Chain of Thought Prompting Elicits Knowledge Augmentation

Dingjun Wu[1], **Jing Zhang**[2] *, **Xinmei Huang**[2]
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]School of Information, Renmin University of China
wudj20@mails.tsinghua.edu.cn
{zhang-jing,huangxinmei}@ruc.edu.cn

Abstract

The knowledge-augmented deep learning paradigm refers to a paradigm in which domain knowledge is identified and integrated into deep models. Conventional methods typically employ task-specific approaches to gather external knowledge from various sources. In contrast, large language models are extensively pre-trained and can serve as a comprehensive source of external knowledge. In this paper, we propose CoT-KA, a Chain-of-Thought-based method that augments knowledge for deep learning. CoT-KA avoids the need for additional knowledge retrieval or knowledge reasoning models, as required in conventional augmentation methods. Our results demonstrate that CoT-KA outperforms both pure CoT-based methods and the non-augmented method across the majority of eleven publicly available benchmarks for various reasoning tasks [1].

1 Introduction

The Knowledge-Augmented deep learning (KADL) (Cui et al., 2022) paradigm refers to the deep learning paradigm in which domain knowledge is identified and integrated into the deep model. Adding domain knowledge makes it possible to develop deep learning that is data-efficient, generalizable, and interpretable (Cui et al., 2022). For example, retrieving external knowledge from an external knowledge pool like Wikipedia is typically required for open domain question answering and dialog generation (Izacard and Grave, 2021; Zhang et al., 2023). Logical equivalence laws such as contraposition and transitive laws help extend the implicit logical information (Yu et al., 2019; Wang et al., 2022a).

External knowledge is derived from various sources. For instance, commonsense knowledge can be extracted from commonsense knowledge bases like ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019). Domain-specific knowledge can be retrieved from knowledge bases such as Wikipedia and Freebase (Bollacker et al., 2008). Logic knowledge, on the other hand, can be in the form of human-defined propositional or first-order logic, which is then utilized as rules for reasoning. In summary, existing knowledge augmentation methods typically involve either creating a retriever to gather relevant knowledge or developing a reasoner to leverage the logical rules within the external knowledge sources (Chen et al., 2017; Izacard and Grave, 2021; Wang et al., 2022a; Zhang et al., 2023).

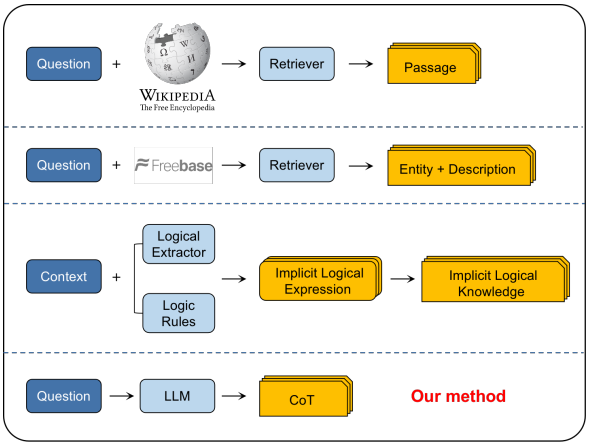

Figure 1: A various sources of external knowledge. We use LLM as our source of knowledge.

Recently, large language models (LLMs) (Zhao et al., 2023) have shown their potential as both the source and the retriever or reasoner of external knowledge. LLMs are pre-trained on a huge scale of datasets. Thus, they have already embedded a large amount of knowledge into their parameters, which can be considered a source of external knowledge. The reasoning ability of LLMs allows them to provide knowledge from their parameters without needing an extra retriever or a reasoner. The latest chain-of-thought (CoT) prompting technique

*Corresponding author: Jing Zhang.
[1]Our code and data are available at https://github.com/RUCKBReasoning/CoT-KA

Findings of the Association for Computational Linguistics: ACL 2023, pages 6519–6534
July 9-14, 2023

(Wei et al., 2022), which elicits LLMs to generate a series of sentences that mimic the reasoning process for arriving at the answers, improves the reasoning ability of LLMs. It has proved to be remarkably effective in a variety of complex reasoning tasks such as math word problems and commonsense question answering (Wei et al., 2022). CoT prompting shows potential as a general technique to retrieve knowledge from LLMs.

In this paper, we propose CoT-KA – a CoT-based method to retrieve knowledge from LLMs for Knowledge-Augmented deep learning. CoT-KA utilizes an LLM as a knowledge source, leveraging CoT prompting to guide the LLM in providing knowledge that can serve as evidence to support downstream reasoning from the input to the answer. Unlike conventional KADL approaches, CoT-KA eliminates the need for additional knowledge retrieval or a separate knowledge reasoning model. Specifically, we begin by extracting CoTs as knowledge from the LLM using either few-shot (Wei et al., 2022) or zero-shot (Kojima et al., 2022) CoT prompting. The former involves providing a few demonstrations to guide the LLM's reasoning, while the latter employs a template such as "let's think step by step" to inspire the LLM. The extracted CoTs are then appended to the original inputs, marked by a special token, to create augmented text. Finally, we fine-tune a small task-relevant pre-trained language model (PLM) on the dataset augmented with CoTs.

We generate CoTs using the public GPT-3 (Brown et al., 2020) (175B parameters) API[2]. For NLU (Natural Language Understanding) tasks, we employ ALBERT (Lan et al., 2019) and DeBERTa (He et al., 2021) as the task-relevant models. T5 (Raffel et al., 2020) is utilized as the task-relevant model for NLG (Natural Language Generation) tasks. We evaluate models' performance using eleven benchmarks, including (i) commonsense reasoning (CSQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021), Date Understanding, Sports Understanding (Srivastava et al., 2022)); (ii) arithmetic reasoning (AQUA-RAT (Ling et al., 2017), GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), MultiArith (Roy and Roth, 2015), SingleEq (Koncel-Kedziorski et al., 2015), AddSub (Hosseini et al., 2014)); (iii) symbolic reasoning (Last Letter Concatenation (Wei et al., 2022)), where all commonsense reasoning benchmarks and AQUA-RAT are formulated as NLU tasks, and the other arithmetic reasoning benchmarks and Last Letter Concatenation are formulated as NLG tasks in this paper. Particularly, we convert all of the multi-choice question answering tasks into NLU tasks. Extensive experimental results show that in the majority of tasks, CoT-KA outperforms the original fine-tuning results without the use of CoTs as augmented knowledge. CoT-KA also surpasses Few-Shot-CoT and Zero-Shot-CoT on LLMs, which directly parse answers from the generated CoTs.

[2]Public API available at https://openai.com/api/

2 Related Work

Knowledge Augmented Technology. The integration of external knowledge into deep learning models through knowledge augmentation approaches has gained significant attention in various NLP tasks, including question answering (Chen et al., 2017; Izacard and Grave, 2021), dialogue generation (Zhang et al., 2023), and logical reasoning (Wang et al., 2022a). For instance, in the context of answering open-domain questions where supporting evidence is not explicitly provided (Izacard and Grave, 2021), Chen et al. (2017) utilized techniques such as bigram hashing and TF-IDF matching to retrieve relevant documents from external knowledge sources. Similarly, Fusion-in-Decoder (Izacard and Grave, 2021) employed methods like BM25 (Robertson et al., 1995) and DPR (Karpukhin et al., 2020) for evidence retrieval. By augmenting the questions with these retrieved pieces of evidence, the models can better reason and provide answers. Logic reasoning is another challenging task that requires a deep understanding of the logical structure within a given text to arrive at the correct answer. To facilitate such logic-level analysis, human-defined logic rules are introduced. Wang et al. (2022a) proposed LReasoner, a logic-driven context extension framework that extends implicit logical information by performing logical reasoning using these predefined rules. The framework enhances the original input by verbalizing and concatenating the implicit logical information, enabling subsequent answer reasoning.

