

AUTOMATIC CHAIN OF THOUGHT PROMPTING IN LARGE LANGUAGE MODELS

Zhuosheng Zhang^{1*}, Aston Zhang², Mu Li², Alex Smola²

¹Shanghai Jiao Tong University, ²Amazon Web Services

2023 LLM Paper seminar

2023 ICLR (POSTER)

Introduction

- Chain-of-Thought (CoT) Prompting (Wei et al. 2022)

- CoT Prompting elicits LLMs to reasoning on complex reasoning task by decomposing multi-step problems into multi-step problems into intermediate steps before giving answers.

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. ✅

Introduction

- Two major flavors in CoT Prompting

- CoT prompting comes in two major flavors: Zero-shot-CoT (Kojima et al., 2022) and Manual-CoT (Wei et al. 2022)

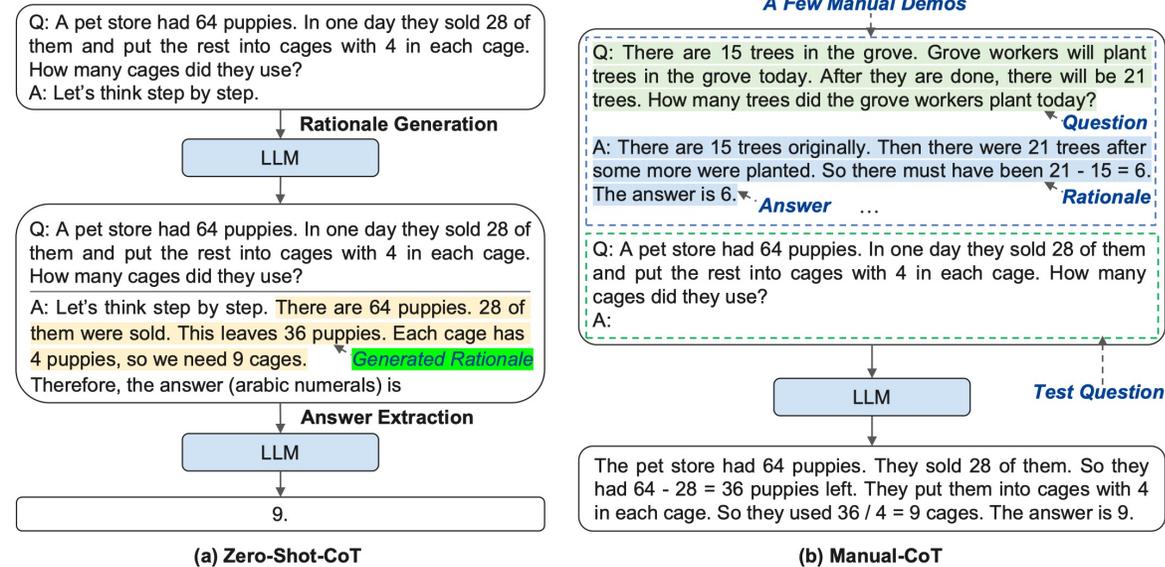


Figure 1: Zero-Shot-CoT (Kojima et al., 2022), using the 'Let's think step by step' prompt, and Manual-CoT (Wei et al., 2022b), using human generated reasoning chains, with example inputs and outputs of an LLM.

Introduction

- Superior performance from humans (Manual-CoT)
 - High-quality hand-crafted demonstrations 필요.
 - More problematic, different tasks such as arithmetic and Commonsense reasoning, require different ways of demonstrations to be manually generated.
- ➔ Automatically constructing demonstrations with questions and reasoning chains (Zero-Shot-CoT) would address this problems

Introduction

- **Limitations of Zero-Shot-CoT**

- A naïve approach is insufficient.
- Zero-Shot-CoT still makes mistakes in reasoning chains, even though retrieving semantically similar questions and generating reasoning chains.

Method

- **Auto-CoT – Main steps**

- First, partition questions of a given dataset into a few clusters.
- Second, select a representative question from each cluster and generate its reasoning chain using Zero-Shot-CoT with simple heuristics.

Method

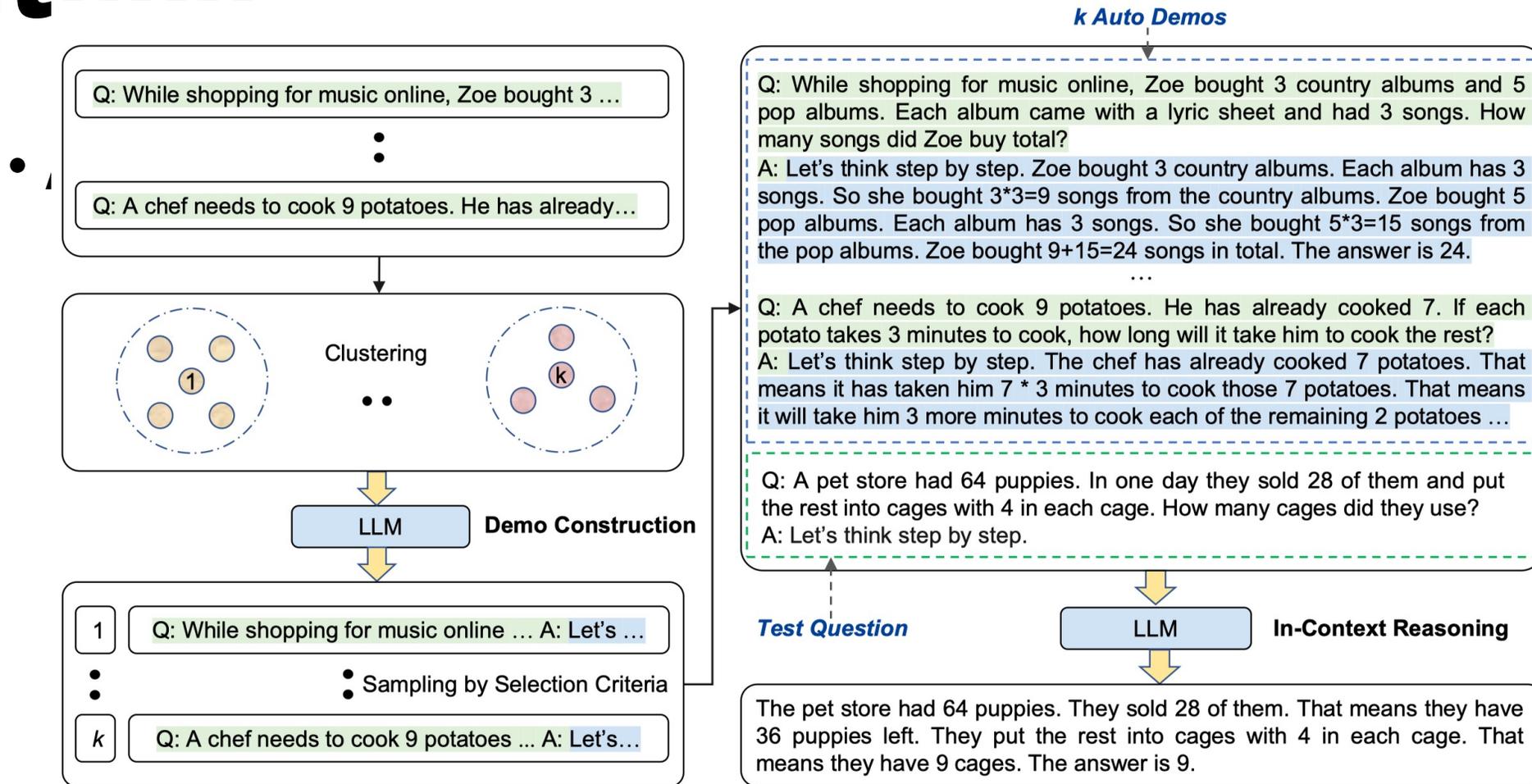


Figure 4: Auto-CoT. Different from Manual-CoT in Figure 1, a total of k demonstrations (on the right) are automatically constructed, using an LLM with the 'Let's think step by step' prompt.

