Tab-CoT: Zero-shot Tabular Chain of Thought

Ziqi Jin and **Wei Lu**
StatNLP Research Group
Singapore University of Technology and Design
ziqi_jin@sutd.edu.sg, luwei@sutd.edu.sg

Abstract

The chain-of-though (CoT) prompting methods were successful in various natural language processing (NLP) tasks thanks to their ability to unveil the underlying complex reasoning processes. Such reasoning processes typically exhibit implicitly structured steps. Recent efforts also started investigating methods to encourage more explicitly structured reasoning procedures to be captured (Zhou et al., 2022). In this work, we propose Tab-CoT, a novel tabular-format CoT prompting method, which allows the complex reasoning process to be explicitly modelled in a highly structured manner. Despite its simplicity, we show that our approach is capable of performing reasoning across multiple dimensions (i.e., both rows and columns). We demonstrate our approach's strong zero-shot and few-shot capabilities through extensive experiments on a range of reasoning tasks.[1]

1 Introduction

The chain-of-thought (CoT) prompting method (Wei et al., 2022) encourages the large language models (LLMs) to engage in a thought process before providing the answer to the given question. Such an approach shows impressive performance improvements in reasoning tasks. Notably, in the zero-shot setting, it was shown that a simple prompt such as "`let's think step by step`" could facilitate the step-by-step thinking process before answering the original question (Kojima et al., 2022). Such a task-agnostic method unveiled that LLMs can be descent zero-shot reasoners.

The reasoning process is inherently structured. This gives rise to some new developments along this line of work recently. Specifically, Zhou et al. (2022) suggests an alternative prompting approach that enables a two-stage structured reasoning process. Gao et al. (2022) proposes an approach that involves code in the prompt design, allowing structured information in the form of formal language to participate in the reasoning process. While effective, such methods require specific prompt engineering for different domains or defining multiple variables, which can be difficult to maintain or keep track of.

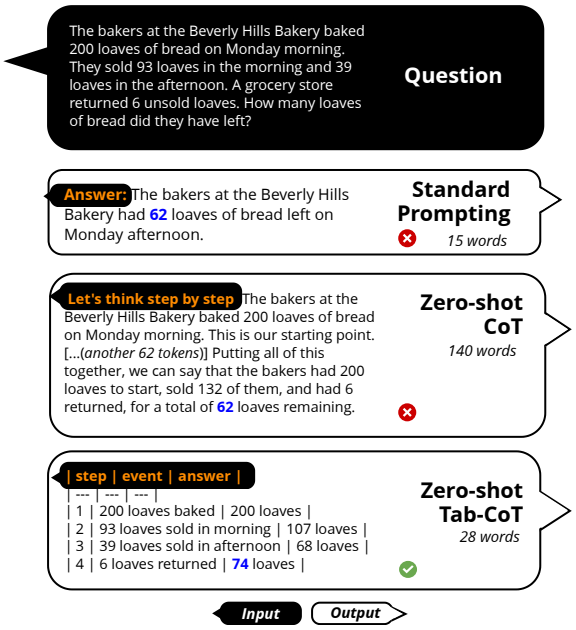

Figure 1: A comparison between Tab-CoT with standard prompting and zero-shot-CoT on the same question. Chain-of-thought prompts are highlighted in orange.

Inspired by the fact that state-of-the-art large language models, such as GPT-3 (Brown et al., 2020) and CodeX (Chen et al., 2021), have the capability of reasoning over tabular structured data (He et al., 2023)[2], we propose a novel framework called *Tabular Chain of Thought* (Tab-CoT) that models the structured reasoning process using a table-filling procedure.

We show that the model can perform step-by-step reasoning by creating a table

[1] Our code is available at https://github.com/Xalp/Tab-CoT

[2] This is because such models are trained on massive data collected from the Internet, which contains a large amount of tabular formed data.

Findings of the Association for Computational Linguistics: ACL 2023, pages 10259–10277

without further fine-tuning by using a table header with column names in the form of "`|step|question|response|`" as a prompt. While conventional natural language texts are generated in a 1-dimensional sequential order, the table has a 2-dimensional structure, allowing inference along both columns and rows to be performed simultaneously. Unlike previous works which focused on extracting information from existing tabular structured data (Gong et al., 2020, He et al., 2023), our approach generates the table while performing the reasoning process (and extracts the answer from the generated table at the end).

Figure 1 shows the results with standard prompting, conventional zero-shot CoT, and our zero-shot Tab-CoT. Our method generates a table as the output, which is more organized and concise than the output from the conventional CoT method. In this example, while zero-shot CoT generates 140 words, our method only generates 28. Besides, we found our method can reason horizontally and vertically at the same time.[3] This demonstrates that our Tab-CoT method benefits from the 2-dimensional structure of the table, where the information can flow in two dimensions.

We summarize our main contributions in this work as follows:

- We propose a new approach called Tabular Chain-of-Thought (Tab-CoT) that utilizes a tabular structured reasoning scheme in combination with state-of-the-art large language models to generate answers. To the best of our knowledge, this is the first method that uses tables in a "chain of thought" process.
- The 2-dimensional tabular structure of Tab-CoT allows for improved unlocking of the step-by-step reasoning capabilities of LLMs, transforming the linear "chain of thought" process into a more structured one.
- Extensive experiments have revealed that our Tab-CoT outperforms traditional CoT techniques in zero and few-shot settings. This indicates that Tab-CoT has strong potential as a superior alternative to current chain-of-thought prompting methods.

[3] "`74 loaves`" is the sum of "`68 loaves`" from the same row and "`6 loaves`" from the same column.

2 Related Work

Chain-of-thought prompting (Wei et al., 2022), a variation of few-shot prompting that adds step-by-step reasoning in those few-shot examples instead of just providing answers, has achieved significant improvements across multiple datasets. The LLMs can generate solutions following the solution format of prompts. Compared to traditional prompting, chain-of-thought prompting decomposes the task into smaller steps, which makes difficult tasks easier to solve.

The chain-of-thought prompting method is not necessarily purely natural language based. Program Aided Language Models (PAL) (Gao et al., 2022) provides few-shot samples that contain executable Python code. Such an approach enables the LLMs to interact with the Python shell, allowing the model to focus on learning how to do mathematical reasoning rather than numerical calculations.

These chain-of-thought methods provide the solution structure and pattern via few-shot samples, but can these be provided without these few-shot samples in the zero-shot setting? Zero-shot CoT (Kojima et al., 2022) is a zero-shot chain-of-thought prompting method. The prompt phrase "`Let's think step by step`" added after the question triggers the explicit reasoning process. However, compared to few-shot CoT (Wei et al., 2022), zero-shot CoT allows more flexibility in the structure of the reasoning process.

Recently, Zhou et al. (2022) proposed Least-to-Most prompting, which is a prompting strategy that reduces a complex problem into a list of sub-questions, and sequentially solves the sub-questions. Each sub-question is solved with the answer to previously solved sub-questions. Compared to zero-shot CoT, this method has more restrictions on the structure of reasoning by decomposing and sequentially answering. Moreover, importing external tools (like calculator and python shell) can further aid the math computation within the arithmetic domain (Gao et al., 2022).

These works reveal the importance of promoting structures in the chain-of-thought process. However, the nature of the zero-shot prompting makes the injection of structures into the generation process challenging. This motivates us to devise a better mechanism to prompt the language models under the zero-shot setting – a new prompting scheme that allows highly structured outputs in the form of tables to be generated.