Fusion-in-Decoder and LReasoner inspire our work to extend the external knowledge into the original input. However, the knowledge in these knowledge augmentation methods is sourced from external knowledge bases or pre-defined logical rules, requiring a retriever for knowledge extraction or a reasoner for rule application in the process. In

contrast, we utilize LLMs that eliminate the need for an additional retriever or reasoner to acquire knowledge for augmentation.

Chain of Thought Prompting on LLMs. A CoT is a series of intermediate natural language reasoning steps that lead to the final output, inspired by how humans use a deliberate thinking process to perform complicated tasks. Experimental results using various LLMs, such as GPT-3 (Brown et al., 2020) and PaLM (Chowdhery et al., 2022), demonstrate that CoT prompting enhances performance across a range of arithmetic, commonsense, and symbolic reasoning tasks (Wei et al., 2022).

Wei et al. (2022) initially propose Few-Shot-CoT, which requires the manual design of a few demonstrations to facilitate the generation of reasoning paths. In contrast, Kojima et al. (2022) propose Zero-Shot-CoT, which employs a single zero-shot prompt that elicits CoTs from LLMs. By simply adding "Let's think step by step" before each answer, Zero-Shot-CoT demonstrates that LLMs are capable zero-shot reasoners without the need for any manually constructed few-shot examples. Furthermore, Wang et al. (2022b) introduce a new decoding strategy called self-consistency, which involves sampling multiple LLM outputs and aggregating them through majority voting. This strategy encourages the model to consider multiple CoTs when generating answers. However, to achieve optimal performance, a large number of reasoning paths (e.g., 40 paths) must be generated, leading to increased computational costs.

All of these CoT prompting methods directly extract the answer from the CoTs. In contrast, our method utilizes the generated CoTs as supplementary knowledge to improve the fine-tuning of task-relevant models. Moreover, our method demonstrates good performance even when a limited number of CoTs are provided, unlike self-consistency, which relies on generating a large number of CoTs.

3 Pilot Study

In this section, we explore the effectiveness of CoT-augmented fine-tuning by simply appending one CoT to the original input. We assess the validity of this approach on two commonsense reasoning datasets, CSQA and StrategyQA.

CoT-augmented Fine-tuning. To perform fine-tuning on ALBERT, we extend the original input text by adding a CoT. We utilize ALBERT-large-v2 for our experiments. Specifically, we generate CoTs using both few-shot and zero-shot CoT methods, known as Few-Shot-CoT and Zero-Shot-CoT, respectively. Few-Shot-CoT employs the same demonstrations as described in (Wei et al., 2022). For Zero-Shot-CoT, we utilize the template "Let's think step by step". As the LLM, we employ GPT-3 with 175-billion parameters (text-davinci-002). Subsequently, we extend the generated CoT into the input of each sample within the CSQA and StrategyQA datasets. Finally, we perform fine-tuning on ALBERT using the augmented datasets.

Method/Dataset	CSQA	StrategyQA
Baseline (ALBERT)	63.4	64.8
Zero-Shot-CoT (ALBERT)	70.1	67.5
Few-Shot-CoT (ALBERT)	**76.2**	**73.1**

Table 1: Accuracy (%) of original fine-tuning (baseline) and CoT-augmented fine-tuning results.

The experiment results in Table 1 show that both the Zero-Shot-CoT and Few-Shot-CoT augmented fine-tuning significantly enhance the performance of the original fine-tuning method.

The Impact of CoT as Additional Knowledge. Given that the answers within CoTs can potentially be incorrect, we hypothesize that this portion of the CoTs will have a negative effect on the fine-tuning and mislead the model's prediction. To further explore the effect of CoTs on fine-tuning, we compare the fine-tuning result of the PLMs before and after adding CoTs through a variety of data analyses.

We investigate the extent to which the prediction results are altered when the model's input is expanded with a CoT. We perform fine-tuning on both the original samples (baseline) and the expanded samples (CoT-extended). Subsequently, we evaluate the fine-tuned models using the validation set. For each instance in the validation set, we compare its predictive result between the originally fine-tuned ALBERT and the CoT-augmented fine-tuning version. Additionally, we define three categories of CoTs during the process.

- A CoT is labeled as a *positive CoT* if the addition of the CoT changes the prediction result from incorrect to correct. This indicates a beneficial

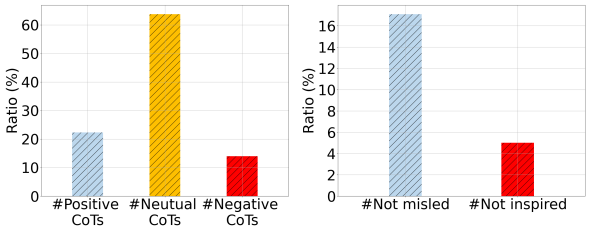

Figure 2: The observation when the original question added a CoT. The figure on the left shows the ratio of *positive*, *neutral*, and *negative CoTs* in the validation set of StrategyQA. The figure on the right shows the proportion of model predictions that do not align with the answer in the CoT. "Not misled" denotes that the answer in the CoT is incorrect, but the model is not misled by the CoT and makes accurate predictions. "Not inspired" denotes that the answer in the CoT is correct, but the model does not follow the correct CoT and makes incorrect predictions.

influence on the model's prediction.

- Conversely, a CoT is labeled as a *negative CoT* if the addition of the CoT changes the prediction result from correct to incorrect. This indicates a misleading effect on the model's prediction.
- Furthermore, a CoT is labeled as a *neutral CoT* if the model's prediction result remains the same after the CoT is added. In such cases, it is not easy to judge the impact of this CoT on the model.

The left figure in Figure 2 illustrates the ratio of *positive*, *neutral*, and *negative CoTs*. It is observed that among the model's prediction results that change after adding a CoT, the ratio is 36.2% (166 out of 458). Within this group, the ratio of *positive CoTs* is 61.4%, while the ratio of *negative CoTs* is 38.6%. These findings suggest that the model successfully resolves 63.3% (102/161, the number of positive CoTs divided by the number of incorrectly predicted samples in the baseline) of the data samples that were incorrectly predicted prior to adding a CoT.

The second objective is to test our hypothesis that an incorrect CoT (the answer in the CoT is incorrect) may have a negative impact on the model and therefore mislead the prediction of the model. If an incorrect CoT is added to the original input text, what impact does it have on the model's prediction? As the right figure in Figure 2 shows, when an incorrect CoT is added to the original input, the model still has a high probability (17.1%) of not being misled by the incorrect CoT and making accurate predictions. Furthermore, we investigate the extent to which the model would mispredict when a correct CoT (the answer in the CoT is correct) is added. As shown in the figure on the right of Figure 2, the model has a low probability (5.0%) of making an incorrect prediction.

In the case of StrategyQA, when the answer in the CoT is incorrect, the alignment ratio is $1 - Ratio$ (`#Not misled`), which equals 82.9%; When the answer in the CoT is correct, the alignment ratio is $1 - Ratio$ (`#Not inspired`), which equals 95.0%. The result demonstrates that CoT is a powerful feature, and the model's predictions tend to align closely with the answers provided in CoT. On the other hand, the fine-tuning strategy employed causes the model's predictions to treat CoT as a secondary feature of the original input, rather than strictly following it. In cases where the answer in CoT is correct, the model is likely to align its predictions with the answers in CoT. Conversely, when the answer in CoT is incorrect, there is a relatively high probability that the model will deviate from the answer in the CoT, preventing misleading from the incorrect CoT.

