는데, 그 중 첫째는 사고 분해(Thought Decomposition)이다. ToT 프롬프팅에서는 먼저 주어진 문제를 유형에 따라 적절하게 작은 사고의 단위로 분해한다.

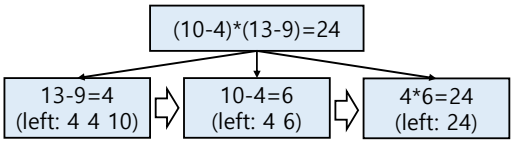
둘째 요소는 사고 생성기(Thought Generator)이다. 사고 생성기는 현재 트리 상태에서 다음 사고 단계를 생성한다. 사고의 생성에는 샘플(Sample)과 제안(Propose)의 두 가지 방식이 있다. 샘플은 현재까지의 트리 상태를 입력으로 한 CoT 프롬프트를 독립적으로 k번 반복하여 다양한 사고를 생성한다. 제안은 제안 프롬프트(Propose Prompt)를 사용해서 k개의 사고를 한 번에 생성한다. 샘플 방식은 다양성을 높이는 반면, 제안은 사고가 한

단어나 한 줄처럼 단순할 때 사용하며 중복을 피할 수 있다.

셋째 요소는 상태 평가기(State Evaluator)이다. 휴리스틱을 이용해 현재 상태를 평가함으로써 트리에서 최적의 경로를 찾는 것이 가능하다. 상태 평가를 위한 전략 역시 각 상태를 독립적으로 평가하는 전략과 주어진 전체 상태에서 상태들 간의 비교를 통해 평가하는 전략 두 가지를 제안했다.

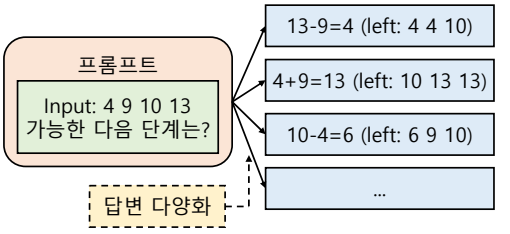
넷째 요소는 탐색 알고리즘이다. 트리를 탐색하기 위한 전형적인 알고리즘인 너비우선탐색과 깊이우선탐색을 사용했다.

논문에서는 24 게임(Game of 24), 창조적 글쓰기, 5x5 크로스워드의 세 작업을 대상으로 실험을 진행했다. 이 중에서 24 게임은 주어진 4개의 숫자로 사칙연산을 해서 24를 만드는 작업이다. 예를 들어 “4 9 10 13”이 입력으로 주어지면 답은 “(10 - 4)*(13-9)=24”이다. <그림 10>은 ToT의 1단계인 사고 분해의 예를 보여준다. 예에서 주어진 문제는 3단계의 사고로 분해되는데, 첫째 단계에서는 13과 9를 이용해서 4를 만들어 4, 10과 함께 세 개의 수가 남는다. 둘째 단계에서는 10과 4를 이용해 6을 만들어 (4, 6)이 남는다. 마지막 단계에서 4와 6으로 24를 만들면 목표를 달성하게 된다. 이와 같이 문제를 사고 단계로 분리하는 것은 주어진 문제의 특성에 맞게 적절한 수로 정의하여 수행한다.



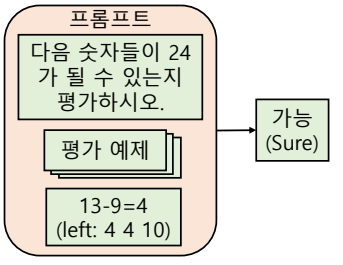
〈그림 10〉 ToT 사고 분해 예제

<그림 11>은 ToT의 2단계인 사고 생성 예제를 보여준다. 숫자가 네 개 있는 초기 단계에서는 프롬프트를 이용해 그림과 같이 다음 사고 단계의 후보들을 생성한다.