Method

- **More challenging assumption**
 - Only a set of test questions are given (w/o a training dataset)
 - utilize for retrieval

Method

- Random vs. Semantic Retrieval

Table 1: Accuracy (%) of different sampling methods. † indicates the use of training sets with *manually* annotated CoT. We report the mean and standard deviations for Random-Q-CoT and Retrieval-Q-CoT over three runs.

Method	MultiArith	GSM8K	AQuA
Zero-Shot-CoT	78.7	40.7	33.5
Manual-CoT	91.7	46.9	35.8
Random-Q-CoT	87.1±1.8	47.3±0.5†	36.4±2.2†
Retrieval-Q-CoT	82.4±0.5	48.4±0.6†	39.6±2.4†

Random Retrieval > Semantic Retrieval?

Method

- Why do Retrieval show lower performances?

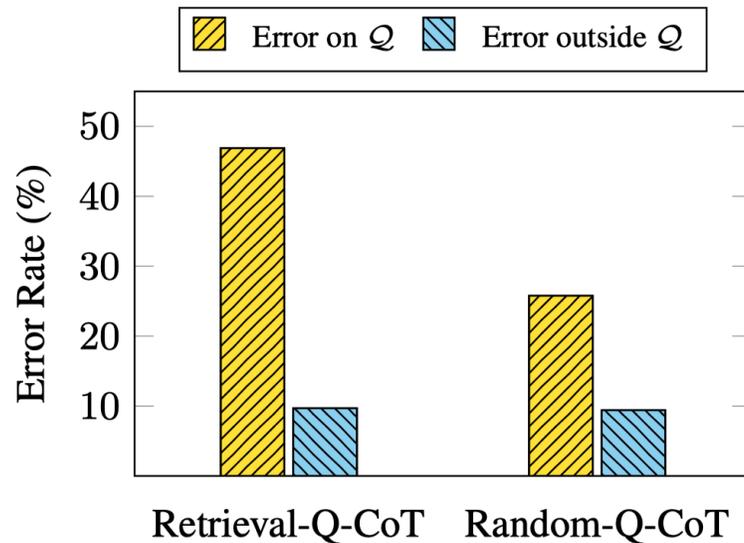


Figure 2: Error on Q and on its complement.

600 question from the MultiArith dataset

Q: 128 questions where Zero-Shot-CoT generates wrong answers

→ Almost same Error Rate outside Q for both,
But larger Error Rate on Q for Retrieval-Q-CoT

→ Retrieval tends to retrieve similarly hard questions
that the model gets wrong

Me 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
<p>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?</p> <p>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. ✗</p> <p>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?</p> <p>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. ✗</p>	<p>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?</p> <p>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, $79 - 44 = 35$ pictures, into 5 different albums. That means each album had $35 / 5 = 7$ pictures in it. The answer is 7. ✓</p> <p>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?</p> <p>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. ✗</p>
<p>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?</p> <p>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. ✗</p>	<p>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?</p> <p>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. ✓</p>

Method

- Errors frequently fall into the same cluster
 - Using k-means, partition all 600 test questions into k=8 clusters.
 - Observed that one of the clusters (Cluster 2) has an abnormally high error rate (52.3%)
 - The phenomenon could be generic as Zero-Shot-CoT may lack some skills to solve some common problems in target tasks

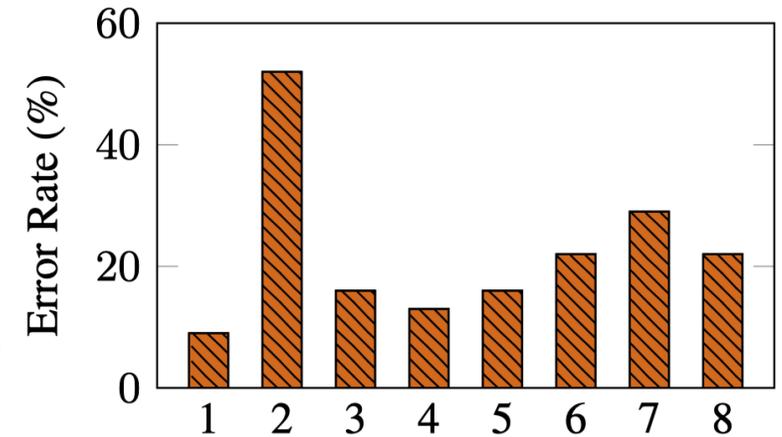


Figure 3: Clusters of similar questions.