In addition, our attempts to preserve the reasoning steps in the CoTs while removing the answers have resulted in a degradation in performance. We recognize that the presence of incorrect answers in some CoTs can have a negative impact. However, we also believe that the inclusion of correct answers in CoTs can yield positive effects, and the answers within CoTs are a more influential factor than the reasoning paths themselves.

4 CoT-KA

In this section, we propose CoT-KA – a CoT-based method for knowledge augmentation. Our method leverages multiple CoTs retrieved from LLMs to provide more auxiliary knowledge for KADL. CoT-KA consists of three steps as shown in Figure 3: (1) CoT Generation: Generating multiple CoTs for each sample in the train, dev, and test sets. (2) Input Augmentation: Taking the generated CoTs as the additional knowledge into the original input text for each sample. (3) Task-relevant Model Training: Fine-tuning a task-relevant model using the CoT-augmented samples.

4.1 CoT Generation

We try both Few-Shot-CoT and Zero-Shot-CoT prompting on LLM f to generate multiple CoTs. Formally, given an original samples (x_i, y_i), where

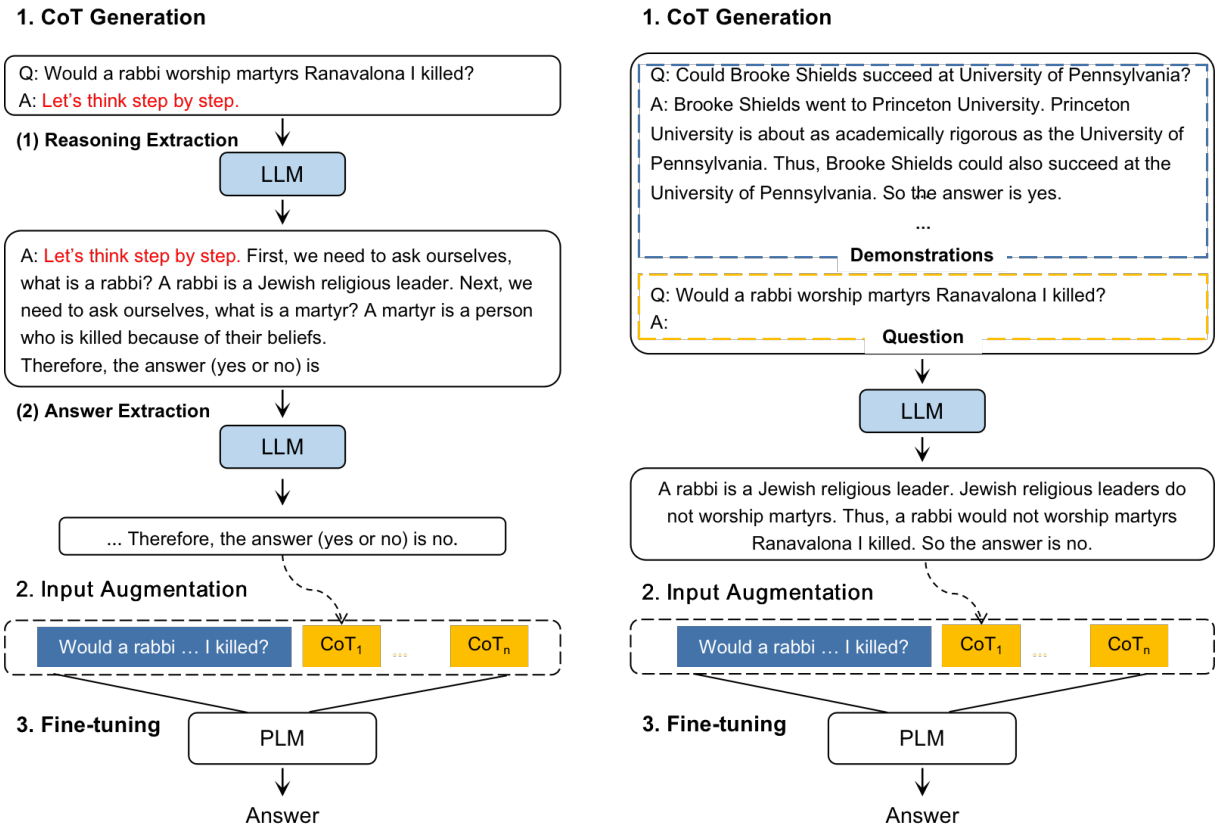

Figure 3: Overview of the CoT-KA method. Both Zero-Shot-CoT (on the left) and Few-Shot-CoT (on the right) can be used in the CoT generation stage for CoT-KA.

x_i is the original input and $y_i \in \mathcal{Y}$ denotes the label. We generate a CoT set consisting of multiple CoTs based on the model f:

$$CoT^{(i)} = f(d, x_i) \tag{1}$$

where d denotes the CoT demonstrations that inspire model f to generate CoTs, and $CoT^{(i)}$ is the generated CoT set of the i-th sample, which consists of m CoTs:

$$CoT^{(i)} = \{CoT_1^{(i)}, CoT_2^{(i)}, ..., CoT_m^{(i)}\} \tag{2}$$

For each sample, we independently generate m CoT outputs from f in each run.

4.2 Input Augmentation

In the second step, we apply the generated CoTs as additional knowledge to enrich the input text of the original samples. The extended input text of each sample is a concatenation of an original input (e.g. a question), and the generated multiple CoTs. For each sample, we construct an extended input text as follows:

$$\tilde{x}^{(i)} = concat(x^{(i)}, CoT^{(i)}) \tag{3}$$

where $\tilde{x}^{(i)}$ is the i-th extended input text, $x^{(i)}$ is the i-th original input, and $CoT^{(i)}$ is the i-th generated CoT set. $concat()$ is a concatenation function that concatenates the original input and the generated CoTs. More concretely:

$$\begin{aligned} &concat(x^{(i)}, CoT^{(i)}) \\ &= x^{(i)} || \, [EXT] \, CoT_1^{(i)} ... \, || \, [EXT] \, CoT_m^{(i)} \end{aligned} \tag{4}$$

where $[EXT]$ is the special token to denote a CoT, and $||$ denotes the concatenation operator.

5 Experiments

5.1 Experimental setup

Tasks and Datasets. We evaluate CoT-KA on the following reasoning benchmarks[3].

[3]By default we use the train, dev, and test split of all the datasets if the labels are available for evaluation. For CSQA and StrategyQA, we only use the train and dev split.

