CoT :
Chain-of-Thought

20190708

김승호

20240643 정윤



https://www.youtube.com/watch?v=mSoNhty4xEg



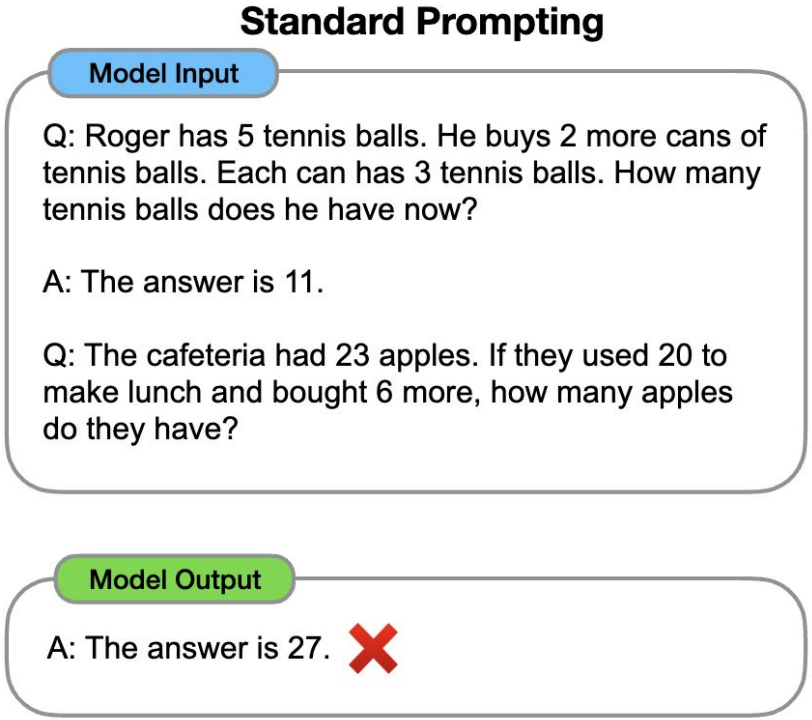
https://www.youtube.com/watch?v=mSoNhty4xEg

Appearance of CoT

LLM의 원리

거대 언어 모델(LLM)은 기본적으로
다음에 올 확률이 높은 단어를 예측

단순한 질문에는 강하지만, 수학 문제나
논리적 추론이 필요한 질문에서는
단번에 정답을 맞히기 어려워하는
경우가 많음



LLM’s Achilles tendon

LLM이 규모를 키우면서 많은 발전. but, 여전히 해결하지 못한 아킬레스건 존재

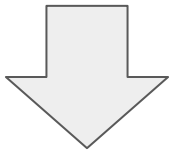
현상 : 모델이 커지면 성능과 샘플 효율성은 좋아지지만, 산술(Arithmetic), 상식(Commonsense), 기호 추론(Symbolic Reasoning) 같은 복잡한 추론 과제에서는 단순히 모델 크기만 키우는 것이 정답이 아님.

결론 : 모델 안에 지식은 많으나, 그 지식을 논리적으로 연결해 결론을 도출하는 추론 능력은 여전히 잠겨있는 상태

Motivation

첫째, Rationale(근거)의 힘: 인간은 수학 문제를 풀 때 자연어로 된 중간 과정을 거치면 훨씬 잘 품. 과거에는 이를 위해 모델을 새로 학습시켜야 해서 비용이 너무 컸음.

둘째, Few-shot Prompting의 발견: GPT-3처럼 예시만 몇 개 주면 모델이 새로운 작업을 수행할 수 있다는 점에 주목.



그렇다면, 예시에 **풀이 과정**을 포함해서 **프롬프팅(Few-shot)**을 하면 모델이 추론을 따라 하지 않을까?

Paper Review

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

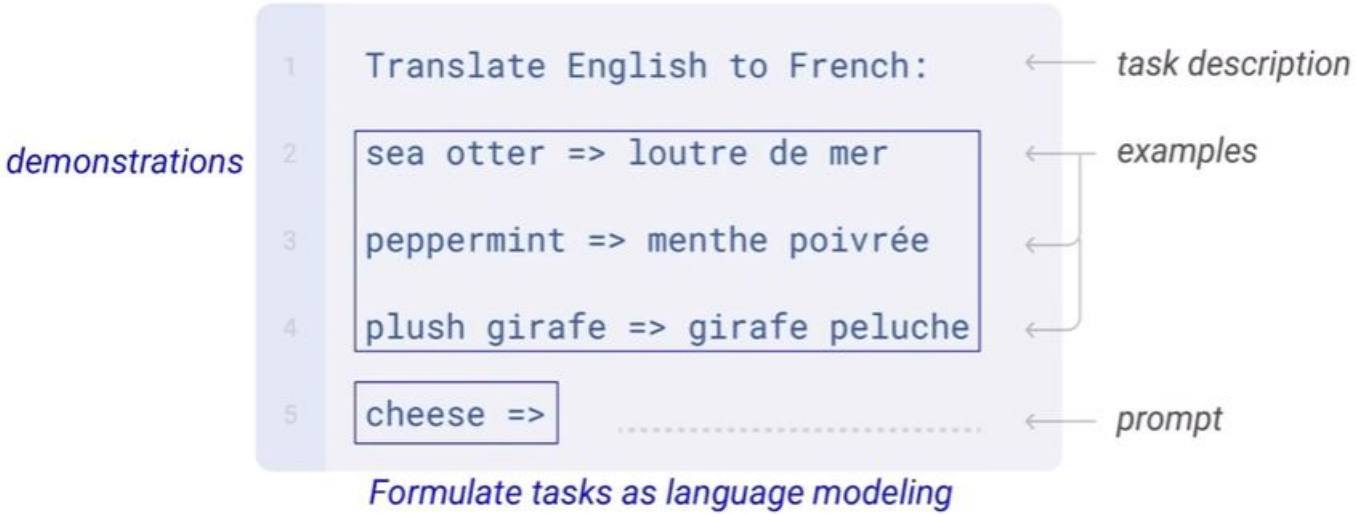
Jason Wei **Xuezhi Wang** **Dale Schuurmans** **Maarten Bosma**

Brian Ichter **Fei Xia** **Ed H. Chi** **Quoc V. Le** **Denny Zhou**

Google Research, Brain Team
{jasonwei,dennyzhou}@google.com

1. Introduction Few-Shot Prompting

Task description + Examples of demonstrations = No model update



2. Manual CoT Method

Chain of Thought

Standard Prompting

Model Input

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: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

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: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

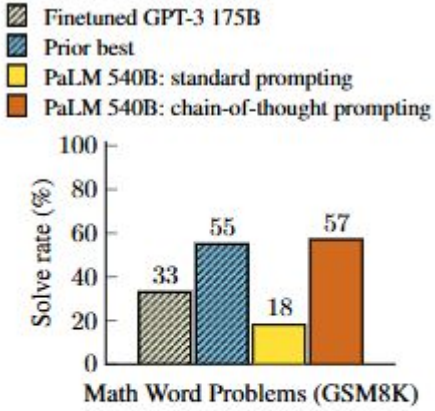


Figure 2: PaLM 540B uses chain-of-thought prompting to achieve new state-of-the-art performance on the GSM8K benchmark of math word problems. Finetuned GPT-3 and prior best are from Cobbe et al. (2021).

2. Manual CoT Experiment Setting

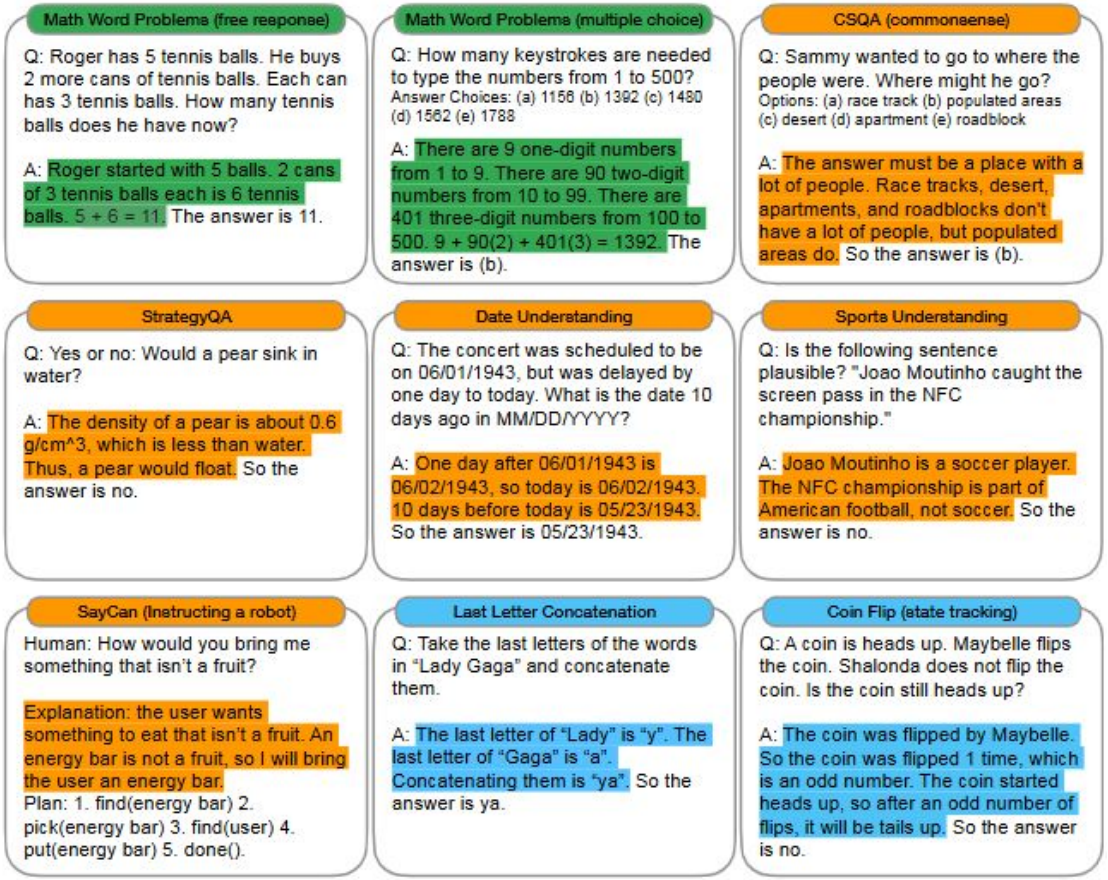
- Evaluation Benchmark
  - Arithmetic inference: math word problems benchmarks such as GSM8K, SVAMP, ASDiv, AQuA, MAWPS, etc.
  - Common sense inference: CSQA, StrategyQA, Date Understanding, Sports Understanding, SayCan (Robot Instruction Mapping) benchmark.
  - Symbolic inference: last letter concatenation, coin flip tracking, et al. toytask.

- Models used: GPT-3 (text-ada-001 to text-davinci-002), LaMDA (422M to 137B), PaLM (8B to 540B), UL220B, Codex.

2. Manual CoT Experiment Setting

- Dataset별 8개의 Manual CoT Prompt를 구축하고 실험 진행
- Manual하게 작성되었지만 CoT Prompt annotator에게 Specific instruction이 주어지지 않음
  - Final answer를 도출할 수 있도록 Step-by-step Reasoning Process를 적어달라고 단순 요청
- Prompting은 Few-shot setting(Standard Prompting)을 기본으로 함
- 3가지 Reasoning에 대해 Final Answer는 짧은 길이의 답변으로 구성됨
  - ex) So the answer is no
  - ex) So the answer is 03/27/2001
  - ex) The answer is (b)

2. Manual CoT Experiment Setting



2. Manual CoT Results

Findings 1) Emergent Ability by Model Size

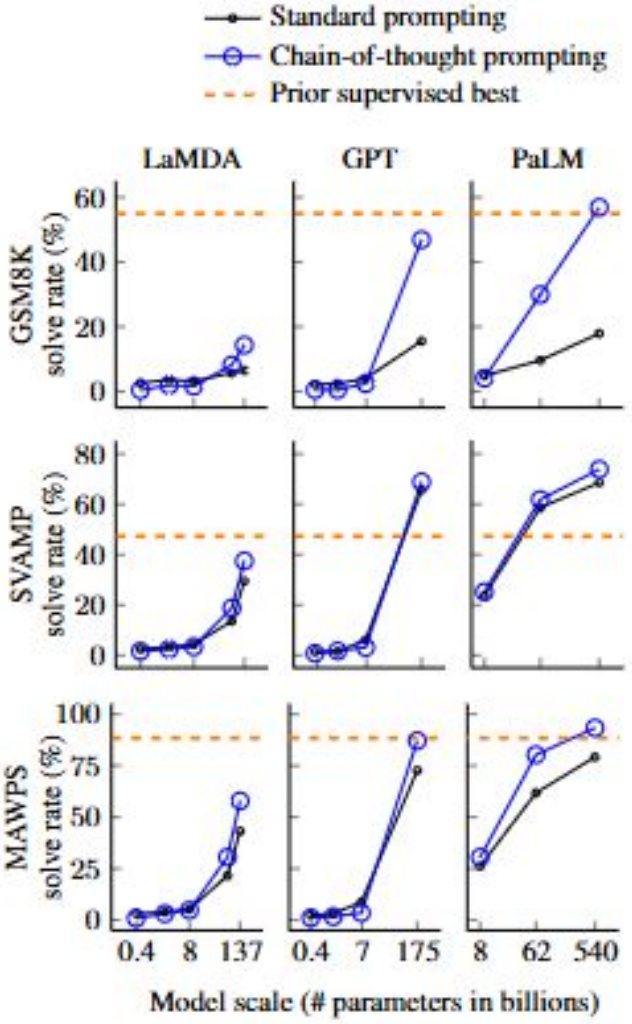
"Chain-of-thought prompting" did not or rather impeded performance improvement in small models, and only showed significant performance improvement in models with parameters above approximately 100B.