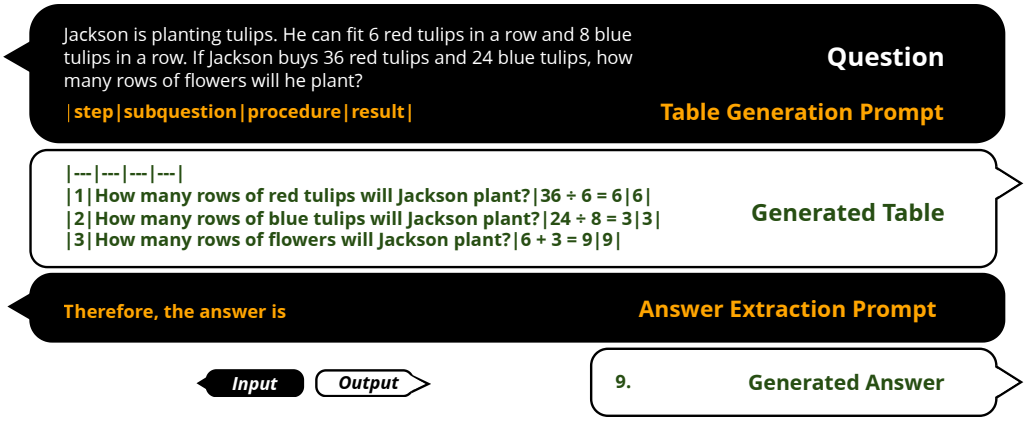

Figure 2: Overview of our zero-shot Tab-CoT method, which contains two steps: (1) table generation and (2) answer extraction. Added prompts are highlighted in orange. Texts generated by the LLM are highlighted in green.

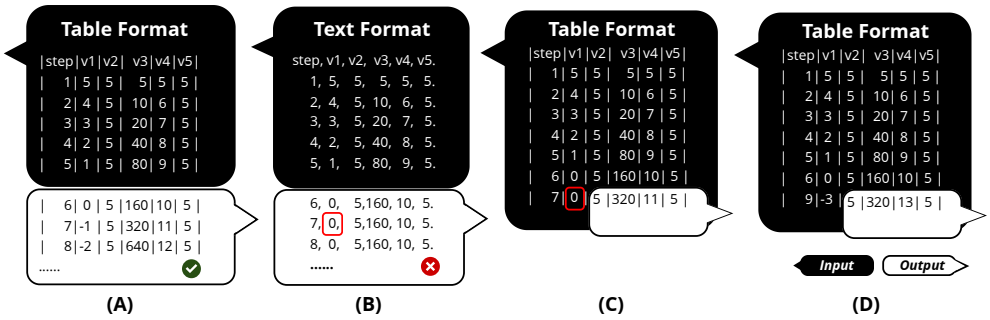

Figure 3: Understanding how state-of-the-art LLM ("code-davinci-002") reason with tabular-structured data.

3 Tab-CoT

Similar to zero-shot CoT (Kojima et al., 2022), our method involves two prompts that can be used in large language models (LLMs), one for table generation and the other for answer extraction. The details are shown in Figure 2. While our method is primarily applied in zero-shot settings, it can also work in few-shot settings.

Tables in LLMs We found that in the official "parse unstructured data" demo provided by OpenAI[4], a table header is provided as part of the prompt, which is as follows: "`|Fruit|Color|Flavor|`". With such a prompt, the underlying LLM can automatically generate a table. This suggests possible formatting for tables in such state-of-the-art LLMs. And "|" is the recognizable delimiter of tables in OpenAI models.

To validate this observation, we queried the LLM "code-davinci-002" (Chen et al., 2021) with the following question: "`a=2, b=3, what is 2*a+3*b?`", and provided another table header: "`|step|solution|`"[5]. We found that it completes a structured table as follows:

```
a=2, b=3, what is 2*a+3*b?
|step|solution|
|:---|:---|
|1|2*a+3*b|
|2|2*2+3*3|
|3|4+9|
|4|13|
```

This experiment essentially unveils how the tables are represented in such LLMs. The results also illustrate how the table can potentially be used for generating a reasoning process. Next, to validate this, we designed several simple experiments to understand how reasoning over such tabular-structured data is performed on such LLMs, as shown in Figure 3. Our first experiment (A) shows that such LLMs are able to perform potential vertical reasoning. However, if we replace '|' with ',' (B), the LLM fails to capture the patterns in the data. This tells us that the correct formatting is crucial when reasoning with tables in such LLMs.

Next, we intentionally insert a mistake into the partial table and ask the model to continue the generation process (circled in C). Surprisingly, the LLM is able to generate the correct entries even though the mistake occurred in the same row. This further confirms the LLM's strong potential in per-

[4] https://beta.openai.com/playground/p/default-parse-data

[5] The temperature is set to 0 for reproducibility.

forming vertical reasoning with tabular-structured data.

Moreover, to prove both vertical and horizontal reasoning exists, we increase the difficulty by directly appending the first two elements from step 9 after step 6 (D). If only vertical reasoning existed, the value under "v4" would have been "11". Instead, the value generated is "13," confirming that the LLMs have the potential to perform a combination of horizontal and vertical reasoning simultaneously.

Table Generation Prompt To make use of the 2-dimensional structure of the table, we replace the natural language prompt with a table-generation prompt (e.g., "|step|question|response|"), which serves as the header of the table. This regulates the context of this table, forcing the LLMs to conduct step-by-step reasoning by completing the table. Meanwhile, the choice of columns can be very specific. If each row of the table is regarded as a step, the row-by-row table generation process will become a step-by-step reasoning process. Within each step (row), we have multiple columns, each of which contributes certain detail towards the current reasoning step.

For any text question x, we have a table generation prompt (all column names) c. Concretely, we add the table generation prompt in the next row of the text question:

$$\mathsf{LLM}(x, c) = \begin{bmatrix} c_1 & c_2 & \cdots & c_{n-1} & c_n \\ t_{1,1} & t_{1,2} & \cdots & t_{1,n-1} & t_{1,n} \\ \vdots & & \ddots & & \vdots \\ t_{m,1} & t_{m,2} & \cdots & t_{m,n-1} & t_{m,n} \end{bmatrix} \quad (1)$$

where $t_{1,1} \cdots t_{m,n}$ are the entries within the generated table, which contains m rows and n columns.

Answer Extraction Prompt After the table content, denoted as T, is generated from the previous step, we perform answer extraction. The answer extraction step helps us to extract the answer from the table, as the final results may not always be in the last cell of the generated table. Following zero-shot CoT (Kojima et al., 2022), we add another answer extraction prompt a: "the answer is" after the generated table, to extract the final answer from the table:

$$Answer = \mathsf{LLM}(x, c, T, a) \quad (2)$$

Structure-Promoting Table Scheme Different table generation prompts (headers) may result in different tables generated (with different content). We propose a "structure-promoting scheme", which maximally unlocks the reasoning abilities of LLMs.