Method/Dataset	*Commonsense*						*Arithmetic*	
	CSQA	StrategyQA	Date		Sports		AQuA	
	Dev	Dev	Dev	Test	Dev	Test	Dev	Test
Zero-Shot-CoT	64.6*	54.8*	67.5*		52.4*		33.5*	
Few-Shot-CoT	- (73.5*)	68.3 (65.4*)	54.7/47.4 (52.1*)		83.2/86.7 (82.4*)		-/37.9 (35.8*)	
Self-Consistency (5 Zero-Shot-CoTs)	71.2	64.6	29.2/35.6		57.6/58.9		33.2/37.0	
Self-Consistency (5 Few-Shot-CoTs)	77.6	73.6	53.4/50.1		85.4/90.5		40.6/40.2	
Baseline (ALBERT)	61.8	62.2	33.2	33.5	57.2	53.2	25.6	22.7
CoT-KA (5 Zero-Shot-CoTs, ALBERT)	73.6	66.1	58.6	64.1	68.8	69.6	42.3	40.2
CoT-KA (5 Few-Shot-CoTs, ALBERT)	**78.8**	**75.7**	**74.2**	**76.6**	**89.9**	**89.8**	**46.9**	**47.6**
Baseline (DeBERTa)	**84.2**	68.8	73.6	72.7	84.5	82.8	27.8	26.5
CoT-KA (5 Zero-Shot-CoTs, DeBERTa)	80.3	72.3	69.2	73.8	91.3	90.5	40.1	40.3
CoT-KA (5 Few-Shot-CoTs, DeBERTa)	82.0	**76.9**	**80.4**	**78.0**	**96.9**	**95.6**	**45.9**	**46.5**

Table 2: Accuracy on five NLU datasets from two categories of reasoning tasks. For CSQA and StrategyQA, we report the evaluation results of the dev set. For the other datasets in which the labels are available, we report the results of both the dev and test. * indicates the results comes from (Wei et al., 2022) and (Kojima et al., 2022). The results of baseline methods and CoT-KA are based on ALBERT-large-v2 and DeBERTa-v3-large. "Baseline" denotes the fine-tuning baseline with original data. "5 Zero-Shot-CoTs" and "5 Few-Shot-CoTs" denotes five CoTs used at Self-Consistency and CoT-KA. Bold denotes the best-performed results. For Few-Shot-CoT, the results before and after the "/" symbol indicate the results of directly parsing the answers from the CoT (from Wei et al. (2022)) for the dev and test set, respectively, under our data partitioning. For Self-Consistency, the results before and after the "/" symbol represent the results obtained by parsing the answer from multiple CoTs (We generated) in the dev and test set, respectively, under our data partitioning and then applying majority voting.

- **Commonsense reasoning.** We evaluate our method on four commonsense reasoning tasks: CSQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021) and two benchmarks from the BIG-bench effort (Srivastava et al., 2022): Date Understanding and Sports Understanding.

- **Arithmetic reasoning.** We use six arithmetic reasoning benchmarks: AQUA-RAT (Ling et al., 2017), GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), MultiArith (Roy and Roth, 2015), SingleEq (Koncel-Kedziorski et al., 2015), AddSub (Hosseini et al., 2014).

- **Symbolic Reasoning.** We use the Last Letter Concatenation from Wei et al. (2022). [4]

Implementation.

- **CoT Generation Models.** We use GPT-3 of the text-davinci-002 engine with 175-billion parameters to generate the CoTs used in CoT-KA.

- **CoT Demonstrations.** For a fair comparison, we perform Few-Shot-CoT with the same demonstrations as in Wei et al. (2022) and use the same zero-shot prompt as in Kojima et al. (2022) to perform Zero-Shot-CoT.

- **Sampling Scheme.** To generate diverse CoTs, we apply temperature sampling during the CoT generation. Specifically, we use the same T=0.7 as in (Wang et al., 2022b) for a fair comparison.

- **Data Preprocessing.** For certain undivided datasets, we divide them into train, dev, and test sets for fine-tuning, following a ratio of 6:2:2. Further details regarding the dataset splits can be found in Appendix A.1. Additionally, as the original questions and demonstrations used for CoT generation may include option information (e.g., Answer Choices: *(a) ignore ...(e) avoid*),

[4]We do not use the Coin Flip dataset for the evaluation because it is a simple classification task for fine-tuning. This is because ALBERT-large-v2 and DeBERTa-v3-large can already achieve 100% accuracy in the evaluation phase.

Method/Dataset	*Arithmetic*										*Symbolic*	
	GSM8K		SVAMP		MultiArith		SingleEq		AddSub		Letter (4)	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Zero-Shot-CoT	40.7*		63.7*		78.7*		78.7*		74.7*		57.6*	
Few-Shot-CoT	-/46.5 (46.9*)		69.2/69.0 (68.9*)		85.8/90.0 (91.7*)		82.4/87.3 (86.6*)		79.7/65.8 (81.3*)		(59.0**)	
Self-Consistency (5 Zero-Shot-CoTs)	51.7/52.2		70.0/73.4		81.7/**96.4**		64.8/92.0		79.7/73.7		66.3/60.2	
Self-Consistency (5 Few-Shot-CoTs)	55.7/56.6		**74.7**/75.5		**94.8**/95.7		**88.5/91.9**		86.8/73.9		59.0/60.5	
Baseline (T5)	5.3	4.4	8.0	8.5	12.5	8.3	5.9	2.9	6.3	6.3	30.0	26.0
CoT-KA (5 Zero-Shot-CoTs, T5)	58.9	57.3	64.2	**82.3**	82.7	93.3	62.9	73.3	80.3	74.9	**75.9**	60.4
CoT-KA (5 Few-Shot-CoTs, T5)	**61.2**	**61.5**	71.8	70.8	81.8	95.3	76.7	75.7	**86.6**	**78.7**	**71.8**	**69.8**

Table 3: Accuracy on six NLG datasets from two categories of reasoning tasks. * indicates the results comes from (Wei et al., 2022) and (Kojima et al., 2022) and ** denotes the result comes from (Zhang et al., 2022).

StrategyQA	**Question**: Would Siduri enjoy an unlimited buffet? **Blink**: Siduri is a character in the "Epic of Gilgamesh". She is an "alewife", a wise female divinity associated with fermentation (specifically beer and wine). **Few Shot CoT**: Siduri is a fairy in Irish mythology. She was known for her hospitality, so she would probably enjoy an unlimited buffet. So the answer is yes.
Sports	**Question**: Will Fuller was perfect from the line? **Blink**: William Vincent Fuller V (born April 16, 1994) is an American football wide receiver for the Houston Texans of the National Football League (NFL). He was drafted by the Texans in the first round of the 2016 NFL Draft. He played college football at Notre Dame. **Few Shot CoT**: Will Fuller is a football player. Being perfect from the line is part of basketball, not football. So the answer is no.

Table 4: Knowledge augmentation examples from commonsense reasoning tasks. The first case comes from StrategyQA. In this case, the description of Siduri does not mention the relationship between Siduri and the unlimited buffet, which is the key to answering the question. The second case comes from Sports Understanding. In this case, we need to know that being perfect from the line is part of basketball, and Will Fuller is a football player, while the entity-knowledge can only provide the latter.

the generated CoT will also contain option markers (e.g., the answer is *(a)*). To provide valuable information within the CoTs, we replace the option markers in the generated CoT with their corresponding textual content (e.g., the answer is *"ignore"*).

- **Classifier Models.** We conduct the main experiments using two backbone PLMs: ALBERT-large-v2 and DeBERTa-v3-large. The hyperparameters for the training process are reported in Appendix A.2.

Baselines. We take three methods as the baselines: Zero-Shot-CoT, Few-Shot-CoT, and Self-Consistency. Furthermore, to demonstrate the extent to which the CoT knowledge elicits the KADL, we also compare our method with the original fine-tuning baselines, which solely employ the original text for fine-tuning.

5.2 Main Results

Table 2 compares the accuracy across eleven datasets from three categories of NLU and NLG tasks. The Zero-Shot-CoT results are taken from Kojima et al. (2022), and the Few-Shot-CoT results are taken from Wei et al. (2022). For Self-Consistency (5 sampled CoTs), we report the result based on a majority vote. The CoT-KA results are averaged over at least five random runs (see Appendix for more details), where we use the different seeds to sample 5 CoTs from a CoT set containing 10 generated CoTs in each run.