Findings 2) Significant performance improvements

For complex problems such as GSM8K, performance is more than doubled on the largest GPT and PaLM models.

Findings 3) The Importance of Natural Language Inference

Most "chain of thought" was also logically and mathematically accurate when the model gave the correct answer. Errors tended to improve with increasing model size.



2. Manual CoT Results

Findings 4) Robustness

Successful "chain of thought" prompting does not depend on a particular linguistic style. It was also solid in the order or number of examples.

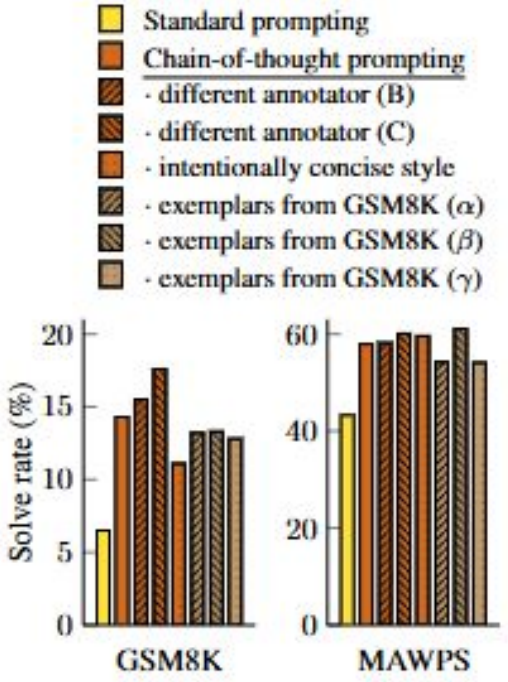


Figure 6: Chain-of-thought prompting has variance for different prompt examples (as expected) but outperforms standard prompting for various annotators as well as for different exemplars.

Large Language Models are Zero-Shot Reasoners

Takeshi Kojima
The University of Tokyo
t.kojima@weblab.t.u-tokyo.ac.jp

Shixiang Shane Gu
Google Research, Brain Team

Machel Reid
Google Research*

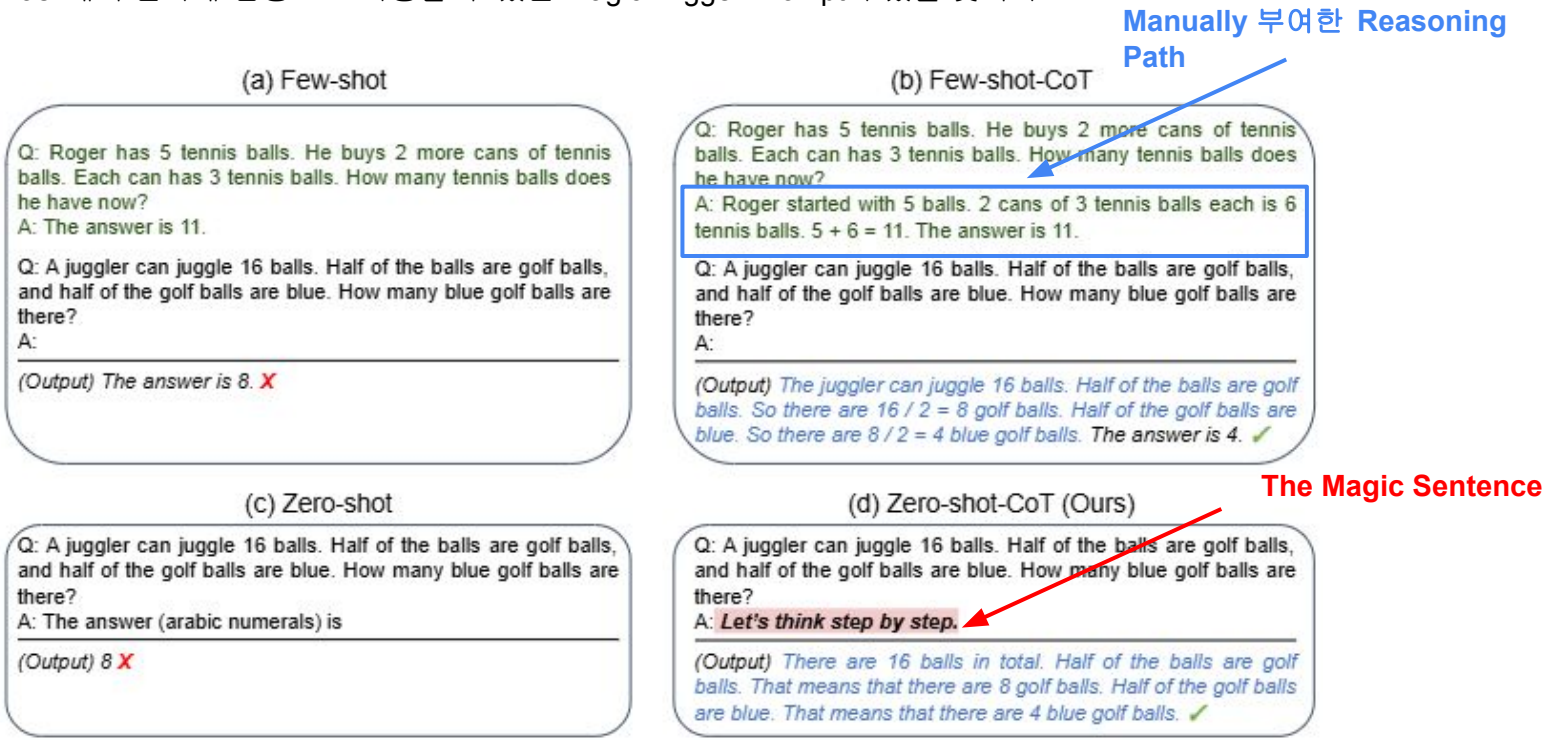
Yutaka Matsuo
The University of Tokyo

Yusuke Iwasawa
The University of Tokyo

3. Zero-Shot CoT Introduction

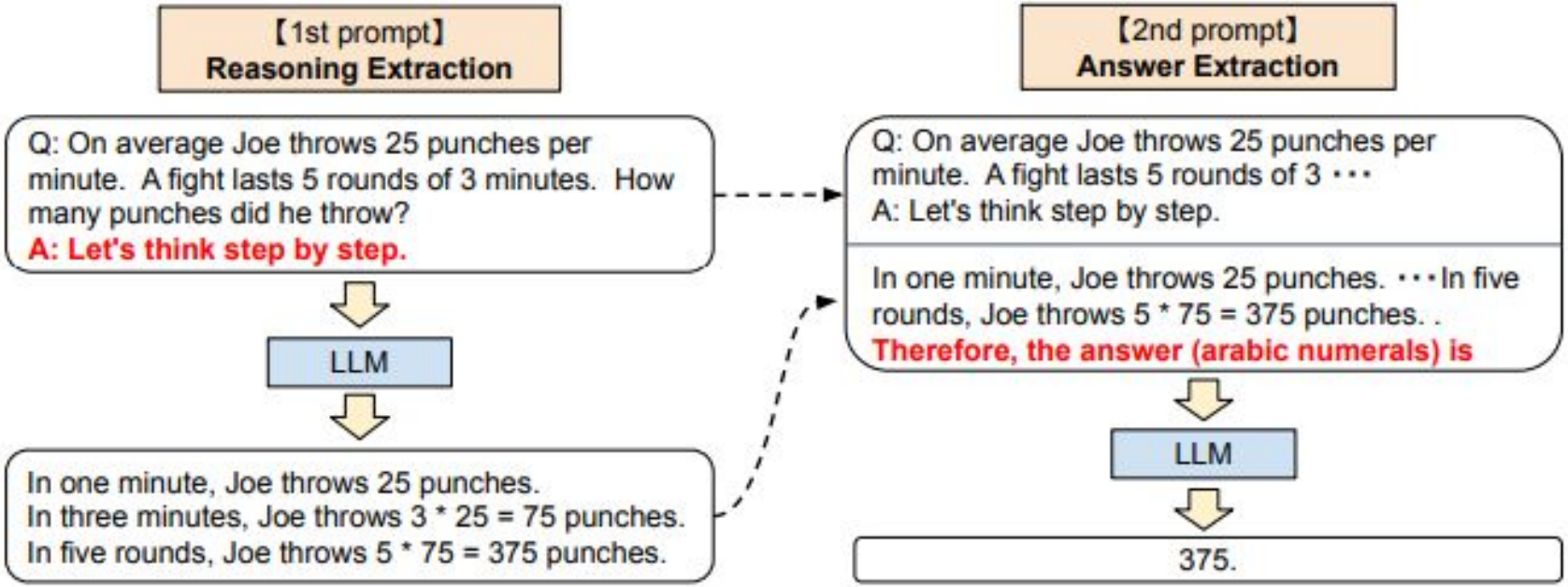
Remark) Few-shot CoT에서 manually task-specific CoT Prompt를 생성했는데, Simple Prompt의 효과를 보임

Conjecture) Task에 무관하게 범용으로 사용할 수 있는 Magic Trigger Prompt가 있을 것이다!

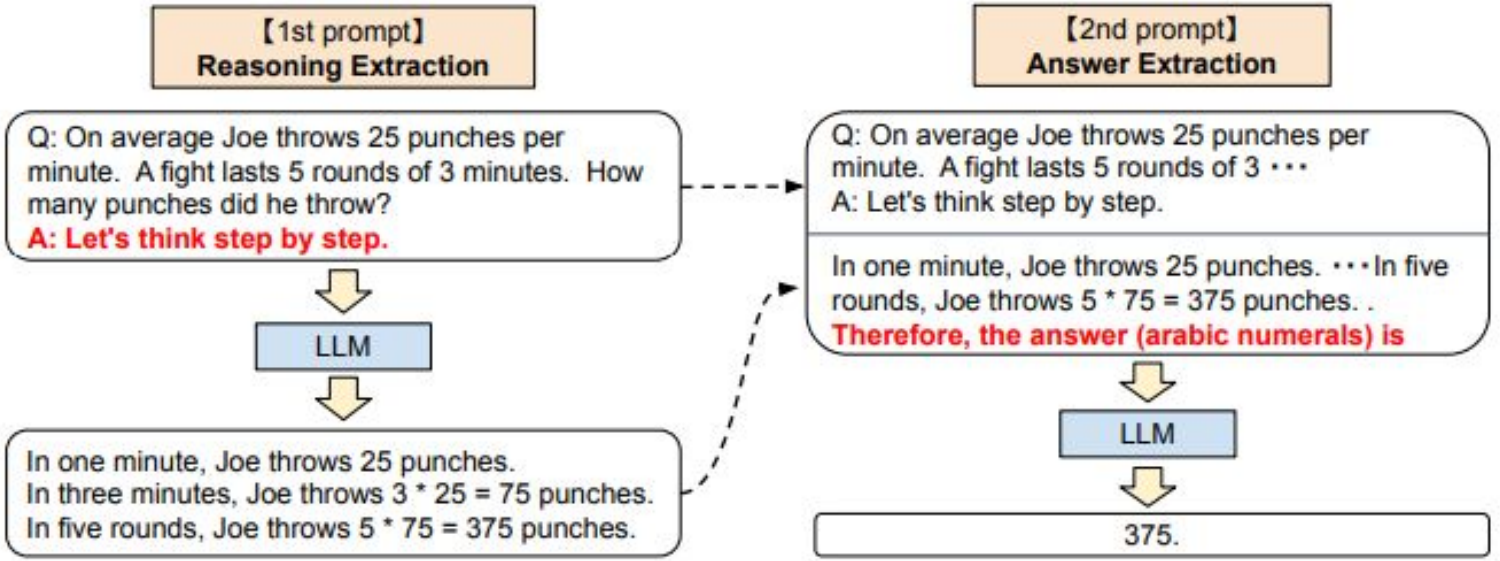


3. Zero-Shot CoT Method

- Task에 무관하게 Answer에 “Let’s think step by step”을 추가하여 Reasoning path를 생성
- Pipeline은 Reasoning Extraction, Answer Extraction의 순서로 진행됨



3. Zero-Shot CoT Method



1st prompt

- Q : [X]
- A : [T]
- T : Trigger sentence

2nd prompt

- [X'], [Z], [A]
- Z : Generation output at 1st step
- A : Trigger sentence to extract answer
- *SELF-AUGMENTED PROMPT*

3. Zero-Shot CoT Results

- **실험 1) Reasoning Ability에 대한 성능 검증**
  - Commonsense Reasoning, Symbolic Reasoning에서 유의미한 성능 향상이 관찰됨

Table 1: Accuracy comparison of Zero-shot-CoT with Zero-shot on each tasks.