→ Single cluster 가 아닌 multiple clusters에서 다양한 skills를 제공해보자

Method

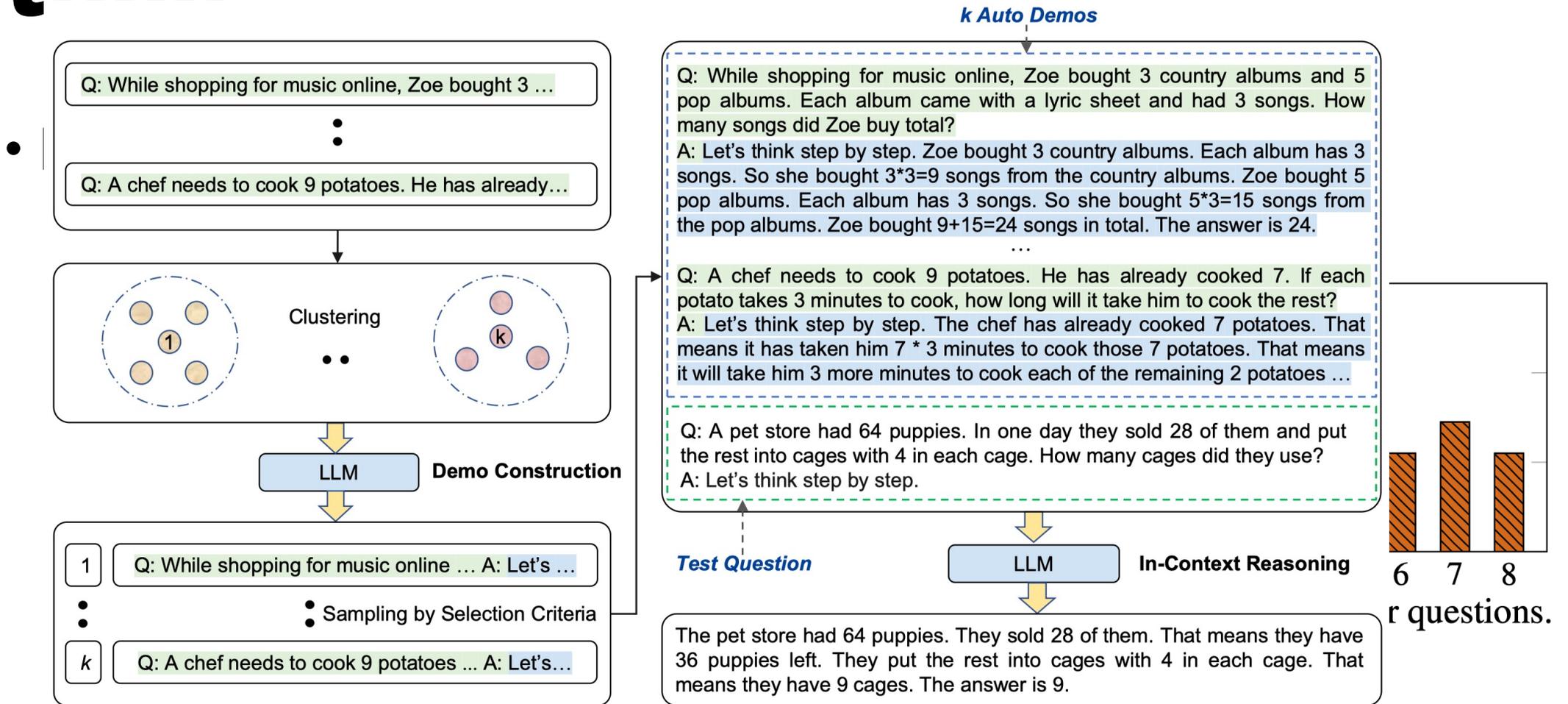


Figure 4: Auto-CoT. Different from Manual-CoT in Figure 1, a total of k demonstrations (on the right) are automatically constructed, using an LLM with the 'Let's think step by step' prompt.

Method

• Auto-CoT: Automatic Chain-of-Thought Prompting

Algorithm 1 Cluster

Require: Questions \mathcal{Q} , number of chains k

Ensure: Sorted questions $\mathbf{q}^i = [q_1^i, q_2^i, \dots]$

for each cluster $i \in \{1 \dots k\}$

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

Algorithm 2 Construct Demonstrations

Require: Sorted question lists \mathbf{q}^i for all k clusters

Ensure: Demonstration list $\mathbf{d} = [d^1, \dots, d^k]$

- 1: **procedure** CONSTRUCT($\mathbf{q}^1, \dots, \mathbf{q}^k$)
 - 2: $\mathbf{d} \leftarrow \emptyset$
 - 3: **for** each cluster $i \in \{1 \dots k\}$ **do**
 - 4: **for** each question $q \in \mathbf{q}^i$ **do**
 - 5: (rationale r , answer a) via Zero-Shot-CoT(q)
 - 6: **if** (q, r) satisfy selection heuristic **then**
 - 7: $\mathbf{d} \leftarrow \mathbf{d} \cup \{(q, r, a)\}$
 - 8: **break**
 - 9: **return** \mathbf{d}
-

Method

- **Auto-CoT: Automatic Chain-of-Thought Prompting**
 - Question clustering: Sentence-BERT 인코딩 기반 k-means clustering.
각 cluster 내의 vector list는 cluster center와의 distance에 기반해서 내림차순 정렬
→ Sampling 시 중심에서 가까운 질의부터 우선적으로 고려
 - Demonstration sampling: 각 클러스터에서 질의 q에 대하여 Zero-Shot-CoT 수행.
생성된 rationale r, answer a의 길이(60 tokens, no more than 5 reasoning steps)를
초과하면 next question으로 넘어가는 방식으로 선택

Experiments

- **Tasks and Datasets**

- (i): Arithmetic reasoning (MultiArith, GSM8K, AddSub, AQUA-RAT, SingleEq, SVAMP)
- (ii): Commonsense reasoning (CSQS, StrategyQA)
- (III) Symbolic reasoning (Last Letter Concatenation, Coin Flip)

Experiments

- Experimental results

10개 reasoning tasks에서 Manual-CoT 보다 좋음

Table 3: Accuracy on ten reasoning tasks. We report mean and standard deviations (\pm). Random-Q-CoT and Auto-CoT with three different random seeds. \uparrow and \uparrow indicate that Auto-CoT is significantly better than Random-Q-CoT at significance level $p < 0.01$ and $p < 0.05$ respectively.

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
Random-Q-CoT	87.1 \pm 1.8	40.4 \pm 0.4	82.7 \pm 1.3	31.5 \pm 1.1	81.5 \pm 0.3	66.7 \pm 1.8	71.9 \pm 0.2	58.0 \pm 0.1	58.2 \pm 0.3	95.9 \pm 0.1
Auto-CoT	92.0 \uparrow \pm 1.7	47.9 \uparrow \pm 3.7	84.8 \uparrow \pm 2.9	36.5 \uparrow \pm 2.2	87.0 \uparrow \pm 1.2	69.5 \uparrow \pm 2.2	74.4 \uparrow \pm 2.5	65.4 \uparrow \pm 0.4	59.7 \uparrow \pm 3.2	99.9 \uparrow \pm 0.1

Experiments

- Effect of Wrong Demonstrations

- Wrong demonstrations의 비율을 높여도 Demonstration의 diversity가 ICL의 효과성을 유지

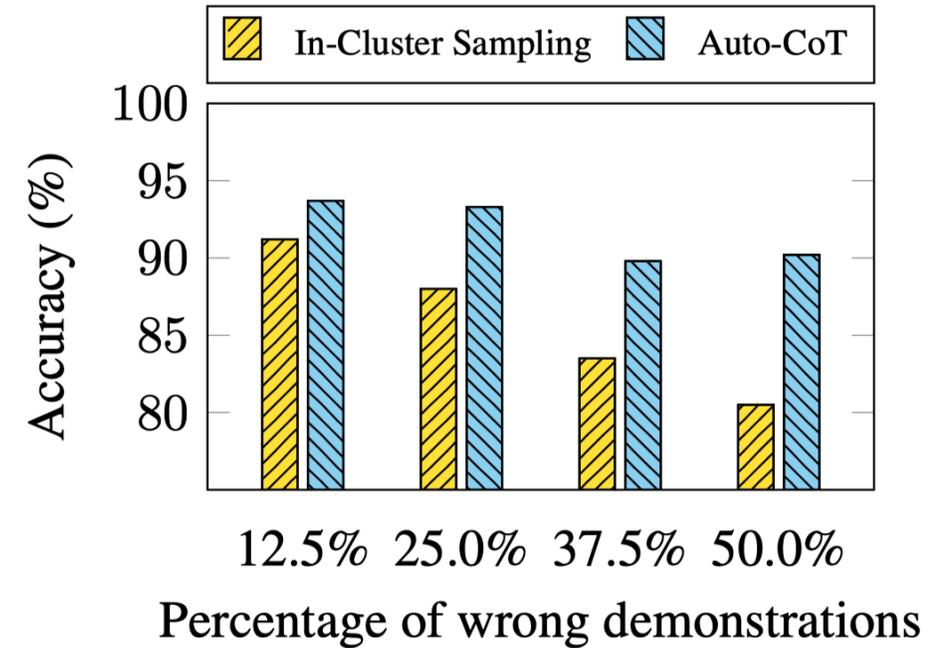


Figure 5: Effect of wrong demonstrations.

Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models

Lei Wang¹ Wanyu Xu² Yihuai Lan Zhiqiang Hu³ Yunshi Lan⁴
Roy Ka-Wei Lee³ Ee-Peng Lim^{1*}

¹Singapore Management University

²Southwest Jiaotong University

³Singapore University of Technology and Design

⁴East China Normal University

2023 LLM Paper seminar

2023 ACL

Introduction

- Zero-Shot-CoT (Kojima et al., 2022)
 - Zero-Shot-CoT eliminates the need for manually crafted examples in prompts by appending “*Let’s think step by step*” to the target problem fed to LLMs such as GPT-3.

