복습과제 : Chain-of Thought Prompting Elicits Reasoning in Large Language Models 논문 리뷰

▼ 1. Introduction

- language model은 사이즈를 높이는 것만으로도 성능이 훨씬 좋아지고 효율적
- BUT 사이즈만 높이는 것의 한계 : 산술, 상식, 상징적 추론(기호 → 논리적 결론 이끌어냄) 과 같은 분야에서는 성능이 떨어졌음
- 두가지 기존 접근 방식
    - 설명 기반 training: 모델이 알아서 자연어로 중간 단계를 생성하거나, pre trained model을 fine tuing하는 방식, 또는 자연어가 아닌 형식 언어를 사용하는 neuro-symbolic한 방법등...
        - 한계 : 중간 과정을 보여주는 설명(rationale) 데이터셋을 만드는 게 너무 복잡하고 비용이 큼
    - 전통적 few-shot prompting: 예시 몇가지를 주는 방식
        - 한계 : 추론 능력이 필요한 작업에서는 모델 규모가 커져도 성능 안좋음

→ 이 두가지 장점만 결합해서 Chain of thought를 생각. chain of thought란 중간 자연어 추론 단계를 의미한다. 즉 ⟨input, chain of thought, output⟩를 few shot prompting에 포함시키자.

- 결과

산술, 상식, 상징적 추론 등의 벤치마크에서 매우 좋은 성과.

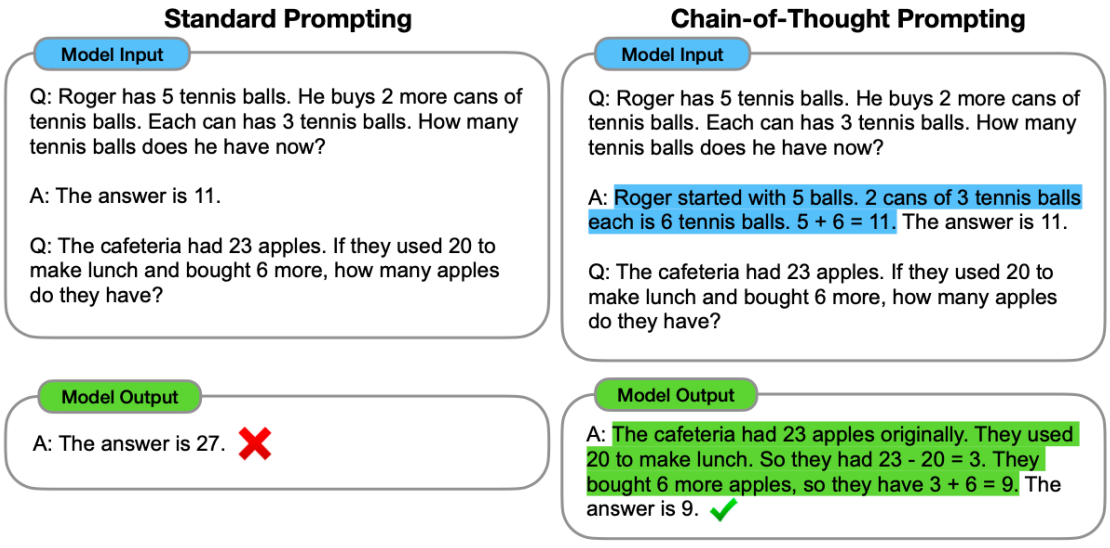
Ex) 수학 문제 해결 벤치마크인 GSM8K에서 PaLM 540B 모델에 CoT를 적용했을 때, 기존 최고 성능을 뛰어넘는 결과

의의 : 대규모 훈련 데이터셋도 없이, 간단한 프롬프팅 만으로도 generality를 유지하며 다양한 작업을 해낼 수 있음

▼ 2. Chain-of-Thought Prompting

> 논문의 목표 : **언어 모델이 Chain-of-Thought을 스스로 만들어내게 하는 것. 즉, 문제를 풀어가는 중간 reasoning(추론) 단계를 일관되게 나열해서 최종 답에 도달하도록 하는 것!**

→ 실험에 따르면 충분히 큰 언어 모델은 few-shot prompting(예시를 보여주며 유도하는 기법) 안에 Chain-of-Thought의 예시를 넣어주면, 이런 사고 과정을 스스로 생성할 수 있더라.