As shown in Table 2 and 3, the performance of CoT-KA surpasses all baselines on most tasks. We have made several findings: (1) The CoTs generated by Zero-Shot-CoT and Few Shot-CoT can be utilized with CoT-KA, resulting in significantly improved performance compared to the fine-tuning baselines. Additionally, the CoTs generated by Few-Shot-CoT exhibit better performance com-

pared to Zero-Shot-CoT when they are used with CoT-KA. (2) CoT-KA achieves better performance on the NLU tasks than on the NLG tasks. (3) CoT-KA shows different robustness on different models. While DeBERTa outperforms ALBERT on most tasks, CoT-KA is more robust on ALBERT and exhibits performance improvements across all tasks.

5.3 Knowledge Augmentation Comparison

To compare CoT-KA with other knowledge augmentation methods, we employ BLINK (Wu et al., 2020) to enrich the entity knowledge in the question. BLINK is a two-stage entity linking approach based on BERT (Kenton and Toutanova, 2019). We use BLINK to link the entities mentioned in the question and retrieve their corresponding entity information. BLINK provides a short description for each entity, which we utilize as extensions to enrich the questions.

Method/Dataset	StrategyQA	Sports	
	Dev	Dev	Test
Baseline (ALBERT)	62.2	57.2	53.2
BLink (ALBERT)	58.0	81.3	77.4
CoT-KA (ALBERT)	75.7	89.9	89.8
Baseline (DeBERTa)	68.8	84.5	82.8
BLink (DeBERTa)	67.7	92.5	87.5
CoT-KA (DeBERTa)	**76.9**	**96.9**	**95.6**

Table 5: Knowledge augmentation comparison.

As shown in Table 5, the entity knowledge-based augmentation method improves performance on Sports Understanding but has a negative impact on StrategyQA, with both performing worse than our method. Additionally, we observe that approximately 29% of questions in StrategyQA and 3% in Sports Understanding could not have entities extracted. Furthermore, the average number of recognized entities in a Sports Understanding question is 1.095, while in StrategyQA, it is 0.928. Moreover, Table 4 demonstrates that entity information may not always include the specific information required by the questions. In contrast, our method can add more useful information, resulting in a more substantial improvement.

5.4 The Effect of CoT Size

To demonstrate the effect of the number of sampled CoTs, we vary the number of sampled CoTs (1, 2, 3, 4, 5) in CoT-KA and evaluate on StrategyQA. The results are shown in Figure 4. The experimental results indicate that as the number of CoTs increases,

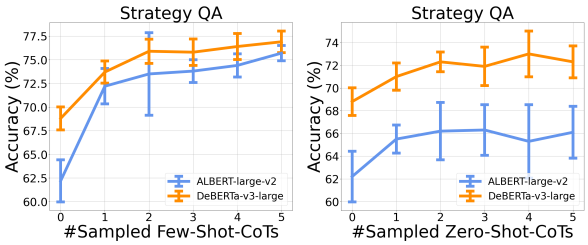

Figure 4: The impact of the sampled CoT size on CoT-KA . We randomly sampled 1 to 5 CoTs from both the CoT set generated by Zero-Shot-CoT and Few-Shot-CoT.

there is a general upward trend in the performance of CoT-KA . This trend becomes more pronounced when the CoTs are generated by Few-Shot-CoT. More results are reported in Appendix B.

5.5 CoT Selection Strategy

CoT-KA can only extend a small number of CoTs due to the maximum length limitation of the input sequence that the language model can handle. Therefore, it is natural to consider designing a CoT selection strategy to choose higher-quality CoTs from the generated CoT set for KADL. Each CoT can be expressed as: $t_i \in \{t_1, t_2, ..., t_K\}$, where t_i is the i-th token. We can get the *log prob* of each generated token when using GPT3 API to generate reasoning chains. The *log prob* refers to the natural logarithm of the probability that the token occurs next given the prompt. To select the 5 reasoning chains with higher confidence from the 10 generated CoTs, we score the generated CoTs using the following formula:

$$score(CoT_j) = \frac{\sum_{i=1}^{K_j} \exp(\log p(t_i))}{K_j} = \frac{\sum_{i=1}^{K_j} p(t_i)}{K_j} \quad (5)$$

where $p(t_i)$ denotes the probability of generating the i-th token, and log denotes the logarithm. and K_j is the total number of tokens in the j-th CoT. The results shown in Table 6 demonstrate that selecting CoTs from the generated set based on the probability of token generation in the sentence does not lead to a significant improvement in the performance of CoT-KA .

6 Conclusion and Future Work

This paper introduces a CoT-based method to retrieve knowledge from LLMs for Knowledge-Augmented deep learning (CoT-KA) that elicits

Method	StrategyQA
CoT-KA (ALBERT)	75.7
CoT-KA (ALBERT) + CoT Selection	75.9
CoT-KA (DeBERTa)	76.9
CoT-KA (DeBERTa) + CoT Selection	76.9

Table 6: CoT selection strategy based on the *log prob*

knowledge augmentation on a variety of NLU and NLG benchmarks. Unlike conventional knowledge augmentation approaches, our method does not require a retriever or a reasoner, yet it surpasses the performance of conventional knowledge-based methods and other CoT-based approaches across a range of public NLP tasks.

In the future, it is worthwhile to investigate other methods that can provide insights from LLMs. Exploring new approaches for leveraging the capabilities of LLMs to enhance knowledge augmentation represents a promising area for future research.

7 Limitations

One limitation of CoT-KA is that it performs fine-tuning based on the PLMs, and the input sequence length limit of the PLMs allows us to add only a limited number of CoTs. Therefore, it is important to explore and develop a CoT selection strategy in future research. A good CoT selection strategy would enable the identification of highly effective CoTs from a set of CoTs, enhancing the efficiency of KADL.

Acknowledgments

This work is supported by National Natural Science Foundation of China 62076245; CCF-Zhipu AI Large Model Fund.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

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

Zijun Cui, Tian Gao, Kartik Talamadupula, and Qiang Ji. 2022. Knowledge-augmented deep learning and its applications: A survey. *arXiv preprint arXiv:2212.00017*.

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*, 9:346–361.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

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*, 3:585–597.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

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 the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3027–3035.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of NAACL-HLT*, pages 4149–4158.

Siyuan Wang, Wanjun Zhong, Duyu Tang, Zhongyu Wei, Zhihao Fan, Daxin Jiang, Ming Zhou, and Nan Duan. 2022a. Logic-driven context extension and data augmentation for logical reasoning of text. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1619–1629.

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

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

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2019. Reclor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations*.

Jing Zhang, Xiaokang Zhang, Daniel Zhang-Li, Jifan Yu, Zijun Yao, Zeyao Ma, Yiqi Xu, Haohua Wang, Xiaohan Zhang, Nianyi Lin, et al. 2023. Glm-dialog: Noise-tolerant pre-training for knowledge-grounded dialogue generation. *arXiv preprint arXiv:2302.14401*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

A Implementation Detail

A.1 Datasets

Dataset	#Number of samples			We divide the dataset
	Train	Dev	Test	
CSQA	9741	1221	1140	No
StrategyQA	1831	458	490	No
Date	221	74	74	Yes
Sports	600	200	200	Yes
AQUA	5000	254	254	Yes
GSM8K	5978	1495	1319	Yes
SVAMP	600	200	200	Yes
MuitiArith	360	120	120	Yes
Single Eq	304	102	102	Yes
Add Sub	237	79	79	Yes
Last Letter	600	200	200	Yes

Table 7: Summary of the datasets we use in this paper. For datasets that are not pre-divided into train, dev, and test sets, we conduct the division ourselves.