Introduction

- Three pitfalls of Zero-Shot-CoT (Kojima et al., 2022)

- Results on a sample of 100 arithmetic test examples

- Calculation errors (7%):
 - wrong answer
- Missing Step errors (12%):
 - missed-out some intermediate steps
- Semantic misunderstanding (27%):
 - semantically misunderstand the problem

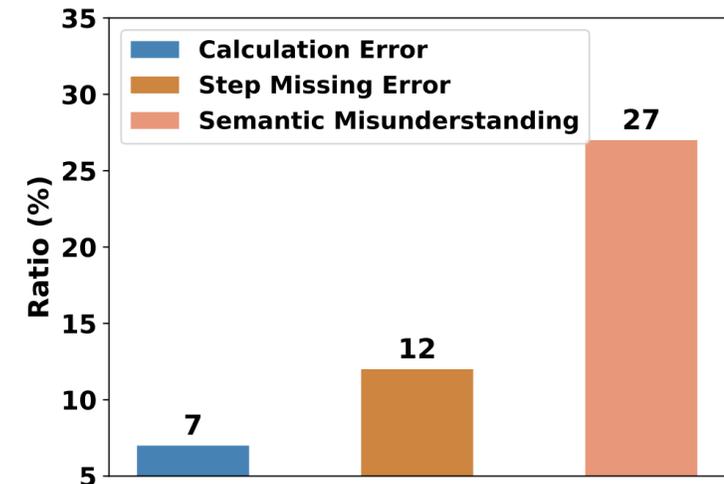


Figure 1: Error analysis of 46 GSM8K problems with incorrect answers returned by Zero-shot-CoT using GPT-3 LLM. Following [Wei et al. \(2022b\)](#) and [Wang et al. \(2022a\)](#), we assign “Calculation Error” (7%), “Step Missing Error” (12%), or “Semantic misunderstanding Error” (27%) to each incorrect answer.

Introduction

- **Plan-and-Solve (PS) Prompting**

- First, devising a plan to divide the entire task into smaller subtasks,
- and then carrying out the subtasks according to the plan.
- + Give more detailed description to solve a variety of problems. **(PS+ prompting)**

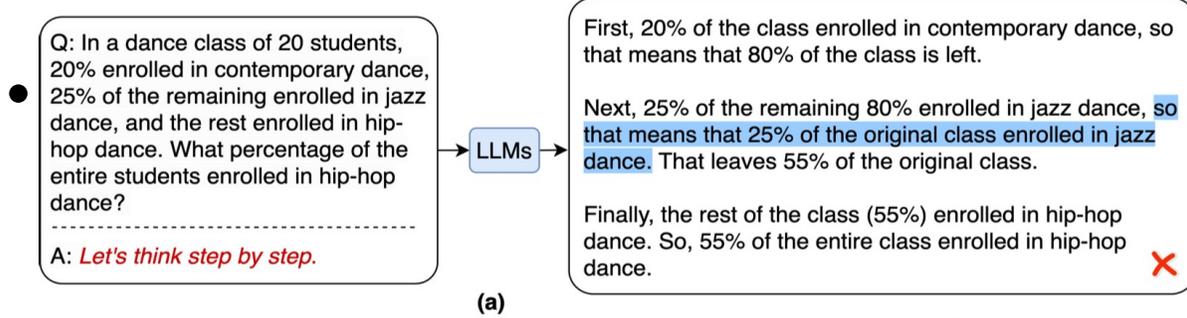
Introduction

- **Plan-and-Solve (PS) Prompting**

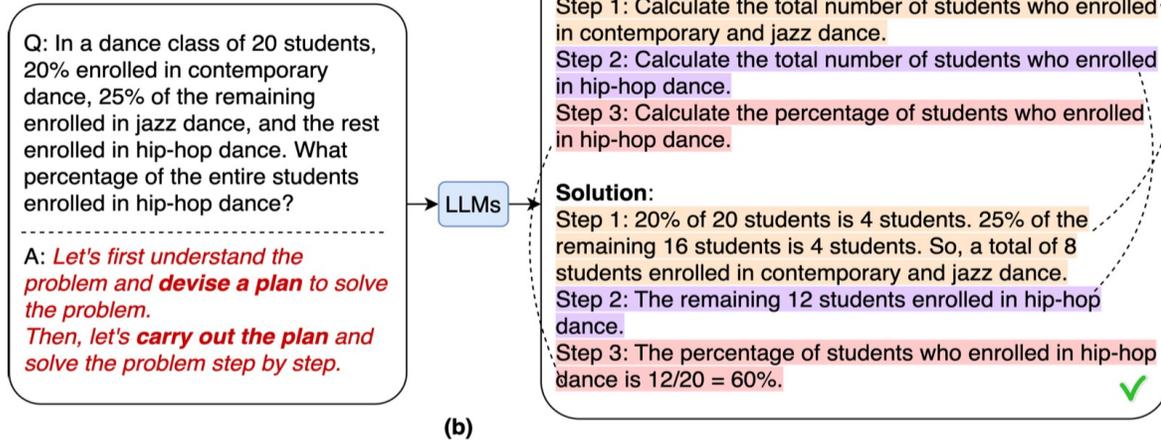
- Zero-Shot-PS+ > Zero-shot-CoT by a large margin
- Zero-shot-PS+ has a similar performance to an 8-shot CoT prompting in arithmetic reasoning (sometimes outperforms).
- A new CoT prompting approaches 개발에 대한 spark를 기대

Method

(a) Zero-Shot-CoT



(b) PS



(c) Answer Extraction

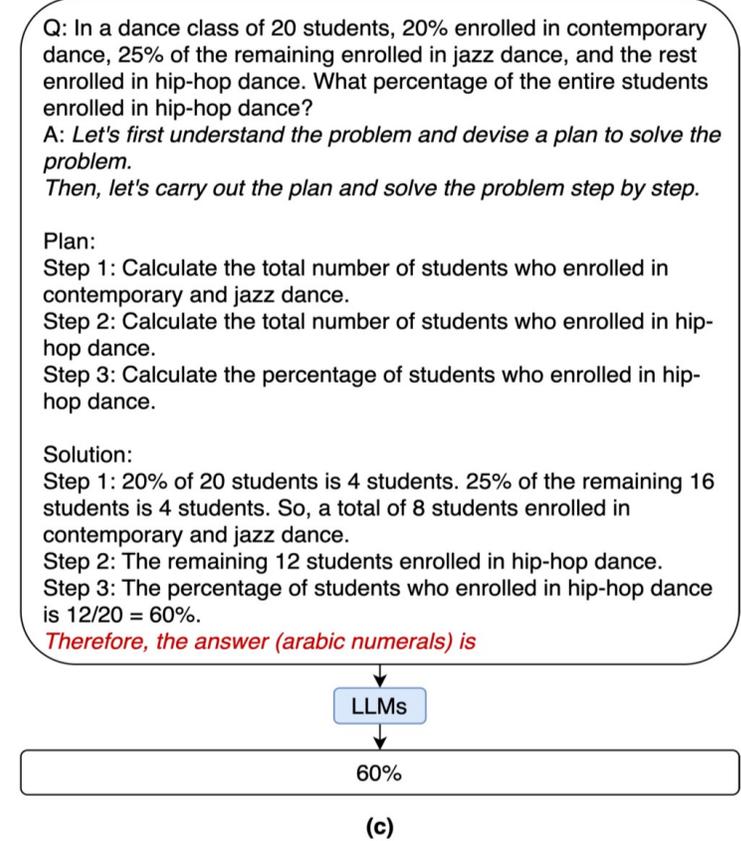


Figure 2: Example inputs and outputs of GPT-3 with (a) Zero-shot-CoT prompting, (b) Plan-and-Solve (PS) prompting, and (c) answer extraction prompting. While Zero-shot-CoT encourages LLMs to generate multi-step reasoning with “Let’s think step by step”, it may still generate wrong reasoning steps when the problem is complex. Unlike Zero-shot-CoT, PS prompting first asks LLMs to devise a plan to solve the problem by generating a step-by-step plan and carrying out the plan to find the answer.

