S^{3}Prompt: Instructing the Model with Self-calibration, Self-recall and Self-aggregation to Improve In-context Learning

Jianting Liu [†,♣], **Junda Chen** [†,♠(✉)]
University of Electronic Science and Technology of China♠ , Shortest Path Technology♣
{liujianting873, junda.chen.27227}@gmail.com

Abstract

Large language models achieve impressive results by inferring conditional probability distributions in the context of user input to generate responses. However, they still have the following limitations in practical applications: 1) User queries are often colloquial and do not conform to the conditional probability distribution of LLM. 2) Unsupervised generation and recall of in-context examples (compared to random sampling) remains an open problem. To alleviate the above problems, we propose a novel **Self-calibration, Self-recall and Self-aggregation prompt pipeline** (**S^{3}Prompt**). Specifically, we first design a question calibration prompt to align colloquial queries with LLM context. Secondly, we construct a candidate recall prompt that allows LLM to generate potential background information, which is different from traditional retrieval-based QA. Finally, we design an information aggregation prompt to generate the final answer by aggregating the recalled information. Notably, we find that the self-generated information by LLM has a smaller gap when fused with LLM. We conducted comprehensive experiments on various datasets, including numerical reasoning, common sense reasoning, logical reasoning, and reading comprehension. The results showed that the performance of LLM can be significantly improved by using question calibration, candidate recall, and information aggregation, without requiring annotated datasets and model parameter updates.

Keywords: prompt tuning, in-context learning, large language model

1. Introduction

Large language models (LLMs) (OpenAI, 2023; Anil et al., 2023) have revolutionized natural language task solving through prompting(Brown et al., 2020) and have demonstrated impressive capabilities in a variety of natural language processing tasks. This technique involves conditioning the language model with an instruction (zero-shot) and sometimes augmenting with a small set of task-specific examples (few-shot), enabling it to generalize and respond effectively to tasks. A rapidly advancing body of research has introduced techniques to enhance these prompting methodologies. Notably, chain-of-thought prompting (Wei et al., 2023; Kojima et al., 2022) has emerged as a powerful method for enhancing language model performance in NLP tasks. LTM prompting (Zhou et al., 2022a), Tree of Thoughts (Yao et al., 2023), Graph of thoughts (Besta et al., 2023), and Algorithm of thoughts (Sel et al., 2023) further support chain-of-thought by breaking down complex problems into simpler sub-problems. Kojima et al. (2022) further demonstrate the LLM's zero-shot reasoning ability by adding "Let's think step by step" before the LLM outputs the answer. MetaICL(Min et al., 2021) find that after allowing the model to experience as many NLP tasks as possible, the model can achieve good results even for tasks it has never seen before, whether it is zero-shot learning or contextual learning.

Although LLMs pre-trained on large corpora can solve certain problems in different domains, their performance in specific vertical domains varies and requires examples from different scenarios. Recently, three techniques have been introduced by rapidly developing research institutions to enhance the ability of LLMs in different domains. The first is supervised fine-tuning: FLAN (Wei et al., 2021) annotates multiple NLP tasks and divides them into multiple clusters based on their task types and objectives, and then, it randomly selects all tasks within one cluster for evaluation. MetaICL (Min et al., 2021) found that after exposing the model to as many NLP tasks as possible, the model can achieve good results even for unseen tasks, whether it is zero-shot learning or in-context learning. InstructGPT (Ouyang et al., 2022) improves the GPT-3's instruction-following ability by fine-tuning it on many diverse crowd-sourced instruction-answer pairs. The second is retrieve&generation: (Liu et al., 2021), Zhang et al. (2022a) and Chen et al. (2023) form annotated data using SentenceBERT-based method (Reimers and Gurevych, 2019; Liang et al., 2019a,b; Song et al., 2022; Wang et al., 2022b; Xue et al., 2023) or BM25 (Robertson et al., 2004) focused retrieval of relevant examples to enhance in-context learning in the LLM. Shi et al. (2022a) Utilize an annotated dataset to train a retriever with LLM feedback to retrieve demonstrations useful for test examples. Li and Qiu (2023) proposes to select a representative set of examples from

[†] The authors contributed equally to this work.

LREC-COLING 2024, pages 14259–14271
20-25 May, 2024.

the training set, which significantly improves ICL over a randomly selected baseline. The third is Fine-tuning with LLM-generated data: Zelikman et al. (2022) propose to let the LLM generate rationales for annotated input/output pairs and train itself to enhance the reasoning ability. Magister et al. (2022) and Fu et al. (2023) use the reasoning paths generated by large LM to improve the small LM's reasoning capability. More recently, Huang et al. (2022) apply self-training with varying data formats on PaLM (Chowdhery et al., 2022) and improve its reasoning performance.

While standard prompting and chain of thought prompting exhibit impressive capabilities and find applications across various domains. However, they still face the following problems: 1) User input is usually colloquial and may contain spelling errors. This colloquial query does not comply with the conditional probability distribution of LLM. 2) LLMs such as ChatGPT and GPT4 usually interact with users through open APIs, which face the challenge of untrainable or expensive training costs. In non-finetune scenarios, the annotation of in-context learning examples faces high costs, and the selection of examples is crucial to the effectiveness of reasoning, the strategies may sometimes lead to repetitive responses and chain-of-thought prompting, is susceptible to symbol mapping errors, hallucinations, and omission of intermediate steps (Kojima et al., 2022).

In order to alleviate the above problems, in this paper we propose S^{3}prompt which is a prompting strategy that builds upon existing prompting approaches. We provide instructions for the S^3 prompt in the ICL settings. Details are as follows: first, S^{3}Prompt draws inspiration from the innate cognitive strategies employed by humans, precisely the act of self-questioning, when answering queries. By checking the input questions and revising them before answering the questions, people usually refine their ideas and even discover the original Errors in the input (Joseph and Ross, 2018; Joseph et al., 2019). Secondly, examples are crucial to LLM reasoning in in-context learning. To address this issue, we designed multiple reasoning paths and cross-validation to obtain high-confidence examples. Then, we constructed a vector retriever using sentence-bert and faiss, which can retrieve the semantically closest examples based on the user's input query. Finally, considering that the semantically closest examples may not be the most needed for LLM reasoning about the target question, we also designed a self-aggregation module , which will select the most useful example for the target question from the top K examples retrieved by semantic. We empirically evaluate our approach against various prompting baselines using a wide variety of model families with different sizes, including Codex (code-davinci-002), GPT-3.5-Turbo[1], Starcoder-15B, Llama-13B, and GPT-J-6B. Our results show that S^{3}Prompt significantly improves the performance of language models on various NLP tasks.