	Arithmetic					
	SingleEq	AddSub	MultiArith	GSM8K	AQUA	SVAMP
zero-shot	74.6/**78.7**	**72.2/77.0**	17.7/22.7	10.4/12.5	22.4/22.4	58.8/58.7
zero-shot-cot	**78.0/78.7**	69.6/74.7	**78.7/79.3**	**40.7/40.5**	**33.5/31.9**	**62.1/63.7**

	Common Sense		Other Reasoning Tasks		Symbolic Reasoning	
	Common SenseQA	Strategy QA	Date Understand	Shuffled Objects	Last Letter (4 words)	Coin Flip (4 times)
zero-shot	**68.8/72.6**	12.7/**54.3**	49.3/33.6	31.3/29.7	0.2/-	12.8/53.8
zero-shot-cot	64.6/64.0	**54.8**/52.3	**67.5/61.8**	**52.4/52.9**	**57.6**/-	**91.4/87.8**

3. Zero-Shot CoT Results

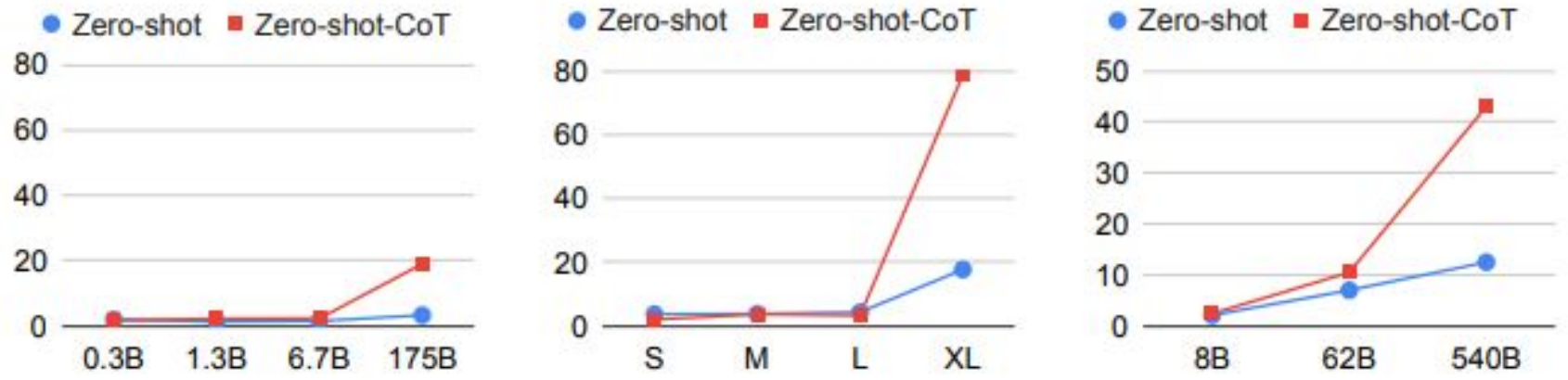
** 답변 형식에 따라 약간씩 다른 트리거 문구를 사용하였음

Table 9: Answer extraction prompts used for Zero-shot experiments in Table 1. C.S.QA : CommonsenseQA, D.U. : Date Understanding, S.O. : Tracking Shuffled Objects

No	Task	Zero-Shot (left)	Zero-Shot (right)
1	SingleEq	The answer (arabic numerals) is	The answer is
2	AddSub	The answer (arabic numerals) is	The answer is
3	MultiArith	The answer (arabic numerals) is	The answer is
4	GSM8K	The answer (arabic numerals) is	The answer is
5	AQUA-RAT	Among A through E, the answer is	The answer is
6	SVAMP	The answer (arabic numerals) is	The answer is
7	C.S.QA	Among A through E, the answer is	The answer is
8	StrategyQA	The answer (Yes or No) is	The answer is
9	D.U.	Among A through F, the answer is	The answer is
10	S.O.	Among A through C, the answer is	The answer is
11	Last Letters	The answer is	The answer is
12	Coin Flip	The answer (Yes or No) is	The answer is

3. Zero-Shot CoT Results

- **실험 2) Model Type, Scale에 따른 Reasoning Ability 검증 실험**
  - Model이 커질수록 Zero-shot-CoT와 Zero-shot의 성능 갭이 커짐



(a) MultiArith on Original GPT-3 (b) MultiArith on Instruct GPT-3 (c) GMS8K on PaLM

Figure 3: Model scale study with various types of models. S: text-ada-001, M: text-babbage-001, L: text-curie-001, XL: text-davinci-002. See Appendix A.3 and E for the detail.

3. Zero-Shot CoT Results

- **실험 3) Best Template을 찾기 위한 실험**
  - Zero-shot의 성능이 17.7임을 감안하면, Misleading한 Prompt를 대입하더라도 일부 경우에서는 더 좋은 성능이 나타남
  - 하지만 ?

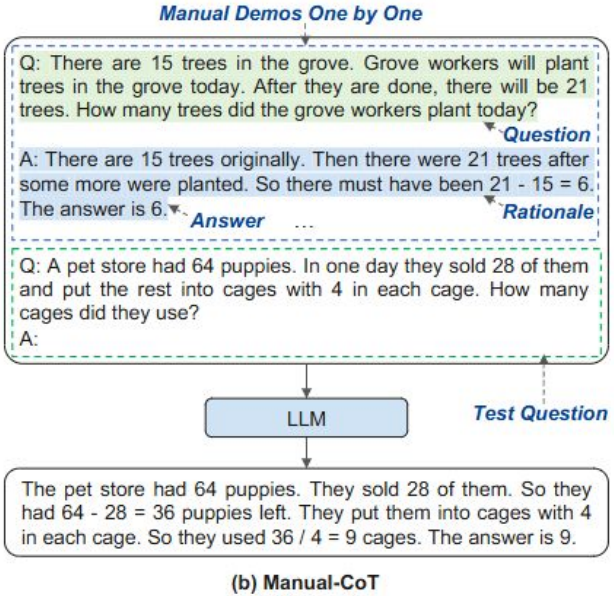
No.	Category	Template	Accuracy
1	instructive	Let's think step by step.	**78.7**
2		First, (*1)	77.3
3		Let's think about this logically.	74.5
4		Let's solve this problem by splitting it into steps. (*2)	72.2
5		Let's be realistic and think step by step.	70.8
6		Let's think like a detective step by step.	70.3
7		Let's think	57.5
8		Before we dive into the answer,	55.7
9		The answer is after the proof.	45.7
10	misleading	Don't think. Just feel.	18.8
11		Let's think step by step but reach an incorrect answer.	18.7
12		Let's count the number of "a" in the question.	16.7
13		By using the fact that the earth is round,	9.3
14	irrelevant	By the way, I found a good restaurant nearby.	17.5
15		Abrakadabra!	15.5
16		It's a beautiful day.	13.1
-		(Zero-shot)	17.7

AUTOMATIC CHAIN OF THOUGHT PROMPTING IN LARGE LANGUAGE MODELS

Zhuosheng Zhang[†]*, **Aston Zhang**[‡], **Mu Li**[‡], **Alex Smola**[‡]
[†]Shanghai Jiao Tong University, [‡]Amazon Web Services

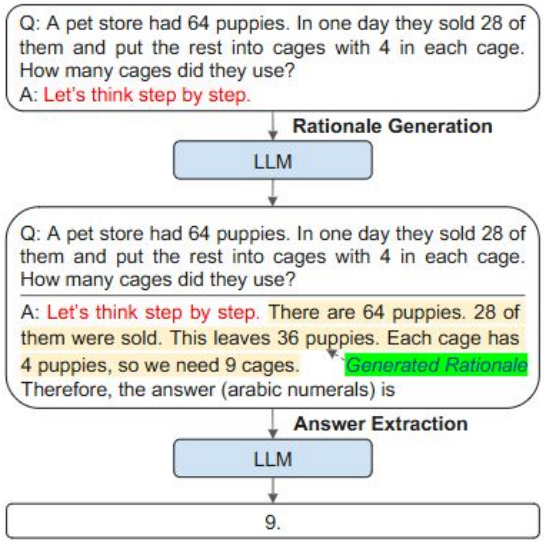
4. Auto CoT Introduction

- **Manual CoT :** 높은 성능이 보장되지만, 추가적인 **effort**가 요구됨



(b) Manual-CoT

- **Zero-shot-CoT : Simple**하지만 성능이 다소 아쉬움



(a) Zero-Shot-CoT

Auto CoT : Manual CoT처럼 보장된 성능을 나타낼 수 있는 좋은 Demonstration을 Automatic하게 Generation

But, How to generate good demonstration?

4. Auto CoT Introduction

- **Two Step approach with *"Clustering"***

Step 1)
Partition questions of given dataset into a few clusters.

Step 2)
Select a representative question from each cluster
&
generate its reading chain using Zero-Shot-CoT

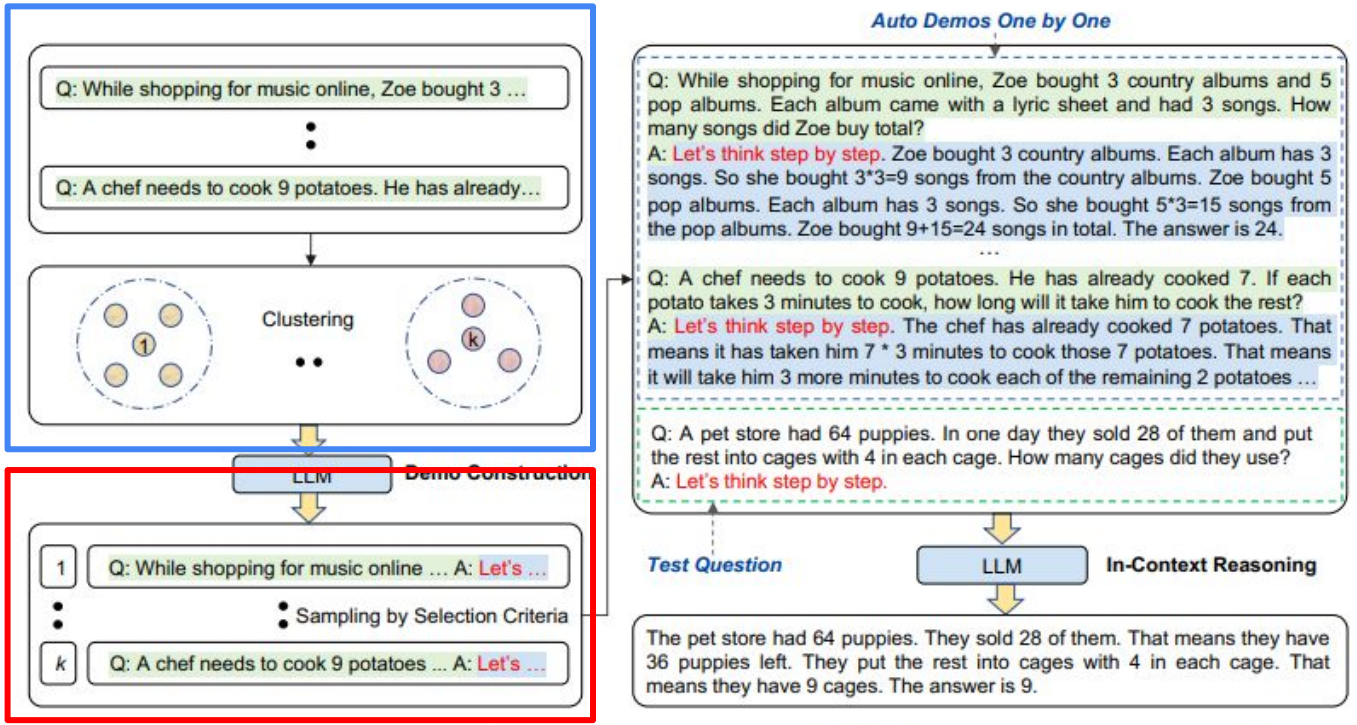


Figure 4: Overview of the Auto-CoT method. Different from Manual-CoT in Figure 1, demonstrations (on the right) are automatically constructed one by one (total: k) using an LLM with the "Let's think step by step" prompt.