Method

- Zero-Shot-PS Prompting

- Q: [X]. A: Let's first understand the problem and devise a plan to solve the problem. Then, let's carry out the plan and solve the problem step by step.
- [PS+] "pay attention to calculation":
 - calculation error를 줄이기 위한 trigger sentence
- [PS+] "extract relevant variables and their corresponding numerals":
 - the input problem statement에 explicit 정보를 놓치지 않도록
- [PS+] "calculate intermediate results":
 - relevant and important intermediate reasoning steps를 놓치지 않도록

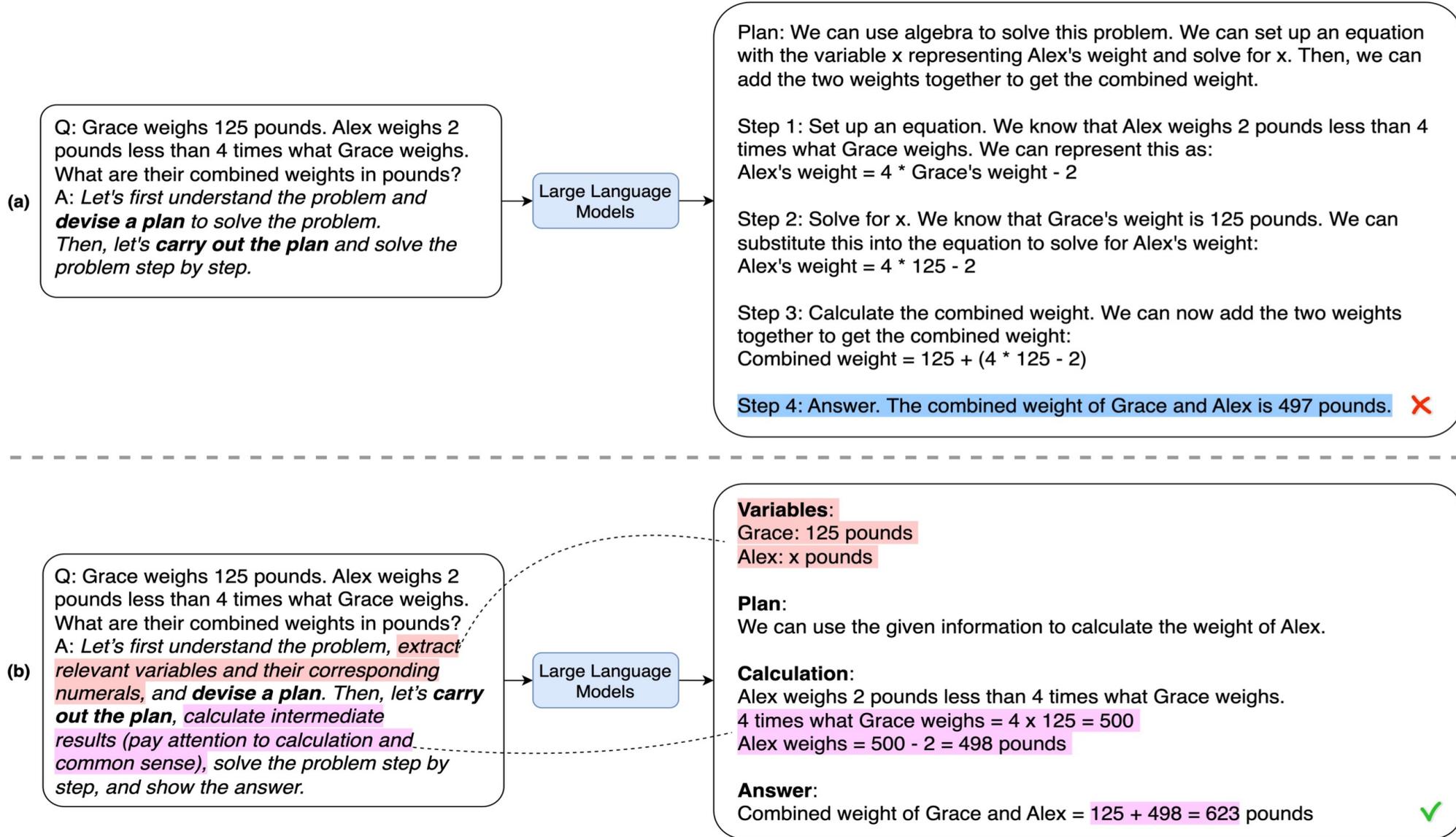


Figure 3: Example inputs and outputs of GPT-3 with (a) Plan-and-Solve (PS) Prompting and (b) Plan-and-Solve prompting with more detailed instructions (PS+ prompting). PS+ prompting greatly improves the quality of the generated reasoning process.

Experiments

- **Benchmark Dataset**

- Arithmetic Reasoning (GSM8K, SVAMP, MultiArith, Addsub, AQUA, SingleEq)
- Commonsense Reasoning (CSQS, StrategyQA)
- Symbolic Reasoning (Last Letter Concatenation, Coin Flip)

Experiments

- Experimental results (Arithmetic Reasoning)

Table 2: Accuracy comparison on six math reasoning datasets. The best and second best results are boldfaced and underlined respectively.

Setting	Method (text-davinci-003)	MultiArith	GSM8K	AddSub	AQuA	SingleEq	SVAMP	Average
Zero-Shot	CoT	83.8	56.4	85.3	38.9	88.1	69.9	70.4
	PoT	92.2	57.0	85.1	<u>43.9</u>	<u>91.7</u>	70.8	<u>73.5</u>
	PS (ours)	87.2	<u>58.2</u>	<u>88.1</u>	42.5	89.2	<u>72.0</u>	<u>72.9</u>
	PS+ (ours)	<u>91.8</u>	59.3	92.2	46.0	94.7	75.7	76.7
Few-Shot	Manual-CoT	93.6	58.4	91.6	48.4	93.5	80.3	77.6
	Auto-CoT	95.5	57.1	90.8	41.7	92.1	78.1	75.9

Experiments

- Experimental results (CS, Symbolic reasonings)

- PS+ consistently outperforms Zero-Shot CoT
- On symbolic reasoning, PS+ shows strong performances

Table 3: Accuracy on commonsense reasoning datasets.