For some undivided datasets used in this paper, we divide them into train, dev, and test sets for fine-tuning, following a ratio of 6:2:2. Table 7 shows the division details of each dataset. In the case of AQUA, the raw training set is too large (97467 samples). To mitigate the computational cost of generating multiple CoTs using the public GPT3 API, we select a subset of 5000 samples (the top 5000) from the raw train set as our train set.

A.2 Hyper-parameters for Fine-tuning

All experiments are conducted in a Linux environment with a single (24G) NVidia RTX 3090 GPU. The model is optimized using the AdamW optimizer. We do not perform an exhaustive hyper-parameter search, but only adjust the learning rate prior to the formal experiment. For most experiments in this paper, a learning rate of 1e-5 is chosen as the final value for fine-tuning ALBERT and DeBERTa, except in the following cases for CSQA and StrategyQA:

- CSQA: A learning rate of 2e-5 is used for CoT-KA (1 Zero-Shot-CoT, ALBERT).
- StrategyQA: A learning rate of 5e-6 is used for CoT-KA (1 Zero-Shot-CoT, ALBERT), CoT-KA (1 Few-Shot-CoT, DeBERTa) and CoT-KA (5 Few-Shot-CoTs, both ALBERT and DeBERTa).

More hyper-parameters are shown in Table 8.

The random seed set utilized for experiments is `[0, 10, 20, 30, 40, 50, 60, 70, 80, 90]`.

	ALBERT/DeBERTa	T5
Batch Size	16	16
Peak Learning Rate	1e-5	1e-5
Training Steps	2000	2000
Warmup Proportion	0.1	0
Weight Decay	0	0
Adam ϵ	1e-8	1e-8
Adam β_1	0.9	0.9
Adam β_2	0.999	0.999

Table 8: Hyper-parameters for fine-tuning.

These seeds are used for both CoT sampling and fine-tuning. For the case of experimental results averaged over five runs, we use the top five seeds from the seed set. For NLU tasks, most experimental results in Table 2 are averaged over ten runs, except for the following cases:

- CoT-KA (5 Zero-Shot-CoTs) on all NLU tasks are averaged over five runs.
- CoT-KA (5 Few-Shot-CoTs) on AQUA is averaged over five runs.

For NLG tasks, most results in Table 3 are averaged over ten runs, with the exception of CoT-KA (5 Zero-Shot-CoTs) and CoT-KA (5 Few-Shot-CoTs), which are averaged over five runs.

The result for Blink in Table 5 are averaged over five runs. All the new results in Section 5.4 and Appendix B, where the number of sampled CoTs ranges from 1 to 4, are averaged over five runs.

B More results about the Effect of CoT Size in CoT-KA

We vary the number of sampled CoTs (1, 5) in CoT-KA and evaluate its performance on ten tasks, excluding StrategyQA. Figures from 5 to 14 indicate that in most of these tasks, increasing the number of CoTs from 0 to 1 significantly improves task performance. However, when using DeBERTa-v3-large as the PLM, the performance gain in CoT-KA for CSQA, Date Understanding, and Sports Understanding is slight and even leads to a degradation. Furthermore, increasing the number of CoTs from 1 to 5 has a relatively small performance gain in CoT-KA (DeBERTa), except for improved Date Understanding and continued degradation in CSQA.

We observe that if the baseline, where the dataset is not augmented by a CoT, starts with a lower performance, the performance gain in CoT-KA becomes more significant as the number of CoTs increases.

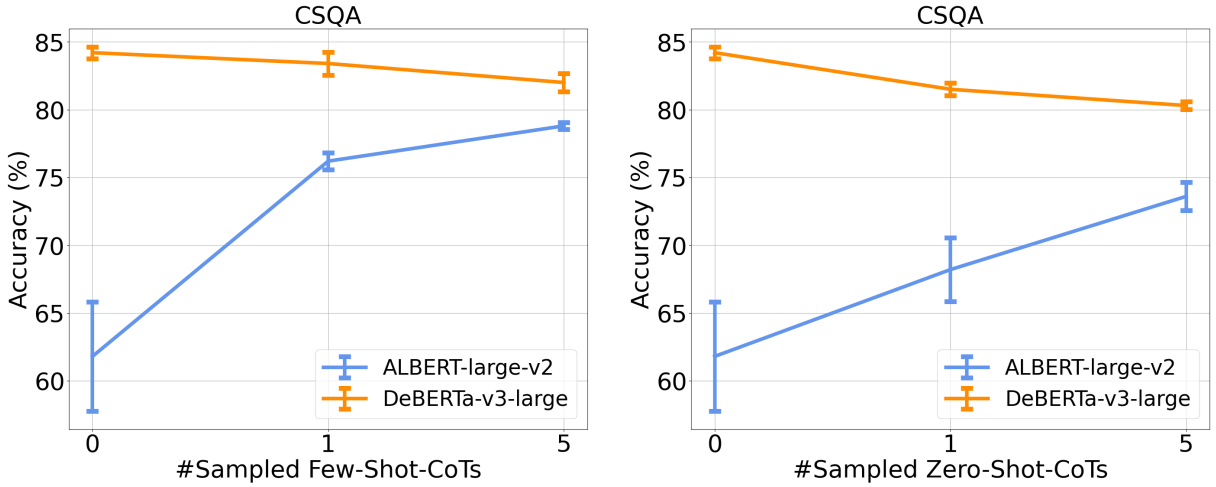

Figure 5: Accuracy of CSQA. Performance over various numbers of CoTs used in CoT-KA.

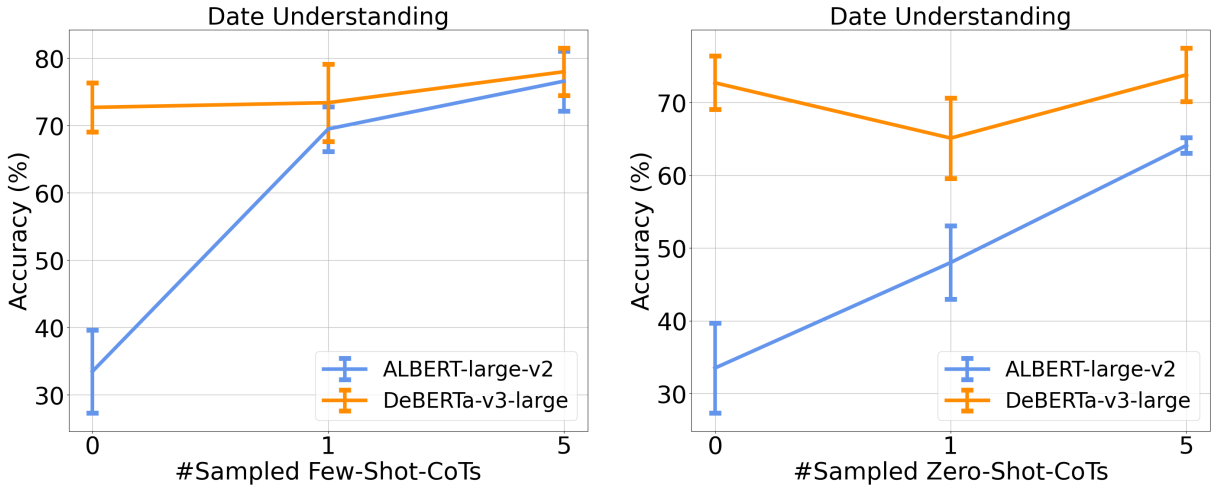

Figure 6: Accuracy of Date Understanding. Performance over various numbers of CoTs used in CoT-KA.