The main contributions of this work can be summarized as follows. First, we systematically discuss the feasibility of query representation calibration , candidate recall and aggregation reasoning in in-context learning. And a novel Self-calibration, Self-recall and Self-aggregation reasoning paradigm is proposed to better understand the problem and deduce the correct answer. Secondly, the proposed S^{3}Prompt paradigm effectively enhances the distribution alignment relationship between the problem and LLM. And through retrieval + aggregation LLM's own knowledge can be better utilized, enabling it to successfully model complex problems and achieve efficient reasoning. Finally, We conducted comprehensive experiments on various datasets, including numerical reasoning, common sense reasoning, logical reasoning, and reading comprehension. The results showed that the performance of LLM can be significantly improved by using question calibration, candidate recall, and information aggregation, without requiring annotated datasets and model parameter updates.

2. RELATED WORK

Prompting The success of large language models has sparked interest in using prompting techniques to improve task performance (Brown et al., 2020). While recent research has focused on task-specific instruction tuning, either by fine-tuning the entire model (Raffel et al., 2020; Wei et al., 2021; Sanh et al., 2021; Wang et al., 2022d) or maintaining task-specific parameters (Li and Liang, 2021; Lester et al., 2021), our work is a general prompting approach that can improve contextual learning ability without any fine-tuning.

LLM Reasoning The use of language models to generate intermediate steps has been widely validated in the context of solving reasoning tasks, whether through training (Nye et al., 2021; Zelikman et al., 2022), zero-shot (Kojima et al., 2022), or few-shot prompting (Wei et al., 2022). Recent work has focused on problem decomposition and teaching language models to answer subtasks, ultimately answering complex questions (Zhou et al., 2022a; Dua et al., 2022; Wang et al., 2022a; Zhou et al., 2022b). S^{3}Prompt is orthogonal to these methods, enhancing input queries rather than assigning generation. Therefore, it can be easily extended by any of these prompting strategies.

[1] https://openai.com/blog/chatgpt/. We use gpt-3.5-turbo-0301 snapshot from May 2023

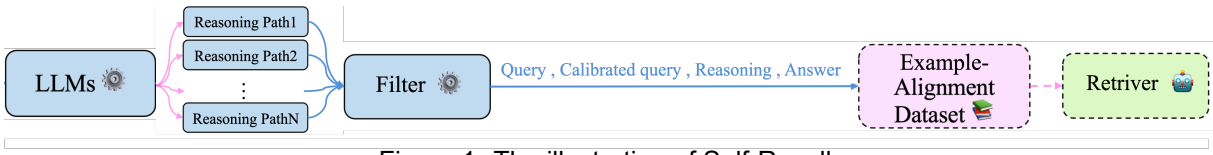

Figure 1: The illustration of Self-Recall.

A closely related research direction involves exploring the interpretability and consistency of the fundamental principles generated by large models. Recent works (Wang et al., 2023; Imani et al., 2023; Madaan and Yazdanbakhsh, 2022) have improved the interpretability of numerical tasks, while (Yao et al., 2022; Jung et al., 2022; Yao et al., 2023; Zheng et al., 2022; Fei et al., 2022) have improved the performance of various arithmetic and logical reasoning tasks by leveraging the consistency between multiple generated principles.

Demonstration Selection in-context learning (ICL) primarily retrieves relevant input/output pairs for a provided test example from an annotated dataset. Liu et al. (2021) proposes to utilize dense semantic embedders to retrieve relevant examples to improve ICL. Agrawal et al. (2022) Example of retrieving machine translation ICL using BM25. Das et al. (2021) and Hu et al. (2022) explore knowledge-based question answering and dialogue state tracking respectively, and design specialized target similarity to train demonstration retrieval. Rubin et al. (2021) propose to train demonstration retrieval via feedback from LM and demonstrate its effectiveness on semantic parsing. Shi et al. (2022b) further explores cross-language semantic parsing using similar training signals as Rubin et al. (2021). Lyu et al. (2022) uses random labels to retrieve relevant examples and proposes heuristics to reduce the negative impact of incorrect labels. Recently, Li et al. (2023b) proposed UDR, a unified demonstration retriever for various NLP tasks, which is trained on approximately 40 annotated datasets with unified LM feedback. Although most of these methods rely on high-quality annotated datasets and only explore contextual learning without underlying principles, But S^3Prompt allows LLM to improve itself without annotated datasets and parameter updates.

Model Augmentation In addition, some works attempt to use data generated by LLMs for model enhancement. Ye et al. (2022b); Gao et al. (2022); Ye et al. (2022a) propose ZeroGen, ProGen, and ZeroGen$^+$ to use LLM-generated datasets to enhance small models such as LSTM or DistilBERT. Ho et al. (2022); Magister et al. (2022); Fu et al. (2023); Xue et al. (2024); Liu et al. (2023a,b) suggest using LLMs to generate reasoning paths and teach small language models to reason. Wang et al. (2022c) and Honovich et al. (2022) use LLMs to generate instruction data and improve the LLM's instruction tracking ability. Schick et al. (2023); Ma et al. (2022) proposes ToolFormer, which learns how to use various tools through self-generated data. Huang et al. (2022) and Zelikman et al. (2022) use LLMs to generate reasoning paths and improve themselves using annotated and unannotated datasets, respectively. Shao et al. (2023); Zhang et al. (2022b) have LLMs generate COT data themselves. Li et al. (2022) uses LLMs to generate knowledge bases and improve their open-domain QA capabilities.

