

[ReAct Prompting]_개념정리+논문

1. ReAct 정의와 등장배경

- chain of thought과 같이 결론에 이르는 추론 과정을 활용하는 것에 더해 언어 모델이 태스크를 해결하는 과정에서 주어진 환경과 상호작용하는 행동에 대한 정보도 프롬프트 내에서 활용하는 것
- 인간이 새로운 작업을 학습하고 의사 결정이나 추론을 할 수 있도록 하는 "행동"과 "추론"의 시너지 효과에서 영감을 받음.
- 생각의 사고(CoT) 프롬프팅은 외부 세계에 대한 접근성이 부족하거나 지식을 업데이트하는 능력이 부족하여 Fact Hallucination이나 오류전파 같은 문제가 발생한다.

2. ReAct의 역할

- 추론과 행동을 대규모언어모델(LLM)과 결합하는 패러다임이다.
- 추론 추적과 행동을 생성하도록 대규모언어모델(LLM)을 유도한다.
 - 시스템은 외부 환경(ex. Wikipedia)과의 상호 작용을 통해 추론에 추가 정보를 통합



< Prompt >

Apple Remote 외에 Apple Remote가 원래 상호 작용하도록 설계된 프로그램을 제어할 수 있는 다른 장치는 어떤 것이 있니?

< 결과 분석 >

모델이 생각/행동함을 볼 수 있다. ReAct는 추론을 지원하기 위해 정보를 검색할 수 있으면, 추론은 다음에 검색할 대상을 타겟팅하는데 도움이 된다.

3. ReAct 작동원리(논문 예시HotpotQA이용)

```
질문 콜로라도 조산 운동의 동쪽 구역이 확장되는 지역의 표고 범위는 얼마이니?
생각 1 콜로라도 조산 운동을 검색하여, 콜로라도 조산 운동의 동쪽 구역이 확장되는 지역을 찾는 다음, 그 지역의 표고 범위를 찾아야 해.
행동 1 검색하기[콜로라도 조산 운동]
관찰 1 콜로라도 조산 운동은 콜로라도와 그 주변 지역에서 산이 형성되는 과정(조산 운동)을 말해.
생각 2 동부에 대한 언급이 없어. 그래서 동부 색터를 찾아봐야겠어.
행동 2 찾아보기[동부 색터]
관찰 2 (결과 1 / 1) 동쪽 색터는 고원 지대까지 뻗어 있고 중앙 평원 조산 운동이라고 불러.
생각 3 콜로라도 조산 운동의 동쪽 색터는 고원 지대까지 뻗어 있어. 따라서 고원 지대를 검색하여 해발 범위를 찾아야 해.
행동 3 검색[고원 지대]
관찰 3 고원 지대는 두 개의 다른 지대 중 하나를 가르켜.
생각 4 대신 고원 지대(미국)를 검색해야해.
행동 4 검색[고원 지대 (미국)]
관찰 4 고원 지대는 대평원(the Great Plains)의 일부 지역이야. 동쪽에서 서쪽으로, 고원 지대는 약 1,800피트에서 7,000피트(550~2,130m)까지 고도가 높아져.[3]
생각 5 고원 지대의 해발 고도는 약 1,800~7,000피트이므로, 정답은 1,800~7,000피트야.
행동 5 완료[1,800~7,000피트]
...
```

문제를 해결하는 과정에서 **생각(Thought)**, **행동(Action)**, **관찰(Observation)**의 일련의 단계들이 순서대로 구성됨

1. **생각(Thought):** 문제를 분석하고, 어떻게 해결할지 전략을 세우는 단계
2. **행동(Action):** 문제를 해결하기 위해 검색하거나 계산하는 등의 행동을 하는 단계
3. **관찰(Observation):** 행동의 결과로 얻은 정보를 바탕으로 새로운 사실을 확인하는 단계

ReAct 프롬프트에서는 이러한 궤적이 여러 생각-행동-관찰 단계를 포함하여, 모델이 어떤 방식으로 문제를 풀어가는지 보여준다.

→추론이 중요한 작업 (예: HotpotQA)

- 여러 "생각(Thought)"과 "행동(Action)" 단계가 반복되면서 문제를 해결한다. 즉, 문제를 여러 부분으로 나누고 각 부분에 대해 생각하고 정보를 찾는 과정을 계속 이어 나가는 방식. 이 과정에서 생각의 과정이 많이 필요합니다.

→많은 행동 단계가 포함되는 의사 결정의 경우

- **의사 결정 중심의 작업**은 행동이 더 많이 요구됩니다. 예를 들어, 어떤 행동을 실행하는 것이 중요한 작업에서는 각 행동이 중요한 역할을 하므로, 행동 단계가 더 많아질 수 있습니다.
- 행동을 하기 전에 복잡한 사고가 필요하지 않을 수도 있습니다. 생각(Thought) 단계는 드물게 사용됩니다.