〈그림 11〉 ToT 사고 생성 예제

생성된 후보들에 대해서는 ToT의 3단계인 사고 평가를 실시한다. <그림 12>와 같이 사고 평가는 사고 생성과 마찬가지로 프롬프트를 이용해 이루어지며 평가 예제들을 이용한 퓨샷 프롬프트를 사용해서 “Sure”/“Likely”/“Impossible” 중에 판별하도록 한다. 평가 결과를 이용해 트리에서 해를 찾기 위한 휴리스틱 기반의 탐색이 이루어진다.



〈그림 12〉 ToT 사고 평가 예제

실험은 GPT-4를 사용하여 이루어졌으며, 실험 결과 CoT 프롬프팅과 자기 일관성 CoT 프롬프팅이 각각 4%와 9%의 성공률을 보인 반면 ToT 프롬프팅은 무려 74%의 성공률을 보였다., 창조

적 글쓰기, 5x5 크로스워드에서도 훨씬 개선된 성능을 보였는데, 이러한 결과는 프롬프팅 구조 혹은 단계의 설계에 따라 거대언어모델이 매우 복잡한 문제도 풀 수 있음을 보여준다.

5. 결론

ChatGPT를 비롯한 거대언어모델은 인공지능 분야에서 일반인이 가장 많이 활용되는 도구가 되었으며, 성능을 개선하기 위한 노력이 다양하게 이루어지고 있다. 그 중에서도 특히 거대언어모델의 추론 능력은 GPT-3이 개발된 이후 문맥 내 학습을 중심으로 급격히 발전해왔다. 이러한 거대언어모델의 추론 능력은 다양한 업무에 인공지능 요소를 도입하고자 하는 기업에게 중요한 요소로 작용하고 있으며 점차 활용의 범위가 넓어지고 있다. 또한 일반인에게도 거대언어모델의 인공지능 능력을 보다 다양하게 활용할 수 있는 기회를 제공하고 있다.

이에 본 논문에서는 추론 능력의 향상을 위한 최근 프롬프트 엔지니어링 기법들의 발전 과정에 대해 조사하고, 각 기법들 간의 관계를 정리하고자 했다. 퓨샷 프롬프팅은 거대언어모델의 추론 능력 연구의 시작이라고 할 수 있으며, 미세조정학습과는 달리 실행 과정에서 몇 개의 예제를 프롬프트에 추가함으로써 기본적인 추론 능력을 가질 수 있음을 보였다. 지식 생성 프롬프팅은 추론에 필요한 지식을 먼저 생성하여 프롬프팅을 단계화함으로써, 프롬프트 기반 추론을 체계화하는 기반을 제공했다고 할 수 있다. CoT 프롬프팅은 추론 과정을 예제에 추가함으로써 이전에 비해 복잡한 추론을 실행할 수 있게 했다. 자기일관성 프롬프팅은 CoT에서 다양한 답변을 생성하고 그 중 일관성이 높은 답변을 선택함으로써 CoT보다 더 높은 성능을 보였는데, 이러한 기법은 이후 많은 프롬프트 엔지니어링에서 활용되었다. 다만 추론과정을 사람이 추가해야 했는데, 이러한 부가적인 노력을 없애기 위해 추론과정에 대한 예제 생성을 자동화하기 위한 제로샷 CoT가 제안되었다. 제로샷 CoT에서는 추론 트리거 문장을 이용해 자동으로 추론과정을 생성할 수 있으나 틀린 추론과정과 답변이 추가될 가능성이 있었다. Auto-CoT와 액티브 프롬프팅은 제로샷 CoT를 개선하기 위해 제안되었으며, Auto-CoT는 군집화를 이용해 추론 성능을 개선한 반면, 액티브 프롬프팅은 불확실성이 높은 추론과정 예제를 골라서 사람이 직접 추론과정을 추가하도록 했다. ToT 프롬프팅은 사고의 분해와 트리 기반의 휴리스틱 탐색을 이용해 기존에는 거의 풀지 못했던 추론 문제를 풀 수 있도록 함으로써 거대언어모델의 추론 수준을 한 단계 더 끌어올렸다고 할 수 있다.