4. Auto CoT Motivation

Why *"Clustering"?*

Manual CoT는 8개의 예시를 Prompt로 활용했음

- Retrieval-Q-CoT : 특정 질문을 기반으로 유사한 8개의 질문을 탐색함 (S-BERT 기반, Similarity 방향 탐색)
- Random-Q-CoT : 특정 질문을 기반으로 랜덤한 8개의 질문을 탐색함 (Diversity 방향 탐색)

실험 결과, Retrieval-Q-CoT에서 조금 더 성능 개선이 이루어짐을 확인할 수 있음. 하지만, Zero-shot에서 잘못된 추론을 하는 비율은 높은 편으로 나타남.

Table 1: Accuracy (%) of different sampling methods. Symbol † indicates using training sets with annotated reasoning chains.

Method	MultiArith	GSM8K	AQuA
Zero-Shot-CoT	78.7	40.7	33.5
Manual-CoT	**91.7**	46.9	35.8†
Random-Q-CoT	86.2	47.6†	36.2†
Retrieval-Q-CoT	82.8	**48.0**†	**39.7**†

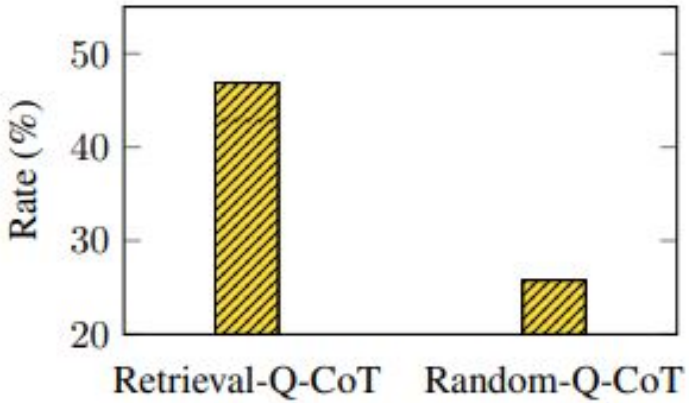


Figure 2: Unresolving Rate.

** S-BERT : 문장을 의미 기반 벡터로 변환해서 문장 간 유사도 비교/검색/클러스터링을 빠르게 할 수 있도록 만든 모델

4. Auto CoT Motivation

- Retrieval-Q-CoT : 특정 질문을 기반으로 유사한 8개의 질문을 탐색함 (S-BERT 기반, Similarity 방향 탐색)
- Random-Q-CoT : 특정 질문을 기반으로 랜덤한 8개의 질문을 탐색함 (Diversity 방향 탐색)

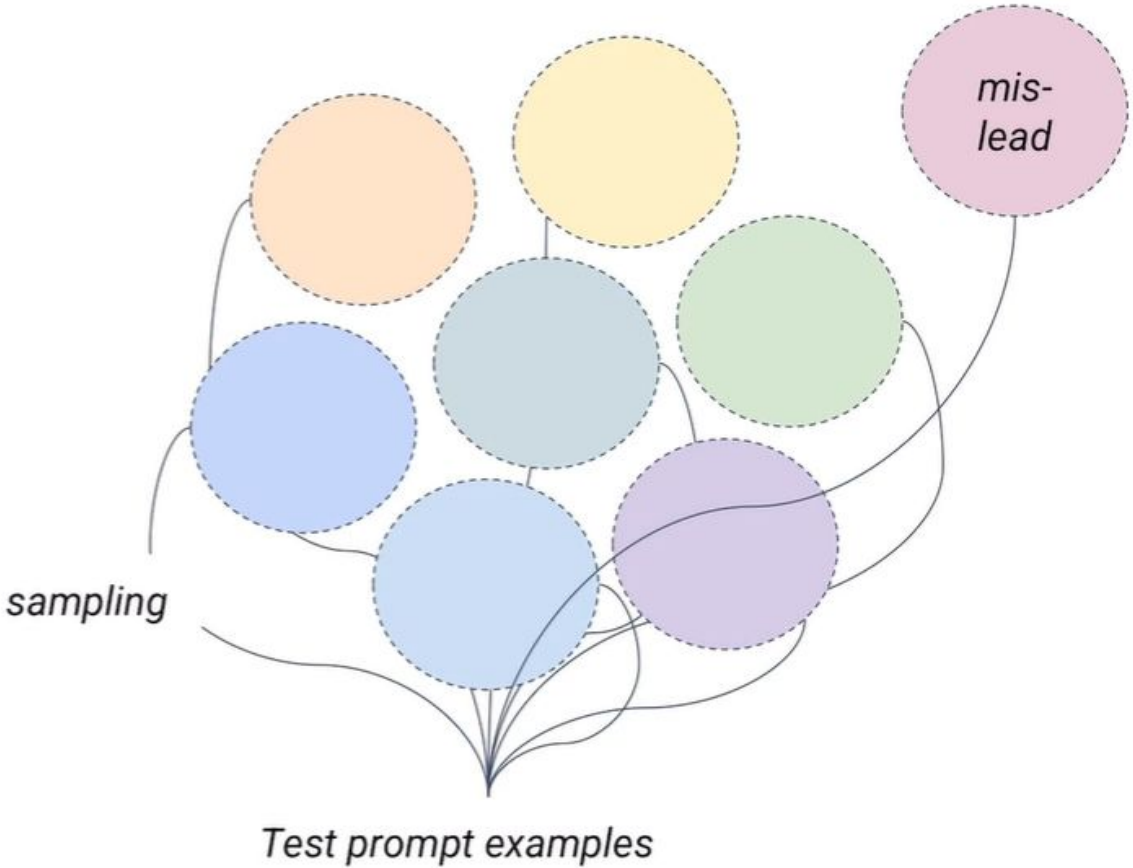
또한, Case Study에서 Retrieval-Q-CoT가 틀리는 문제는 비슷한 유형임을 발견하였음.

만약 Misleading Example을 하나의 Cluster에 배제시킬수 있다면?

Table 2: Examples of Retrieval-Q-CoT and Random-Q-CoT for constructing demonstrations and prediction. The upper part contains the automatically constructed demonstrations, and the lower part contains test questions along with predicted reasoning chains.

Retrieval-Q-CoT	Random-Q-CoT
Q: A chef needs to cook 9 potatoes. He has already cooked 7. If each potato takes 3 minutes to cook, how long will it take him to cook **the rest**? **A:** Let's think step by step. The chef has already cooked 7 potatoes. That means it has taken him 7 * 3 minutes to cook those 7 potatoes. That means it will take him 3 more minutes to cook each of the remaining 2 potatoes. That means it will take him a total of 7 * 3 + 2 * 3 = 21 minutes to cook all 9 potatoes. The answer is 21. ✗ **Q:** A chef needs to cook 13 potatoes. He has already cooked 5. If each potato takes 6 minutes to cook, how long will it take him to cook **the rest**? **A:** Let's think step by step. The chef has already cooked 5 potatoes. That means it has taken him 30 minutes so far. There are 13 potatoes in total. 5 have been cooked so far. That means there are 8 left to cook. Each potato takes 6 minutes to cook. That means it will take the chef 48 minutes to cook the 8 remaining potatoes. In total, it will take the chef 78 minutes to cook all 13 potatoes. The answer is 78. ✗	**Q:** Wendy uploaded 79 pictures to Facebook. She put 44 pics into one album and put the rest into 5 different albums. How many pictures were in each album? **A:** Let's think step by step. Wendy uploaded 79 pictures to Facebook. She put 44 pictures into one album. That means she put the rest of the pictures into 5 different albums. That means each album would have (79-44)/5 = 15 pictures in each album. The answer is 15.✓ **Q:** At a restaurant each adult meal costs $5 and kids eat free. If a group of 15 people came in and 8 were kids, how much would it cost for the group to eat? **A:** Let's think step by step. There are 8 kids, so that means there are 7 adults. Each adult meal costs $5, so that means that the 7 adults will cost $5*7=$35. Now we just need to add the two numbers together. $35+8=43$. So it would cost $43 for the group to eat. The answer is 43. ✗
Q: A chef needs to cook 15 potatoes. He has already cooked 8. If each potato takes 9 minutes to cook, how long will it take him to cook **the rest**? **A:** Let's think step by step. The chef has already cooked 8 potatoes. That means it has taken him 8 * 9 minutes to cook those 8 potatoes. That means it will take him 9 minutes to cook each of the remaining 7 potatoes. That means it will take him a total of 8 * 9 + 7 * 9 = 135 minutes to cook all 15 potatoes. The answer is 135. ✗	**Q:** A chef needs to cook 15 potatoes. He has already cooked 8. If each potato takes 9 minutes to cook, how long will it take him to cook **the rest**? **A:** Let's think step by step. The chef needs to cook 15 potatoes. He has already cooked 8. That means he has to cook 15-8=7 more potatoes. Each potato takes 9 minutes to cook. That means it will take him 9*7=63 minutes to cook **the rest** of the potatoes. The answer is 63. ✓

4. Auto CoT Motivation



4. Auto CoT Method

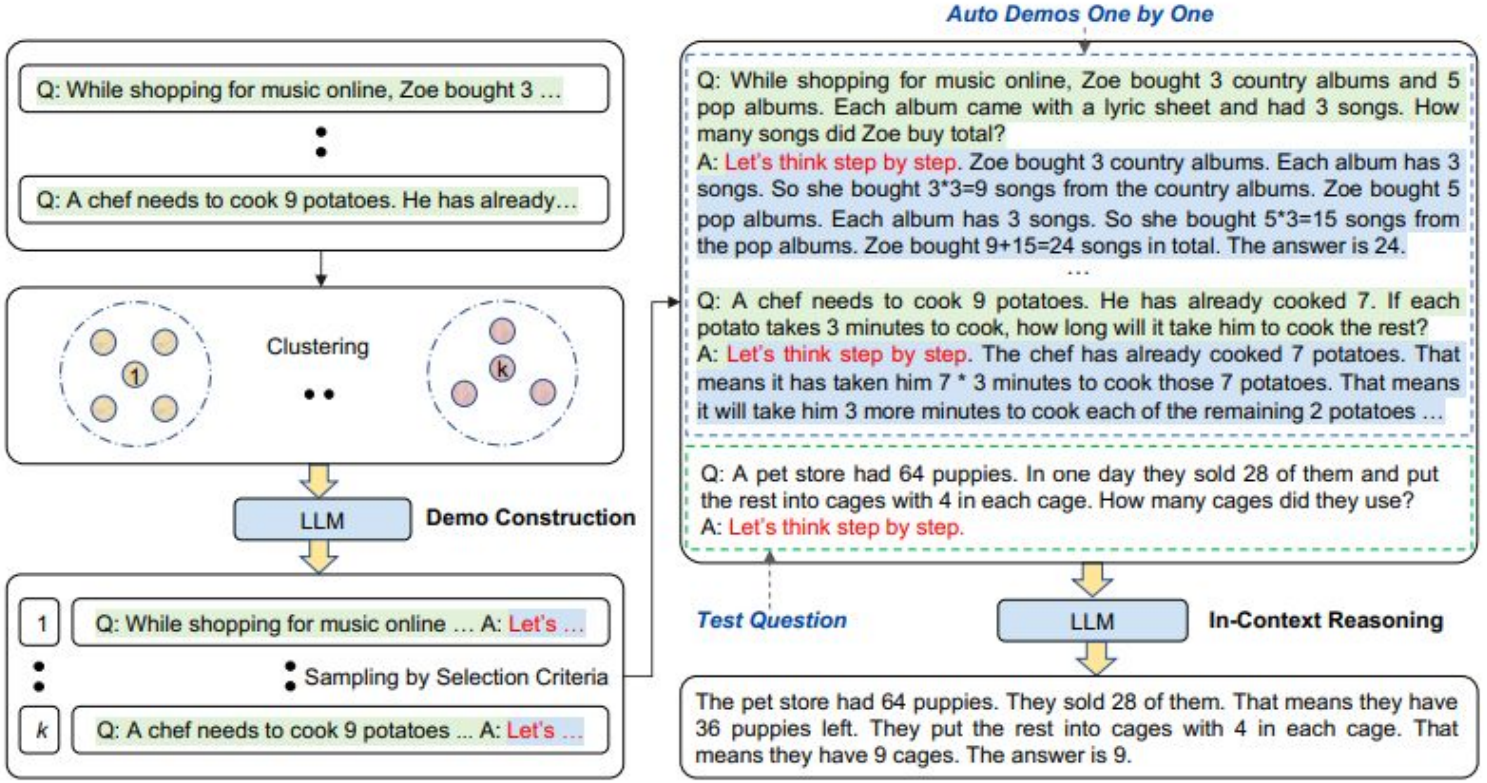


Figure 4: Overview of the Auto-CoT method. Different from Manual-CoT in Figure 1, demonstrations (on the right) are automatically constructed one by one (total: k) using an LLM with the "Let's think step by step" prompt.