왜 chain of thought인가? : solution이라고 볼수도 있지만, 답을 내기 전에 인간처럼 최종 답을 도출하기 위한 사고 단계를 흉내내는 것이기 때문.

CoT의 장점

1. **문제를 작은 단계들로 쪼개서 풀 수 있다.** 더 많은 reasoning 단계가 필요한 문제에는 더 많은 계산 자원을 쓸 수 있다.
2. **모델의 해석이 가능해진다. 정답만 나오는게 아니라, 중간 추론 과정을 볼 수 있기에 디버깅도 가능.** 물론 모델의 실제 내부 계산 전체를 완전히 규명하는 것은 아직 풀리지 않은 과제다.
3. **수학, 상식,** symbolic manipulation(기호를 규칙에 맞게 계산, 전개, 단순화 하는 등의 작업) 등 **다양한 과제에 활용 가능하다.** 원칙적으로는 사람들이 언어를 통해 풀 수 있는 모든 문제에 적용 가능하다.
4. **충분히 큰 LLM이라면, 쉽게 유도할 수 있다.**

▼ 3. Arithmetic Reasoning

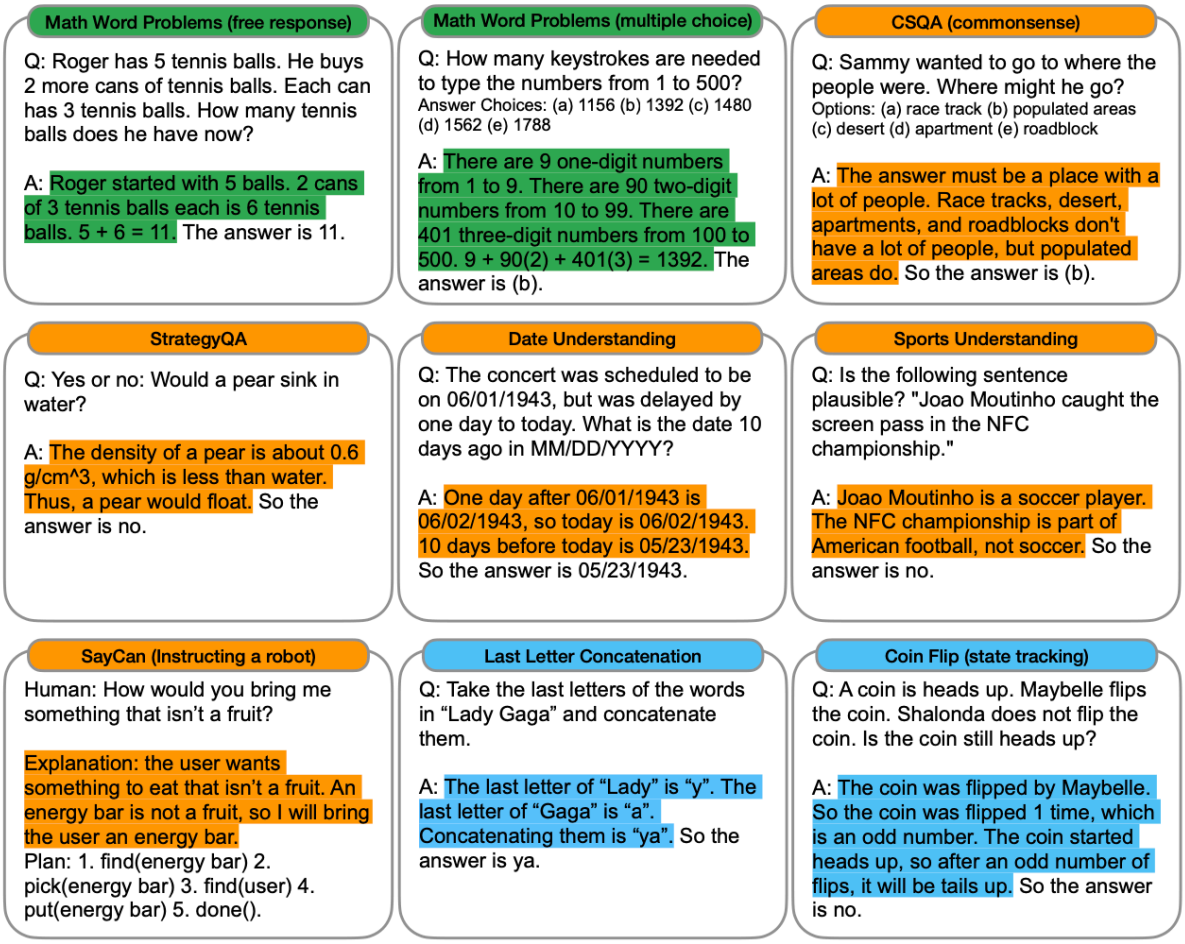
수학 서술형 문제를 통해 언어 모델의 **산술적 추론 능력**을 평가해보자.