These works reveal the importance of promoting high-quality reasoning in the thinking chain process. However, generating and utilizing high-quality examples that fit the LLM distribution makes this process challenging. This prompted us to design a better mechanism (S^3Prompt) to prompt language models in in-context learning. It is a new prompting scheme that allows LLM to output high-quality answers in a structured way through self-calibration, self-recall, and self-aggregation.

3. Background: Chain of Thought

The large language model has shown impressive reasoning abilities on various tasks. Chain-of-Thoughts (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) is the most prevailing way to let the LLM reason, i.e., generate a coherent series of intermediate reasoning steps that lead to the final answer. Chain-of-Thoughts (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) is the most common way to inspire LLM reasoning, and its core idea is that by simulating the reasoning process of human thinking, we can better understand the essence and mechanism of human thinking, and thus provide a more in-depth and comprehensive understanding for problem solving. And the Zero-shot COT and Few-shot COT are both reasoning methods based on the COT thinking chain model. Few-shot COT is a reasoning method that learns from a small number of examples, aiming to control the model's reasoning direction by letting the LLM learn the step-by-step reasoning process of few-shot examples, thus achieving significant improvements in complex tasks. In contrast, Zero-shot COT does not use examples, and its core is to extract the step-by-step reasoning process of the current problem using "Let's think step by step". In other words, the main difference between them is that Zero-shot COT is suitable for situations where there is no sample reference at all, while Few-shot COT is suitable for situations where there are few sample references but still some. Specifically, the

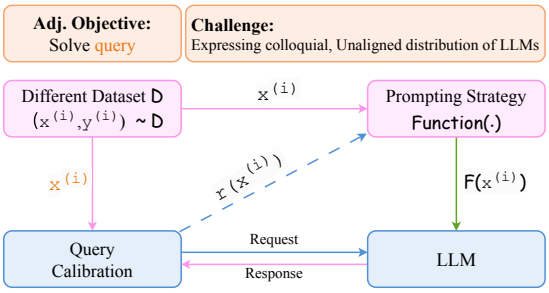

Figure 2: The illustration of Self-Calibration.

Few-Shot-CoT gets the answer as follows:

$$s = \text{LLM}(e_1, e_2, \cdots, q_{test}) \quad (1)$$

where $e_i = [x_i, r_i, a_i]$ is the ith demonstration, including input, reasons and answers. It first decodes s from the LLM via the few-shot Chain-of-Thought prompting, and parses s to get the final answer. Since the demonstration are usually in the format: "[Input] [Reason] *The answer is* [Answer]", so the answer can be easily parsed from s via the trigger "*The answer is*". Similarly, Zero-Shot-CoT uses answer triggers, e.g., "*Therefore, the answer is*", to extract the final answer from the zero-shot reasoning path generated by LLM (Wei et al., 2022).

In this work, we focus on enabling LLM to enhance reasoning capabilities and in-context learning through self-calibration, self-recall, and self-aggregation, without the need for parameter updates and annotated data sets.

4. Method

4.1. Self-Calibration

According to human language habits, there is a huge difference between written language and spoken language. Written language is generally more formal, with direct content and clear meaning, while spoken language often includes many modal particles or oral language habits. We found that there is a significant difference in semantic understanding of sentences using LLM in the two scenarios. Queries in spoken language scenarios contain more irrelevant and chaotic information, which leads to erroneous prediction results in LLM output. To address the above issue, we designed a self-calibration module as shown in Figure 2. Specifically, we use the task-independent prompt "Please help me revise and rephrase this query to make it clearer." to rewrite the query. The goal is to make the regenerated query more clear in meaning and help users to initially clarify their intentions.

4.2. Self-Recall

The selection of examples is crucial in in-context learning, different selection schemes can cause significant differences in results. LLM can provide reference "learning" samples based on examples to guide model inference and complete thinking chains. However, there are two problems with example construction in real-world scenarios: one is the lack of thinking chains, and the other is the gap between manually constructed thinking chains and LLM itself, which may exacerbate the model's hallucination. To address these issues, we designed a self-recall module, which includes two modules: example generation and example recall.

As show in 1. Firstly, in the example generation module, we let LLM perform extensive adaptive inference on the training dataset and save the question/reason/answer to an external knowledge base. For each data, we sample multiple inference paths by controlling the temperature coefficient, represented as $[r_1, r_2 \cdots, r_n]$ and $[a_1, a_2 \cdots, a_n]$. Then, we use majority voting and intra-cluster answer divergence to select the examples that need to be saved.

Secondly, in the example recall module, we select examples that are semantically closer to the current query by recalling through semantic vectors. We use sentence-BERT to vectorize the questions in the recall candidates and save them in a vector retrieval database. We recall the TOP-N semantically closest examples from the vector database using the query to be inferred.

Finally, we encapsulate self-recall as a retrieval interface, with the specific formula as follows:

$$\textit{Examples} = \text{Retriver}(q_{test}) \quad (2)$$

which can recall the top N most relevant examples ($e_i = [x_i, r_i, a_i]$ < input, reasons, answers>) based on the input query at the semantic level.

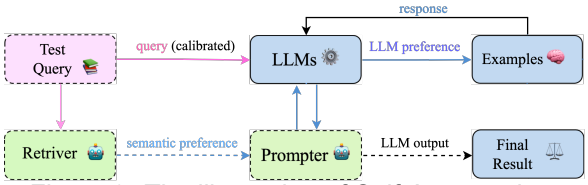

Figure 3: The illustration of Self-Aggregation.

4.3. Self-Aggregation

In order to better use the LLM to understand the goal of finding useful examples, we designed a self-aggregation module in addition to semantic retrieval. Its goal is to select the most relevant examples from the semantic recalled examples and make their output easy to parse using a predefined format. We designed a prompt for selecting examples, such as "You must select the most useful example [idx] from the input examples", as shown in the following formula:

$$e_{best} = \text{LLMRetriver}(e_1, e_2, \cdots, q_{test}), \quad (3)$$

$$\textit{Answer} = \text{LLM}(e_{best}, q_{test}) \quad (4)$$

The final *Answer* is the result obtained by concatenating the most relevant example with q_{test} and requesting LLM.

