Cybernetics and Intelligence

Review

Advances in reasoning by prompting large language models: A survey

Ruixin Hong[1,2,3], Xinyu Pang[1,2,3], Changshui Zhang[1,2,3]✉

Cite this article: Hong RX, Pang XY, Zhang CS. *Cybernetics and Intelligence* 2026, **1**(1): 9390004. https://doi.org/10.26599/CAI.2024.9390004

ABSTRACT: Reasoning is not only an essential aspect of human intelligence but also one of the main research topics in artificial intelligence. With the recent revolutionary developments in natural language processing, it has been observed that large language models possess a degree of reasoning capabilities. When elicited by prompting, these models can exhibit impressive performance in various reasoning tasks. In this paper, we survey the recent advances in reasoning by prompting large language models. We provide an overview of key benchmarks and categorize the different reasoning methods. Our survey focuses on the most recent advancements in this field and seeks to provide a comprehensive understanding of the current state-of-the-art.

KEYWORDS: reasoning; large language model (LLM); natural language processing (NLP)

Enabling machines with complex reasoning ability is one of the long pursuits in artificial intelligence [1–6]. Most initial reasoning systems formulate complex reasoning as multi-step operations over knowledge in symbolic or probabilistic representations [7, 8]. However, such formulation typically encounters difficulties when encoding massive amounts of knowledge into the structured representations, inducing reasoning rules, and dealing with ambiguity [9].

Recently, some researchers have proposed a different approach to reasoning, one that operates directly over natural language [10, 11]. This method treats reasoning as a question-answering task and utilizes language models to handle it. By using natural language, this approach can circumvent some of the difficulties of the traditional method, due to the flexibility and unstructured nature of language. Furthermore, the use of neural network-based language models allows for more effective reasoning over natural language, as they can better handle ambiguity and implicitly learn reasoning rules.

The rapid advancement of large language models (LLMs) has led to significant improvements in the performance of LLMs in various reasoning tasks. Current research indicates that the reasoning ability of LLMs could be elicited through prompting [12]. A wide variety of approaches have been proposed to prompt LLMs for better reasoning, and diverse datasets have been proposed to evaluate different aspects of the model's reasoning ability. Nonetheless, a comprehensive review of these endeavors is currently lacking.

In this paper, we conduct a survey of recent developments in reasoning by prompting LLMs. We begin by discussing reasoning over natural language and presenting relevant benchmarks in Section 1. Then, we provide an overview of LLMs in Section 2. In Section 3, we categorize the recent methods for reasoning by prompting LLMs. In Section 4, we discuss the open challenges and future directions. Section 5 concludes the paper.

1 Reasoning over natural language

Reasoning typically refers to the process of thinking about things in a logical, systematic way in order to form a conclusion, using available evidence, facts, and knowledge to analyze a situation and make an informed decision [2, 3]. Although widely used, the term "reasoning" is an abstract concept that can refer to a wide range of things. The definition of reasoning can be broad and vague. In this paper, we focus on the recent works that reason with LLMs over natural language.

Reasoning over natural language refers to reasoning directly over text sequences without first transforming them into formal structures or symbolic representations. This kind of reasoning is usually informal and is not based on strict logical rules or formal methods. It could rely on intuition, common sense, and past experiences and is often used in daily life. Multiple benchmarks have been established to evaluate different aspects of a model's reasoning abilities, such as arithmetic reasoning, commonsense reasoning, logical reasoning, and symbolic reasoning. These benchmarks generally model the reasoning task as a question-answering task.

In this section, we begin with a brief overview of reasoning tasks that require different abilities with examples, and then summarize different types of reasoning tasks from a higher level and discuss various forms of the task.

Arithmetic reasoning is the capability to use mathematical concepts and theorems to solve mathematical problems, which usually involves arithmetic operations (such as addition, subtraction, multiplication, and division). Arithmetic reasoning is

[1] Institute for Artificial Intelligence (THUAI), Tsinghua University, Beijing 100084, China. [2] Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China. [3] Department of Automation, Tsinghua University, Beijing 100084, China.