	Reasoning Type	Dataset	Size	Answer Type
Main	Arithmetic	SingleEq	508	Numeral
		AddSub	395	Numeral
		MultiArith	600	Numeral
		GSM8K	1,319	Numeral
		AQUA	254	Multiple Choice
		SVAMP	1,000	Numeral
Additional	Symbolic	Coin Flip	1,000	Yes or No
		Last Letter	254	String
	Commonsense	StrategyQA	2,290	Yes or No
		CommonsenseQA	1,221	Multiple Choice

Table 1: Tasks and Data

We define each row as a reasoning step. A table containing multiple rows will depict the step-by-step reasoning procedure leading to the final answer. Thus, our first column is "step", containing a number that indicates which reasoning step the current row represents.

Least-to-most prompting (Zhou et al., 2022) contains two stages: problem reduction and sequential solving. In problem reduction, they decompose a question into multiple subquestions. Similarly, we add "subquestion" as our second column. At the beginning of each step, the LLMs will generate a subquestion under this column, which demonstrates the objective of the current reasoning step.

The conventional zero-shot CoT (Kojima et al., 2022) shows that allowing the model to generate some reasoning process before answering can achieve a better result. Inspired by this observation, we add a third column, "process", into our table. Given a subquestion in the previous column, we expect to generate the reasoning process in the current column before answering.

The last column is named "answer". As the previous reasoning process under the "process" column may not necessarily provide an answer, we hope to use the "answer" column to explicitly request an (intermediate) answer at the end of each reasoning step.

With the above considerations, our primary scheme for the table header is designed as follows, which serves as our main table generation prompt:

|step|subquestion|process|result|

4 Experimental Setup

Large Language Models We consider two state-of-the-art large language models under the GPT-

	Method	CoT Prompt	LLM	SingleEq	AddSub	MultiArith	GSM8K	AQUA	SVAMP	Average
Zero-shot	Standard Prompting	—	text	74.6	72.2	17.7	10.4	22.4	58.8	42.7
			code	46.3	51.4	7.2	4.1	23.6	29.5	27.0
	CoT	`Let's think step by step`	text	78.0	69.6	78.7	40.7	33.5	**62.1**	60.4
			code	65.6	65.6	64.8	31.8	29.5	39.9	49.5
	Tab-CoT	`|step|subquestion|process|result|`	text	74.6	**71.9**	72.2	39.3	36.6	57.0	58.6
			code	**81.9**	70.9	**81.2**	**44.4**	**37.0**	60.5	**62.6**

Table 2: Zero-shot results on the arithmetic datasets. All methods use the same answer extraction prompt in these datasets for a fair comparison. All methods are evaluated under the zero-shot setting.

3 family (Brown et al., 2020) in our experiments, namely "code-davinci-002" and "text-davinci-002", whose APIs are made available by OpenAI[6]. For brevity, we use "code" to refer to the model "code-davinci-002" and "text" to refer to "text-davinci-002" in our experiments.

Tasks and Datasets We primarily focus on mathematical reasoning in this work. Thus, we evaluate our method on 6 arithmetic reasoning datasets. Specifically, they are SingleEq (Koncel-Kedziorski et al., 2015), AddSub (Hosseini et al., 2014), MultiArith (Roy and Roth, 2015), GSM8K (Cobbe et al., 2021), AQUA-RAT (Ling et al., 2017), and SVAMP (Patel et al., 2021), which are standard datasets widely used in the community.

We also conducted additional experiments on datasets involving other types of reasoning tasks. Specifically, we evaluate our method on two symbolic reasoning tasks: Last letter and Coin Flip[7]: the former is the task that asks for the concatenation of the last letters of 4 words, and the latter asks for the state of the coin after being flipped a few times. We investigate how the specificity of column names affects the performance and report in our ablation study. We also evaluate our method on two commonsense reasoning tasks: CommonsenseQA (Talmor et al., 2019) and StrategyQA (Geva et al., 2021).

Following zero-shot CoT (Kojima et al., 2022), we set the first generated number as the numeral answer, the first capitalized letter as the answer for multiple-choice questions, and the first "yes" or "no" as the answer for "Yes or No" questions.

5 Results

5.1 Main Results

Our main experiments are conducted on arithmetic reasoning tasks under the zero-shot setting. We tested the performance of both text-based and code-based LLMs on all methods. The results are shown in Table 2. Under the scheme "`|step|subquestion|process|result|`", our zero-shot Tab-CoT approach significantly outperformed the standard prompting in all tasks. Furthermore, our best-performing Tab-CoT model (using code-based LLM) outperforms the best conventional CoT model in 5 out of 6 tasks (with an average improvement of 2.2%).

When the standard prompting method is considered, using the text-based LLM leads to significantly better results than the code-based counterpart (15.7% on average). Similarly, when zero-shot CoT is considered, using the former also outperforms the latter by 10.9% on average. However, for our Tab-CoT approach, "code" outperforms "text" by 4.0%, leading to the best overall performance among all configurations.

From such results, we can see that the conventional CoT method responds differently from our Tab-CoT method with different types of underlying LLMs involved. The conventional CoT method (and the standard prompting method) strongly favors a text-based LLM under the zero-shot setting. In contrast, our approach works well with both types of LLMs, but the code-based version can give it an additional boost in performance. Compared with "text", the "code" model is further fine-tuned on code (Chen et al., 2021). We conjecture that table generation resembles the code generation process – both involve structured procedures that are highly organized and follow a step-by-step process. Comparing our Tab-CoT approach with conventional CoT, we can conclude that our proposed table-generation prompt is able to significantly better unlock the strong reasoning abilities within the code-based LLM.

Based on the above main experiments, we choose to use "code" as the default LLM for all subsequent experiments unless otherwise specified.

5.2 Importance of Scheme Design

To understand the significance of our proposed table scheme design, we evaluate the performance of

[6] `https://openai.com/api/`

[7] We use the file generated by Kojima et al. (2022).

	Scheme	SingleEq	AddSub	MultiArith	GSM8K	AQUA	SVAMP	Average
Zero-shot	Standard Prompting	46.3	51.4	7.2	4.1	23.6	29.5	27.0
	|step|subquestion|process|result|	81.9	70.9	81.2	44.4	37.0	60.5	62.6
	|step|subquestion|procedure|result|	83.7	69.1	77.8	43.4	38.2	60.4	62.1
	|step|question|response|	77.6	73.9	79.0	38.1	34.3	63.9	61.1
	Self-consistency (using above)	**86.4**	**78.2**	**85.2**	**48.2**	**44.1**	**66.9**	**68.2**

Table 3: Zero-shot performance comparison between the three schemes (and with self-consistency).

	Scheme	Average
Zero-shot	|subquestion|process|result|	54.3
	|step|process|result|	57.2
	|step|subquestion|result|	61.3
	|step|subquestion|process|	60.9
	|step|subquestion|process|result|	**62.6**

Table 4: Performance if a column is removed from the scheme (detailed results are in Appendix A).

	Method	Standard Prompting	CoT	Tab-CoT
Few-shot	SingleEq	86.8	**93.1**	92.1
	AddSub	**90.9**	89.1	89.1
	MultiArith	44.0	96.2	**96.3**
	GSM8K	19.7	**63.1**	61.6
	AQUA	29.5	45.3	**46.9**
	SVAMP	69.9	76.4	**82.9**
	Average	68.2	77.2	**78.2**

Table 5: Few-shot results on the arithmetic datasets.