작업의 성격에 따라 ReAct 방식에서 "생각"과 "행동"의 사용 빈도가 달라진다

4. Results on Knowledge-Intensive Tasks

Prompt Method ^a	HotpotQA (EM)	Fever (Acc)
Standard	28.7	57.1
CoT (Wei et al., 2022)	29.4	56.3
CoT-SC (Wang et al., 2022a)	33.4	60.4
Act	25.7	58.9
ReAct	27.4	60.9
CoT-SC → ReAct	34.2	64.6
ReAct → CoT-SC	35.1	62.0
Supervised SoTA^b	67.5	89.5

- CoT는 사실 착각에 빠짐
- ReAct의 구조적 제약은 추론 단계를 공식화할 때 유연성이 떨어진다
- ReAct는 검색하는 정보에 크게 의존하여 정보가 없는 검색 결과는 모델 추론을 방해하고 생각을 복구하고 재구성하는데 어려움을 초래
- ReAct가 Act(행동만 포함)보다는 더 나은 성능을 보임
- CoT-SC와 ReAct 방식이 성능 향상이 크고, 특히 **CoT-SC → ReAct : 뛰어남**

5 . Results on Decision Making Tasks

: 의사결정 작업에서의 ReAct 성능평가

(2b) ReAct (Reason + Act)

```

Act 1: Think[First I need to find a pepper shaker...more likely to appear in cabinets (1-6), countertops (1-3), ... ]
Act 2: Go to cabinet 1
Obs 2: On the cabinet 1, you see a vase 2.
(Here the agent go to cabinet 1, then cabinet 2, 3, then countertop 1 and 2)
Act 6: Go to countertop 3
Obs 6: On the countertop 3, you see a apple 1, a bread 1, ..... a pepper shaker 1, and a vase 3.

Act 7: Take pepper shaker 1 from countertop 3
Obs 7: You pick up the pepper shaker 1 from the countertop 3.

Act 8: Think[Now I find a pepper shaker 1. Next, I need to put it in/on drawer 1. ]
Act 9: Go to drawer 1
Obs 9: Drawer 1 is closed.

Act 10: Open drawer 1
Obs 10: You open Drawer 1 ...

Act 11: Put pepper shaker 1 in/on drawer 1
Obs 11: You put pepper shaker 1 in/on the drawer 1.

```

✓

→ALFWorld(텍스트 기반 게임)와 WebShop(온라인 쇼핑 환경)에서 평가

: 복잡한 환경에서 목표를 하위 목표로 정확하게 분해하는데 Act보다 성능이 우수함

→ReAct의 이점

- 목표를 더 효과적으로 달성하기 위해 생각과 행동을 결합

- 단순한 행동 기반 방식인 **Act**보다 더 나은 성과를 보여줌

→한계

- ReAct가 성능이 뛰어나지만, 여전히 이러한 작업에서 전문 인력의 성과에는 미치지 못함

→핵심

- ReAct는 복잡한 의사 결정 작업에서 목표 달성을 위한 최적의 추론-행동 조합을 통해 성능을 높임

6. LangChain ReAct Usage(코드 예시)

```
!pip install --upgrade openai
!pip install --upgrade langchain
!pip install --upgrade python-dotenv
!pip install google-search-results

# import libraries
import openai
import os
from langchain.llms import OpenAI
from langchain.agents import load_tools
from langchain.agents import initialize_agent
from dotenv import load_dotenv
load_dotenv()

# load API keys; you will need to obtain these if you haven't yet
os.environ["OPENAI_API_KEY"] = os.getenv("OPENAI_API_KEY")
os.environ["SERPER_API_KEY"] = os.getenv("SERPER_API_KEY")

llm = OpenAI(model_name="text-davinci-003", temperature=0)
tools = load_tools(["google-serper", "llm-math"], llm=llm)
agent = initialize_agent(tools, llm, agent="zero-shot-react-description", verbose=True)

agent.run("Olivia Wilde의 남자 친구는 누구이니? 0.23 거둬제곱을 하면 현재 그의 나이는 얼마이니?")

> Entering new AgentExecutor chain...
나는 Olivia Wilde의 남자 친구가 누구인지 알아내고 0.23 거듭제곱을 한 그의 나이를 계산해야해.
행동 : 검색
행동 입력 : "Olivia Wilde 남자 친구"
관찰 : Olivia Wilde는 Jason Sudeikis와 수년간의 약혼을 끝낸 후 Harry Styles와 사귀기 시작했어. (두 사람간의 관계 타임라인 참조)
생각 : Harry Styles의 나이를 알아야해.
행동 : 검색
행동 입력 : "Harry Styles 나이"
관찰 : 29 세
```

생각 : 나는 29 에 0.23 거듭제곱을 계산해야 해.

행동 : 계산기

행동 입력 : $29^{0.23}$

관찰 : 답변 : 2.169459462491557

생각 : 나는 이제 마지막 답변을 알고 있어.

최종 답변 : Olivia Wilde의 남자 친구인 Harry Styles는 29 세이고, 그의 나이에 0.23 거듭제곱한 값은 2.169459462491557 이야.

> Finished chain.

"Olivia Wilde의 남자 친구인 Harry Styles는 29 세이고, 그의 나이에 0.23 거듭제곱한 값은 2.169459462491557 이야."