✉ Corresponding author. E-mail: zcs@mail.tsinghua.edu.cn

Received: February 27, 2023; Revised: August 30, 2023; Accepted: December 4, 2023

Table 1 Examples of different types of reasoning tasks

Reasoning type	Question	Answer
Arithmetic reasoning	Mrs. Lim milks her cows twice a day. Yesterday morning, she got 68 gallons of milk and in the evening, she got 82 gallons. This morning, she got 18 gallons fewer than she had yesterday morning. After selling some gallons of milk in the afternoon, Mrs. Lim has only 24 gallons left. How much was her revenue for the milk if each gallon costs $3.50?	616
Commonsense reasoning	Did Aristotle use a laptop?	No
Logical reasoning	Metals conduct electricity. Insulators do not conduct electricity. If something is made of iron then it is metal. Nails are made of iron. Is it true: nails conduct electricity?	True
Symbolic reasoning	Take the last letters of the words in 'Natural Language Processing' and then concatenate them.	Leg

an important aspect of human intelligence and has been proven to be quite challenging for language models. Table 1 shows a question that requires arithmetic reasoning. To solve this question, the model must understand natural language and master basic arithmetic operations. One solution to the example question is "Mrs. Lim got 68 − 18 = 50 gallons this morning. So she was able to get a total of 68 + 82 + 50 = 200 gallons. She was able to sell 200 − 24 = 176 gallons. Thus, her total revenue for the milk is $3.50/gallon × 176 gallons = $ 616."

Commonsense refers to knowledge that humans are expected to know without extra introduction, such as location relationship, temporal relationship as well as properties and interactions of things in the real world. Commonsense reasoning involves using commonsense and everyday knowledge to make informed decisions about the world. For example, commonsense reasoning would allow us to infer that the ground would become wet when it rains, or that someone should eat food if he/she is hungry. The example provided in Table 1 requires an understanding of several commonsense: "Aristotle was an ancient Greek philosopher (384 BC–322 BC)", "the first laptop was invented in the modern era in 1980, later than Aristotle's death", and "a person cannot use a laptop if he died before its invention".

Logical reasoning is the process of analyzing and deriving arguments using a set of facts and rules. A common logical reasoning task is to use deduction to determine whether a conclusion follows logically from the premises. While traditional logical reasoning tasks are usually based on formal logical language, some works [10] attempt to reason directly based on informal natural language. Table 1 shows an example.

Symbolic reasoning is a type of reasoning that manipulates information in symbolic or abstract representation. We focus on the symbolic reasoning tasks that use the natural language to describe the required symbolic operations or state changes. For the example in Table 1, it is a Last Letter Concatenation task that asks the model to concatenate the last letters of words in the input words. Information on the relevant datasets is summarized in Table 2.

From a higher level, the reasoning can be summarized mainly into three categories: deductive, inductive, and abductive reasoning [34]. Deductive reasoning draws a conclusion from known premises or assumptions, and the conclusion is logically related to the premises, i.e., if the premises are valid, then the

Table 2 Representative benchmarks for arithmetic, commonsense, logical, and symbolic reasoning

Type	Benchmark	Year	Size			
			All	Train	Dev	Test
Arithmetic reasoning	Addsub [13]	2014	395	—	—	—
	MultiArith [14]	2015	600	—	—	—
	SingleOP [15]	2015	562	—	—	—
	SingleEq [16]	2015	508	—	—	—
	MAWPS [17]	2016	3320	2656	—	664
	AQuA [18]	2017	101,449	100,949	250	250
	DROP [19]	2019	96,000	77,000	9500	9500
	IsarStep [20]	2020	830,000	820,000	5000	5000
	ASDiV [21]	2020	2305	—	—	—
	GSM8K [22]	2021	8500	7500	—	1000
	MATH [23]	2021	12,500	—	—	—
	SVAMP [24]	2021	1000	—	—	—
	Lila [25]	2022	134,000	93,800	13,400	26,800
Commonsense reasoning	ARC [26]	2018	7787	3370	869	3548
	OpenBookQA [27]	2018	5957	4957	500	500
	CommonsenseQA [28]	2019	12,102	9741	1221	1140
	StrategyQA [29]	2021	2780	2290	—	490
Logical reasoning	CLUTRR [30]	2019	6016	—	—	—
	ReCLor [31]	2020	6138	4638	500	1000
	LogiQA [32]	2020	8678	7376	651	651
	RuleTaker [10]	2020	500,000	350,000	50,000	100,000
	ProofWriter [11]	2020	1,000,000	700,000	100,000	200,000
Symbolic reasoning	Last letter concatenation [33]	2022	—	—	—	—
	Coin flip [33]	2022	—	—	—	—

conclusion is valid. For example, given that "all species of cats are mammals" and that "Ragdoll cats are a species of cat", we can deductively conclude that, "therefore, Ragdoll cats are mammals". Inductive reasoning is the drawing of a general conclusion based on a large number of specific observations and instances. It helps us to extract universal laws from a large number of instances. For example, if we observe that "eagles can fly" and "swallows can fly", we could try to inductively draw the conclusion that, "therefore, birds can fly". Abductive reasoning [35] is the process of seeking the most plausible causes or explanations based on a set of observations. For example, observing that the "sky is darkening and clouds are gathering", we could try to abductively infer that "it is going to rain". Unlike deductive reasoning, the conclusions given by inductive and abductive reasoning are not guaranteed to be logically correct. As more observations are added, the conclusions of inductive and abductive reasoning are defeasible. Among the three categories, however, inductive and abductive reasoning are much more under-researched than deductive reasoning in the context of reasoning over natural language. The majority of the datasets in Table 2 focus on deductive reasoning.