5. Evaluation Setup

5.1. Benchmarks

We conducted a comprehensive evaluation of S^{3}Prompt on a range of natural language processing tasks, with a particular focus on four types, including 14 widely recognized benchmarks. We also evaluated S^{3}Prompt in the presence of perturbed sentences in queries. To ensure a thorough and comprehensive evaluation, we used four different causal language models. In this section, we will delve into the details of the evaluation setup.

Numerical Reasoning We evaluate numerical reasoning tasks from (Wei et al., 2023) for a fair comparison between the methods including, *GSM8K* (Cobbe et al., 2021), *SVAMP* (Patel et al., 2021), *AQUA-RAT* (Ling et al., 2017), *SingleOp* and *MultiArith* subsets from (Roy and Roth, 2016) , which can measure the reasoning ability of LLM. Additionally, we examine the performance of S^{3}Prompt on the high school mathematics subset of the *MMLU* dataset (Hendrycks et al., 2021a,b) and the *GSMIC-4k* dataset (Shi et al., 2023), which focuses explicitly on queries containing perturbations.

Logical Reasoning For logical reasoning, we assess the *Shuffled Objects* (tracking three objects), *Date Understanding* tasks from bigBench (Ghazal et al., 2013), *LogiQA* (Liu et al., 2020) and generate 1000 random samples with two trials of flipping the coin for *Coin Flipping* task (Wei et al., 2023) , which can measure the logical reasoning ability of LLM.

Reading Comprehension While we evaluate multiple numerical subsets of *DROP* (Dua et al., 2019), (including Football, Non-football, Census, and Break(Wolfson et al., 2020) from the *QDMR* dev subset) and could also be included in the arithmetic benchmarks, we group it with *SQuAD* (Rajpurkar et al., 2016) based on the query style. We evaluate S^{3}Prompt on *DROP* (Dua et al., 2019) and *SQuAD* (Rajpurkar et al., 2016) as two standard reading comprehension benchmarks. The Football subset of the DROP dataset was curated by applying keyword-based filtering with the keyword ”yard” (Zhou et al., 2022a), and the Census subset was created by selectively filtering passages that contained the terms ”population” and ”census.” , which can measure the reading comprehension ability of LLM.

Commonsense Reasoning For commonsense reasoning, we use *StrategyQA* (Geva et al., 2021), *Winogrande* (ai2, 2019) datasets to assess the performance of S^{3}Prompt on tasks that involve simpler queries but require factual knowledge.

5.2. Language models

For our experiments, we use code-davinci-002 (Chen et al., 2021) as the primary model for all tasks since this model is free to evaluate and has a strong in-context learning ability. And we present the results on a subset of datasets on GPT-3.5-Turbo, a model comparable to the size of code-davinci-002. We also experiment with the smaller and publicly available models such as StarCoder-15B (Li et al., 2023a), Llama-13B (Touvron et al., 2023), and GPT-J-6B (Wang and Komatsuzaki, 2021) specifically on synthetic and simpler tasks, including coin flipping, SingleOp, SVAMP and date understanding. To experiment with smaller models, we utilize the HuggingFace[2] models.

5.3. Prompts

Few-Shot Exemplars For a fair comparison of methods, we use the same exemplars introduced in (Wei et al., 2023) for the GSM8K, SVAMP, SingleOp, MultiArith, Date Understanding, and Coin-Flipping tasks across all models. Additionally, we evaluate with the prompts suggested by (Zhou et al., 2022a) in the least-to-most prompting method for the GSM8K, SVAMP, MultiArith, and DROP subsets. Furthermore, we propose a new set of prompts specifically for the DROP Census subset since no prior proposals exist.

Zero-Shot-CoT Prompts As proposed in (Kojima et al., 2022), we employ the prompt ”Let’s think step by step.” in stage 1. In stage 2, we extract the answer using different prompts depending on the type of task. For multiple-choice tasks, we utilize prompts like ”From (a) through (e), the answer is.” For other tasks, we use the phrase ”So, the answer is.”

5.4. Implementation Details

We use the same few-shot CoT examples as Wei et al. (2022) and Zhou et al. (2022a), respectively. For the left datasets, we randomly select questions from the training set and use LLM to generate reasoning paths and get their few-shot CoT examples. And we use the temperature $T = 1.2$ to encourage more diverse reasoning paths and use $\tau = 0.3$ as uncertainty threshold unless otherwise specified. For self-recall, we use Sentence-BERT (Reimers and Gurevych, 2019) and the vector retrieval tool