4. Auto CoT Method

Algorithm 1 Cluster

Require: A set of questions Q and the number of demonstrations k

Ensure: Sorted questions $\mathbf{q}^{(i)} = [q_1^{(i)}, q_2^{(i)}, \ldots]$ for each cluster i $(i = 1, \ldots, k)$

1: **procedure** CLUSTER(Q, k)
2: **for** each question q in Q **do**
3: Encode q by Sentence-BERT
4: Cluster all the encoded question representations into k clusters
5: **for** each cluster $i = 1, \ldots, k$ **do**
6: Sort questions $\mathbf{q}^{(i)} = [q_1^{(i)}, q_2^{(i)}, \ldots]$ in the ascending order of the distance to the cluster center
7: **return** $\mathbf{q}^{(i)}$ $(i = 1, \ldots, k)$

Algorithm 1) Question Clustering

목표 : 주어진 데이터셋 Q 내의 질문들을 의미론적으로 유사한 그룹으로 묶어 샘플링 시 다양성을 확보한다.

1) 데이터셋 내의 질문 q에 S-BERT 모델을 사용하여 밀집 벡터 표현으로 인코딩한다.
2) Test question vector들은 K-means Algorithm에 입력되어, 생성하고자 하는 Demonstration의 종류 수인 k개만큼의 클러스터를 생성한다.
3) 각 클러스터 내의 질문들은 해당 클러스트의 중심(Centroid)으로부터의 거리에 따라 오름차순으로 정렬된다.

4. Auto CoT Method

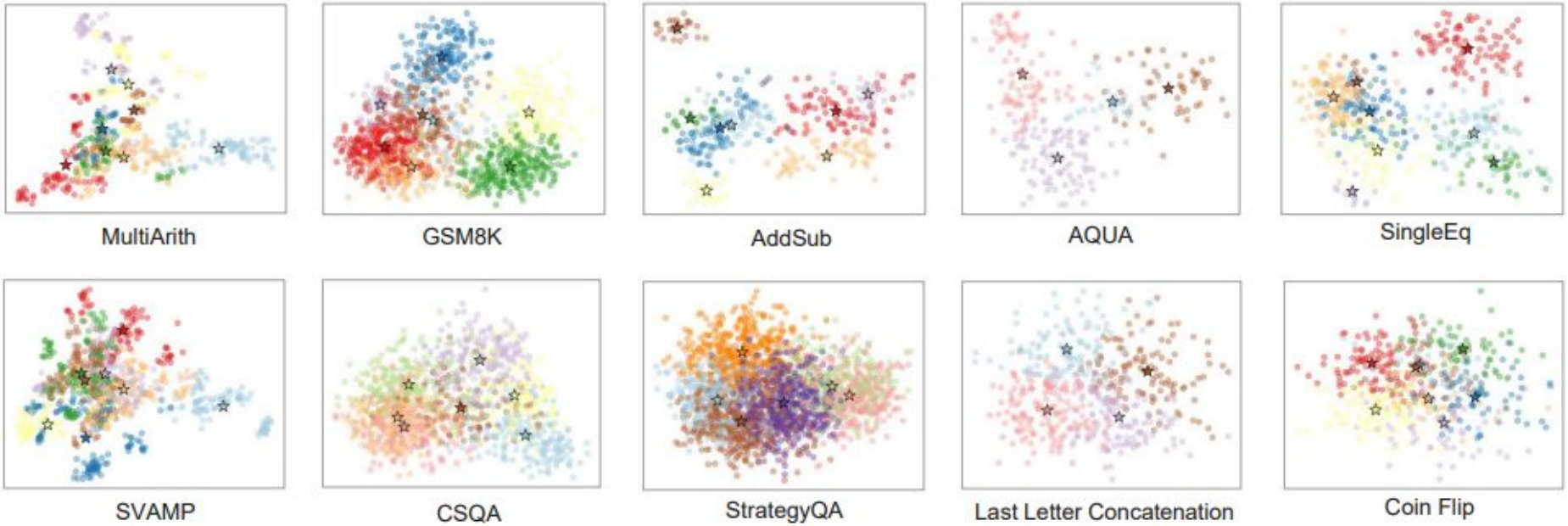


Figure 5: Question clustering on ten datasets of reasoning tasks. Stars denote cluster centers.

4. Auto CoT Method

Algorithm 2) Demonstration Sampling

목표 : 각 Cluster마다 Demonstration [question, rationale, answer] 구축

1) k개의 모든 Cluster에 대해 정렬된 질문 목록 q를 순회한다.
2) 후보 질문에 대해 프롬프트를 구성하여 LLM에 질문한다.
   a) Q : qi , A :Let's think step by step
3) LLM은 추론 사슬을 생성하고, 이 출력에서 rationale(중간 단계)과 추출된 answer를 parsing한다.
4) 휴리스틱 셀렉션된 q,r,a가 demonstration 목록에 넘어가고, 알고리즘은 다음 클러스터를 탐색한다. 만약, 선택되지 못했다면 현재 클러스터 내 다음 q로 이동한다.

Rationale : 특정 답변을 만들어낼 때 생성할 수 있는 Reasoning path.

Algorithm 2 Construct

Require: Sorted questions $\mathbf{q}^{(i)} = [q_1^{(i)}, q_2^{(i)}, \ldots]$ for each cluster i $(i = 1, \ldots, k)$, empty demonstration list $\mathbf{d}$
Ensure: Demonstration list $\mathbf{d} = [d^{(1)}, \ldots, d^{(k)}]$

procedure CONSTRUCT($\mathbf{q}^{(i)}, \ldots, \mathbf{q}^{(k)}$)
  for each cluster $i = 1, \ldots, k$ **do**
    for each question $q_j^{(i)}$ in $\mathbf{q}^{(i)}$ **do**
      Generate rationale $r_j^{(i)}$ and answer $a_j^{(i)}$ for $q_j^{(i)}$ using Zero-Shot-CoT
      if $q_j^{(i)}, r_j^{(i)}$ satisfy selection criteria **then**
        Add $d^{(i)} = [\text{Q: } q_j^{(i)}, \text{A: } r_j^{(i)} \circ a_j^{(i)}]$ to $\mathbf{d}$
        break
  return $\mathbf{d}$

4. Auto CoT Method

Algorithm 2) Demonstration Sampling

- 자동 생성된 demonstration의 품질이 지나치게 낮거나 복잡한 경우를 필터링하기 위한 휴리스틱 selection이 적용된다.
  a) q의 토큰 개수가 60개를 초과하지 않아야한다.
  b) 추론 단계 수가 5단계를 초과하지 않아야한다.
  c) rationale-answer 데이터셋에서 a가 비어있지 않고, 생성된 rationale 안에 존재해야 한다.

Algorithm 2 Construct

Require: Sorted questions $\mathbf{q}^{(i)} = [q_1^{(i)}, q_2^{(i)}, \ldots]$ for each cluster i $(i = 1, \ldots, k)$, empty demonstration list $\mathbf{d}$
Ensure: Demonstration list $\mathbf{d} = [d^{(1)}, \ldots, d^{(k)}]$

procedure CONSTRUCT($\mathbf{q}^{(i)}, \ldots, \mathbf{q}^{(k)}$)
  for each cluster $i = 1, \ldots, k$ **do**
    for each question $q_j^{(i)}$ in $\mathbf{q}^{(i)}$ **do**
      Generate rationale $r_j^{(i)}$ and answer $a_j^{(i)}$ for $q_j^{(i)}$ using Zero-Shot-CoT
      if $q_j^{(i)}, r_j^{(i)}$ satisfy selection criteria **then**
        Add $d^{(i)} = [\text{Q: } q_j^{(i)}, \text{A: } r_j^{(i)} \circ a_j^{(i)}]$ to $\mathbf{d}$
        break
  return $\mathbf{d}$

4. Auto CoT Method

Revisit)

Auto CoT : Manual CoT처럼 보장된 성능을 나타낼 수 있는 좋은 Demonstration을 Automatic하게 Generation

But, How to generate good demonstration?

1) Demonstration을 자동으로 생성한다면 좋은 Demonstration이 어떤 것인지 알지 못하기 때문에
2) 클러스터링을 진행하고
3) Zero-shot-CoT를 통해 reasoning path를 생성한 후
4) 다시 한 번 Demonstration을 구성해서
5) 새로운 Question이 들어왔을 때도 Rationale을 잘 생성하고 Accurate한 Answer를 얻음

4. Auto CoT Result

Table 3: Accuracy on ten datasets from three categories of reasoning tasks.

Model	*Arithmetic*						*Commonsense*		*Symbolic*	
	MultiArith	GSM8K	AddSub	AQuA	SingleEq	SVAMP	CSQA	Strategy	Letter	Coin
Zero-Shot	22.7	12.5	77.0	22.4	78.7	58.8	72.6	54.3	0.2	53.8
Zero-Shot-CoT	78.7	40.7	74.7	33.5	78.7	63.7	64.6	54.8	57.6	91.4
Few-Shot	33.8	15.6	83.3	24.8	82.7	65.7	**79.5**	**65.9**	0.2	57.2
Manual-CoT	91.7	46.9	81.3	35.8	86.6	68.9	73.5	65.4	59.0	97.2
Auto-CoT	**92.0**	**47.9**	**84.8**	**36.5**	**87.0**	**69.5**	74.4	65.4	**59.7**	**99.9**

- 대부분의 **Reasoning**에서 **Auto-CoT**의 성능이 우수했음
- 하지만 **Commonsense Reasoning**에서는 **Few-shot**의 성능에 미치지 못하였음
  - **Reasoning path**를 잘못 생성한 경우에 발생하는 **Error propagation**의 결과로 해석됨