The specific forms of the task of reasoning over natural language are various. Existing datasets typically model reasoning tasks as question-answering tasks on account of the flexibility of this form. In addition, reasoning tasks can also be performed as natural language inference tasks, classification tasks, dialogue tasks, and so on. For example, the natural language inference task involves classifying statements into three categories: entailment, contradiction, and neutral. The goal is to determine whether a given premise supports a hypothesis (e.g., the premise "a soccer game with multiple males playing" entails the hypothesis "some men are playing a sport").

Early methods of reasoning tend to use symbolic systems, where the natural language is first mapped to symbolic or probabilistic representations. In this approach, the natural language is initially transformed into symbolic or probabilistic representations. These methods, which employ symbols and rules to represent and manipulate knowledge, have persisted as a subject of research up to this day [36–38]. However, a potential drawback of symbolic methods is the difficulty of dealing with complex natural language scenarios. Symbolic methods struggle to grapple with the complexity inherent in the richness of natural language, often requiring an explicit representation of every possible rule and relationship. This can result in an overwhelming number of rules and symbols, making it difficult to handle the intricacies of language effectively. Reasoning with LLMs, on the other hand, enables direct processing of unstructured natural language and allows implicit learning of rules and knowledge. This is due to their ability to learn from vast amounts of data, which grants them an inherent understanding of natural language. Meanwhile, as LLMs continue to advance rapidly and find applications across various fields, there arises a pressing need to inspect and comprehend their reasoning capabilities in-depth. Ensuring the efficiency and reliability of reasoning with LLMs becomes crucial, particularly when their outputs influence critical decisions. Consequently, the exploration of reasoning with LLMs has gained significant traction in recent times.

2 Large language models

The development of LLMs has been a major milestone in the field of natural language processing (NLP). These models, which commonly adopt transformer architecture, are pre-trained on a massive amount of text and exhibit an impressive tendency to scale rapidly in size. As the model scale grows larger, the large-scale language models exhibit different behaviors compared to small-scale language models, and show emergent abilities [39] in solving a range of complex tasks. For example, the 175 billion-parameter GPT-3 [40] can solve few-shot tasks through in-context learning, while the 1.5 billion-parameter GPT-2 [41] performs poorly. Thus, the term "large language models" is introduced by the research community to refer to these large-scale language models [33, 39, 42]. In the following, we present an overview of the architecture, pre-training technique, and scale of LLMs. Then, we briefly summarize the key challenges and limitations of LLMs.

2.1 Model architecture

One of the key breakthroughs in language models is the introduction of transformer architecture. The transformer architecture uses the self-attention mechanism, which allows the model to focus on the specific parts of the input sequence, making it well-suited for processing natural language. It was first introduced by Vaswani et al. [43] and then opened the way for the development of various language models. These models have achieved state-of-the-art performances on a wide range of NLP tasks, such as text classification, information extraction, and summarization. Two of the most representative models in terms of model architecture could be BERT [44] and GPT [45]. BERT, short for bidirectional encoder representations from transformers, uses the transformer encoder to obtain the representations of input sequences. Based on BERT, various language models are proposed gradually, such as RoBERTa [46], ERNIE [47], and XLNet [48]. GPT, short for generative pre-trained transformer, is a unidirectional language model based on the decoder-only transformer. It regards the previous text sequence as context and predicts the next token to complete the sequence. GPT can achieve satisfying results on a lot of tasks, including machine translation, text generation, and question answering. Inspired by the success of GPT, the larger and more powerful GPT-2 [41] and GPT-3 [40] are proposed to further improve and expand GPT.

2.2 Pre-training

Pre-training is a machine learning technique that trains models on a large amount of unannotated text data before applying them to a specific task. Initially, language models are trained from scratch for each specific task, resulting in limitations in terms of the amount of annotated training data and the overall performance. Pre-training enables language models to learn from huge amounts of unannotated data. By pre-training on a large corpus of unannotated text data, the models can learn to understand the structure and patterns of language, which is useful for downstream tasks and could substantially improve performance. The pre-training methods (e.g., masked language modeling [44]) depend on the task and the architecture of the model being used.

