25-W NLP 논문 요약 과제

- NLP week 6 복습과제
- 21기 강서연
- **Principled Instructions Are All You Need for Questioning LLaMA-1/2, GPT-3.5/4**

Abstract

- 26가지 원칙(Principles) 을 통해 LLM(Large Language Models)의 프롬프트 최적화 전략 제시
- LLaMA-1/2 (7B, 13B, 70B), GPT-3.5, GPT-4에 대한 광범위한 실험을 통해 효과 검증
- LLM을 활용하여 더 나은 질문을 구성하고, 보다 효과적인 답변을 받을 수 있도록 돕고자 한다.
- https://github.com/VILA-Lab/ATLAS (프로젝트 페이지)

Introduction

- 프롬프트 엔지니어링: LLM과 효과적으로 상호작용하기 위한 기술
- LLM
  - 질문 응답, 수학적 추론, 코드 생성 등 다양한 작업 수행 가능
  - 그러나 최적의 프롬프트, 지시를 설계하는 건 어려움
  - 직접 파인튜닝하는 것은 비용과 시간 문제로 인해 비효율적

    ⇒ ⭐ 프롬프트 최적화 = 중요 연구 분야

    - 구체적 업무 지시, 제공 예시의 신중한 선택 필요 - 해당 연구에서 최적화 전략을 제시하고자 함
- 연구 소개
    - 다양한 프롬프트 구성 시도를 통해 LLM이 최적의 응답을 생성하도록 하는 최적의 프롬프트 설계 방법 제시 (ex. LLM에게 역할 지시 등)
    - 모델 크기가 클수록 프롬프트 개선에 따른 성능 개선 효과가 큼

Related Work

- 대형 언어 모델 (LLMs)
    - Google의 BERT: 문맥 이해 강화 (양방향 학습)
    - GPT-1 ~ GPT-4: 파라미터 증가 → 언어 생성 능력 발전 (transformer → 비지도 학습)
    - Meta의 LLaMA: 더 적은 리소스로 강력한 성능 발휘
    - Chinchilla: 작은 모델의 효과적 학습 증명
    - Mistral: 효율성과 성능을 모두 향상
- 프롬프트 엔지니어링
    - 프롬프트의 문맥적 뉘앙스 차이에 따른 답변 차이 등
    - 프롬프트 설계 발전 방향
    - Few-shot learning: 예제를 통한 학습
    - Ask-Me-Anything: 다수의 불완전한 프롬프트 → 통합 (질문-답변 형식)
    - Chain-of-Thought(CoT): 복잡한 문제를 단계적으로 해결하도록 유도
    - least-to-most prompting: 복잡한 문제 → 간단한 subproblem으로 분해

      ⇒ 자세한 설명이 복잡한 문제에 대한 해결 능력 향상
    - Directional Stimulus Prompting: 특정 목표 방향으로 출력을 유도하는 방법

Principles

3.1 Motivation

- LLM의 출력 품질은 프롬프트의 품질에 직접적으로 영향을 받음 ⇒ 좋은 프롬프트 설계의 중요성
- 프롬프트는 단순한 질문이 아니라, LLM과 상호작용을 설계하는 프로그래밍 도구
- 최적의 프롬프트를 설계하기 위한 구체적인 원칙을 제시하여 출력 품질을 향상시키고자 함

3.2 Overview

- **26가지 프롬프트 원칙**을 5가지 카테고리로 분류

카테고리	설명
프롬프트 구조 및 명확성	청중 설정, 명확한 구문 사용, 부정어 대신 긍정어 사용
구체성 및 정보 제공	편향 없는 답변 요구, 추가적인 맥락 제공
사용자 상호작용 및 참여 유도	모델이 사용자에게 추가 질문하도록 유도
문맥 및 언어 스타일	모델에게 특정 역할 부여, 특정 문체 유지
복잡한 작업 및 코드 프롬프트	복잡한 문제를 세부 단계로 나누기

💡 26가지 원칙 정리

1. 정중한 표현 생략 (ex. "please", "thank you" 등 포함 X)
2. 청중을 명확히 설정 (ex. 청중은 해당 분야의 전문가)
3. 복잡한 작업을 작은 단계로 분할 → 상호작용적인 프롬프트 구성
4. 긍정적 지시문 사용 (don't X, do O)
5. 분명하고 깊은 설명 요청 시 직관적 설명 요청 (ex. 초보자를 위해~, 5살 아이에게 설명하듯~)
6. 보상 시스템 추가 (ex. tip $XX를 줄게)
7. Few-shot prompting - 예제 추가
8. 프롬프트 포맷 설정 (###Instruction### 다른 줄에 ###Question### 등)
9. 명확한 작업 지시문, 강조 (Your task is ~, You MUST 등)
10. 페널티, 벌칙 조건 포함
11. 자연스러운 답변 유도 (Answer naturally, 인간 같이 등)
12. "step by step" 사고 유도
13. 편향 방지 요청 포함
14. 모델이 질문하도록 유도 (ex. 충분한 정보를 모을 때까지 질문해줘)
15. 이해도 테스트 포함 - 사용자가 학습할 수 있도록 퀴즈 요청 (ex. 개념 설명 후 마지막에 퀴즈 포함시켜줘)
16. 모델에게 역할 부여
17. 구분자 사용 (### 등으로 중요한 부분 강조)
18. 특정 단어 또는 문구 반복 - 모델이 해당 단어에 집중하도록
19. Chain-of-Thought(CoT) + Few-Shot - 논리적 사고 + 예제 결합
20. 출력 예측 설정 (Output Priming) - 기대하는 답변 시작 형식을 프롬프트 끝에 일부 제공
21. 세부적인 글 형태 지정 (ex. essay, article 등)
22. 텍스트 교정 요청 시 기존 형식 유지 부탁 (원래 글 형식을 유지하도록 요청)