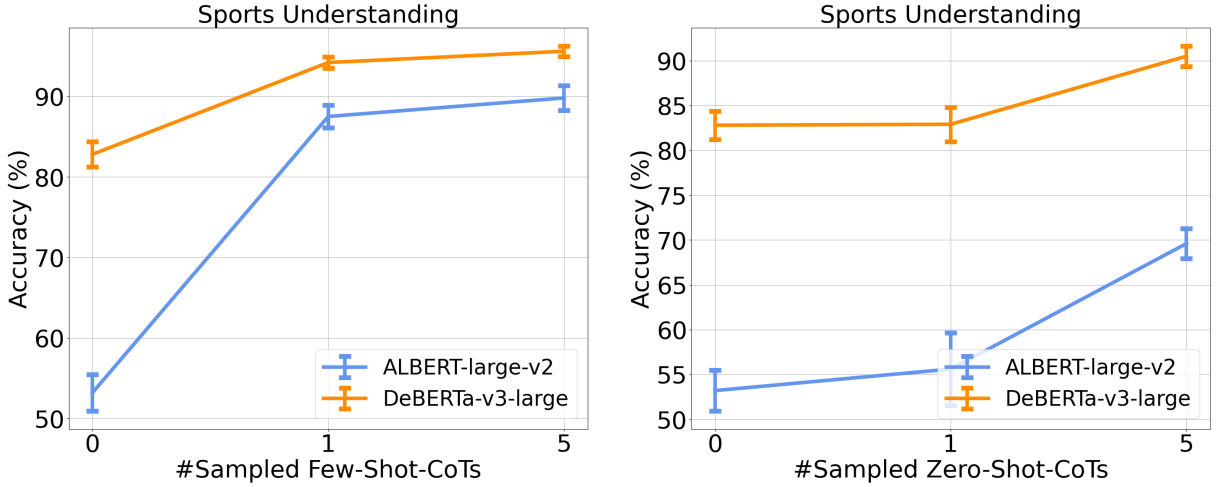

Figure 7: Accuracy of Sports Understanding. Performance over various numbers of CoTs used in CoT-KA.

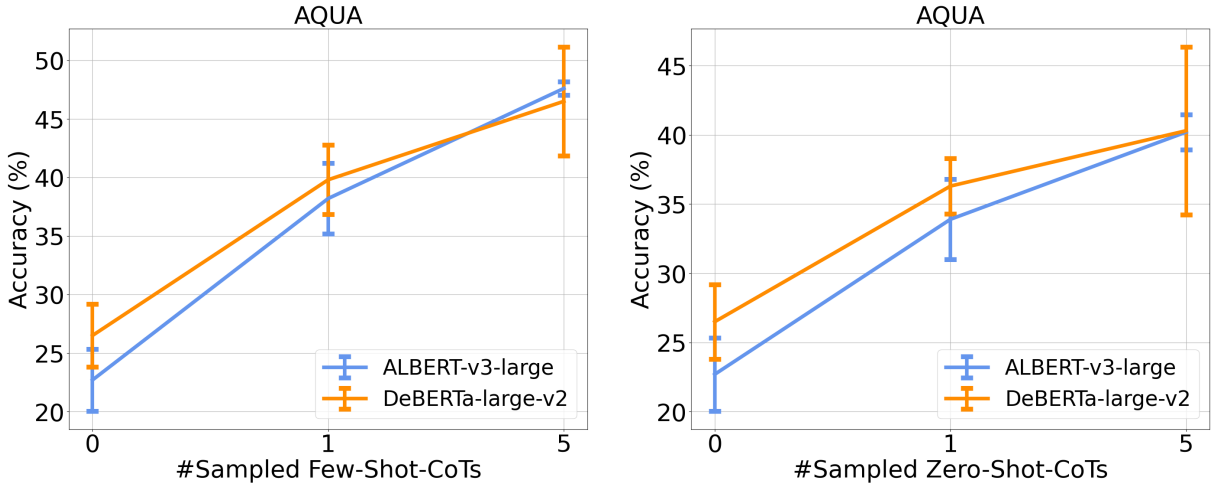

Figure 8: Accuracy of AQUA. Performance over various numbers of CoTs used in CoT-KA.

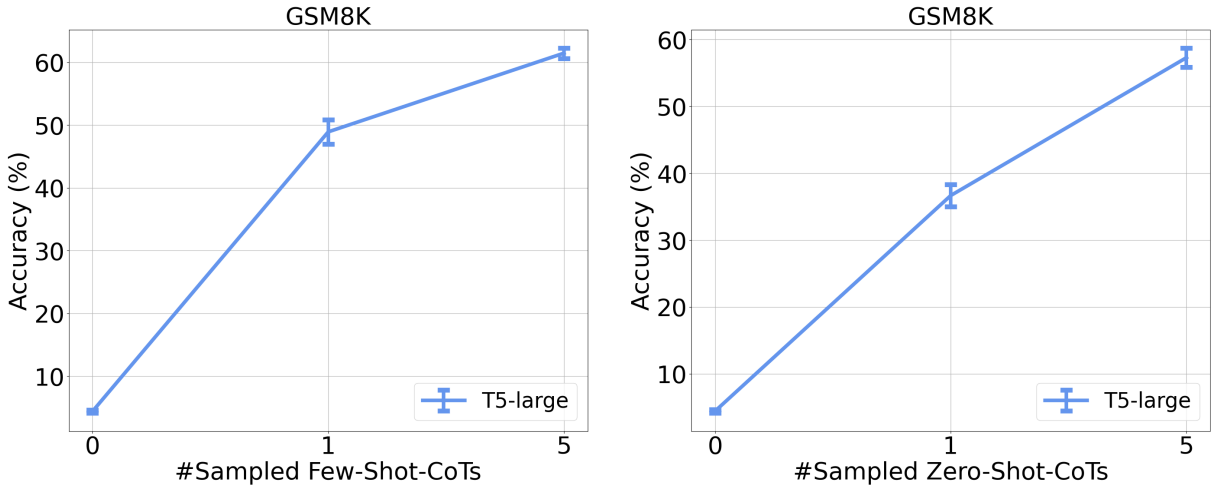

Figure 9: Accuracy of GSM8K. Performance over various numbers of CoTs used in CoT-KA.