2.3 Model scale

The scale of a language model refers to its number of parameters. The model scale could have a significant impact on its performance, since larger models typically have more capacity to learn from more data and capture more complex patterns, resulting in more accurate predictions. Early language models were relatively small, with only a few million parameters. With progress in computational power and the accessibility of large amounts of text data, the scale of language models has become much larger. Figure 1 shows the trend of language model scales in recent years. To briefly show the changes, we include a small set of the largest or representative language models, rather than encompassing all variants of models. The models we include are

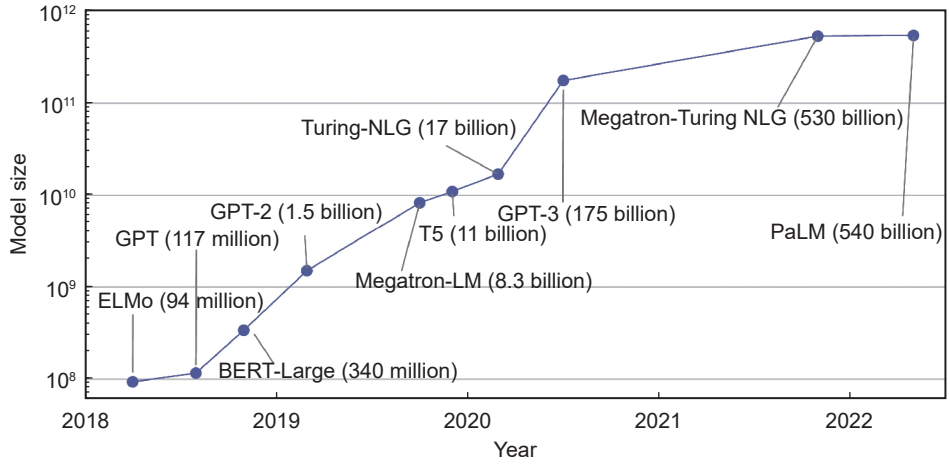

Fig. 1 Model scale curve with years. Note that the vertical axis is displayed on a logarithmic scale.

ELMo [49], GPT [45], BERT-large [44], GPT-2 [41], Megatron-LM [50], T5 [51], Turing-NLG [52], GPT-3 [40], Megatron-Turing NLG [53], and PaLM [54]. The scale of language models has been growing almost exponentially (from 94 million in 2018 to 540 billion in 2022).

As the model scale becomes larger, the model may exhibit some emergent capabilities [39] that are not present in smaller-scale models. For example, scaling up the language models could significantly improve the sample efficiency. Large language models can be adapted to specific tasks and achieve promising performance using only a small number of samples.

2.4 Key challenges

Although LLMs have achieved considerable success across various tasks, they have also introduced certain challenges that require resolution. For example, training effective LLMs in practice is tough due to extensive computation consumption [55–57]. How to develop cost-effective training methods that optimize LLMs steadily and effectively is one of the key challenges. Moreover, how to tackle security issues of LLMs [58] (e.g., privacy, overreliance, and disinformation) is also a key challenge. And it is also crucial to understand why certain emergent abilities would occur when the parameter of LLMs increases to a critical size (e.g., 10 billion) [33, 39]. Above are some of the general key challenges of LLMs, and we will discuss the detailed challenges in the context of reasoning with LLMs in Section 4.

3 Methodology

This section provides an overview of the advanced techniques for reasoning by prompting LLMs, which are categorized into several aspects. The first aspect, referred to as the learning paradigm (Section 3.1), delves into how LLMs acquire reasoning abilities. Regarding prompts, we explore the optimal content for eliciting reasoning from LLMs in Section 3.2 (Prompt content). Then, we discuss how we get these prompts in Section 3.3 (Prompt source). Additionally, Section 3.4 (Prompt strategy) covers the methodologies for utilizing these prompts effectively. Furthermore, the post-processing of LLM output is discussed in Section 3.5 (Post-processing), followed by an exploration of how external modules can be integrated to enhance LLM's inference capabilities in Section 3.6 (External modules). We present each dimension in detail and summarize the categories in Table 3.