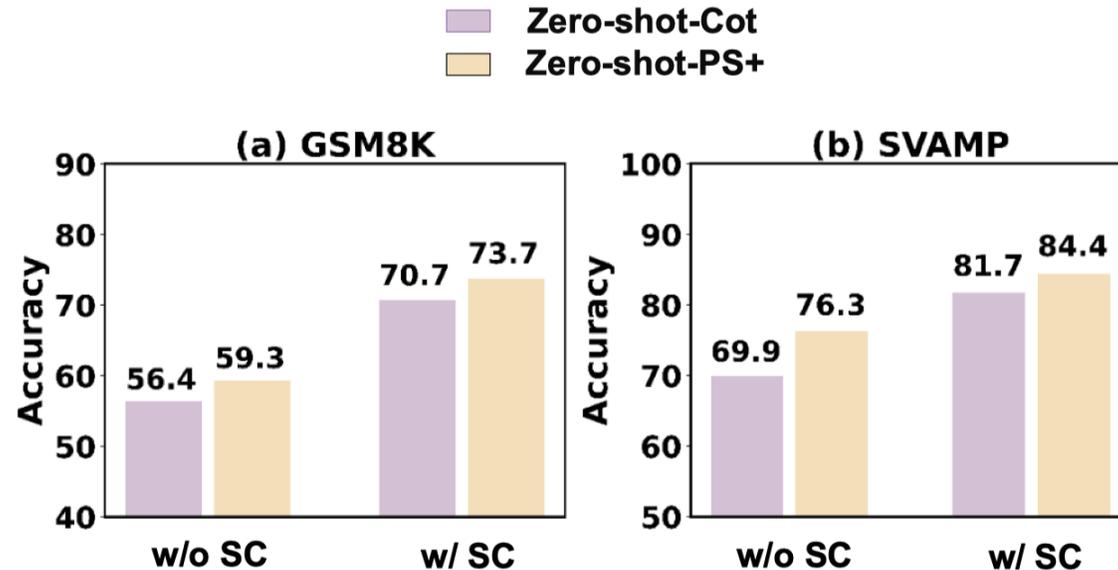
Method	CSQA	StrategyQA
Few-Shot-CoT (Manual)	78.3	71.2
Zero-shot-CoT	65.2	63.8
Zero-shot-PS+ (ours)	71.9	65.4

Table 4: Accuracy on symbolic reasoning datasets.

Method	Last Letter	Coin Flip
Few-Shot-CoT (Manual)	70.6	100.0
Zero-shot-CoT	64.8	96.8
Zero-shot-PS+ (ours)	75.2	99.6

Experiments

- Prompting with Self-Consistency



Self-Consistency (Wang et al., 2022): LLM's output의 randomness 완화를 위해, N reasoning results 생성하고 majority. Voting으로 final answer 결정하는 방법론 (2023 ICLR)

Figure 4: Results of methods with and without self-consistency (SC) on GSM8K and SVAMP.

Experiments

• Effect of Prompts

Table 5: Performance comparison of trigger sentences measured on GSM8K and SVAMP datasets with text-davinci-003 except for No. 2 (code-davinci-002). (*1) means the trigger sentence used in Zero-shot-CoT (Kojima et al., 2022). (*2) means the trigger sentence used in Zero-shot-PoT (Chen et al., 2022).

No.	Trigger Sentence		GSM8K	SVAMP	
1	Let's think step by step.	(*1)	56.4	69.9	→ Zero-Shot-CoT
2	<pre>import math import numpy as np # Question: example['question'] # Answer this question by implementing a solver() function. def solver(): # Let's write a Python program step by step, and then return the answer # Firstly, we need define the following variable:</pre>	(*2)	57.0	70.8	→ Zero-Shot-PoT
3	Extract variables and assign their corresponding numerals to these variables first and then solve the problem step by step.		50.5	69.5	→ PS
4	Firstly, extract variables and their corresponding numerals. Then, calculate intermediate variables. Finally, solve the problem step by step.		54.8	70.8	
5	Let's first understand the problem and devise a plan to solve the problem. Then, let's carry out the plan and solve the problem step by step.		58.2	72.0	→ PS+ triggers
6	Let's first understand the problem, extract relevant variables and their corresponding numerals, and make a plan. Then, let's carry out the plan, calculate intermediate variables (pay attention to correct numerical calculation and commonsense), solve the problem step by step, and show the answer.		59.3	75.7	→ PS+

Verify-and-Edit: A Knowledge-Enhanced Chain-of-Thought Framework

Ruochen Zhao ^{1*} Xingxuan Li ^{1,2*†} Shafiq Joty ^{1,3‡} Chengwei Qin ¹ Lidong Bing ²

¹ Nanyang Technological University, Singapore

² DAMO Academy, Alibaba Group

³ Salesforce AI

{ruochen002, chengwei003}@e.ntu.edu.sg

{xingxuan.li, l.bing}@alibaba-inc.com

srjoty@ntu.edu.sg

2023 LLM Paper seminar

2023 ACL

Introduction

- CoT prompting in Knowledge-intensive tasks
 - CoT prompting improves complex reasoning capabilities of LLMs by generating interpretable reasoning chains, but still suffers from factuality concerns in knowledge-intensive tasks such as open-domain question-answering.
 - As a major use case of LLMs is the prospect of replacing traditional search engines and usage for more direct information access through question answering, factuality concerns could largely undermine their validity and degrade users' level of trust (Marcus, 2022).
 - As LLMs could fail to recall accurate details when functioning as a knowledge base (Ye and Durrett, 2022; Creswell et al., 2022)
- ➔ Human process? They often search (or revisit) external knowledge sources for supporting facts in order to refresh their memory.

Introduction

- **Verify-and-Edit Framework (VE)**

- VE Framework to post-edit the reasoning chains for more factually aligned predictions

1. Find uncertain predictions
2. Edit their rationales by searching for supporting facts
3. Generate final answer based on edited rationales