1) Experimental Setup

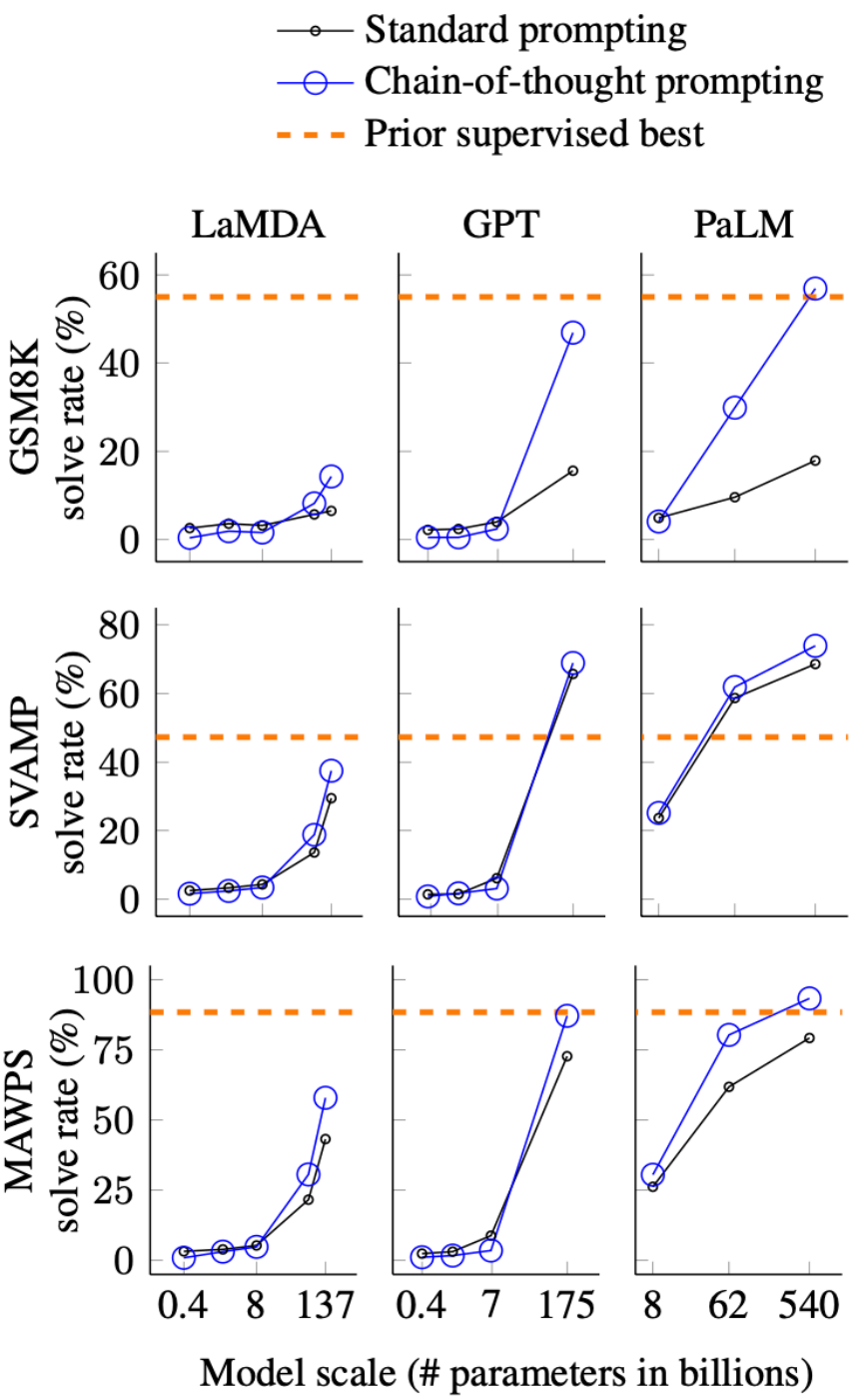
- 사용한 다섯가지 수학 서술형 문제 벤치마크(Benchmarks) : GSM8K , SVAMP (구조가 다양), ASDiv (문제 유형이 다양), AQuA (대수학, 객관식), MAWPS
- 두가지 prompting 방식 비교