[2]Source code at https://huggingface.co/EleutherAI

Model	Dataset	Zero-shot				Few-shot			
		Standard		CoT		Standard		CoT	
	S^{3}Prompt	✗	✓	✗	✓	✗	✓	✗	✓
Code-davinci-002	GSM8K	16.4	22.6(+5.2)	49.3	54.1(+4.8)	19.2	23.5(+4.3)	61.1	68.6(+7.5)
	SVAMP	66.8	74.1(+7.3)	66.5	74.3(+7.8)	69.8	76.5(+6.7)	75.2	81.3(+6.1)
	MultiArith	31.0	48.8(+17.8)	76.0	81.6(+5.6)	44.0	58.8(+14.8)	96.1	98.6(+2.5)
	SingleOp	91.6	92.3(+0.7)	82.9	92.7(+9.8)	93.2	94.6(+1.4)	92.8	95.1(+2.3)
	Shuffled Objects	36.4	37.4(+1.0)	42.4	58.9(+15.5)	34.8	36.6(+1.8)	66.0	69.7(+3.7)
	Coin Flip	47.7	47.6(−0.1)	58.5	62.1 (+3.6)	99.6	100(+0.4)	100	100(+0.0)
	Date	44.2	43.9(−0.3)	39.0	47.4(+8.4)	49.3	51.0(+1.7)	65.6	69.8(+4.2)
	DROP(Football)	50.8	59.5(+8.7)	44.1	60.4(+16.3)	63.7	70.6(+6.9)	67.3	73.8(+6.5)
	DROP(Nonfootball)	43.2	57.3(+14.1)	39.7	52.7(+13.0)	57.1	64.5(+7.4)	69.2	74.2(+5.0)
	DROP(Census)	45.9	64.3(+18.4)	30.0	52.3(+22.3)	56.8	66.9(+10.1)	69.6	77.4(+7.8)
	DROP(Break)	43.7	56.9(+13.1)	38.2	52.5(+14.3)	55.5	63.7(+8.2)	65.3	69.7(+4.4)
	SQuAD(F1)	65.7	68.9(+3.2)	52.6	54.8(+2.2)	88.7	91.9(+3.2)	90.5	90.9(+0.4)
	AQUA-RAT	21.2	23.4(+2.2)	37.0	36.4(−0.6)	30.3	31.9(+1.6)	43.7	44.3(+0.6)
	MMLU-h	31.8	36.6(+4.8)	42.5	42.7(+0.2)	36.7	39.3(+2.6)	44.1	43.7(−0.4)
	logiQA	42.5	42.1(−0.4)	37.0	40.7(+3.7)	45.3	46.9(+1.6)	40.9	41.3(+0.4)
GPT-3.5 (Turbo)	GSM8K	5.6	24.8(+19.2)	75.7	77.6(+1.9)	31.3	34.5(+3.2)	75.1	84.4(+9.3)
	SVAMP	51.9	76.4(+24.5)	80.5	83.8(+3.3)	76.1	78.4(+2.3)	77.4	82.3(+5.9)
	MultiArith	76.5	83.7(+7.2)	93.4	96.6(+3.2)	83.4	90.5(+7.1)	97.8	98.8(+1.0)
	SingleOp	92.6	96.8(+4.2)	91.4	94.8(+3.4)	93.9	96.8(+2.0)	95.7	96.8(+1.1)
	Shuffled Objects	26.9	26.7(−0.2)	79.5	82.5(+3.0)	30.6	36.4(+5.8)	68.8	75.2(+6.4)
	Coin Flip	76.7	86.8 (+10.1)	99.8	99.8(+0.0)	90.0	95.6(+5.6)	100	100(+0.0)
	Date	45.7	45.1(+0.4)	46.6	47.0(+0.4)	50.4	51.3(+1.9)	64.5	66.8(+2.3)
	DROP(Break)	47.1	52.8(+5.7)	51.9	51.7(−0.2)	59.9	62.7(+2.8)	61.6	66.7(+5.1)
	SQuAD(F1)	79.1	80.6(+1.5)	62.1	59.4(−2.7)	76.4	84.0(+7.6)	85.3	86.8(+1.5)
	AQUA-RAT	27.9	28.6(+0.7)	51.1	51.0(−0.1)	33.4	35.8 (+2.4)	39.7	57.6(+17.9)
	MMLU-h	25.6	31.5(+5.9)	51.1	53.3(+2.2)	34.1	36.4 (+2.2)	28.9	41.1(+12.2)
	logiQA	36.2	38.5(+2.3)	37.6	39.0(+1.4)	45.1	44.6(−0.5)	32.5	32.3 (−0.2)
Starcoder (15B)	SingleOp	63.1	66.9(+3.8)	53.5	66.6 (+13.1)	64.0	70.2(+6.2)	68.8	74.7(+5.9)
	SVAMP	35.6	37.5(+1.9)	30.9	37.6(+6.7)	32.4	38.7(+6.3)	30.2	41.1(+10.9)
	Coin Flip	55.4	55.3(−0.1)	51.6	52.0(+0.4)	98.6	99.8(+1.2)	100.0	100.0(+0.0)
	Date	15.9	19.2(+3.3)	20.6	21.8(+2.2)	24.4	26.6(+2.2)	38.4	38.8(+0.4)
Llama (13B)	SingleOp	78.4	81.0(+2.6)	64.9	75.0(+10.1)	81.1	85.4(+4.3)	81.3	86.6(+5.3)
	SVAMP	36.4	46.3(+9.9)	30.7	38.3(+7.6)	39.2	46.5(+7.3)	38.7	47.1(+8.4)
	Coin Flip	53.2	55.3(+2.1)	51.0	51.2(+0.2)	89.8	92.9(+2.1)	100.0	100.0(+0.0)
	Date	24.9	26.6(+1.7)	22.5	27.0(+4.5)	32.8	35.1(+12.3)	42.3	45.9 (+3.6)
GPT-J (6B)	SingleOp	-	-	-	-	37.2	39.9(+2.7)	45.3	48.5(+3.2)
	SVAMP	-	-	-	-	8.9	10.1(+1.2)	21.1	24.8(+3.7)
	Coin Flip	-	-	-	-	81.3	81.3(+0.0)	80.6	96.4(+15.8)
	Date	-	-	-	-	13.2	13.8(+0.6)	11.1	15.8(+4.7)

Table 1: Performance Summary of S^{3}Prompt on all models. S^{3}Prompt consistently improves performance across different prompting strategies, showing significant improvements in zero-shot prompting scenarios. It outperforms the prior state of the art in numerical reasoning and reading comprehension tasks. However, we do not see consistent improvements on multiple choice tasks.

Faiss for semantic filtering. And we set the number of cluster l and the number of each cluster's memory candidates k as 4 and 10, respectively.

6. Results

6.1. Model Performance

As we focus on whether S^{3}Prompt can help LLM improve reasoning accuracy, we extensively compared our method with zero-shot, zero-shot CoT, few-shot, and few-shot CoT prompt strategies. Table 1 provides the overall results of S^{3}Prompt.

Result based on Code-davinci-002: We find that S^{3}Prompt significantly outperforms baselines on most datasets, which shows S^{3}Prompt's best comprehensive reasoning capability on a series of NLP tasks. It is worth noting that S^{3}Prompt shows significant improvements in the zero-shot prompt scenario, especially for tasks with long query contexts, such as the DROP and SQuAD datasets. We observed an 18.4% improvement in accuracy for the zero-shot prompt DROP (census subset) dataset. Similarly, S^{3}Prompt using Zero-shot-CoT achieved an accuracy improvement (7.3%) on SVAMP, which makes the overall accuracy comparable to the few-shot CoT prompt. However, at the same time, S^{3}Prompt also achieved improvements when the baseline methods could not solve the task. For example, in the Shuffled Objects task involving three objects, S^{3}Prompt showed a slight improvement in zero-shot performance (from 36.4% to 37.4%). Nevertheless, it greatly improved the accuracy of Zero-shot-CoT (from 42.4% to 58.9%).