"|step|subquestion|process|result|", along with four variations, each of which is obtained by removing one of the four columns as ablation. The results in Table 4 show that each column of "|step|subquestion|process|result|" is crucial. From the result, we notice that removing the column "step" from our scheme results in the most significant performance drop. This implies although the step only contains a number indicating "which step this is", it organized the table in sequential order over rows. The column "subquestion" is also important. Removing "subquestion" from the scheme also shows an average performance drop of 5.4%. The "subquestion" column forms step-by-step instructions vertically, indicating the subquestion under consideration for each step. The "step" and "subquestion" columns play important roles in maintaining the structure of the table, building vertical connections across rows.

5.3 Effectiveness of Self-Consistency

The self-consistency (Wang et al., 2022) decoding strategy was shown to obtain better results by generating and exploring multiple, diverse reasoning paths. We also adopt a similar approach here. In the original self-consistency paper, up to 40 reasoning paths were considered. We show the feasibility of using only 3 paths in our work.[8] This is conveniently achieved by using 3 different prompts – we select another two table schemes besides the standard scheme. One is a highly similar prompt, which we expect to perform similarly well, and the other is less similar, which we expect to yield a worse performance (based on Sec 5.2). They are shown in Table 3. We then perform majority voting based on the outputs from these 3 prompts. Interestingly, although a prompt with worse performance is used in the voting process, the overall performance improves. This shows the benefits of integrating different table schemes for such tasks, which helps improve the overall robustness of the approach.

5.4 Few-shot Tab-CoT

Tab-CoT shows impressive reasoning ability under the zero-shot setting. It can generate a structured output in the form of a table that enables the chain-of-thought reasoning process without few-shot samples. Tables are capable chain-of-thought carriers, but can they also serve as good chain-of-thought teachers? To answer this question, we evaluated Tab-CoT under the few-shot setting.[9]

For a fair comparison, we use the same few-shot sample questions described in Wei et al. (2022) (listed in Appendix D). We use "|step|subquestion|process|result|" as the table scheme when representing few-shot samples. The results are reported in Table 5, our method outperformed few-shot CoT by 1% on average. While the performance difference between Tab-CoT and CoT on other datasets is below 2%, the performance difference on SVAMP is 6.5%. The large improvement on SVAMP is likely related to the selection of few-shot samples because Wei et al. (2022) select 8 sample questions from SVAMP for all arithmetic reasoning tasks except AQUA[10].

[8]The self-consistency decoding method did not show significant improvement when the number of reasoning paths is below 5 in their paper.

[9]We did not compare with least-to-most prompting (Zhou et al., 2022) as it requires task-specific supervision, it only evaluated on GSM8K and provide task-specific prompt for GSM8K in the paper.

[10]AQUA has its own few-shot sample questions as they are

Question 1:	*Gretchen has some coins. There are 30 more gold coins than silver coins. If she had 70 gold coins, how many coins did Gretchen have in total?* (GT: **110**)			
CoT:	Let's think step by step. If Gretchen had 70 gold coins, then she would have 30 silver coins (70 - 30 = 40). So Gretchen would have a total of **100** coins (70 + 30 = 100).			
Model	Step	Subquestion	Process	Result
text	1	How many silver coins does Gretchen have?	There are 30 more gold coins than silver coins. If she had 70 gold coins, how many coins did Gretchen have in total?	100 coins
	2	How many gold coins does Gretchen have?	There are 30 more gold coins than silver coins. If she had 70 gold coins, how many coins did Gretchen have in total?	**70** gold coins
		Error Analysis: The "text" model overwrites the question in the process, thus answering the wrong question.		
code	1	How many gold coins does Gretchen have?	30 more gold coins than silver coins	70 gold coins
	2	How many silver coins does Gretchen have?	30 more gold coins than silver coins	40 silver coins
	3	How many coins does Gretchen have in total?	70 gold coins + 40 silver coins	**110** coins
Question 2:	*Adam bought 2 boxes of chocolate candy and 5 boxes of caramel candy. If each box has 4 pieces inside it, how much candy did he have total?* (GT: **28**)			
CoT:	Adam bought 2 boxes of chocolate candy. That means he has 8 pieces of chocolate candy. Adam bought 5 boxes of caramel candy. That means he has 20 pieces of caramel candy. To find out how much candy Adam has in total, we add 8 pieces of chocolate candy and 20 pieces of caramel candy. Adam has **28** pieces of candy in total.			
Model	Step	Subquestion	Process	Result
text	1	What is the known information?	Adam bought 2 boxes of chocolate candy and 5 boxes of caramel candy. If each box has 4 pieces inside it, how much candy did he have total?	2 boxes of chocolate candy and 5 boxes of caramel candy, each with 4 pieces inside
	2	What is the unknown information?	How much candy did he have total?	—
	3	What information is not needed?	—	—
	4	What is the equation?	boxes of chocolate candy and 5 boxes of caramel candy, each with 4 pieces inside	2(4)+5(4)
	5	What is the answer?	—	**24** pieces of candy
		Error Analysis: The "text" model contains redundant steps in the table and wrong calculation.		
code	1	How many pieces of chocolate candy did Adam buy?	2 boxes x 4 pieces per box	8 pieces
	2	How many pieces of caramel candy did Adam buy?	5 boxes x 4 pieces per box	20 pieces
	3	How many pieces of candy did Adam buy?	8 pieces + 20 pieces	**28** pieces
Question 3:	*A pet store had 102 puppies. In one day they sold 21 of them and put the rest into cages with 9 in each cage. How many cages did they use?* (GT: **9**)			
CoT:	We know that the pet store had 102 puppies and that 21 of them were sold. That means we have 81 puppies left. Now, if 9 puppies fit in each cage, we can divide 81 by 9 to find out how many cages are needed.81 / 9 = 9. So, the pet store used **9** cages.			
Model	Step	Subquestion	Process	Result
text	1	How many cages are needed for 102 puppies?	102 / 9 = 11.3333	11
	2	How many cages are needed for 21 puppies?	21 / 9 = 2.3333	2
	3	How many cages are needed in total?	11 + 2 = 13	**13**
		Error Analysis: The "text" model has a wrong reasoning order, and should add up before dividing.		
code	1	How many puppies are left?	102 - 21 = 81	81
	2	How many puppies are in each cage?	9	9
	3	How many cages are needed?	81 / 9 = 9	**9**

Table 6: Case studies (on MultiArith) of the tables generated from "code-davinci-002"/"text-davinci-002". The results returned after applying the answer extraction prompts are in **bold**. Additional case studies are in Appendix C.

5.5 Case Studies

The main experimental results show that "code" under-performs "text" with conventional CoT but yields better results in our Tab-CoT. To understand this better, we conduct case studies to compare their generated tables in Table 6.

While "code" only generated short text snippets or formulas under "`process`", the words generated by "text" under the same column tend to form complete sentences whenever possible. As we mentioned earlier, "code" is an LLM that is further fine-tuned on code (Chen et al., 2021). This explains why it appears more amenable to the tabular-structured format of the output. In question 1, the model with "text" overwrites the generated "`subquestion`" by asking another question. Thus, the "`result`" fails to answer the "`subquestion`" in the same row. In question 2, "text" generated 5 steps while "code" only took 3. The "`subquestion`" generated by "text" is also ambiguous (e.g., "`what is the known information?`"). In question 3, "text" presents a wrong reasoning order. Overall, "code" shows better reasoning ability by demonstrating a more concise and straightforward reasoning process.

5.6 Additional Experiments

We further evaluate our methods on symbolic reasoning and commonsense reasoning tasks. We also conducted some new experiments based on the GPT-3.5 model to understand our approach's effectiveness on such newer models [11]. With such additional experiments, we hope to draw further insights into our approach.