Standard prompting	**Chain-of-Thought prompting**
표준화된 few-shot prompting(Brown et al. (2020)) 이용. 문제-정답 쌍 예시를 몇개 보여주고, 새 문제를 주면 모델이 바로 답을 출력하는 방식	few-shot에서 각 예시들에 정답 뿐 아니라 CoT를 포함시키는 방식.
	대부분의 데이터셋은 평가용 split만 있기 때문에, 저자들이 직접 **8개의 Chain-of-Thought 예시**를 만들어 모든 벤치마크에서 동일하게 사용. - 예시들은 특별한 prompt engineering 없이 만들어짐 - AQuA(객관식)만은 4개의 예시와 training set의 정답을 사용

- 사용한 language models : GPT-3, LaMDA, PaLM, UL2 20B, COdex
  - 모델 출력 : greedy decoding으로 샘플링
    - greedy decoding : 매번 가장 확률이 높은 단어를 바로 선택하는 방식
    - 후속 연구에 따르면, 여러 답을 생성하고 다수결을 취하는 방식이 성능을 더 높일 수 있음
  - LaMDA의 경우, 예시 순서를 무작위로 섞은 5개의 seed에 대해 평균 성능을 보고함. 여러번 돌려도 시드 간 변동이 크지 않았기 때문에, 나머지 모델들은 계산 자원을 절약하기 위해 단일 시드 결과만 보고함.



2) Results



행 : 벤치마크, 열 : 사용한 언어 모델.

CoT의 solve rate이 훨씬 높은 것을 확인할 수 있으며, 전반적으로 모델 크기가 커질수록 차이가 많이 난다.

GSM8K라는 벤치마크에서 SOTA 성능을 보이고 있다.

구체적인 결과 :

1. CoT는 작은 모델에서는 유창하게 말하나 논리가 이상한 문제가 있었다.

   그러나 모델이 클수록 (100B 파라미터 이상의 거대모델일수록) 효과가 갑자기 크게 나타난다.

2. 쉬운문제보다 어렵고 복잡한 문제를 풀 때 CoT의 성능이 2배 이상 좋아졌다.

3. 초대형 모델(PaLM 540B)에 CoT를 쓸 시, fine tuning된 모델 이상의 성능을 보임. 즉, **데이터셋별 맞춤 학습 없이도 CoT만 잘 쓰면 최신 성능에 도달 가능했다.**

4. LaMDA 137B에서 틀린 문제를 분석한 결과 계산이나 기호 매핑실수 같은 작은 실수와 개념, 의미 이해를 못하는 큰 실수가 각각 46%, 54%로 나타났음.

   모델이 클수록 이러한 CoT에서 생기는 오류 유형이 개선되었다.

   - PaLM62B → PaLM540B 스케일링시 62B 모델의 한 단계 누락 및 의미 이해 오류의 대부분이 수정됨.

3) **Ablation Study**

모델의 성능에 가장 큰 영향을 미치는 요소를 찾기 위해 모델의 구성요소 및 feature들을 단계적으로 제거 하거나 변경해가며 성능의 변화를 관찰해보자.