3.1 Learning paradigm

Supervised learning by fine-tuning. A typical way to empower language models with reasoning ability is supervised learning, i.e., fine-tuning the language models with annotated data. For example, Clark et al. [10] proposed to fine-tune a pre-trained transformer to perform logical reasoning over synthetic sentences. Liang et al. [92] trained a BERT model as an encoder to solve math word problems. Dalvi et al. [93] fine-tuned a T5 model to generate the intermediate reasoning steps for the answers and Hong et al. [94, 95] further proposed a module-based improvement. Nye et al. [96] trained transformers to perform multi-step computations by asking them to generate intermediate computation steps before producing the final answers. More recently, Lewkowycz et al. [97] proposed Minerva, an LLM pretrained on general natural language data and further tuned on quantitative reasoning problems. Choudhary and Reddy [98] integrated symbolic methods and language models for complex logical reasoning over knowledge graphs.

However, the supervised learning paradigm requires large amounts of annotated data, which can be laborious and time-consuming. Moreover, when the model size becomes larger, the computational cost of fine-tuning increases and may be unaffordable. In addition, fine-tuning only on the domain-specific data may lead to overfitting, thus limiting models to generalize to other domains.

To alleviate these issues, some efforts focus on improvements with pre-training techniques and fine-tuning strategies. For the pre-training techniques, the source [99, 100], quality [101], quantity [102] of the training data, and the hyperparameters used for training, would affect the final capability of LLMs. For the fine-tuning strategies, instruction tuning and alignment tuning are two of the most critical and popular strategies. Instruction tuning is a method for fine-tuning a pre-trained LLM on a collection of instances in a natural language format. It is widely used in existing LLMs, such as GPT-4 [57, 58]. To conduct instruction fine-tuning, we gather or create instances formatted as instructions and subsequently employ these instances to supervise the fine-tuning process of the LLM. Following instruction tuning, the LLM could demonstrate remarkable generalization capabilities for novel tasks. The most commonly used technique for alignment tuning is reinforcement learning from human feedback (RLHF) [103, 104]. RLHF comprises three primary phases. The initial step involves training the LLM in a supervised manner, enabling it to execute the intended behavior from the start. Subsequently, a reward model is constructed through human feedback data. Then, a reinforcement learning strategy is adopted to further tune the LLM, leveraging the rewards provided by the previously developed reward model.

Table 3 Organization of works on reasoning with LLMs. For each work, we present the learning paradigm (FT = fine-tuning, P = prompting), prompt content (Std = standard, CoT = chain-of-thought, fixed), prompt source (HC = hand-crafted, MG = model-generated, RB = retrieval-based), prompt strategy (single-stage, multi-stage), post-processing (ensemble, validation), external module, and used LLMs

Work	Learning paradigm	Prompt content	Prompt source	Prompt strategy	Post processing	External module	LLM
GPT-3 [40]	P	Std	HC	Single	—	—	GPT-3, T5, RAG
Liu et al. [59]	P	Std	HC, MG	Multi	—	—	GPT-3, T5, UQA, Unicorn
Chain-of-Thought (CoT) [33]	P	CoT	HC	Single	—	—	GPT-3, LaMDA, PaLM, UL2, Codex
Wiegreffe et al. [60]	P	CoT	HC	Single	Validation	—	GPT-3, T5
Zero-shot-CoT [61]	P	Fixed	HC	Multi	—	—	GPT-3, InstructGPT, PaLM, GPT-2, GPT-Neo, GPT-J, T0, OPT
Selection-Inference [62]	P	CoT	HC	Multi	—	—	Gopher
Least-to-Most [63]	P	CoT	HC	Multi	—	—	GPT-3
MAIEUTIC [64]	P	Std, CoT	HC	Multi	Validation	—	GPT-3
Ye and Durrett [65]	P	CoT	HC	Single	Validation	—	RoBERTa, InstructGPT, GPT-3, OPT
STaR [66]	P, FT	CoT	HC	Multi	—	—	GPT-J
DIVERSE [67]	P	CoT	HC	Single	Ensemble	—	GPT-3
Creswell and Shanahan [68]	P, FT	CoT	HC	Multi	—	—	Chinchilla
Prystawski et al. [69]	P	CoT	HC	Single	—	—	GPT-3
Dynamic Least-to-Most [70]	P	Std	RB	Multi	—	—	Codex
PROMPTPG [71]	P, FT	Std	RB	Single	—	—	GPT-3
Counterfactual Prompting [72]	P	Std	HC	Single	—	—	GPT-3, PaLM, Codex
Complex CoT [73]	P	CoT	HC	Single	Ensemble	—	GPT-3, Codex
Auto-CoT [74]	P	CoT	MG	Single	—	—	GPT-3
Chen [75]	P	CoT	HC	Single	Ensemble	—	Codex, GPT-3
Decomp [76]	P	CoT	HC	Multi	—	—	GPT-3
Self-Ask [77]	P	CoT	HC	Multi	—	Search engine	GPT-3
Self-consistency [78]	P	CoT	HC	Single	Ensemble	—	UL2, GPT-3, LaMDA, PaLM
LMSI [79]	P, FT	CoT	HC	Single	Ensemble	—	PaLM
COCOGEN [80]	P	Std	HC	Single	—	Python interpreter	Codex
Algorithmic prompting [81]	P	CoT	HC	Single	—	—	Codex
PAL [82]	P	CoT	HC	Single	—	Python interpreter	Codex
PoT [83]	P	CoT	HC	Single	—	Python interpreter	Codex, GPT-3, PaLM, LaMDA
TSGP [84]	P	Std	HC	Multi	—	—	GPT-2
Successive prompting [85]	P, FT	CoT	HC	Multi	—	Mathematical module	GPT-J
LMLP [86]	P	CoT	HC	Multi	—	—	GPT-2, Sent-BERT
LAMBADA [87]	P	Std, CoT	HC	Multi	—	—	PaLM
Self-Verification [88]	P	CoT	HC	Multi	Validation	—	Codex, InstructGPT, GPT-3
Rethinking with Retrieval [89]	P	CoT	HC	Single	Validation	—	GPT-3
Faithful CoT [90]	P	CoT	HC	Multi	—	Python interpreter	Codex
Synthetic prompting [91]	P	CoT	MG	Single	Ensemble	Python interpreter	InstructGPT