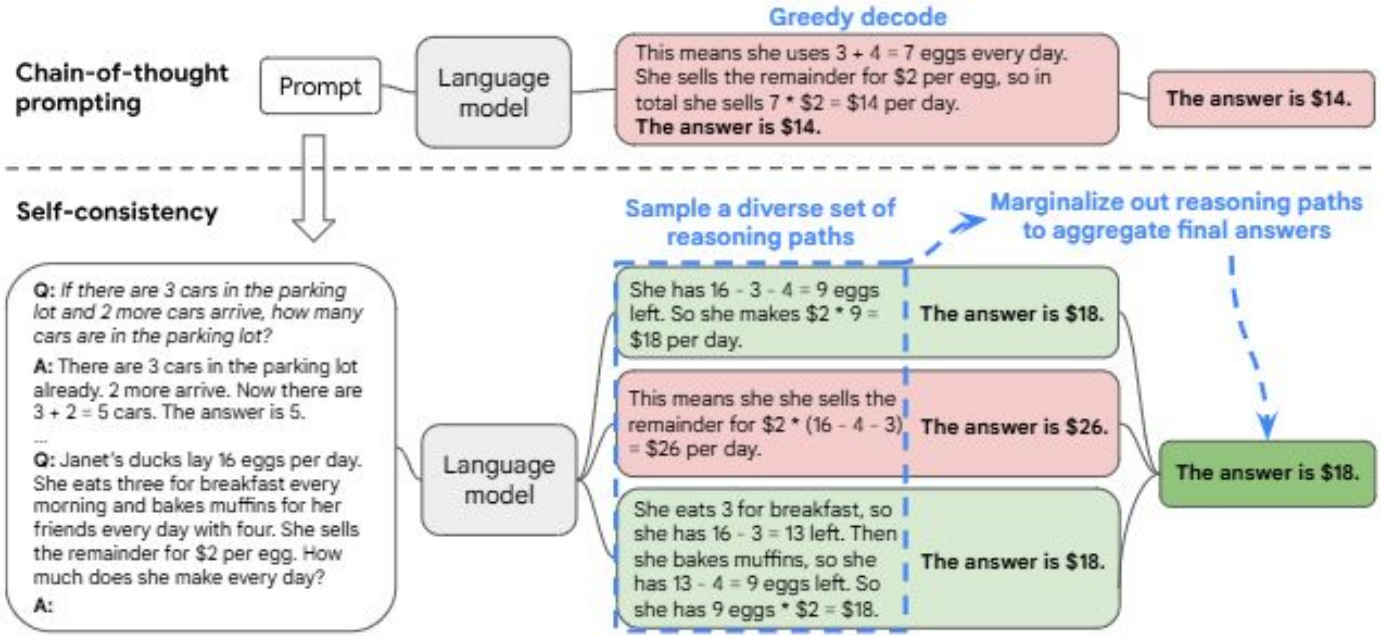
- **Auto CoT**에서는 **Reasoning path**를 **Design**하는 방법이 매우 중요하며 , **Task**에 따라 성능이 저하될 수도 있음

SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS

Xuezhi Wang[†‡] **Jason Wei**[†] **Dale Schuurmans**[†] **Quoc Le**[†] **Ed H. Chi**[†]
Sharan Narang[†] **Aakanksha Chowdhery**[†] **Denny Zhou**[†§]
[†]Google Research, Brain Team
[‡]xuezhiw@google.com, [§]dennyzhou@google.com

5. Self-Consistency Introduction

- **Self - Consistency :** 기존 **CoT** 프롬프팅에서 사용되는 **greedy decoding** 전략을 대체하는 새로운 디코딩 전략
- 여러 가지 다른 사고방식이 동일한 답변으로 이어질 때, 최종 답변에 대한 확신이 더 커진다 (다수결 투표)



5. Self-Consistency Result

Findings 1)

- **Arithmetic Reasoning** : 모든 4가지 LM에서 CoT 프롬프팅 대비 상당한 성능 향상을 보였습니다. 모델 스케일이 커질수록 성능 향상 폭이 더 커지는 경향을 보였습니다. 대부분의 태스크에서 SoTA 성능을 달성
- **Commonsense / Symbolic Reasoning** : 유사하게, Self-Consistency는 모든 LM에서 큰 성능 향상을 가져왔으며, 6개 태스크 중 5개에서 SoTA를 기록함. 특히, Out-of-Distribution (OOD) 설정의 상징 추론 태스크에서도 상당한 개선을 보임

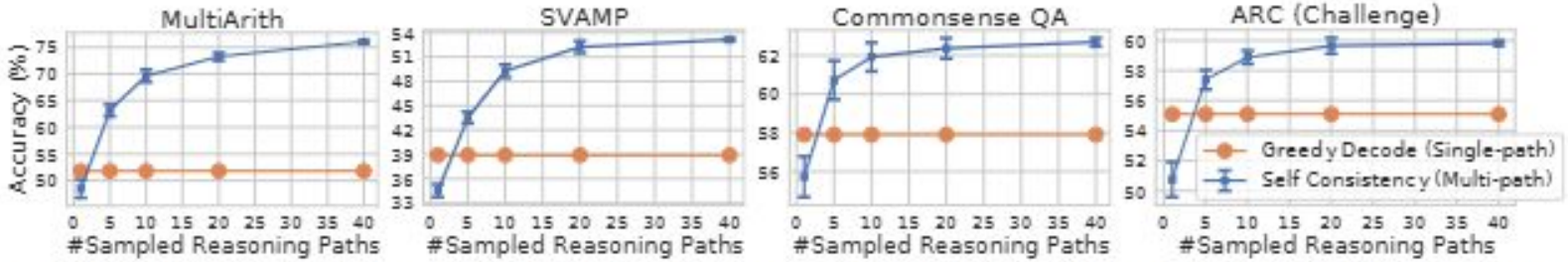


Figure 2: Self-consistency (blue) significantly improves accuracy over CoT-prompting with greedy decoding (orange) across arithmetic and commonsense reasoning tasks, over LaMDA-137B. Sampling a higher number of diverse reasoning paths consistently improves reasoning accuracy.

5. Self-Consistency Result

Findings 2)

- **샘플링 경로 수의 영향**
  - 더 많은 추론 경로를 샘플링할수록 더 나은 추론 정확도를 보여 추론 경로의 다양성이 중요
  - 더 많은 계산 비용을 수반한다는 한계가 있으나, 대부분의 경우 적은 수의 경로(예: 5개 또는 10개)를 샘플링하는 것으로도 큰 이점을 얻을 수 있었음



Figure 3: Self-consistency significantly outperforms sample-and-rank with the same # of samples.

6. Conclusion

- Chain-of-Thought를 활용하는 세 가지 방법론을 살펴봄
- Manual CoT는 Question마다 Prompting을 해야하지만, 성능이 우수함
- Zero-shot-CoT에서는 “Let’s think step by step”이라는 Magic trigger sentence를 추가함으로써 Demonstration 을 Manually 작성하지 않아도 CoT가 가능함을 보임
- Manual CoT와 Zero-shot-CoT가 결합된 형태인 Auto CoT는 Arithmetic과 Symbolic reasoning에서 우수한 성능을 보임

CoT advantage

CoT advantages

1. **연산 자원 할당** - 복잡한 문제를 여러 단계로 쪼개어, 모델이 더 많은 연산을 수행할 수 있게 함
2. **해석 가능성** - 모델이 왜 그 답을 냈는지 Window를 통해 들여다볼 수 있어, 어디서 논리가 틀렸는지 디버깅이 가능
3. **범용성** - 수학뿐만 아니라 상식, 기호 조작 등 인간이 언어로 풀 수 있는 모든 영역에 적용 가능
4. **간편함** - 별도의 미세 조정(Fine-tuning) 없이, 기존의 거대 모델(Off-the-shelf)에 프롬프트 예시만 몇 개 추가하면 바로 작동

Limitations

Hallucination

사고 과정(Reasoning steps) 자체는 논리적으로 보이나, 중간에 잘못된 사실을 기반으로 추론을 이어가는 경우 최종 답이 틀릴 뿐만 아니라 사용자를 더 강하게 속일 수 있음

질문: "철수는 사과 5개를 가졌고, 영희는 철수보다 3배 많은 사과를 가졌어. 민수는 영희보다 2개 적게 가졌다면, 세 사람이 가진 사과는 총 몇 개야?"

1. 철수는 5개를 가졌습니다.
2. 영희는 철수의 3배이므로 $5 \times 3 = 15$개를 가졌습니다.
3. 민수는 영희보다 2개 적으므로 **$15 - 2 = 12$개를 가졌습니다.** ← **Hallucination!**
4. 따라서 세 사람의 합은 $5 + 15 + 12 = 32$개입니다.

Cost problem

중간 과정을 모두 생성해야 하므로 출력 토큰 수가 급증

→ API 비용 증가와 응답 속도 저하로 직결

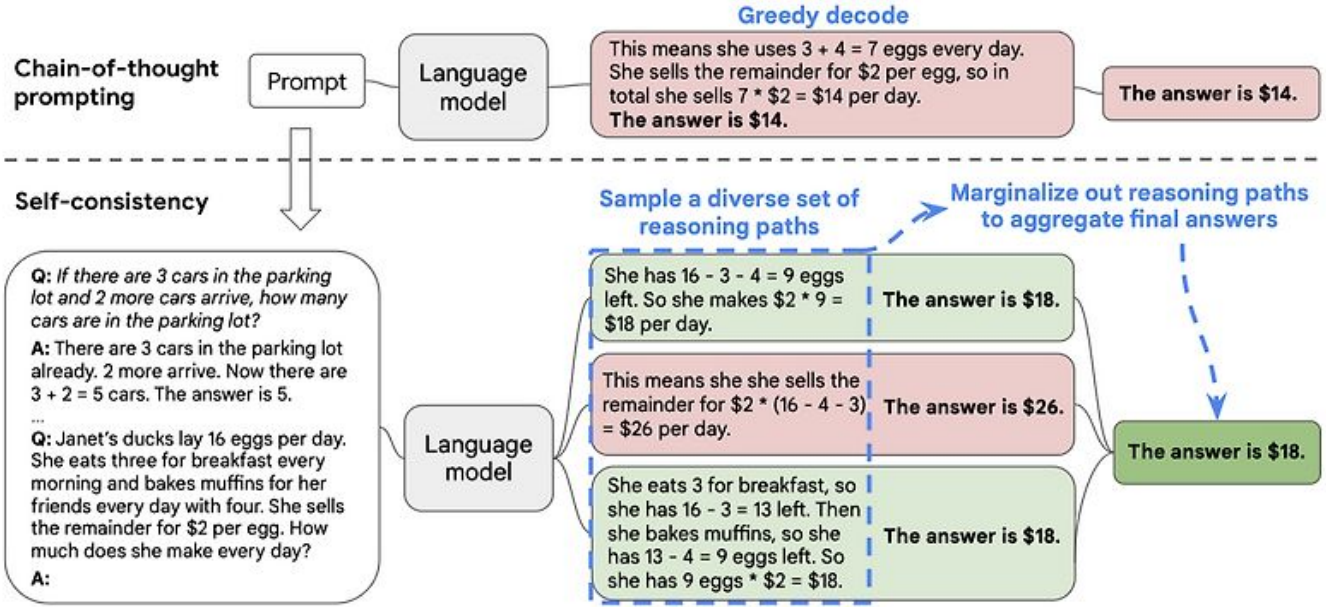
```
for i in range(1, 101):
    if i % 3 == 0 and i %
        print("FizzBuzz")
    elif i % 3 == 0:
        print("Fizz")
    elif i % 5 == 0:
        print("Buzz")
    else:
        print(i)
```



Advanced CoT Methods

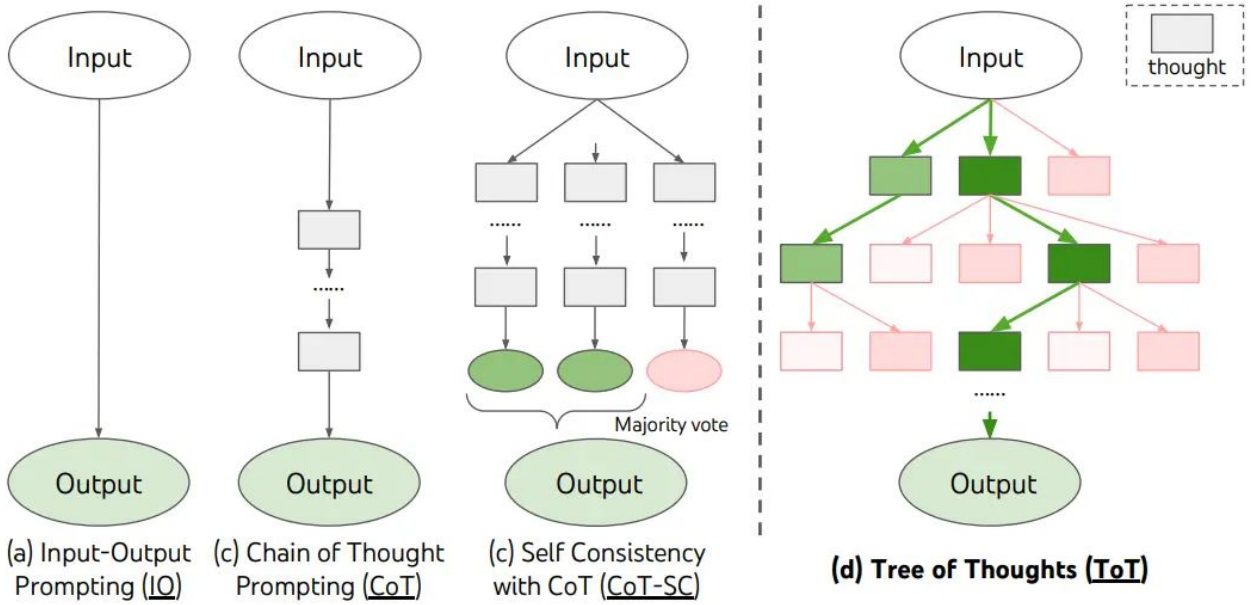
Self-Consistency

단순히 한 번의 CoT가 아니라, 여러 번의 사고 경로를 생성한 뒤 다수결(Majority Voting)로 답을 정하는 방식으로 추론의 노이즈를 제거하는 역할