1. equation only : 중간 추론 과정 없이 수식만 쓰게 하자.
   - 결과 : GSM8K같은 어려운 문제에는 거의 효과가 없었다. (자연어 추론 단계를 거치지 않으면 문제의 의미를 제대로 수식으로 옮기기 어렵기 때문)
   - 1-2단계로 풀수 있는 간단한 문제는 성능이 향상됐다.
2. variable compute only : 모델이 답하기 전에 ... 을 출력하게 하자.
   - 단순히 계산을 오래해서 CoT 성능이 좋은 건지 확인하기 위함
   - 점 개수는 필요한 수식 길이에 맞춰 조정
   - 결과 : 기본 prompting과 비슷하게 효과가 없었다.
3. CoT after answer : 답하고 난 뒤에 CoT를 붙이게 하자.
   - 모델이 기억한 지식을 꺼내기 쉬워서 CoT 성능이 좋은 건지 확인하기 위함

- 결과 : 마찬가지로 기본 prompting과 비슷하게 효과가 없었다.

> 결론 : CoT의 효과는 단순히 계산 많이 해서나 기억하고 있던 지식을 꺼내기 쉬워서가 아니라, 실제로 답을 내는 과정에서 자연어로 중간 reasoning을 순차적으로 적는 과정이 있었기에 나타났다.

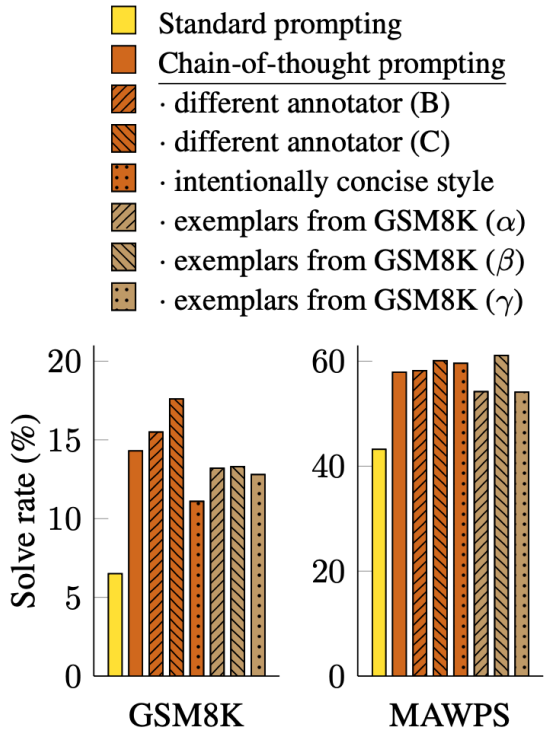
4) Robustness of Chain of Thought

- Robustness : 견고성. 모델이 노이즈에 크게 영향을 받지 않는 정도.
- 기존 baseline prompting : few shot 예시들을 뭘로 주느냐에 따라 성능이 크게 달라진다. (robust ⬇️)
  - GPT-3가 SST-2(감정분류 데이터셋)에서 예시 순서를 어떻게 섞느냐에 따라 정확도가 54 ~ 93%까지 왔다 갔다 한다는 연구 (Zhao et al., 2021).
- CoT prompting의 robustness 실험
  1. Annotator 실험 : 논문 저자 A,B,C가 같은 문제에 대해 서로 다른 CoT 예시 작성. A는 추가적으로 간결한 버전의 CoT도 작성.

     → 결과 : 사람이 다 다르더라도 모두 baseline보다 성능이 좋았음.
  2. Exemplar 다양성 실험 : GSM8K 데이터셋에서 무작위로 뽑은 8개의 예시 세트 사용해서 CoT prompting.

     → 결과 : 사람이 직접 작성한 예시와 비슷하게 잘 작동함.
  3. 이외에 예시 순서와 개수에 따라서도 robust했다.

  ✅ 특정 언어적 스타일, 예시의 순서, 개수 등이 달라도 성능은 여전히 robust함을 입증.



→ standard prompting (노란색) 이 가장 성능이 낮고, 저자 A,B,C가 쓴 예시들과 A가 쓴 간결한 예시, GSM8K에서 랜덤으로 뽑은 예시들은 세부 성능 차이는 있으나 모두 baseline보다 월등히 좋은 성능을 보여준다.