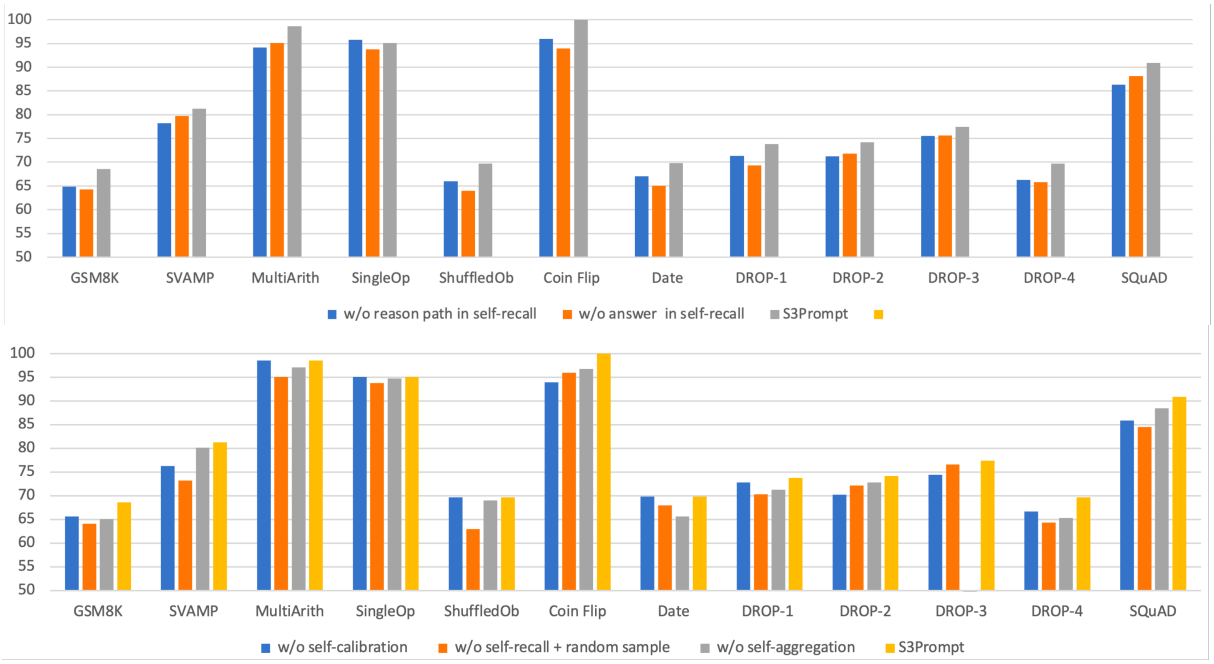

Figure 4: Results of component ablation experiment.

Result based on GPT-3.5-Turbo: We also evaluated the performance of S^{3}Prompt on non-code training models of similar size to Code-davinci-002, using GPT-3.5-Turbo for experiments on task subsets. Overall, these results are consistent with our previous experiments on code-davinci-002. For example, S^{3}Prompt significantly improved the accuracy on GSM8K, increasing from 75.1% to 84.4% in Fewshot-CoT. Meanwhile, we observed that the performance improvement in reading comprehension tasks (DROP and SQuAD) in the zero-shot scenario was not significant. After qualitative analysis, we observed that the model was less affected by semantically related examples and had higher relevance to the passage, which may be the reason for the insignificant performance improvement.

Result based on StarCoder-15B, Llama-13B and GPT-J-6B: And we evaluated the performance of S^{3}Prompt on smaller public models. Our evaluation included coin-flipping, SingleOp, and date-understanding tasks, as these smaller models are less capable of challenging reasoning tasks. The set includes a toy task and two relatively simple datasets, while date understanding is considered a challenging task on Bigbench. We can observe that the performance of the small model is significantly weaker than that of the large model. And we also observed inconsistent results in the causal chain reasoning, but performance was still improved with the S^{3}Prompt-enhanced.

6.2. Ablation Study

To evaluate the contribution of each component in S^{3}Prompt, we conducted a series of ablation studies as show in Figure 4. First, we examined whether the accuracy improvement brought by S^{3}Prompt was solely due to query calibration. Our research results show that both the original query and the calibrated query are crucial for achieving performance improvements. Second, Zero-Shot-CoT showed impressive performance on ChatGPT and outperformed Few-Shot-CoT on several datasets, indicating that introducing irrelevant CoT examples may not be necessary for LLM and may introduce noise. Meanwhile, S^{3}Prompt consistently outperformed Zero-Shot-CoT on almost all datasets, demonstrating the usefulness of example retrieval. Additionally, to verify the importance of reason and answer, we removed them from S^{3}Prompt. Overall, despite directly outputting the reasoning answer, their performance on multiple datasets is still better than Zero-Shot and Few-Shot. Furthermore, although (no reasoning) provides the question and answer pair, and (no answer) does not provide the reasoning process, they exhibit similar performance overall. That is, relevant examples and explicit reasoning in inference are important for significant performance improvement of S^{3}Prompt. In addition, we also compared the effect of random sampling and found that compared with semantic retrieval, there is still a decline, which is consistent with human intuition that queries that are more similar are more likely to have similar answers. Finally, we removed the self-aggregation module and used the output of semantic retrieval as an example. We can see that the performance still declines, indicating that LLM needs to perform example selection reasoning to fully utilize the retrieved examples. In summary, we can see that S^{3}Prompt fundamentally improves the performance of in-context learning and has broad applicability as a new emerging complex technology that utilizes prompting at mul-

tiple stages.