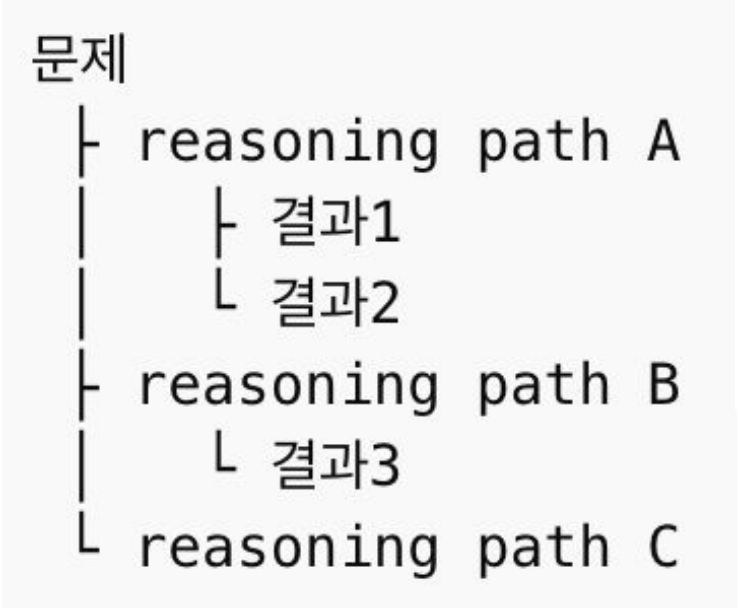


Tree of Thoughts (ToT)

선형적인 사슬이 아니라, 여러 갈래의 선택지를 나무(Tree)처럼 뻗어나가며 검토하고 **막히면 되돌아가는 (Backtracking)** 방식



Tree of Thoughts (ToT)



Tree of Thoughts: Deliberate Problem Solving with Large Language Models

Shunyu Yao
Princeton University

Dian Yu
Google DeepMind

Jeffrey Zhao
Google DeepMind

Izhak Shafran
Google DeepMind

Thomas L. Griffiths
Princeton University

Yuan Cao
Google DeepMind

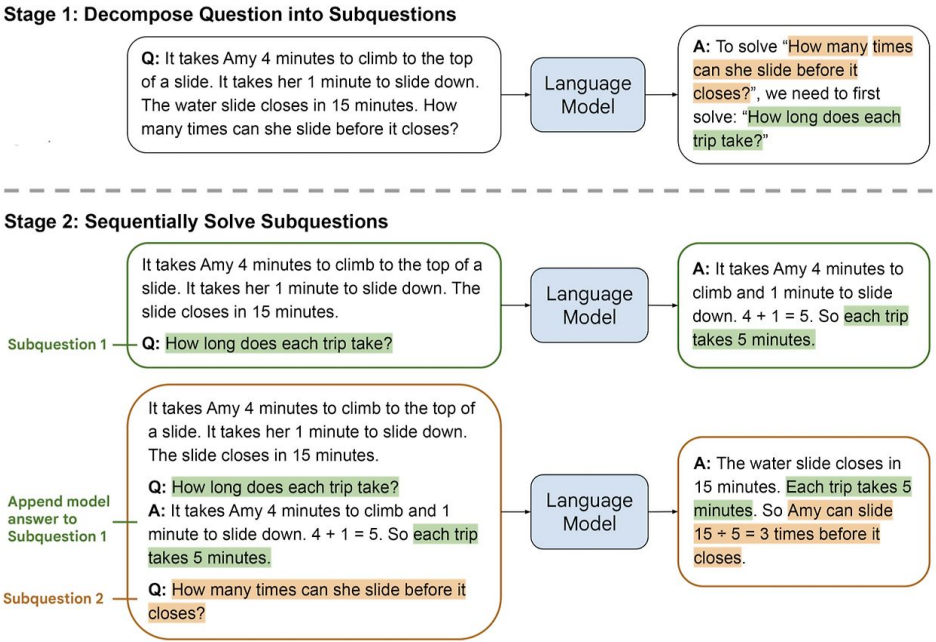
Karthik Narasimhan
Princeton University

Game of 24	Generate/Prompt tokens	Cost per case	Success
IO (best of 100)	1.8k / 1.0k	$0.13	33%
CoT (best of 100)	6.7k / 2.2k	$0.47	49%
ToT	5.5k / 1.4k	$0.74	74%

Table 7: Cost analysis on Game of 24.

Least-to-Most Prompting

아주 복잡한 문제를 여러 개의 쉬운 하위 문제(Sub-problems)로 먼저 분해한 뒤 하나씩 해결하는 기법



Least-to-Most Prompting

Q: “think, machine”
A: The last letter of “think” is “k”. The last letter of “machine” is “e”. Concatenating “k”, “e” leads to “ke”. So, “think, machine” outputs “ke”.

Q: “think, machine, learning”
A: “think, machine” outputs “ke”. The last letter of “learning” is “g”. Concatenating “ke”, “g” leads to “keg”. So, “think, machine, learning” outputs “keg”.

Q: “think, machine”
A: The last letter of “think” is “k”. The last letter of “machine” is “e”. Concatenating “k”, “e” leads to “ke”. So, “think, machine” outputs “ke”.

Q: “think, machine, learning”
A: The last letter of “think” is “k”. The last letter of “machine” is “e”. The last letter of “learning” is “g”. Concatenating “k”, “e”, “g” leads to “keg”. So, “think, machine, learning” outputs “keg”.

(좌) least-to-most prompting (우) Chain-of-Thought

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., ... & Chi, E. (2022). Least-to-most prompting enables complex reasoning in large language models.

Method	Non-football (DROP)	Football (DROP)	GSM8K
Zero-Shot	43.86	51.77	16.38
Standard prompting	58.78	62.73	17.06
Chain-of-Thought	74.77	59.56	60.87
Least-to-Most	**82.45**	**73.42**	**62.39**

ReAct (Reasoning + Action)

CoT를 **외부 행동과 결합**한 방식

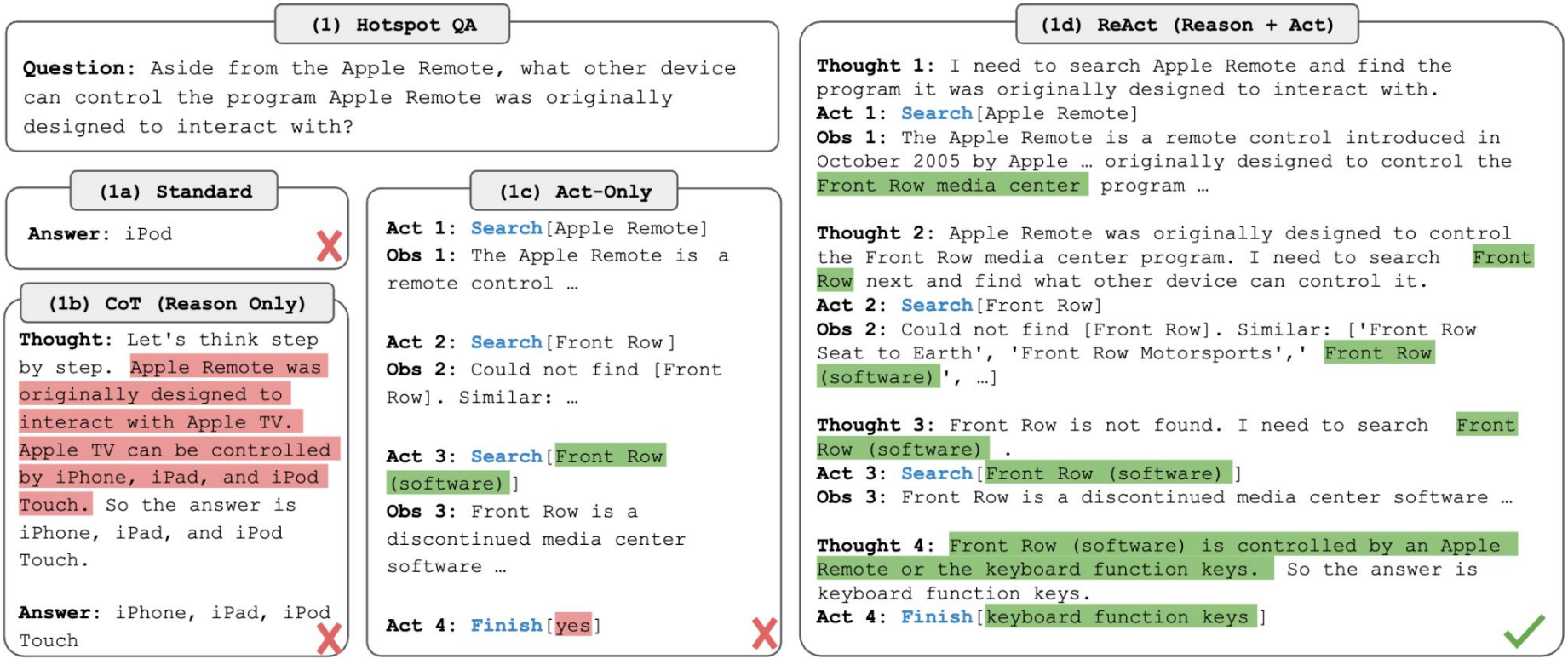
```
Thought: 필요한 정보를 찾아야 한다
Action: Wikipedia 검색
Observation: 결과 확인
Thought: 정보를 종합
Answer: 최종 답
```

REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Shunyu Yao[*,1], Jeffrey Zhao[2], Dian Yu[2], Nan Du[2], Izhak Shafran[2], Karthik Narasimhan[1], Yuan Cao[2]

[1]Department of Computer Science, Princeton University
[2]Google Research, Brain team
[1]{shunyuy,karthikn}@princeton.edu
[2]{jeffreyzhao,dianyu,dunan,izhak,yuancao}@google.com

ReAct (Reasoning + Action)



ReAct (Reasoning + Action)

Thought 1

```
Apple Remote가 어떤 프로그램과 상호작용하는지 찾아야 한다.
```

Action 1

```
Search[Apple Remote]
```

Observation

```
Apple Remote → Front Row media center program
```

Thought 2

```
Front Row가 무엇인지 찾아야 한다.
```

Action 2

```
Search[Front Row]
```

Thought 3

```
Front Row software를 검색해야 한다.
```

Action 3

```
Search[Front Row (software)]
```

Observation

```
Front Row is controlled by Apple Remote or keyboard function keys
```

Thought 4

```
Front Row는 Apple Remote 또는 keyboard function keys로 제어된다.
```

Final

```
Answer: keyboard function keys
```

Appendix

창발적 능력

CoT는 모든 모델에서 작동하지 X

약 **1,000억 개(100B) 이상의 파라미터**를 가진 거대 모델에서만 성능이 비약적으로 상승하는 '임계점'이 존재함.

작은 모델에게 CoT를 시키면 오히려 횡설수설하며 성능이 떨어지는 '역효과'가 나기도 함. → **추론은 거대 모델만의 특권**

CoT applications in multiple situations

단순히 "말을 잘하게 하는 것"을 넘어,
논리적 단계가 필요한 모든 지적 작업에 혁신적인 도구

1. 전문 영역에서의 의사결정 지원 (High-Stakes Reasoning)

실수가 치명적인 전문분야에서의 사용

- 의료 진단 보조
- 법률 조언 및 판례 분석



법률 및 정책 분석 프롬프트 예시

"당신은 전문 법률 자문가입니다. 아래 상황이 '개인정보 보호법' 위반인지 **단계별로 논리를 전개하여** 판단해 주세요.

상황: A 쇼핑몰은 이벤트 당첨자 발표를 위해 당첨자의 이름 전체와 휴대폰 번호 중간 자리를 제외한 나머지를 홈페이지에 공지했습니다. 별도의 사전 동의는 받지 않았습니다.

사고 절차:

1. 해당 정보가 '개인정보'의 정의에 부합하는지 분석합니다.
2. 정보 주체의 동의 없이 공개 가능한 예외 사유가 있는지 검토합니다.
3. 식별 가능성 측면에서 '비식별화'가 충분히 이루어졌는지 평가합니다.
4. 최종 결론을 내립니다."