> 결론 : 누가 쓰든, 어떻게 쓰든, 심지어 데이터셋에서 뽑은 예시를 쓰든, CoT는 baseline보다 훨씬 강력하다.

▼ 4. Commonsense Reasoning

CoT는 일반적인 배경지식을 전제로 하는, 더 넓은 범주의 상식 추론 문제에도 적용 가능!

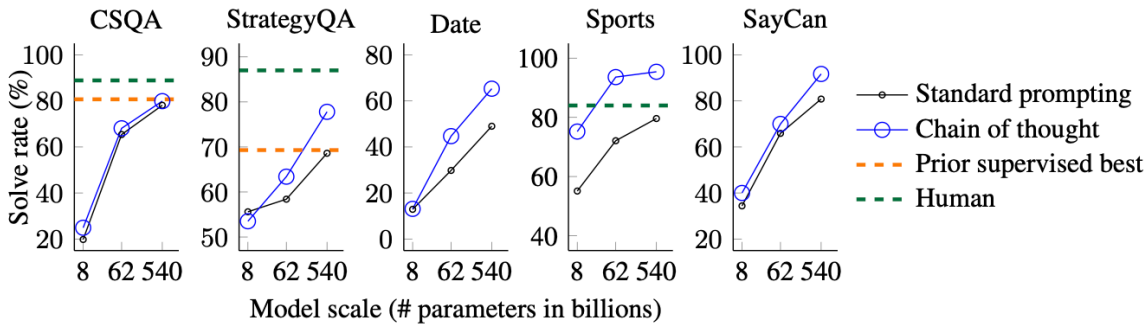
Experimental Setup

- 사용한 다섯가지 수학 서술형 문제 벤치마크(Benchmarks) : CSQA , StrategyQA, Big-bench 평가세트 - Date Understanding(주어진 문맥으로부터 날짜를 추론하는 문제), Sports Understanding (스포츠와 관련된 문장이 그럴듯한지 판별하는 문제), SayCan (자연어 지시 → 로봇 행동 시퀀스 매핑 문제)
- 두가지 prompting 방식 비교 : Arithmetic reasoning과 동일
    - CSQA와 StrategyQA : 데이터셋에서 예시 무작위 선택, 수작업으로 CoT 작성

- BIG-bench : 학습 데이터셋이 없어 처음 10개 예시를 few-shot 예시로 사용 / 나머지 평가 세트에 대해 결과 보고
- SayCan : Ahn et al. (2022)에서 사용된 학습 데이터셋에서 6개의 예시를 사용, 수작업으로 CoT 작성

Results

- 모든 문제에서 모델 크기 클수록 baseline도 성능 향상, CoT는 더욱 향상
  - 특히 PaLM 540B에서 효과가 가장 컸다.
- StrategyQA: 기존 SOTA가 69.4%였는데, PaLM 540B + CoT는 75.6% 달성
- Sports Understanding: 스포츠 애호가의 평균 정확도가 84%였으나 PaLM 540B + CoT에서는 95.4% → 사람보다 성능이 좋음
- CSQA : 성능 향상 미미했음



PaLM 결과. **CoT가 표준 prompting보다 성능이 좋고, 모델 크기가 커질수록 성능 향상되는 것 확인 가능**

Table 4: Standard prompting versus chain of thought prompting on five commonsense reasoning benchmarks. Chain of thought prompting is an emergent ability of model scale—it does not positively impact performance until used with a model of sufficient scale.