Symbolic Reasoning We evaluate Tab-CoT on two symbolic reasoning datasets: Coin Flip (CF)[12] and Last Letter (LL)[13]. Unlike the arithmetic reasoning tasks, these tasks focus on some specific problems. This also opens up the opportunity for us to examine whether the specificity of the table

multiple choice questions. We use the same few-shot samples following Wei et al. (2022).

[11] GPT-3.5 is released on Mar 2023.

[12] An example for Coin Flip: "`A coin is heads up. Vinny does not flip the coin. Landon flips the coin. Miguel flips the coin. Caitlyn does not flip the coin. Is the coin still heads up? Note that "flip" here means "reverse".`"

[13] An example for Last Letter: "`Take the Last Letter of each word in "Vinny Landon Miguel Caitlyn" and concatenate them.`"

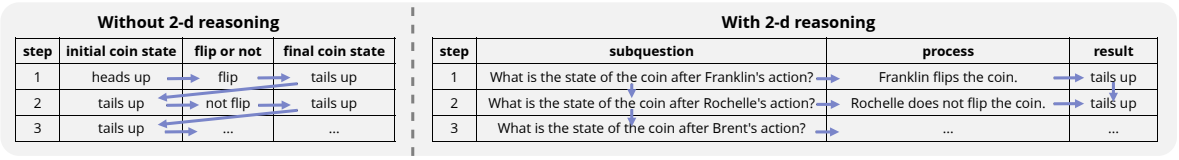

Figure 4: The schemes that disable (left) and enable (right) potential 2-dimensional reasoning.

	Task	Cat	Prompt	Result
Zero-shot	CF		`let's think step by step`	91.4
		1	`|step|subquestion|process|result|`	85.0
		2	`|step|initial state|action|next state|`	80.4
		3	`|step|name|flip or not|result|`	**96.2**
	LL		`let's think step by step`	57.6
		1	`|step|subquestion|process|result|`	25.2
		2	`|step|original answer|action|updated answer|`	50.8
		3	`|step|word|last letter|answer|`	**72.8**

Table 7: Effect of different specificity of schemes. We use Zero-shot CoT with the "text" model as our baseline (as Zero-shot CoT works better with "text" model).

	Method	CommonsenseQA	StrategyQA	Avg
Zero-shot	Standard Prompting	69.0	3.3	36.2
	CoT	54.6	38.9	46.8
	Tab-CoT	68.4	50.4	59.4

Table 8: Results on commonsense reasoning.

scheme may have an impact on the reasoning process in such tasks.

To this end, we split table schemes into three categories: (1) *general*: the table scheme that can be generally applied to most text questions. (2) *domain-specific*: the table scheme that can be adapted to a specific domain. (3) *task-specific*: the scheme that can only be adopted by a single task.

Our experiments in Table 7 illustrate that the specificity of the table schemes highly affects the performance of symbolic reasoning tasks. One may expect the performance to increase as the table scheme becomes more task-specific. Our task-specific scheme outperformed the zero-shot CoT in both tasks. However, the increased specificity does not always lead to higher accuracy. In the Coin Flip task, we noticed that another task-specific scheme "`|step|initial coin state|flip or not|next coin state|`" only achieves an accuracy of 68.0%. To understand this, we investigate their reasoning flows in Figure 4. Although the left scheme is more task-specific, it largely disabled the vertical reasoning in the table. While the right scheme is general, it effectively enables reasoning along both vertical and horizontal directions, leading to significantly better results. [14]

[14]We further evaluate the general scheme under the one-shot setting, and the results are in Appendix A

	Method	CoT	Tab-CoT
Zero-shot	SingleEq	85.6	**87.8**
	AddSub	83.3	**85.8**
	MultiArith	**90.5**	89.3
	GSM8K	68.7	**78.2**
	AQUA	50.8	**51.2**
	SVAMP	79.0	**81.1**
	Average	76.3	**78.9**

Table 9: Results with GPT-3.5.

	Task	code-cushman-001 (13B)	code-davinci-002 (175B)
Zero-shot	SingleEq	6.3	81.9
	AddSub	6.3	70.9
	MultiArith	2.0	81.2
	GSM8K	0.9	44.4
	AQUA	16.9	37.0
	SVAMP	5.0	60.5
	Average	6.2	62.6

Table 10: A comparison between the different sizes of "code", "Average" is the average score across six datasets.

Commonsense Reasoning As another set of additional experiments, we further evaluate our method on commonsense reasoning, including CommonsenseQA (Talmor et al., 2019) and StrategyQA (Geva et al., 2021). The results are in Table 8. Tab-CoT obtained the highest average accuracy. However, the results of our method did not show significantly improved performance compared with Standard Prompting in a few-shot setting[15]. These results imply that commonsense reasoning tasks do not have a fixed answering pattern. Therefore, providing chain-of-thought samples is not enough to make up for the lack of commonsense knowledge. For a fair comparison, we use the same few-shot questions listed in (Wei et al., 2022).

Results on GPT-3.5 We test our method on the recent model "GPT-3.5-turbo-0301" in Table 9[16]. We found that our method is applicable to GPT-3.5, and achieves better performance compared to conventional Zero-shot CoT. Another interesting observation is when prompting the GPT-3.5 model with "Let's think step by step", a large number of the generated texts already contain a table in their

[15]The few-shot results are in Appendix B.

[16]Our experiment is conducted in May 2023. The "GPT-3.5-turbo-0301" may be updated in the future.

	Scheme	SingleEq	AddSub	MultiArith	GSM8K	AQUA	SVAMP	Average
Zero-shot	Standard Prompting	46.3	51.4	7.2	4.1	23.6	29.5	27.0
	`|subquestion|process|result|`	69.9	51.9	84.0	40.1	35	44.7	54.3
	`|step|process|result|`	77.0	55.7	84.2	41.5	37.8	46.9	57.2
	`|step|subquestion|result|`	76.0	77.9	76.8	40.1	36.2	60.6	61.3
	`|step|subquestion|process|`	78.0	75.9	76.3	39.7	34.3	60.9	60.9
	`|step|subquestion|process|result|`	81.9	70.9	81.2	44.4	37.0	60.5	**62.6**

Table 11: Performance if a column is removed from the scheme.

CoT process. [17]

5.7 Ablation Studies

Model Sizes Kojima et al. (2022) evaluated the family of GPT-3 models of four different sizes: 2.7B, 6.7B, 13B, and 175B parameters. The results show that only the largest model ("text-davinci-002") shows the chain-of-thought reasoning ability.

We compare the performance of the smaller model "code-cushman-001" (13B) with "code-davinci-002" (175B). Similar to zero-shot CoT, smaller models do not show the ability to conduct chain-of-thought reasoning. The performance of "code-cushman-001" cannot reach 10%, except AQUA (a multiple choice dataset with 5 choices for each question). The experimental results are reported in Table 10.

Structure-Promoting Scheme As mentioned in Table 4, we compare the performance when we remove any column from "`|step|subquestion|process|result|`". The detailed experimental results are reported in Table 11. Results suggest that each column of our proposed scheme is important because removing any column will lead to a drop in performance.

6 Discussion

Our experimental results confirmed the effectiveness of our proposed tabular chain-of-thought method under both zero-shot and few-shot settings. We summarize several advantages of our method compared to conventional chain-of-thought methods and list them below.