의료 진단 및 가설 생성

"아래 환자의 증상을 바탕으로 **가능성 있는 질환들을 차례대로 추론(Differential Diagnosis)** 해 보세요.

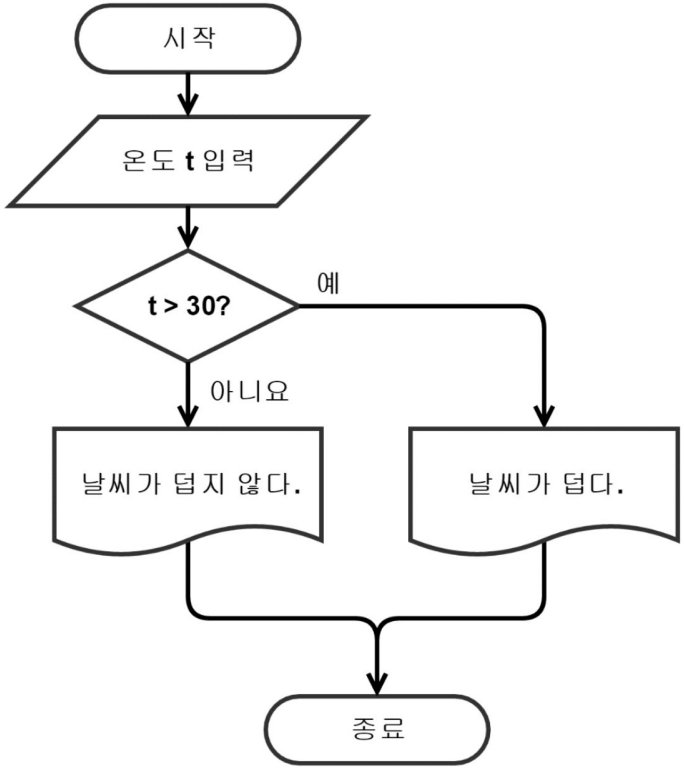
환자 정보: 45세 남성, 갑작스러운 가슴 통증, 통증이 왼쪽 어깨로 퍼짐, 식은땀 동반.

Let's think step by step:

- 먼저, 가장 생명에 치명적인 원인(심혈관 질환)부터 검토하세요.
- 통증의 양상과 방사통의 방향이 시사하는 바를 분석하세요.
- 소화기 계통이나 근골격계 질환일 가능성을 배제할 수 있는 근거를 찾으세요.
- 추가로 확인이 필요한 검사 항목을 제안하며 마무리하세요."

2. 복잡한 프로그래밍 및 알고리즘 설계

- 코드 디버깅
- 시스템 아키텍쳐 설계



알고리즘 최적화 및 복잡도 분석 프롬프트 예시

질문(Standard): "중복된 숫자가 포함된 배열에서 가장 많이 나타나는 숫자 K개를 찾는 파이썬 코드를 짜줘."

CoT 프롬프트(Advanced):

"배열 내 상위 K개 빈도 요소를 찾는 알고리즘을 설계해 주세요. **다음 단계를 거쳐 사고해 주세요:**

1. **데이터 구조 선택:** 각 요소의 빈도를 저장하기 위해 어떤 자료구조(예: Hash Map)가 가장 효율적인지 판단하세요.
2. **정렬 vs 힙:** 모든 요소를 정렬하는 방식과 우선순위 큐(Heap)를 사용하는 방식의 시간 복잡도($O(N \log N)$ vs $O(N \log K)$)를 비교하세요.
3. **단계별 로직:** 1) 빈도 카운팅, 2) 힙 구성, 3) 결과 추출 과정을 나누어 설명하세요.
4. **코드 구현:** 위 사고 과정을 바탕으로 최적화된 Python 코드를 작성하세요."

3. 개인화된 교육 및 튜터링 (EdTech)

CoT는 교육 분야에서 '정답 제조기'가 아닌 '선생님' 역할을 수행

- **단계별 학습 가이드:** 학생이 수학 문제를 틀렸을 때, 답만 알려주는 것이 아니라 AI가 CoT를 활용해 "1단계: 이 공식이 필요해", "2단계: 이 숫자를 대입해봐" 식으로 풀이 과정을 유도
- **오개념 교정:** 학생의 잘못된 풀이 과정을 입력하면, AI가 그 사고 경로를 분석하여 어느 지점에서 논리가 꼬였는지 정확히 짚어줌



교육적 튜터링 - 소크라테스식 대화의 프롬프트 예시

정답을 바로 주지 않고 학습자의 사고를 유도하는 방식

> "학생이 피타고라스의 정리를 활용해 문제를 풀려고 합니다. **학생의 실수를 바로 지적하지 말고, CoT를 활용해 스스로 깨닫게 유도하세요.**
>
> **학생의 입력:** '밑변이 3, 높이가 4인 직각삼각형의 빗변은 $3 + 4 = 7$이야.'
>
> **당신의 사고 과정:**
>
> - 학생이 길이를 단순히 더했다는 점을 파악합니다.
> - 빗변의 제곱이 각 변의 제곱의 합과 같다는 원리를 상기시킵니다.
> - 학생에게 직접 계산해볼 수 있도록 단계적인 질문을 던집니다."

4. 데이터 분석 및 비즈니스 전략

숫자 뒤에 숨겨진 의미를 해석하는 데 유용합니다.

- **비정형 데이터의 구조화:** 수많은 고객 리뷰를 분석할 때, 단순히 '긍정/부정'으로 나누는 게 아니라 "고객이 언급한 단점 → 그 이유 → 개선 방향"의 흐름으로 전략적인 보고서를 작성
- **시나리오 플래닝:** "유가가 10% 상승했을 때 우리 회사의 물류비에 미칠 영향을 단계별로 추론해줘"와 같은 복잡한 예측 작업에 활용

비즈니스 전략 및 시나리오 플래닝 프롬프트 예시

"현재 반도체 공급망 위기가 자동차 제조업체 B사에 미칠 영향을 **연쇄적 사고(Chain of Thought)**를 통해 분석하고 대응책을 제시하세요.

분석 경로:

1. 반도체 부족이 생산 라인 가동률에 미치는 직접적 영향 계산.
2. 생산 지연이 고객 인도 시간 및 브랜드 신뢰도에 미치는 영향 추론.
3. 신차 공급 부족이 중고차 가격 및 수익 구조에 미치는 2차 효과 분석.
4. 위 상황을 타개하기 위한 단기/장기 공급망 다변화 전략 도출."

CoT applications in LLM

Internalized Reasoning

CoT에 대해 설명해줘

2초 동안 생각함

추론 전용 모델(chatGPT의 o1 모델 등)들은 내부적으로 매우 긴 CoT 과정을 거침

작동 방식: 사용자가 질문을 던지면, 모델은 답변을 내놓기 전에 수초에서 수십 초간 내부적으로 수만 개의 추론 경로를 탐색하고, 이를 '생각 중 (Thinking...)'이라는 상태로 표시

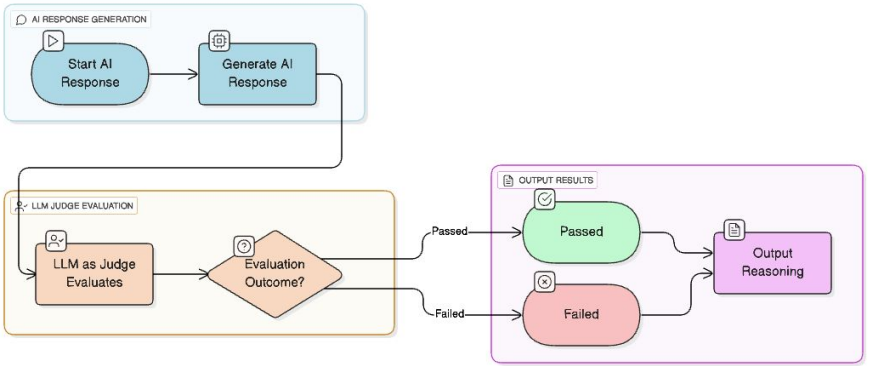
토큰 낭비, hallucination 가능성, 보안 때문에 CoT를 생성하지 않기도 함.

Self-Correction & Verification

AI가 자신이 낸 답이 맞는지 스스로 검토할 때 CoT를 사용

Step-by-Step Verification: 긴 코드를 짠 후, AI는 내부적으로 "1행부터 논리적으로 문제가 없는지" CoT 방식으로 한 줄씩 검토. 만약 중간에 모순을 발견하면 스스로 사고 경로를 수정하여 최종 답을 수정.

Self-Consistency: 동일한 문제에 대해 여러 개의 사고 사슬을 생성한 뒤, 가장 일관된 결론을 선택하여 답변의 신뢰



Today’s CoT usability

Tool-augmented CoT : LLM이 CoT로 reasoning을 하고 필요하면 **외부 도구를 호출**

```
Thought: 계산이 필요하다
Action: calculator
Observation: 결과 확인
Thought: 답 도출
```

Prompt → Reasoning generation(CoT) → Multiple reasoning → Consistency check → Tool usage → Answer

내부적으로

1️⃣ CoT 스타일 reasoning 생성

2️⃣ 여러 reasoning path 생성

3️⃣ self-consistency

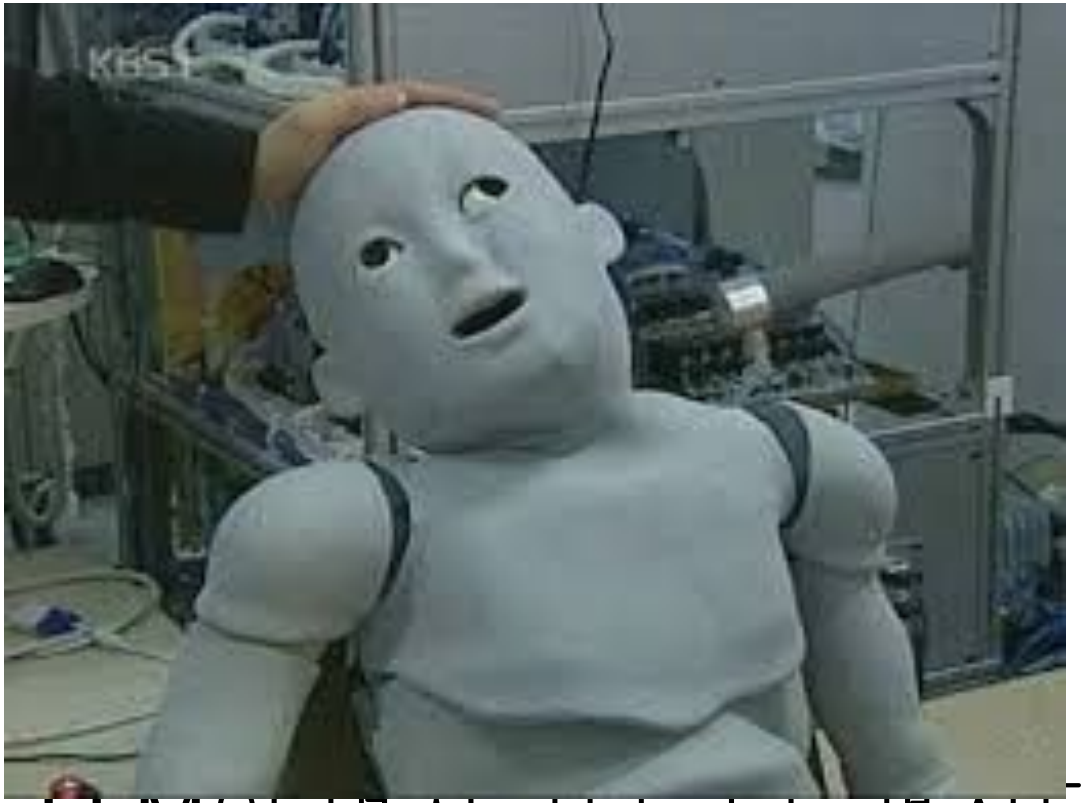
4️⃣ tool 사용

CoT application summary

결국 CoT의 응용은 **'결과'보다 '과정'이 중요한 모든 분야**에 해당

AI가 정답만 맞히는 계산기를 넘어, 인간과 논리적으로 대화하며 문제를 해결하는 '파트너'가 되는 과정





여러분의 LLM이 멍청하다고 느껴진다면 화만 내지 말고 LLM에게 푸는 방법을 제시해 보는것은 어떤가요?

Q&A

E.O.D