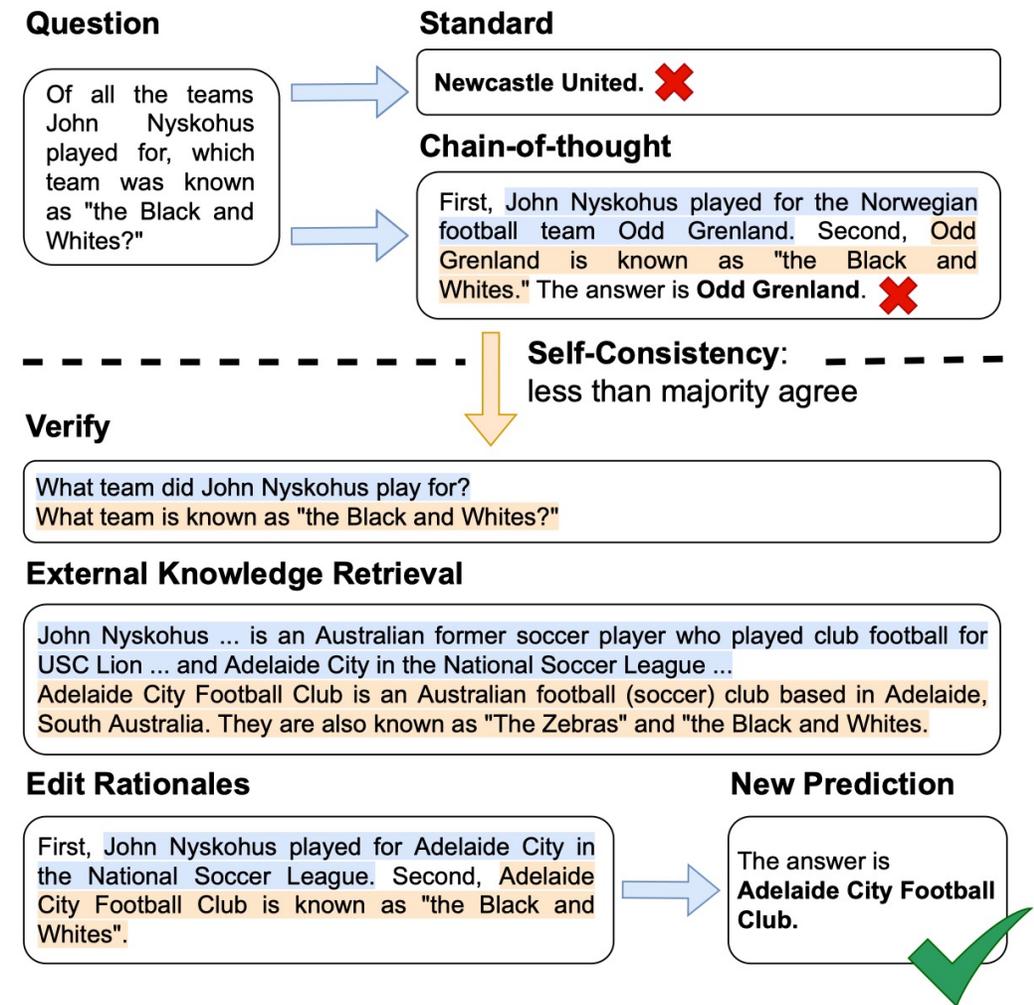


Figure 1: The Verify-and-Edit framework consists of five steps: (1) pass predictions with lower-than-average consistency to the next stages while leaving highly consistent predictions as-is; (2) produce verifying questions; (3) retrieve external knowledge; (4) edit rationales with informed answers; and (5) produce new predictions.

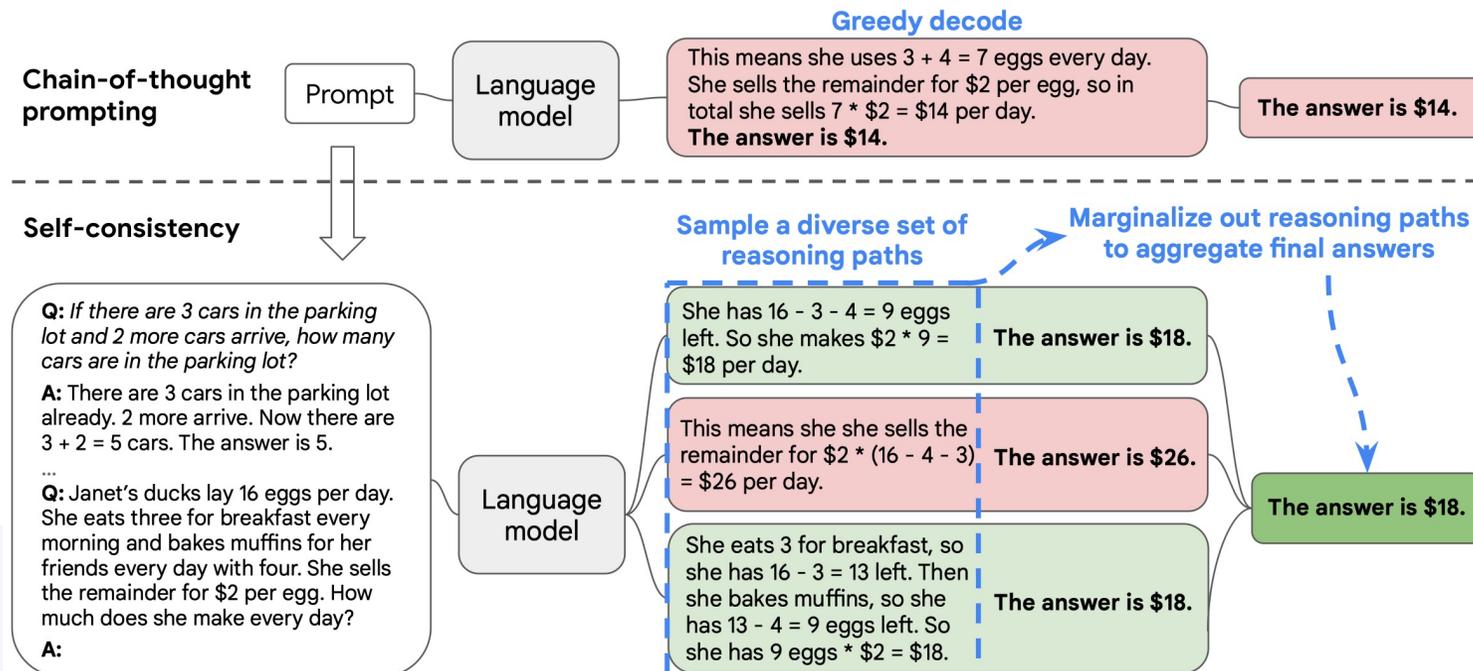
Introduction

- **Verify-and-Edit Framework (VE)**
 - 3.8x improvement compared to retrieval-augmented models on AdvHotpot.
 - On 2WikiMultiHop, VE reaches 33.6% accuracy with open-domain search, while CoT stands at 27.7%

Method

- Deciding when to edit

- Self-Consistency* (Wang et al., 2022): A high correlation between self-consistency with accuracy \rightarrow it could provide an uncertainty estimation for the model.
- A high consistency predictions are left as-is. When consistency is lower than $\lceil n/2 \rceil$, label it as “uncertain”



Method

- **How to edit a specific rationale?**
 - To verify the rationale, generate verifying question using the original question and the rationale.
 - Search for supporting facts in external knowledge sources (*e.g., Wikipedia, Google*) based on the verifying question.
 - Generate verifying answer for the verifying question based on the supporting facts.
 - Edit original rationale with the verifying answer.

Method

• Algorithm for Verify-and-Edit

Algorithm 1 Verify-and-Edit

Require: The original question q ; An n -shot CoT prompt p_{cot}

Require: An LLM $f(\cdot)$; LM number of completions n ; LM decoding temperature τ

Require: An external knowledge retrieval model $g(\cdot)$

Require: n -shot prompts for verifying question generation (p_{vq}) and answer generation (p_{va})

```
 $R, A \leftarrow f(p_{cot}, q, n, \tau)$                                 ▷ Generate a set of reasonings (R) and answers (A).  
 $s_{sc}^* \leftarrow \max P(a|p_{cot}, q), a \in A$                 ▷ The highest self-consistency score among all answers.  
 $r^*, a^* \leftarrow \arg \max P(a|p_{cot}, q), a \in A$         ▷ Reasoning and answer with highest self-consistency.  
if  $s_{sc}^* < \lceil \frac{n}{2} \rceil$  then                                ▷ Edit reasoning with a less-than-majority-agree consistency.  
  for  $o_i \in r^*$  do                                       ▷ Edit each sentence in the reasoning.  
     $u \leftarrow f(p_{vq}, q, o_i)$                                ▷ Generate verifying question.  
     $v \leftarrow g(u)$                                          ▷ Retrieve external knowledge.  
     $w \leftarrow f(p_{va}, u, v)$                                ▷ Generate verifying answer.  
     $o_i \leftarrow w$                                          ▷ Edit original reasoning sentence with verifying answer.  
  end for  
   $a^* \leftarrow f(p_{cot}, q, r^*)$                                ▷ Generate final answer with edited reasoning.  
  return  $a^*$   
else if  $s_{sc}^* \geq \lceil \frac{n}{2} \rceil$  then                        ▷ Answer with high consistency is left as-is.  
  return  $a^*$   
end if
```

Experiments

- **Benchmark datasets**

- **Adversarial HotpotQA**: a multi-hop question answering dataset (the challenging subset proposed by Ye and Durrett (2022))
- **2WikiMultihop**: a multi-hop question answering dataset exploiting the structured format in Wikidata
- **Fever**: a fact verification dataset

Experiments

- **Baselines**

- **Standard Prediction (Standard)**: directly predict using LLM
- **CoT**: Predicting the label after generating the explanation
- **CoT with Self-Consistency (CoT-SC)**: Sampling 5 CoT reasoning paths with a decoding temperature of 0.7
- **Calibrator (Calib.)**: A calibrator that tunes the probabilities of a prediction based on the score of its prediction
- **ReAct**: A reason-and-act framework that utilizes an external Wikipedia API. It uses the PaLM model (Chowdhery et al., 2022), whose performance is similar to GPT-3.