Tab-CoT generates a table illustrating the reasoning process, which is more *organized*. This nature of the generated text, as can be seen from Table 6, makes the reasoning process much easier.

Additionally, from Figure 4, we conclude that Tab-CoT encourages a more *structured* reasoning process to be explicitly modelled. As a 2-dimensional data structure, tables enable both horizontal reasoning along rows and vertical reasoning along columns.

Practically, table schemes are also *easy to craft*. Designing a specific table generation prompt typically involves deciding concise header names without concerning grammar. It is thus less cumbersome than choosing a natural language prompt from a diverse set of candidates.

Overall, we argue that under current state-of-the-art LLMs, table schemes are *natural prompts* that are well suited for zero-shot learning.

7 Conclusion

In this paper, we propose Tab-CoT, a novel prompting framework that performs effective zero-shot reasoning by generating a table.

Tab-CoT shows competitive results on arithmetic reasoning tasks under both zero-shot and few-shot settings. We further conducted comprehensive experiments across different reasoning tasks under different settings. Our comprehensive experiments revealed some specific benefits of our method and identify the optimal way to use it. We hope that, through our work, we can sparkle new ideas and provide some inspiration to our community.

In the future, we would like to explore methods to automate the scheme selection process, using the generated schemes to meet task-specific requirements. Future work also includes integrating external calculators (Gao et al., 2022), or task-specific supervision (Zhou et al., 2022) into the learning process, under both zero-shot and few-shot settings.

Our Tab-CoT also provides a straightforward decomposition of the intermediate thought process. This highly structured chain of thought produced by our approach may help people to observe and interpret how large language models decompose complex problems. We believe our proposed method can help reveal the underlying mechanisms associated with the emergence of certain complex behaviours associated with large language models.

[17] Based on our observations, those tables generated in conventional Zero-shot CoT under GPT 3.5 can be different from those generated with our method. They appear to be mostly used to organize information related to the question but do not appear to be used for presenting reasoning steps.

Limitations

We identify a few limitations of this work. First, our approach is applicable to language models pretrained with tables, which may not always be included in all language models (especially small ones). Second, our approach's limited improvement in commonsense reasoning tasks suggests that its effectiveness may depend on the specific task and the level of structured reasoning required.

Acknowledgement

We would like to thank the anonymous reviewers, our meta-reviewer, and senior area chairs for their constructive comments and support on our work. This research/project is supported by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Program (AISG Award No: AISG2-RP-2020-016), and the Ministry of Education, Singapore, under its Tier 3 Programme (The Award No.: MOET32020-0004).

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*.

Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. TableGPT: Few-shot table-to-text generation with table structure reconstruction and content matching. In *Proceedings of COLING*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2023. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of EMNLP*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of ACL*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of NAACL*.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of EMNLP*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of NAACL*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A One-shot Reasoning on Symbolic Reasoning

We evaluate our method on Coin Flip (Table 21) and Last Letter (Table 22) under the one-shot setting. As shown in Table 13, by adding one few-shot sample, LLMs can gain a significant performance boost in both tasks with general scheme "`|step|subquestion|process|result|`".

B Additional Few-shot Results

We evaluate our method on commonsense reasoning tasks under a few-shot setting. Our model performs slightly better in terms of average accuracy. The results are reported in Table 12.

C Additional Case Studies

We show some errors our method made in arithmetic reasoning tasks through further case studies. The results are reported in Table 14 and 16.

D Few-Shot Samples

We list our few-shot samples for all arithmetic reasoning (Table 17 and Table 18), CommonsenseQA (Table 19) and StrategyQA (Table 20). We use the same few-shot sample questions from Wei et al. (2022)

	Method	CommonsenseQA	StrategyQA	Avg
Few-shot	Standard Prompting	82.3	67.1	74.7
	CoT	77.9	73.2	75.6
	Tab-CoT	80.8	72.0	76.4

Table 12: Few-shot results on commonsense reasoning.

	Task	Prompt	Result
Zero-shot	CF	`|step|subquestion|process|result|`	85.0
	LL	`|step|subquestion|process|result|`	25.2
One-shot	CF	`|step|subquestion|process|result|`	100.0
	LL	`|step|subquestion|process|result|`	96.0

Table 13: One-shot results on symbolic reasoning.

Case Study

Tom bought a skateboard for $ 9.46 , and spent $ 9.56 on marbles . Tom also spent $ 14.50 on shorts . In total , how much did Tom spend on toys ?

step	subproblem	process	result
—	—	—	—
1	How much did Tom spend on a skateboard?	$9.46	$9.46
2	How much did Tom spend on marbles?	$9.56	$9.56
3	How much did Tom spend on shorts?	$14.50	$14.50
4	How much did Tom spend on toys?	$9.46 + $9.56 + $14.50	$33.52	Therefore, the answer (arabic numerals) is $33.52. (GT: 19.02)

Error Analysis: Misleading by redundant information. Regard shorts as toys.

Tom found 15 seashells and Fred found 43 seashells on the beach . When they cleaned them , they discovered that 29 were cracked . How many seashells did they find together ?

step	subproblem	process	result
—	—	—	—
1	How many seashells did Tom find?	15	15
2	How many seashells did Fred find?	43	43
3	How many seashells were cracked?	29	29
4	How many seashells did they find together?	15 + 43	58
5	How many seashells did they keep?	58 - 29	29
Therefore, the answer (arabic numerals) is 29. (GT: 58)

Error Analysis: Misleading by redundant information. Cracked seashells are still seashells.

The grasshopper, the frog and the mouse had a jumping contest. The grasshopper jumped 19 inches. The grasshopper jumped 4 inches farther than the frog and the mouse jumped 44 inches lesser than the frog. How far did the frog jump?

step	subproblem	process	result
—	—	—	—
1	How far did the grasshopper jump?	19	19
2	How far did the grasshopper jump farther than the frog?	4	4
3	How far did the mouse jump lesser than the frog?	44	44
4	How far did the frog jump?	19 - 4	15
5	How far did the frog jump?	19 + 44	63
6	How far did the frog jump?	63 - 15	48
Therefore, the answer (arabic numerals) is 48. (GT: 15)

Error Analysis: Misleading by redundant information. The mouse does not matter.

Table 14: Few-shot case studies in arithmetic reasoning datasets.

Case Study

A coin is heads up. Whitney flips the coin. Erika does not flip the coin. Tj does not flip the coin. Benito flips the coin. Is the coin still heads up? Note that "flip" here means "reverse".

step	initial coin state	flip or not	next coin state
—	—	—	—
1	heads up	no flip	heads up
2	heads up	no flip	heads up
3	heads up	flip	tails up
Therefore, the answer (Yes or No) is No. (GT: Yes)

Error Analysis: The Table failed to simulate all 4 steps and was early-stopped.

Table 15: Case studies in symbolic reasoning datasets.

Case Study

A bakery produces 60 loaves of bread each day. Two-thirds of the loaves are sold in the morning and half of what is left is sold equally in the afternoon and evening. How many loaves of bread are sold in the afternoon?

step	subquestion	process	result
1	How many loaves of bread are produced each day?	60 loaves	60 loaves
2	How many loaves of bread are sold in the morning?	2/3 * 60 loaves = 40 loaves	40 loaves
3	How many loaves of bread are left?	60 loaves - 40 loaves = 20 loaves	20 loaves
4	How many loaves of bread are sold in the afternoon and evening?	1/2 * 20 loaves = 10 loaves	10 loaves
5	How many loaves of bread are sold in the afternoon?	10 loaves / 2 = 5 loaves	5 loaves

Therefore, the answer (arabic numerals) is 5.(GT: 10)

Error Analysis: Wrong subquestion in step 4. This result in /2 in step 5.