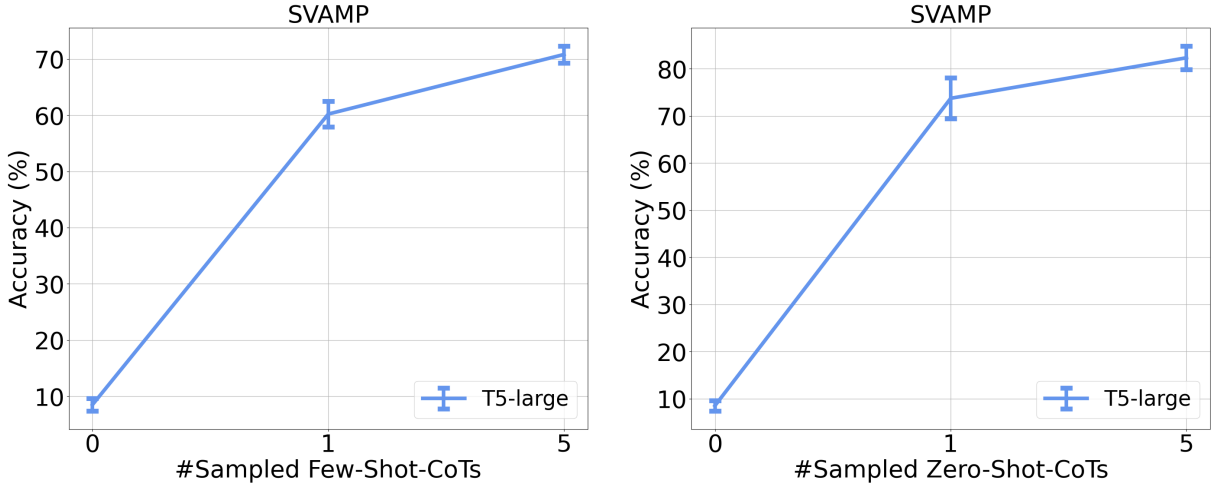

Figure 10: Accuracy of SVAMP. Performance over various numbers of CoTs used in CoT-KA.

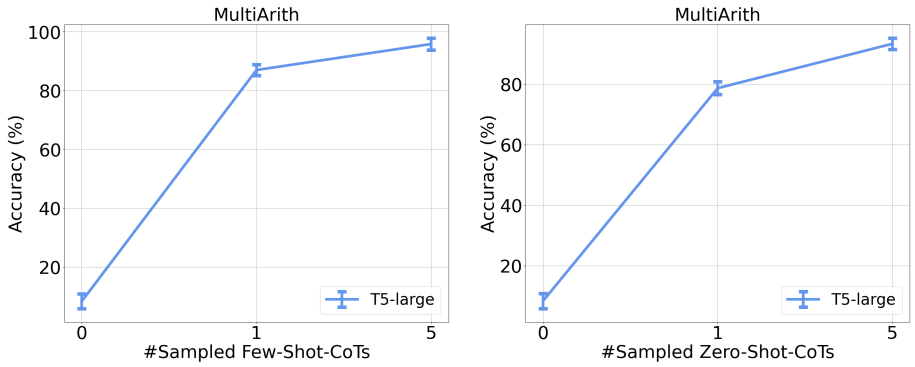

Figure 11: Accuracy of MultiArith. Performance over various numbers of CoTs used in CoT-KA.

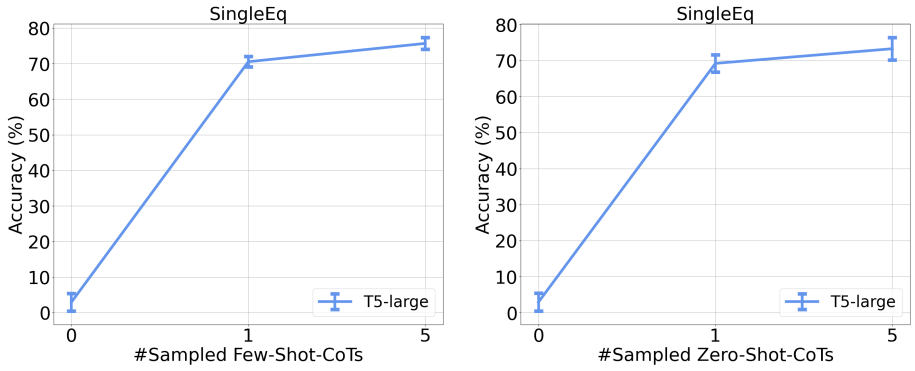

Figure 12: Accuracy of SingleEq. Performance over various numbers of CoTs used in CoT-KA.

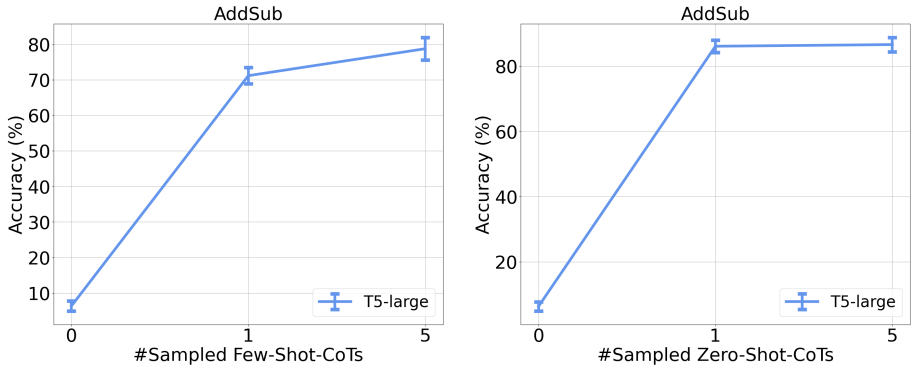

Figure 13: Accuracy of AddSub. Performance over various numbers of CoTs used in CoT-KA.

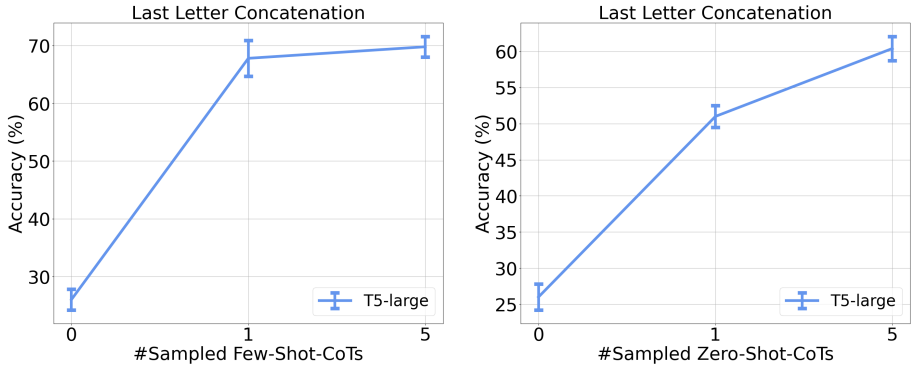

Figure 14: Accuracy of Last Letter Concatenation. Performance over various numbers of CoTs used in CoT-KA.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 5 Experiments, Section 7 Limitations.

☒ A2. Did you discuss any potential risks of your work?
Our paper mainly used GPT3 and chain-of-thought prompting for application, the risk of large language models and chain-of-thought prompting were discussed in references we cited.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract, Section 1 introduction.

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

B ☑ Did you use or create scientific artifacts?

Section 3 Pilot Study, Section 5 Experiments.

☑ B1. Did you cite the creators of artifacts you used?
Section 1 introduction, Section 3 Pilot Study, Section 5 Experiments.

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
We used the same publicly available dataset as in the existing work, and we did not discuss this matter specifically.

☒ 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)?
We used the same publicly available dataset as in the existing work, and we did not discuss this matter specifically.

☒ 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?
We used the same publicly available dataset as in the existing work, and we did not discuss this matter specifically.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
We used the same publicly available dataset as in the existing work, and we did not discuss this matter specifically.

☑ 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.
Appendix: A.2 Datasets.

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

C ☑ Did you run computational experiments?

Section 3 Pilot Study, Section 5 Experiments.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix: A.1 Implementation.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5 Experiments, Appendix: A.1 Implementation.

☑ 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?
Section 5 Experiments, Appendix B.

☑ 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.)?
Section 1 Introduction, Section 5 Experiments, Appendix: A.1 Implementation.

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.