		CSQA		StrategyQA		Date		Sports		SayCan	
Model		standard	CoT	standard	CoT	standard	CoT	standard	CoT	standard	CoT
UL2	20B	34.2	**51.4**	59.0	53.3	13.5	**14.0**	57.9	**65.3**	20.0	**41.7**
LaMDA	420M	20.1	19.2	46.4	24.9	1.9	1.6	50.0	49.7	7.5	7.5
	2B	20.2	19.6	52.6	45.2	8.0	6.8	49.3	57.5	8.3	8.3
	8B	19.0	20.3	54.1	46.8	9.5	5.4	50.0	52.1	28.3	33.3
	68B	37.0	**44.1**	59.6	**62.2**	15.5	**18.6**	55.2	**77.5**	35.0	**42.5**
	137B	53.6	**57.9**	62.4	**65.4**	21.5	**26.8**	59.5	**85.8**	43.3	**46.6**
GPT	350M	14.7	15.2	20.6	0.9	4.3	0.9	33.8	41.6	12.5	0.8
	1.3B	12.0	19.2	45.8	35.7	4.0	1.4	0.0	26.9	20.8	9.2
	6.7B	19.0	**24.0**	53.6	50.0	8.9	4.9	0.0	4.4	17.5	**35.0**
	175B	79.5	73.5	65.9	65.4	43.8	**52.1**	69.6	**82.4**	81.7	**87.5**
Codex	-	82.3	77.9	67.1	**73.2**	49.0	**64.8**	71.7	**98.5**	85.8	**88.3**
PaLM	8B	19.8	**24.9**	55.6	53.5	12.9	13.1	55.1	**75.2**	34.2	**40.0**
	62B	65.4	**68.1**	58.4	**63.4**	29.8	**44.7**	72.1	**93.6**	65.8	**70.0**
	540B	78.1	**79.9**	68.6	**77.8**	49.0	**65.3**	80.5	**95.4**	80.8	**91.7**

전체 모델 결과

▼ 5. Symbolic Reasoning

Tasks

- 마지막 글자 이어붙이기 : 이름 속 단어들의 마지막 글자를 이어 붙이기

  ex) “Amy Brown” → “yn”

  - CoT 없이도 첫글자 이어붙이기는 할 수 있으나, 더 어려운 과제
  - 전체 이름 : name census 데이터에서 상위 1000개의 이름과 성을 무작위로 결합하여 생성

- 동전 뒤집기 : 사람들이 동전을 뒤집거나 뒤집지 않았을 때, 동전이 여전히 앞면인지 아닌지를 묻기

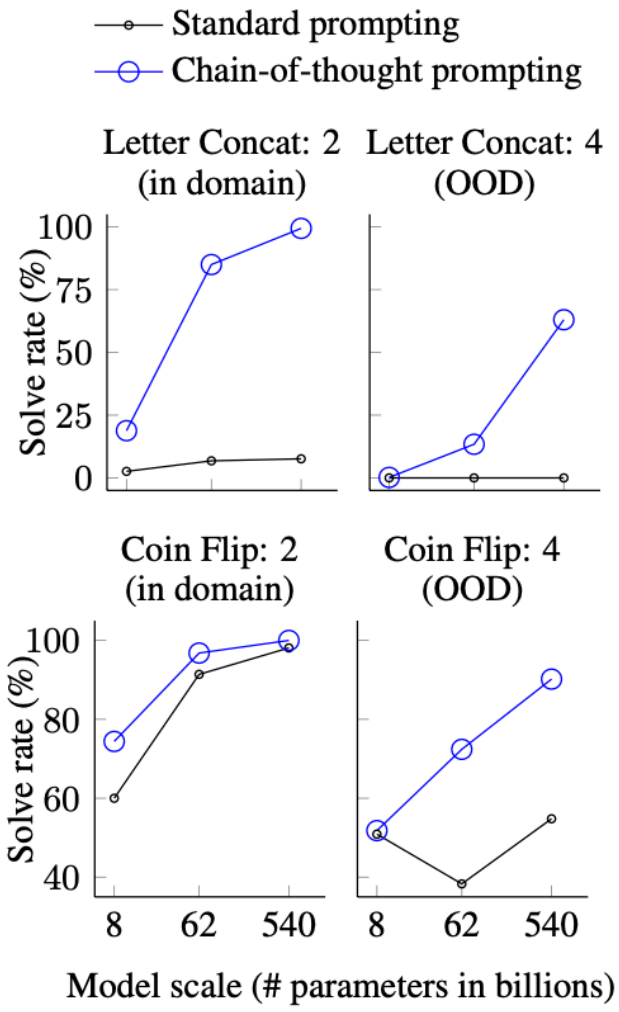
  ex) “A coin is heads up. Phoebe flips the coin. Osvaldo does not flip the coin. Is the coin still heads up?” → “no”

test set

각 과제에 대해 두가지 test set 고려.