John hires a driving service to get him to work each day. His work is 30 miles away and he has to go there and back each day. He goes to work 5 days a week for 50 weeks a year. He gets charged $2permiledrivenandhealsogiveshisdrivera$**150 bonus per month. How much does he pay a year for driving?**

step	subquestion	process	result
1	How many miles does he drive each day?	30 miles	30 miles
2	How many days a week does he go to work?	5 days	5 days
3	How many weeks a year does he go to work?	50 weeks	50 weeks
4	How many miles does he drive a year?	30 miles * 5 days * 50 weeks = 7,500 miles	7,500 miles
5	How much does he pay per mile?	$2	$2
6	How much does he pay for driving a year?	$2 * 7,500 miles = $15,000	$15,000
7	How much does he pay for the bonus?	$150 * 12 months = $1,800	$1,800
8	How much does he pay a year for driving?	$15,000 + $1,800 = $16,800	$16,800

Therefore, the answer (arabic numerals) is $16,800. (GT: 31800)

Error Analysis: Forget that John need to go there and back, so it is a two-way-trip.

There are four birds at the Caboose. Sally Two is three years older than Granny Red. Granny Red is two times as old as Sally Four. If Sally Four is the same age as Sally Thirtytwo, and Sally Thirtytwo is 8 years old, what's the total age of the four birds?

step	subquestion	process	result
1	How old is Sally Four?	8 years	8 years
2	How old is Sally Thirtytwo?	8 years	8 years
3	How old is Granny Red?	8 years / 2 = 4 years	4 years
4	How old is Sally Two?	4 years + 3 years = 7 years	7 years
5	What's the total age of the four birds?	8 years + 8 years + 4 years + 7 years = 27 years	27 years

Therefore, the answer (arabic numerals) is 27.(GT: 27)

Error Analysis: Step 3, Granny is 2 times older, so it is *2.

Sam 's dog had puppies and 8 had spots . He gave 2 to his friends . He now has 6 puppies . How many puppies did he have to start with ?

step	subproblem	process	result
1	How many puppies did Sam have to start with?	8	8
2	How many puppies did he give to his friends?	2	2
3	How many puppies does he have now?	6	6
4	How many puppies did he have to start with?	8 - 2	6

Therefore, the answer (arabic numerals) is 6.(GT: 8)

Error Analysis: The question ask for existing value, but the table still do the calculation.

The school is planning a field trip. The school has 87 classrooms. There are 58 students in the school with each classroom having the same number of students. If there are 2 seats on each school bus. How many buses are needed to take the trip?

step	subproblem	process	result
1	How many classrooms are in the school?	87	87
2	How many students are in the school?	58	58
3	How many students are in each classroom?	58 / 87	0.67
4	How many seats are on each bus?	2	2
5	How many students can fit on each bus?	2 * 0.67	1.34
6	How many buses are needed to take the trip?	58 / 1.34	43

Therefore, the answer (arabic numerals) is 43.(GT: 29)

Error Analysis: Redundant steps by redundant information, 58/2=29 .

Table 16: Additional few-shot Case studies in arithmetic reasoning datasets.

Few-shot Samples For 5 Arithmetic Reasoning Tasks

There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?

step	subquestion	process	result
—	—	—	—
1	How many trees are in the grove?	15	15
2	How many trees will be in the grove after the workers are done?	21	21
3	How many trees did the workers plant?	21 - 15	6
Therefore, the answer (arabic numerals) is 6.

If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

step	subquestion	process	result
—	—	—	—
1	How many cars are in the parking lot?	3	3
2	How many cars arrive?	2	2
3	How many cars are in the parking lot?	3 + 2	5
Therefore, the answer (arabic numerals) is 5.

Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total?

step	subquestion	process	result
—	—	—	—
1	How many chocolates did Leah have?	32	32
2	How many chocolates did her sister have?	42	42
3	How many chocolates did they eat?	35	35
4	How many chocolates do they have left?	32 + 42 - 35	39
Therefore, the answer (arabic numerals) is 39.

Jason had 20 lollipops. He gave Denny some lollipops. Now Jason has 12 lollipops. How many lollipops did Jason give to Denny?

step	subquestion	process	result
—	—	—	—
1	How many lollipops did Jason have?	20	20
2	How many lollipops does Jason have now?	12	12
3	How many lollipops did Jason give to Denny?	20 - 12	8
Therefore, the answer (arabic numerals) is 8.

Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now?

step	subquestion	process	result
—	—	—	—
1	How many toys does Shawn have?	5	5
2	How many toys did he get from his mom?	2	2
3	How many toys did he get from his dad?	2	2
4	How many toys does he have now?	5 + 2 + 2	9
Therefore, the answer (arabic numerals) is 9.

There were nine computers in the server room. Five more computers were installed each day, from monday to thursday. How many computers are now in the server room?

step	subquestion	process	result
—	—	—	—
1	How many computers were in the server room?	9	9
2	How many computers were installed each day?	5	5
3	How many computers were installed from monday to thursday?	5 * 4	20
4	How many computers are now in the server room?	9 + 20	29
Therefore, the answer (arabic numerals) is 29.

Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday?

step	subquestion	process	result
—	—	—	—
1	How many golf balls did Michael have?	58	58
2	How many golf balls did he lose on tuesday?	23	23
3	How many golf balls did he lose on wednesday?	2	2
4	How many golf balls did he have at the end of wednesday?	58 - 23 - 2	33
Therefore, the answer (arabic numerals) is 33.

Olivia has $23. She bought five bagels for $3 each. How much money does she have left?

step	subquestion	process	result
—	—	—	—
1	How much money does Olivia have?	$23	$23
2	How much does each bagel cost?	$3	$3
3	How many bagels did she buy?	5	5
4	How much money did she spend on bagels?	$3 * 5	$15
5	How much money does she have left?	$23 - $15	$8
Therefore, the answer (arabic numerals) is $8.

Table 17: Few-shot samples for 5 arithmetic reasoning tasks, including SingleEq, AddSub, MultiArith, GSM8K, and SVAMP

Few-shot Samples For AQUA

John found that the average of 15 numbers is 40. If 10 is added to each number then the mean of the numbers is? Answer Choices: (A) 50 (B) 45 (C) 65 (D) 78 (E) 64

step	subquestion	process	result
—	—	—	—
1	How much did the new mean change?	If 10 is added to each number, then the mean of the numbers also increases by 10.	10
2	What is the new mean?	So the new mean would be 40 + 10 = 50.	50
Therefore, among A through E, the answer is A.

If a / b = 3/4 and 8a + 5b = 22, then find the value of a. Answer Choices: (A) 1/2 (B) 3/2 (C) 5/2 (D) 4/2 (E) 7/2

step	subquestion	process	result
—	—	—	—
1	What equation we have have if we substitute b with a?	a / b = 3/4, then b = 4a / 3. So 8a + 5(4a / 3) = 22.	8a + 5(4a / 3) = 22
2	What is the value of a?	8a + 5(4a / 3) = 22 simplifies to 8a + 20a / 3 = 22, which means 44a / 3 = 22. So a is equal to 3/2.	2/3
Therefore, among A through E, the answer is B.