In-context learning by prompting. Advanced works have demonstrated that LLMs can achieve remarkable performance in various tasks through in-context learning. We could prompt the LLMs with a few examples, which illustrate the input and output of the task, and the test question. The LLMs could learn from the examples and then predict the answer. The in-context learning paradigm is the mainstream paradigm for reasoning with LLMs. By prompting, we only need a few examples (usually 2 to 10) and do not need to fine-tune the large-scale models. Furthermore, we can use the same model for multiple tasks. In the following sections, we focus on this in-context paradigm.

3.2 Prompt content

Researchers have explored various methods of presenting a task to LLMs, with the aim of eliciting the desired output. Three main types of prompt content are the standard prompt, the Chain-of-

Thought prompt, and the fixed prompt. These prompts could vary in their composition and the amount of information they provide to the model.

Standard prompt. The standard prompt [39] is composed of a demonstration and a test question. The demonstration provides a few example input-output pairs that serve to illustrate a particular task. The test question then asks the model to generate an output based on the demonstration and complete the task. A standard prompt for arithmetic reasoning is shown in Fig. 2(a). The first Q and A (in green) is the input question and output answer of the demonstrative example. The second Q is the test question expected to be answered.

Chain-of-thought prompt. The standard prompt enables the model to predict the answer directly. However, when solving a complicated reasoning task, humans typically decompose the problem into intermediate steps and solve each before giving the final answer. To endow the LLMs to generate a similar chain of thought, Wei et al. [33] proposed the chain-of-thought (CoT) prompt. As shown in Fig. 2(b), the CoT prompt incorporates the intermediate reasoning steps (i.e., reasoning chain) within the demonstration examples, which are indicated with underlining. Including these steps in the prompt encourages the model to generate similar intermediate steps when solving the test question. This can also be interpreted as the rationales [105, 106] or explanations [64] that support the predicted answer and explain why it is correct.

Fixed prompt. Several studies investigate prompts that do not rely on demonstration examples. Kojima et al. [60] proposed Zero-shot-CoT prompt which does not require any input-output pairs. Instead, they use a fixed prompt for all tasks (as shown in Fig. 2(c)) and discover that such a straightforward prompt can still elicit the model to generate intermediate steps and answers. Brown et al. [39] also experimented with using a natural language description of the task instead of demonstration examples.

3.3 Prompt source

Hand-crafted. In many works, the prompt's content is hand-crafted. For instance, Wei et al. [33] manually created eight demonstration examples for each task. For all test questions within the same task, they utilize the same demonstrative samples for prompting.

Model-generated. While manually creating prompts is intuitive, it is hard to guarantee that the manual prompts are the best ones to elicit the model's capabilities completely. Moreover, the selection and ordering of the prompts could affect the model's performance. Some researchers turn to using language models to generate demonstrations. Zhang et al. [74] proposed Auto-CoT to automatically construct demonstrative examples. They first partition a small set of test questions into a few clusters, select a representative question from each cluster, and then generate its reasoning chain using Zero-Shot-CoT [61] to create the prompt. Wang et al. [105] leveraged the ability of LLMs to generate high-quality explanations and use the model-generated rationales to replace the human-written rationales in the prompts. Shao et al. [91] proposed synthetic prompting to utilize a few hand-crafted examples to prompt the model to generate more examples on its own and then carefully select effective demonstrations to elicit better reasoning.