본 논문의 이론적 기여점으로는, 최근의 프롬프트 엔지니어링 연구들을 대상으로 하여 각 기법의 중요한 특징과 단계 그리고 기법들의 발전 과정과 상호 간의 관계를 도식화하고 예제를 이용해 설명함으로써, 거대언어모델의 추론 능력에 관심을 가진 연구자가 프롬프트 엔지니어링 기법들을 보다 쉽고 명확하게 이해하도록 했다는 점이다. 이와 같은 이해를 통해 연구자는 보다 향상된 추론 기법의 설계를 위해 필요한 기존 기법들의 조합이나, 기존 기법들을 보다 효과적으로 활용할 수 있는 방법에 대해 연구할 수 있을 것으로 기대된다. 또한 프롬프트 엔지니어링 기법들을 실제 업무에 활용하고자 하는 실무자는 본 연구를 통해 대상 업무가 요구하는 추론 성능에 따라 적절한 프롬프팅 기법을 선택하거나 조합할 수 있을 것으로 기대된다.

본 논문에서는 가급적 최신 경향에 대해 조사

하기 위해 아카이브(arXiv)[3]의 최근 논문들을 포함하여 정리했다. 이로 인해 최신성은 높아졌으나 학술지에 게재된 논문들과는 달리 검증이 되지 않았다는 문제가 있을 수 있다. 이와 같은 문제점을 보완하기 위해 최근 논문임에도 불구하고 인용수가 높은 논문들만을 선별했으며, 비교적 중요하다고 판단되는 논문의 내용을 상세하게 소개하고자 했다.

향후 학술지나 학술대회에서 거대언어모델의 추론 능력 향상에 대해 보다 많은 논문들이 발표된다면, 검증된 연구를 대상으로 보다 다양하고 체계적인 연구 현황을 조사할 수 있을 것으로 기대된다. 또한 거대언어모델의 추론 능력과 관련하여 최근의 연구 동향은 주로 문맥내 학습과 예제 중심으로만 연구가 이루어지고 있으나, 과거 인공지능 연구에서 1차 논리, 2차 논리와 같은 술어 논리(Predicate Logic)와 시맨틱 웹의 기반이 된 서술 논리(Description Logic) 등의 연구가 활발히 이루어졌으므로, 향후에는 이와 같은 기존 연구들과의 결합 혹은 적용방안이 연구될 수 있을 것으로 기대된다.

참고문헌(References)

[국내 문헌]

김맹근. (2023). ChatGPT 활용 사례 및 전망. 디지털 비즈온. http://www.digitalbizon.com/news/articleView.html?idxno=2331610

[국외 문헌]

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877-1901.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., ... & Fiedel, N. (2023). Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240), 1-113.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., ... & Schulman, J. (2021). Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.

Diao, S., Wang, P., Lin, Y., & Zhang, T. (2023). Active prompting with chain-of-thought for large language models. arXiv preprint arXiv:2302.12246.

Fan, A., Lewis, M., & Dauphin, Y. (2018). Hierarchical neural story generation. *Annual Meeting of the Association for Computational Linguistics*, 56(1), 889-898.

Ficler, J., & Goldberg, Y. (2017). Controlling linguistic style aspects in neural language generation. arXiv preprint arXiv:1707.02633.

Geva, M., Khashabi, D., Segal, E., Khot, T., Roth, D., & Berant, J. (2021). Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9, 346-361.

Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019). The curious case of neural text degeneration. arXiv preprint arXiv:1904.09751.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in Neural*

3) https://info.arxiv.org/about/index.html

Information Processing Systems, 35, 22199-22213.

Koncel-Kedziorski, R., Hajishirzi, H., Sabharwal, A., Etzioni, O., & Ang, S. D. (2015). Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3, 585-597.

Koncel-Kedziorski, R., Roy, S., Amini, A., Kushman, N., & Hajishirzi, H. (2016, June). MAWPS: A math word problem repository. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 1152-1157).

Lampinen, A. K., Dasgupta, I., Chan, S. C., Matthewson, K., Tessler, M. H., Creswell, A., ... & Hill, F. (2022). Can language models learn from explanations in context?. arXiv preprint arXiv:2204.02329.