A person is traveling at 20 km/hr and reached his destiny in 2.5 hr then find the distance? Answer Choices: (A) 53 km (B) 55 km (C) 52 km (D) 60 km (E) 50 km

step	subquestion	process	result
—	—	—	—
1	What is the distance this person traveling?	The distance that the person traveled would have been 20 km/hr * 2.5 hrs = 50 km.	50km
Therefore, among A through E, the answer is E.

How many keystrokes are needed to type the numbers from 1 to 500? Answer Choices: (A) 1156 (B) 1392 (C) 1480 (D) 1562 (E) 1788

step	subquestion	process	result
—	—	—	—
1	How many one-digit numbers are there?	There are 9 one-digit numbers from 1 to 9.	9
2	How many two-digit numbers are there?	There are 90 two-digit numbers from 10 to 99.	90
3	How many three-digit numbers are there?	There are 401 three-digit numbers from 100 to 500.	401
4	How many keystrokes are needed to type the number from 1 to 500?	9 + 90(2) + 401(3) = 1392.	1392
Therefore, among A through E, the answer is B.

Table 18: Few-shot samples for AQUA

Few-shot Samples For CommonsenseQA

What do people use to absorb extra ink from a fountain pen? Answer Choices: (A) shirt pocket (B) calligrapher's hand (C) inkwell (D) desk drawer (E) blotter

|step|subquestion|process|result|
|—|—|—|—|
|1|What can we know of answer?|The answer must be an item that can absorb ink.|(E)| Therefore, Among A through E, the answer is E.

What home entertainment equipment requires cable? Answer Choices: (A) radio shack (B) substation (C) television (D) cabinet

step	subquestion	process	result
—	—	—	—
1	What can we know of answer?	The answer must require cable.	(C)
Therefore, Among A through E, the answer is C.

The fox walked of city into the forest, what was it looking for? Answer Choices: (A) pretty flowers (B) hen house (C) natural habitat (D) storybook

step	subquestion	process	result
—	—	—	—
1	What can we know of answer?	The answer must be something in the forest.	(B)
Therefore, Among A through E, the answer is B.

Sammy wanted to go to where the people were. Where might he go? Answer Choices: (A) populated areas (B) race track (C) desert (D) apartment (E) roadblock

step	subquestion	process	result
—	—	—	—
1	What can we know of answer?	The answer must be a place with a lot of people.	(A)
Therefore, Among A through E, the answer is A.

Where do you put your grapes just before checking out? Answer Choices: (A) mouth (B) grocery cart (C)super market (D) fruit basket (E) fruit market

step	subquestion	process	result
—	—	—	—
1	What can we know of answer?	The answer should be the place where grocery items are placed before checking out.	(B)
Therefore, Among A through E, the answer is B.

Google Maps and other highway and street GPS services have replaced what? Answer Choices: (A) united states (B) mexico (C) countryside (D) atlas

step	subquestion	process	result
—	—	—	—
1	What can we know of answer?	The answer must be something that used to do what Google Maps and GPS services do, which is to give directions.	(D)
Therefore, Among A through E, the answer is D.

Before getting a divorce, what did the wife feel who was doing all the work? Answer Choices: (A) harder (B) anguish (C) bitterness (D) tears (E) sadness

step	subquestion	process	result
—	—	—	—
1	What can we know of answer?	The answer should be the feeling of someone getting divorced who was doing all the work.	(C)
Therefore, Among A through E, the answer is C.

Table 19: Few-shot samples for CommonsenseQA

Few-shot Samples For StrategyQA

Do hamsters provide food for any animals?

step	subquestion	process	result
—	—	—	—
1	What is the evidence?	Hamsters are prey animals. Prey are food for predators.	yes
Therefore, the answer (yes or no) is yes.

Could Brooke Shields succeed at University of Pennsylvania?

step	subquestion	process	result
—	—	—	—
1	What is the evidence?	Brooke Shields went to Princeton University. Princeton University is about as academically rigorous as the University of Pennsylvania. Thus, Brooke Shields could also succeed at the University of Pennsylvania.	yes
Therefore, the answer (yes or no) is yes.

Yes or no: Hydrogen's atomic number squared exceeds number of Spice Girls?

step	subquestion	process	result
—	—	—	—
1	What is the evidence?	Hydrogen has an atomic number of 1. 1 squared is 1. There are 5 Spice Girls. Thus, Hydrogen's atomic number squared is less than 5.	no
Therefore, the answer (yes or no) is no.

Yes or no: Is it common to see frost during some college commencements?

step	subquestion	process	result
—	—	—	—
1	What is the evidence?	College commencement ceremonies can happen in December, May, and June. December is in the winter, so there can be frost. Thus, there could be frost at some commencements.	yes
Therefore, the answer (yes or no) is yes.

Yes or no: Could a llama birth twice during War in Vietnam (1945-46)?

step	subquestion	process	result
—	—	—	—
1	What is the evidence?	The War in Vietnam was 6 months. The gestation period for a llama is 11 months, which is more than 6 months. Thus, a llama could not give birth twice during the War in Vietnam.	no
Therefore, the answer (yes or no) is no.

Yes or no: Would a pear sink in water?

step	subquestion	process	result
—	—	—	—
1	The density of a pear is about 0.6g/cm3, which is less than water. Objects less dense than water float. Thus, a pear would float.	no
Therefore, the answer (yes or no) is no.

Table 20: Few-shot samples for StrategyQA

One-shot Sample Used on Coin Flip

A coin is heads up. Dorian flips the coin. Mayra flips the coin. Freddie does not flip the coin. Magaly flips the coin. Is the coin still heads up? Note that "flip" here means "reverse".

step	subquestion	process	result
—	—	—	—
1	Is the coin heads up?	Dorian flips the coin.	The coin is tails up.
2	Is the coin heads up?	Mayra flips the coin.	The coin is heads up.
3	Is the coin heads up?	Freddie does not flip the coin.	The coin is heads up.
4	Is the coin heads up?	Magaly flips the coin.	The coin is tails up.
Therefore, the answer (Yes or No) is "No".

Table 21: One-shot sample used on Coin Flip

One-shot Sample Used on Last Letter

Take the last letters of each words in L̈ucky Mireya Jj Kcänd concatenate them.

step	subquestion	process	result
—	—	—	—
1	What is the last letter of "Lucky"?	"Lucky"[-1] = 'y'	answer = 'y'
1	What is the last letter of "Mireya"?	"Mireya"[-1] = 'a'	answer = 'y' + 'a' = 'ya'
1	What is the last letter of "Jj"?	"Jj"[-1] = 'j'	answer = 'ya' + 'j' = 'yaj'
1	What is the last letter of "Kc"?	"Kc"[-1] = 'c'	answer = 'yaj' + 'c' = 'yajc'
Therefore, the answer is "yajc".

Table 22: One-shot sample used on Last Letter

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
limitation is attached

☑ A2. Did you discuss any potential risks of your work?
risks/drawback are discussed in the limitation

☑ A3. Do the abstract and introduction summarize the paper's main claims?
abstract and introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

for all experiments across multiple sections

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Not applicable. CodeX is currently unavailable, and it is free trail when I am conducting my experiment.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 3

☒ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Table is good enough to present the performance

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
yes, the code is on github

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.