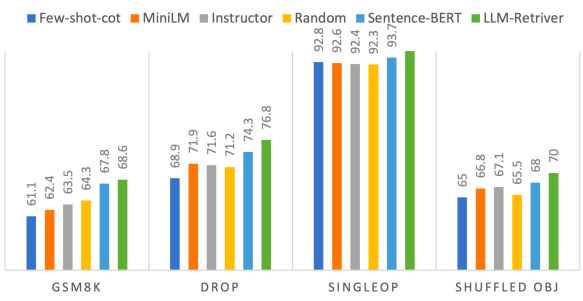

Figure 5: Performance comparison across different retrieval methods.

Ratio	0-0.4	0.4-0.8	0.8-1.2	1.2-1.6	1.6-2
Json Format					
1/1	0.74	0.63	0.55	0.43	0.31
1/3	0.71	0.60	0.52	0.40	0.29
1/5	0.69	0.59	0.48	0.39	0.27
3/1	0.76	0.68	0.57	0.49	0.35
5/1	0.75	0.68	0.60	0.51	0.37
Length Adaptive					
1/1	0.67	0.62	0.56	0.51	0.46
1/3	0.64	0.60	0.53	0.48	0.43
1/5	0.65	0.60	0.55	0.48	0.42
3/1	0.71	0.64	0.62	0.55	0.47
5/1	0.71	0.65	0.60	0.57	0.50

Table 2: Caption

Method	GSM8K	DROP	MMLU
Decoding Paths=8			
Zero-Shot-CoT$_{T=0.7}$	63.2	56.9	31.8
Zero-Shot-CoT$_{T=1}$	63.9	57.4	32.4
Zero-Shot-CoT$_{T=1.2}$	63.5	53.8	32.2
Few-Shot-CoT$_{T=0.7}$	73.7	69.5	39.7
Few-Shot-CoT$_{T=1}$	73.6	70.3	39.8
Few-Shot-CoT$_{T=1.2}$	73.4	70.4	40.2
S^{3}Prompt$_{T=0.7}$	**86.5**	76.0	**43.6**
S^{3}Prompt$_{T=1}$	84.9	**76.5**	42.5
S^{3}Prompt$_{T=1.2}$	86.3	76.3	42.4
Decoding Paths=16			
Zero-Shot-CoT$_{T=0.7}$	68.6	60.7	33.8
Zero-Shot-CoT$_{T=1}$	69.2	59.6	35.3
Zero-Shot-CoT$_{T=1.2}$	67.9	58.5	34.7
Few-Shot-CoT$_{T=0.7}$	72.3	71.1	41.4
Few-Shot-CoT$_{T=1}$	73.5	72.4	41.3
Few-Shot-CoT$_{T=1.2}$	71.4	71.9	40.9
S^{3}Prompt$_{T=0.7}$	86.3	**76.7**	43.7
S^{3}Prompt$_{T=1}$	**86.4**	76.4	**44.2**
S^{3}Prompt$_{T=1.2}$	86.1	76.6	44.1

Table 3: Performance comparison on self-consistency across different decoding temperatures and paths.

7. Analysis

7.1. Varying Decoding Budget and Temperatures

In this section, we evaluate S^{3}Prompt under self-consistency strategy (Wang et al., 2022c), across varying sampling times and temperatures. The result is shown in Table 3. We can see that S^{3}Prompt consistently outperforms Zero-Shot-CoT and Few-Shot-CoT across different decoding temperatures and budgets, which indicates the generality and stability of S^{3}Prompt. We notice the improvements slightly diminish when the decoding budget is more abundant, i.e., the number of decoding paths is higher. This may be because when there are more decoding paths, there are more ideas for solving problems, resulting in unconventional problem-solving ideas, which may cause LLM confusion during the example recall and aggregation stage.

7.2. Retrieval Method

To evaluate the effect of memory retrieval in S^{3}Prompt, we conduct experiments with varying retrieval methods on GSM8K , DROP, SQuAD and Shuffled Objects as shown in Figure 5. Besides the sentence-Bert used in MoT, we further compare two other semantic embedders, Instructor-base and "all-MiniLM-L6-v2". We observe that using only the SBERT for memory retrieval brought performance improvements over Few-Shot-CoT, which shows S^{3}Prompt' generality in the budget-limited inference scenario. After adding the LLM to retrieve memory, the performance gets further improvements, and this straightly indicates the effectiveness of the component of LLM-retrieval. Additionally, we see that LLM-retrieval outperforms all compared semantic embedders, which shows that the LLM can better capture the complicated reasoning logic than semantic embeddings. This also proves the effectiveness of self-aggregation in our method.

8. Conclusion

In this paper, we propose a novel self-calibrating, self-recall, and self-aggregation prompting pipeline **(S^{3}Prompt)** that can effectively align user query to LLM distributions and retrieve the most relevant contextual examples in an unsupervised manner without annotated datasets and parameter updates. To validate the effectiveness of S^{3}Prompt, we conducted intensive experiments on more than 10 public datasets in four major categories, including Numerical Reasoning, Logical Reasoning, Reading Comprehension, and Commonsense Reasoning. The experimental results show that our method consistently outperforms several strong baselines and demonstrating the effectiveness of S^{3}Prompt.

9. References

2019. Winogrande: An adversarial winograd schema challenge at scale.

Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2022. In-context examples selection for machine translation. *arXiv preprint arXiv:2212.02437*.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, et al. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

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. Zieglxer, 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.

Derek Chen, Kun Qian, and Zhou Yu. 2023. Stabilized in-context learning with pre-trained language models for few shot dialogue state tracking. *arXiv preprint arXiv:2302.05932*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

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, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. *arXiv preprint arXiv:2104.08762*.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. *arXiv preprint arXiv:2212.04092*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuan-Jing Huang. 2022. Cqg: A simple and effective controlled generation framework for multi-hop question generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. *arXiv preprint arXiv:2301.12726*.

Jiahui Gao, Renjie Pi, Yong Lin, Hang Xu, Jiacheng Ye, Zhiyong Wu, Xiaodan Liang, Zhenguo Li, and Lingpeng Kong. 2022. Zerogen ^+: Self-guided high-quality data generation in efficient zero-shot learning. *arXiv preprint arXiv:2205.12679*.