Liu, J., Liu, A., Lu, X., Welleck, S., West, P., Bras, R. L., ... & Hajishirzi, H. (2021). Generated knowledge prompting for commonsense reasoning. arXiv preprint arXiv:2110.08387.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9), 1-35.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35, 27730-27744.

Patel, A., Bhattamishra, S., & Goyal, N. (2021). Are NLP models really able to solve simple math word problems?. arXiv preprint arXiv: 2103.07191.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.

Reynolds, L., & McDonell, K. (2021, May). Prompt programming for large language models: Beyond the few-shot paradigm. *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (pp. 1-7).

Roy, S., & Roth, D. (2016). Solving general arithmetic word problems. arXiv preprint arXiv:1608.01413.

Srivastava, A., Rastogi, A., Rao, A., Shoeb, A. A. M., Abid, A., Fisch, A., ... & Wang, G. (2022). Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. arXiv preprint arXiv:2206.04615.

Talmor, A., Herzig, J., Lourie, N., & Berant, J. (2018). Commonsenseqa: A question answering challenge targeting commonsense knowledge. arXiv preprint arXiv:1811.00937.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H. T., ... & Le, Q. (2022). Lamda: Language models for dialog applications. arXiv preprint arXiv:2201.08239.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., ... & Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. arXiv preprint arXiv:2203.11171.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35, 24824-24837.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. arXiv preprint arXiv: 2305.10601.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2019). Bertscore: Evaluating text generation with bert. arXiv preprint arXiv: 1904.09675.

Zhang, Z., Zhang, A., Li, M., & Smola, A. (2022). Automatic chain of thought prompting in large language models. arXiv preprint arXiv:2210. 03493.

Abstract

Analysis of Prompt Engineering Methodologies and Research Status to Improve Inference Capability of ChatGPT and Other Large Language Models

Sangun Park* · Juyoung Kang**

After launching its service in November 2022, ChatGPT has rapidly increased the number of users and is having a significant impact on all aspects of society, bringing a major turning point in the history of artificial intelligence. In particular, the inference ability of large language models such as ChatGPT is improving at a rapid pace through prompt engineering techniques. This reasoning ability can be considered as an important factor for companies that want to adopt artificial intelligence into their workflows or for individuals looking to utilize it. In this paper, we begin with an understanding of in-context learning that enables inference in large language models, explain the concept of prompt engineering, inference with in-context learning, and benchmark data. Moreover, we investigate the prompt engineering techniques that have rapidly improved the inference performance of large language models, and the relationship between the techniques.

Key Words : ChatGPT, Prompt Engineering, Large Language Models, Inference, GPT

Received : December 10, 2023 Revised : December 17, 2023 Accepted : December 18, 2023
Corresponding Author : Juyoung Kang

* Department of Industrial Management Information Engineering
** Corresponding Author: Juyoung Kang
e-Business Department, School of Business, Ajou University
206 Worldcup-ro Suwon, 16499, Korea
Tel: +82-31-219-2910, Fax: +82-31-219-1616, E-mail: jykang@ajou.ac.kr

저 자 소 개

박상언
한국과학기술원 전산학사, 경영공학 석사와 박사학위를 취득하였다. 현재 경기대학교 경영정보전공 교수로 재직하고 있으며, 주요 관심분야는 인공지능, 머신러닝, 딥러닝, 텍스트 마이닝 등이다.

강주영
현재 아주대학교 경영대학 e-비즈니스학과 교수로 재직중이며, 포항공과대학교 컴퓨터공학과에서 학사, 서울대학교 컴퓨터공학과에서 석사, 한국과학기술원 경영공학전공에서 경영공학 박사학위를 취득하였다. 주요 관심분야는 빅데이터, 텍스트마이닝, 시맨틱 웹, 지능형전자상거래, 클라우드 컴퓨팅, ERP 등이다. 관련 분야에서 몇편의 저서를 기술하고, 국내외 학회 및 해외 저명 학술지 등에 100여건 이상의 논문을 발표 및 게재하였다.