23. 여러 개 파일 바탕으로 한 코딩 프롬프트 - 자동 생성 스크립트 요청 (ex. 자동으로 파일을 생성하는 스크립트를 [언어] 형식으로 함께 만들어줘)
24. 특정 문장으로 시작하도록 요청
25. 모델의 응답 규칙 명확히 지정 - 키워드, 힌트, 지시사항 등
26. 샘플 텍스트 기반 응답 요청

✔️ 카테고리 분류

카테고리	원칙
프롬프트 구조 및 명확성	2. 청중 설정 4. 긍정적 지시문 (부정적 표현 X) 12. “step by step” 와 같은 지시문 20. 출력 시작 부분 (출력 예측) 제시 (Output Priming) 17. 구분자 사용 8. 프롬프트 포맷 설정
구체성 및 정보 제공	7. Few-shot prompting - 예제 추가 5. 직관적, 쉬운 설명 요청 (대상을 초보자로 설정) 13. 편향 방지 요청 26. 샘플 텍스트 기반 응답 요청 24. 특정 문장으로 시작하도록 요청 25. 모델의 응답 규칙 명확히 지정 15. 이해도 테스트 요청 21. 세부적인 글 형태 지정
상호작용 및 참여 유도	14. 모델이 질문하도록 유도 21. 세부적인 글 형태 지정
문맥 및 언어 스타일	22. 텍스트 교정 요청 시 기존 형식 유지 부탁 9. 명확한 작업 지시문, 강조 (너의 업무는 ~, 무조건 ~) 10. 페널티, 벌칙 조건 포함 16. 모델에게 역할 부여 11. 자연스러운 답변 유도 (사람같이 ~) 1. 정중한 표현 생략 18. 특정 단어 또는 문구 반복 6. 보상 시스템 추가
복잡한 작업 및 코드 프롬프트	3. 복잡한 작업을 작은 단계로 분할 23. 복잡한 코딩 작업 요청 - 자동 생성 스크립트 요청 19. Chain-of-Thought(CoT) + Few-Shot

3.3 Design Principles

- Conciseness and Clarity (간결성 및 명확성)
  - 지나치게 긴 프롬프트는 모델을 혼란스럽게 할 가능성
  - **핵심 정보만** 제공 + **명확하게 작성**해야 함
- Contextual Relevance (관련 맥락 제공)
  - 필요한 **추가 배경 지식 제공하기**
  - 키워드, 도메인 용어, 상황 설명 등 제공
- Task Alignment (작업과의 정렬)
  - 질문의 형식을 **작업에 맞게 조정**
  - 질문, 명령, 빈칸 채우기와 같은 형식 선택
- Avoiding Bias (편향 방지)
  - 모델에 **내제한 편향 방지하도록** 설계
  - 중립적 언어 사용, 민감한 주제 주의
- Incremental Prompting (단계적 프롬프트)
  - 단계 필요한 작업은 **세부 단계로 나눠서** 해결
  - 모델의 성능과 반복적 피드백에 따라 **조정 필요** (초기 출력에 따라 개선 가능해야 함)
  - 고급 프롬프트 - 프로그래밍과 같은 로직을 통합하여 복잡한 작업 수행 (ex. 조건문, 논리 연산자, 의사 코드 등)

Experiments

4.1 Setup and Implementation Details

- ATLAS 벤치마크 활용 → 프롬프트 원칙 적용 여부에 따른 성능 비교 (각 원칙 당 20개 질문)

4.2 Models and Metrics

- 실험 대상: LLaMA-1/2 (7B, 13B, 70B), GPT-3.5, GPT-4

- 평가 방식
  - Boosting - 단순한 업무로 테스트 / 제안된 원칙 적용 시 생성 답변의 성능 향상 비율

    ⇒ 응답 품질 향상 효과 분석
  - Correctness - 복잡한 업무로 테스트 / 모델 출력의 정확성 (기대하는 방향과 맞는지 여부)