Retrieval-based. Some work proposes to retrieve different demonstrative examples for different test questions instead of using the same examples for all test questions. Drozdov et al. [70] proposed a dynamic exemplar selection, which dynamically retrieves a set of relevant exemplars from a pre-selected pool for each input. Lu et al. [71] proposed to learn to select performing in-context examples using a policy gradient strategy.

3.4 Prompt strategy

Single-stage. The Single-stage strategy is a method that obtains answers directly through a single prompt stage. This approach typically involves calling the model only once for each question. The main advantage of this strategy is that it is simple and straightforward, as it requires a minimum number of interactions between the model and the user.

Multi-stage. Obtaining answers to complex questions through a single call can be challenging. As a result, it is becoming increasingly popular to decompose the solution into multiple stages and arrive at the final answer through the resolution of each stage, as illustrated in Fig. 3.

One such approach is question decomposition, where the complex question is broken down into simpler sub-questions [62, 76, 77, 85]. For example, the least-to-most prompting [62] involves two stages: first reducing a complex question into a list of

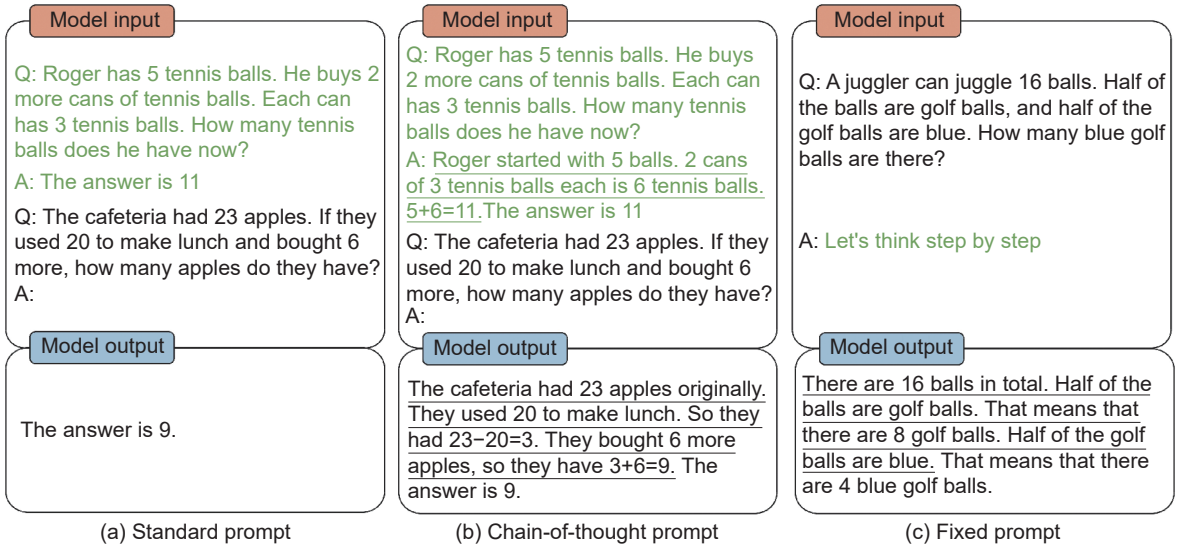

Fig. 2 Illustrations of different types of prompt contents. The input consists of questions to be answered (in black), and a few demonstrative examples or a small fixed piece of text (in green). For brevity, we only show one demonstrative example here. The intermediate reasoning chains in the inputs and outputs are indicated with underlining.