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.

Ahmad Ghazal, Tilmann Rabl, Minqing Hu, Francois Raab, Meikel Poess, Alain Crolotte, and Hans-Arno Jacobsen. 2013. Bigbench: Towards an industry standard benchmark for big data analytics. In *Proceedings of the 2013 ACM SIGMOD international conference on Management of data*, pages 1197–1208.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021a. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and

Jacob Steinhardt. 2021b. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. *arXiv preprint arXiv:2203.08568*.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Laurice M Joseph, Sheila Alber-Morgan, Leigh Ann Amspaugh, Kelsey Ross, Maria Helton, Moira Konrad, and Carrie Davenport. 2019. Stop to ask and respond: Effects of a small-group self-questioning intervention on reading comprehension performance. *Research and Practice in the Schools: The Official Journal of the Texas Association of School Psychologists*, 6(1):27.

Laurice M Joseph and Kelsey M Ross. 2018. Teaching middle school students with learning disabilities to comprehend text using self-questioning. *Intervention in School and Clinic*, 53(5):276–282.

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic prompting: Logically consistent reasoning with recursive explanations. *arXiv preprint arXiv:2205.11822*.

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*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Junlong Li, Zhuosheng Zhang, and Hai Zhao. 2022. Self-prompting large language models for open-domain qa. *arXiv preprint arXiv:2212.08635*.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy, Jason Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nour Fahmy, Urvashi Bhattacharyya, Wenhao Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2023a. Starcoder: may the source be with you!

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023b. Unified demonstration retriever for in-context learning. *arXiv preprint arXiv:2305.04320*.

Xiaonan Li and Xipeng Qiu. 2023. Finding supporting examples for in-context learning. *arXiv preprint arXiv:2302.13539*.

Di Liang, Fubao Zhang, Qi Zhang, and Xuanjing Huang. 2019a. Asynchronous deep interaction network for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2692–2700, Hong Kong, China. Association for Computational Linguistics.

Di Liang, Fubao Zhang, Weidong Zhang, Qi Zhang, Jinlan Fu, Minlong Peng, Tao Gui, and Xuanjing Huang. 2019b. Adaptive multi-attention network incorporating answer information for duplicate question detection. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 95–104.

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.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Yonghao Liu, Di Liang, Fang Fang, Sirui Wang, Wei Wu, and Rui Jiang. 2023a. Time-aware multiway adaptive fusion network for temporal knowledge graph question answering. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Yonghao Liu, Di Liang, Mengyu Li, Fausto Giunchiglia, Ximing Li, Sirui Wang, Wei Wu, Lan Huang, Xiaoyue Feng, and Renchu Guan. 2023b. Local and global: Temporal question answering via information fusion. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 5141–5149. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Xinxi Lyu, Sewon Min, Iz Beltagy, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. Z-icl: Zero-shot in-context learning with pseudo-demonstrations. *arXiv preprint arXiv:2212.09865*.

Ruotian Ma, Yiding Tan, Xin Zhou, Xuanting Chen, Di Liang, Sirui Wang, Wei Wu, Tao Gui, and Qi Zhang. 2022. Searching for optimal subword tokenization in cross-domain ner. *arXiv preprint arXiv:2206.03352*.

Aman Madaan and Amir Yazdanbakhsh. 2022. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

OpenAI. 2023. Gpt-4 technical report.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems?

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Stephen Robertson, Hugo Zaragoza, and Michael Taylor. 2004. Simple bm25 extension to multiple weighted fields. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 42–49.

Subhro Roy and Dan Roth. 2016. Solving general arithmetic word problems.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Lu Wang, Ruoxi Jia, and Ming Jin. 2023. Algorithm of thoughts: Enhancing exploration of ideas in large language models. *arXiv preprint arXiv:2308.10379*.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. *arXiv preprint arXiv:2302.00618*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context.

Peng Shi, Linfeng Song, Lifeng Jin, Haitao Mi, He Bai, Jimmy Lin, and Dong Yu. 2022a. Cross-lingual text-to-SQL semantic parsing with representation mixup. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5296–5306, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Peng Shi, Rui Zhang, He Bai, and Jimmy Lin. 2022b. Xricl: Cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-sql semantic parsing. *arXiv preprint arXiv:2210.13693*.

Jian Song, Di Liang, Rumei Li, Yuntao Li, Sirui Wang, Minlong Peng, Wei Wu, and Yongxin Yu. 2022. Improving semantic matching through dependency-enhanced pre-trained model with adaptive fusion. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 45–57, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Boshi Wang, Xiang Deng, and Huan Sun. 2022a. Iteratively prompt pre-trained language models for chain of thought. *arXiv preprint arXiv:2203.08383*.

Sirui Wang, Di Liang, Jian Song, Yuntao Li, and Wei Wu. 2022b. DABERT: Dual attention enhanced BERT for semantic matching. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1645–1654, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. 2022d. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

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*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198.

Chao Xue, Di Liang, Pengfei Wang, and Jing Zhang. 2024. Question calibration and multi-hop modeling for temporal question answering. *arXiv preprint arXiv:2402.13188*.

Chao Xue, Di Liang, Sirui Wang, Jing Zhang, and Wei Wu. 2023. Dual path modeling for semantic matching by perceiving subtle conflicts. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Jiacheng Ye, Jiahui Gao, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022a. Progen: Progressive zero-shot dataset generation via in-context feedback. *arXiv preprint arXiv:2210.12329*.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022b. Zerogen: Efficient zero-shot learning via dataset generation. *arXiv preprint arXiv:2202.07922*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022a. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Rui Zheng, Rong Bao, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Robust lottery tickets for pre-trained language models. *arXiv preprint arXiv:2211.03013*.

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

Hattie Zhou, Azade Nova, Hugo Larochelle, Aaron Courville, Behnam Neyshabur, and Hanie Sedghi. 2022b. Teaching algorithmic reasoning via in-context learning. *arXiv preprint arXiv:2211.09066*.