1. In-domain: 훈련/예제로 본 것과 동일한 단계 수를 가진 사례
2. Out-of-domain (OOD): 훈련 예제보다 더 많은 단계가 필요한 사례
    - 마지막 글자 이어붙이기 : 2단어짜리 이름만 학습하다가 test에서 3~4 단어 이름 주기
    - 동전 뒤집기 : 일정 개수의 flip만 학습하다가 테스트에서는 더 많은 flip 단계 주기

Results



PaLM 결과

- PaLM 540B에서 CoT는 거의 정답률 100%에 도달

    - coin flip은 PaLM 540B에서 표준 prompting만으로도 이미 풀 수 있었지만, LaMDA 137B에서는 불가능했음
- in-domain : toy tasks (few shot 예제에 완벽한 CoT를 제시해놨고, 모델은 그냥 symbol을 이용해서 그 과정만 반복하면 되는 과제) 인데도, 작은 모델들은 실패함

  → 보지 못한 기호들에 대해 추상적 조작을 수행하는 능력은 최소 100B 이상의 파라미터 규모에서만 나타난다.
- OOD : 표준 prompting은 두 task 모두 실패. CoT 사용시 성능 증가하는 scaling curve 보임
    - in-domain보다는 성능 낮음

  → CoT가 few-shot 예제에서 본 것보다 더 긴 CoT 입력에 대해서도 length generalization 가능하더라.

▼ 6. Discussion

LLM에서 여러 단계의 추론을 이끌어내는 단순한 매커니즘으로서, CoT prompting을 탐구했다.

결과 해석

- arithmetic reasoning : 큰 성능 향상. ablation, annotator, exemplar, 다양한 모델들에 대해 robust하게 유지됨.
- commonsense reasoning : 일반적으로 적용 가능하게 됨
- symbolic reasoning : 더 긴 입력에 대해서 OOD generalization 가능하게 됨
- CoT는 scaling이 핵심! 모델이 클수록 성능 확연히 좋아진다.
- 표준 prompting이 성능 증가 곡선이 flat한데에 비해, CoT는 급격히 상승하는 scaling curve를 보임. 즉, LLM이 수행할 수 있는 task의 범위를 CoT가 확장해줄 수 있음. 즉, 표준 prompting은 LLM의 하한선만을 제공하는 것.

이후 탐구 가능한 질문

- 모델 규모가 더 커지면 추론 능력은 얼마나 더 개선될 수 있을까?
- 언어 모델이 풀 수 있는 과제의 범위를 확장시킬 다른 prompting 기법은 무엇일까?

한계점

1. 신경망이 실제로 추론하는 것인지는 미지수
2. few shot 에서 예제에 CoT를 수동으로 추가하는 건 쉽지만, fine tuning을 위해 대규모로 주석을 달면 비용이 매우 커지는 문제
    - 합성 데이터 생성, zero shot 일반화 등으로 극복 가능성
3. CoT가 항상 올바르진 않다. 잘못된 추론 과정을 거쳐 정답이나 오답 모두 나올 수 있음
    - fact기반 생성을 개선 필요
4. LLM에서만 CoT 추론이 가능하기에 실제로 응용하기엔 비용이 높음.
    - 소규모 모델에서 추론 유도하는 방법 연구 필요

▼ 8. Conclusions

- chain of thought : LLM이 복잡한 추론을 더 잘하게 하는 프롬프트 엔지니어링 기법. 단순히 질문과 답변 예시를 주고 새로운 질문에 답변하게하는 few shot이 아닌, 어떤 과정으로 답이 나오는지 를 알려주고 답변 예시를 주면, 더 정확한 답을 내더라.
- CoT는 단순하지만 광범위하게 적용 가능 → 언어 모델의 추론 능력을 크게 향상 시켰다.
- 3가지 LLM 모델에 실험해본 결과, 산술, 상식, 상징적 추론 3가지 분야에서 훨씬 뛰어난 성능 보임.
- 핵심 : CoT는 대규모 모델에서만 나타나는 ‘emergent property’. 즉, 충분히 모델이 커야만 CoT를 통해 flat curve를 급격한 scaling curve로 바꿀 수 있다.
- 향후 언어 기반 접근으로 더 다양한 추론 과제를 풀 수 있도록 연구 확장 가능성