    ⇒ 정확성 향상 효과 분석

4.3 Results

4.3.1 Results on small, medium and large-scale LLMs

- 모델 크기에 따른 성능 비교
- Boosting (응답 품질 향상 효과 분석)
  - 원칙 적용 시 모든 모델에서 응답 품질이 유의미하게 향상
- Correctness (정확성 분석)
  - 절대 정확도 (Absolute Accuracy)
    - 원칙 적용 시 모델의 정확성이 평균적으로 20%~40% 향상
    - 중소형 모델 - 10~40% / 대형 모델 - 40% 이상
  - 상대 정확도 향상 (Relative Accuracy)
    - 원칙 적용 시 모든 모델에서 10% 이상 향상
    - 대형 모델에서 20% 이상

4.3.2 Results on Individual LLMs

- 각 LLM 모델에서의 결과
- Boosting (응답 품질 향상 분석)
  - 전반적으로 50% 이상의 향상률 (전반적으로 비슷한 듯)
  - 특히 특정 원칙에서의 응답 품질 유의미하게 향상 (14번 원칙, 26번 원칙 등)

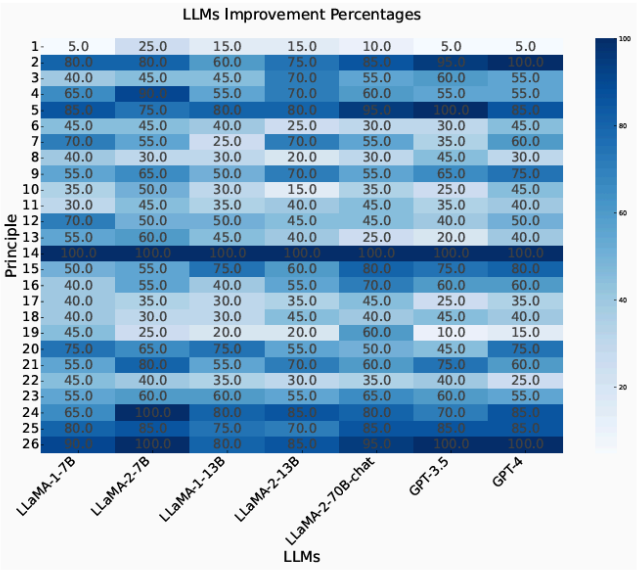

Figure 10: Illustration of heatmap for LLMs boosting percentages.

- Correctness (정확성 분석)
  - **... < LLaMA-2-13B < LLaMA-2-70B-chat < GPT-3.5 < GPT-4** 순으로 **절대 정확도 증가**
  - 모델이 클수록 정확성 향상도 큼

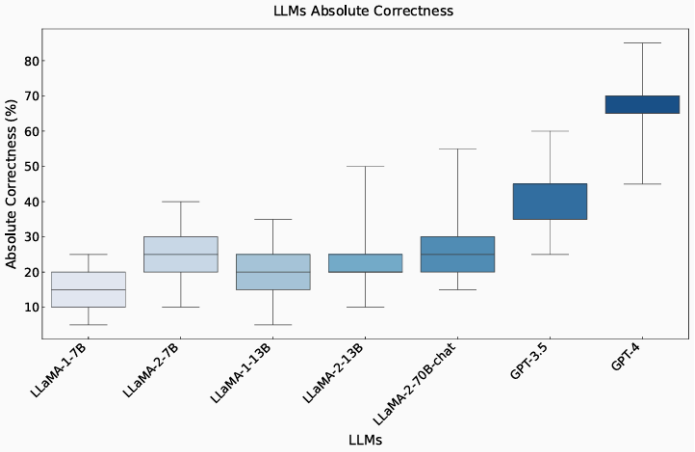

Figure 8: Absolute correctness score on the ATLAS dataset.

4.3.3 More examples on various scales of LLMs

- small-scale LLaMA-2-7B (Fig 13-14)
  - 프롬프트에서 예시 제공 후 답변 correctness 향상 예시 (Principle 7 Few-shot learning - 단어 개수 세기)

    - 프롬프트에서 힌트 제공 후 답변 correctness 향상 예시 (Principle 25 Requirement?)
- medium-scale LLaMA-2-13B (Fig 15-16)
    - 프롬프트에서 단계별 해결 지시 후 correcteness 향상 예시 (Principle 12 Think step by step 원칙)
    - 프롬프트에서 방향 제공 후 답변 correctness 향상 예시 (Principle 25 Requirement?)

Conclusion

- 결론
    - 실험 결과, 프롬프트 원칙을 적용하면 LLM의 응답 품질(Boosting)과 정확성(Correctness)이 크게 향상됨.
    - 특히, 대형 모델에서 가장 큰 성능 향상
- 향후 연구 방향성
    - fine-tuning, reinforcement learning, direct preference optimization 등을 통한 개선 시도 가능
    - 기존 LLM operation과 통합 가능

Limitations and Discussion

- 복잡하거나 고도로 전문적인 질문에서는 이 원칙들의 효과가 감소할 가능성
- 개별 모델의 추론 능력(reasoning capabilities) 및 사전 훈련(training data)에 크게 의존하기 때문
- 한계
    - 특정 LLM 모델에서 성능이 달라질 수도
    - 질문 범위의 한계 - 더 다양한 유형의 질문 시도 필요
    - 평가 기준의 차이 - 더 객관적인 자동 평가 방법 개발이 필요