Experiments

- Knowledge retrieval systems

- **Wikipedia-API (Wiki)**: Searching for the query entities and selecting top sentences from their Wikipedia pages
- **DrQA**: A pre-trained open-domain QA model that combines bigram hashing, TF-IDF matching, and a multi-layer recurrent networks (only utilize retriever from it)
- **Google**: Using top-k search results produced by Google as assistive contexts
- **Dataset**: Selecting from the set of paragraphs provided each dataset
→ Since this includes gold supporting context with distractor paragraphs, this is similar to an oracle setup (upper bound of knowledge retrieval system)

Experiments

- Using Self-Consistency: know when it doesn't know
 - Low consistency \rightarrow more uncertain by using Self-consistency \rightarrow Right?
 - To test this, AdvHotpoQA dataset의 consistency distribution을 plotting
 - Incorrect samples \rightarrow left-skewed (low consistency)
 - Correct samples \rightarrow right-skewed (higher consistency)
 - This effectively validates the hypothesis

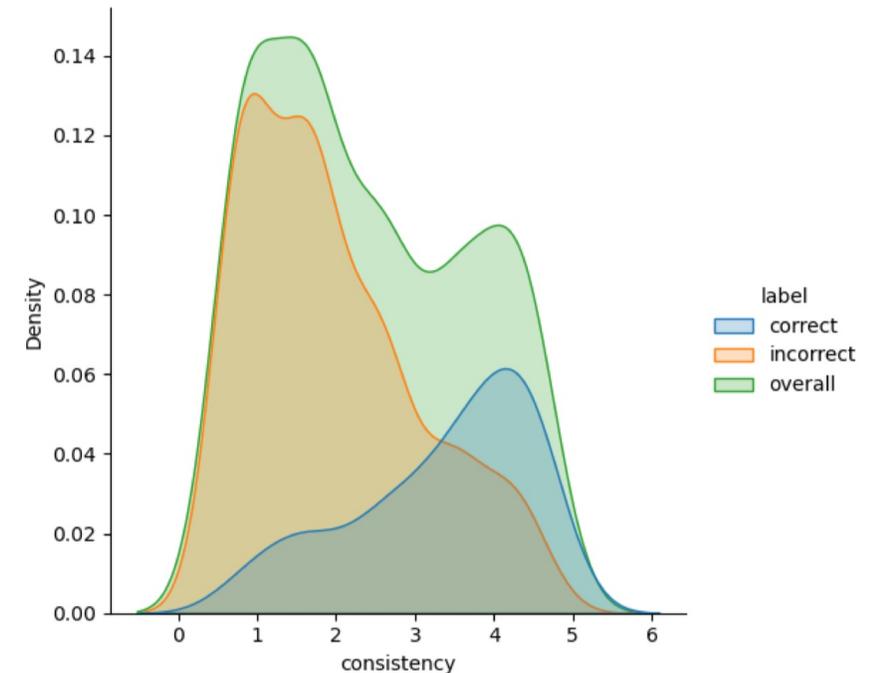


Figure 2: Kernel density estimation plots for consistency on the Adversarial **HotpotQA** dataset. With kernel estimation, the curve extends its true distribution's range, which is from 0 to 5 (as we sampled 5 paths).

Experiments

Prompt Method ^a	HotpotQA (EM)
Standard	28.7
CoT (Wei et al., 2022)	<u>30.4</u>
CoT-SC (Wang et al., 2022a)	33.4

• Experimental Results

- ReAct → CoT-SC: ReAct로 먼저 answer, 실패하면 CoT-SC
- CoT-SC → ReAct: all consistencies가 $n/2$ 보다 낮으면 ReAct로 answer

Method	knowledge	EM	Δ EM	AUC
CoT-SC → ReAct	Wiki.	34.2%	+0.8%	-
ReAct → CoT-SC	Wiki.	35.1%	<u>+1.7%</u>	-
Standard	-	23.1%	-	43.24
CoT	-	31.8%	-	38.30
CoT-SC	-	31.2%	-	34.97
CoT-SC + Calib.	Dataset	-	-	<u>49.00</u>
CoT-SC + VE	Wiki.	35.7%	+4.5%	45.62
CoT-SC + VE	DRQA	36.0%	+4.8%	46.06
CoT-SC + VE	Google	<u>37.7%</u>	<u>+6.5%</u>	47.98
CoT-SC + VE	Dataset	56.8%	+25.6%	60.94

Table 1: Results on the Adversarial **HotpotQA** dataset. The best result for each model is underlined and the best result overall is bolded. Δ EM represents the improvement on Exact Match from the CoT-SC baseline. The top two rows uses the PaLM model and the rest uses the GPT-3 davinci-003 model.

Experiments

- Experimental Results

Method	knowledge	EM	Δ EM	AUC
Standard	-	16.9%	-	35.89
CoT	-	28.4%	-	16.64
CoT-SC	-	27.7%	-	17.16
CoT-SC + Calib.	Dataset	-	-	24.13
CoT-SC + VE	Wiki.	33.1%	+5.4%	28.32
CoT-SC + VE	DRQA	31.1%	+3.4%	27.75
CoT-SC + VE	Google	33.6%	+5.9%	30.06
CoT-SC + VE	Dataset	37.2%	+9.5%	32.28

Table 2: Results on **2WikiMultiHopQA** dataset. Δ EM represents the improvement on Exact Match from the CoT-SC baseline. All experiment uses the GPT-3 davinci-003 model.

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 \rightarrow ReAct	34.2	64.6
ReAct \rightarrow CoT-SC	35.1	62.0

Method	knowledge	Accuracy	Δ Accuracy
CoT-SC \rightarrow ReAct	Wiki.	-	+4.2%
ReAct \rightarrow CoT-SC	Wiki.	-	+1.6%
Standard	-	46.8%	-
CoT	-	50.0%	-
CoT-SC	-	52.0%	-
CoT-SC + Calib.	-	33.7%	-
CoT-SC + VE	Wiki.	53.6%	+1.6%
CoT-SC + VE	DRQA	53.3%	+1.3%
CoT-SC + VE	Google	53.9%	+1.9%

Table 3: Results on **Fever** dataset. Δ Accuracy represents the improvement on Accuracy from the CoT-SC baseline. The top two rows uses the PaLM model and the rest uses the GPT-3 davinci-003 model.

Conclusion

- 다양한 CoT 기반의 연구가 활발하게 진행되고 있고, 특히 Verify-and-Edit (2023 ACL)이나 ReAct (2023 ICLR), Selection-Inference (2023 ICLR) 같은 연구에서 CoT 에 Knowledge 를 결합하려는 시도가 보임
- Arithmetic이나 reasoning 같은 Task 혹은 HotpotQA나 WikiMultiHopQA 같은 Open-domain Multi-hop QA 에서만 연구가 진행됨
- 뭔가 접근해보기 쉬운 분야라고 생각이 들면서, 동시에 ChatGPT나 PaLM 처럼 추론 능력이 충분히 좋은 모델에 한정된 연구생각도 드는데, 흠..

Thank you!