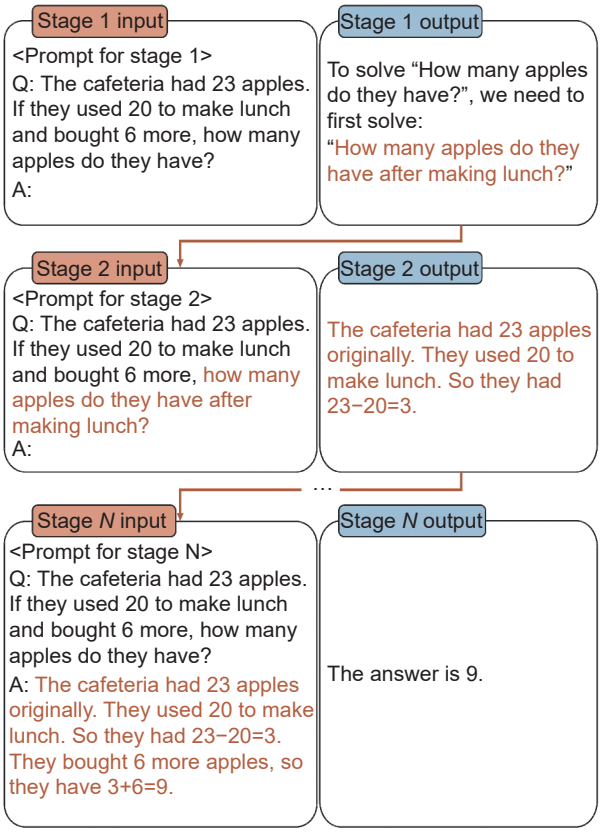

Fig. 3 An example of multi-stage prompting. The final answer is derived from the resolution of multiple stages. Note that different prompts could be used for different stages.

easier sub-questions, and then sequentially solving these sub-questions. In this process, solving a sub-question is facilitated by the answers to previously solved sub-questions, and the answer is derived through the final sub-question. Both stages are implemented by in-context learning but with different prompts.

In addition, Selection-inference [62, 68] decomposes logical reasoning into two stages: the selection stage chooses a subset of relevant information sufficient to make a single step of inference, and the inference stage uses only the limited information provided by the selection to infer a new intermediate conclusion on the way to the final answer. Self-verification [88] first generates candidate reasoning chains and answers and then uses the LLM to verify whether the chains meet the answers and rank the candidate answers based on a verification score. Sun et al. [84] proposed to first use knowledge generation prompts to generate the knowledge required for questions, and then utilize answer generation prompts to predict the answers based on the generated knowledge. Jung et al. [64] recursively prompted the model multiple times to generate diverse hypotheses. LAMBADA [87] decomposes reasoning into four sub-modules, each of which is implemented by a prompted LLM, and adaptively calls the sub-modules to answer questions based on backward chaining.

3.5 Post-processing

Ensemble. The ensemble strategy combines the results of multiple outputs to get the final answer. This is relevant to the ensemble methods in machine learning, which combine multiple systems to make predictions [107, 108]. The different outputs could be obtained via two approaches. (1) Sampling multiple outputs of the same prompt during decoding. The commonly used sampling strategies are top-p sampling [109] and top-k sampling [41, 110]. (2) Using multiple different prompts. Li et al. [67] proposed to use diverse prompts by randomly sampling multiple times from an example base.

The main adopted ensemble method is majority voting [105], where the answer with the most occurrences in outputs is selected as the final answer. Since different outputs may be of different importance, one can assign a score for voting to each output based on a certain criterion. The score could come from (1) the probability of the output given by the generation decoder [78, 79]; (2) the validity of the reasoning chain and the answer estimated by a trained verifier [67, 85]; (3) the complexity of the reasoning chain [73].

Verification and calibration. Since the output of the model is not free from noise and mistakes, some work proposes to verify or calibrate the output. He et al. [89] proposed to verify each generated reasoning step by retrieving relevant external knowledge. Jung et al. [64] used the model to generate multiple propositions and specify their strengths and logical relationships. They then employ the weighted MAX-SAT solver [111] to collectively infer the truth values of all the propositions, verify the propositions with the model's own belief, and select the proposition that best satisfies the set of observed relations. Ye and Durrett [65] train calibrators to calibrate the reliability score of predicted explanations.

3.6 External modules

While LLMs could generate reasoning chains, they could make logical and arithmetic mistakes in the intermediate solution. For example, LLMs often fail when dealing with complex arithmetic or large numbers [72, 112]. To address this issue, incorporating external modules to handle specific parts of the reasoning process is a potential solution. The external modules could be a Python interpreter [80, 82, 83, 90], a search engine [77], mathematical calculation modules [85], etc.

For instance, Chen et al. [83] proposed Program-of-Thought, which uses LLMs to generate text and Python statements. The computation is delegated to a Python interpreter, which is used to execute the generated program and get the final answer. In this way, the complex computation is decoupled from reasoning and language understanding. Press et al. [77] called a search engine to solve decomposed simple sub-questions and perform further reasoning based on the searched answers.

4 Challenges and future directions

4.1 Prompt engineering

The prompt is a crucial part of reasoning with LLMs. Several studies have found that models are sensitive to the format of the prompt, the specific choice of demonstrative examples, and the number of demonstrative examples [113, 114]. The content, source, and strategy of the prompt are likely to affect the performance of the model. How to obtain the optimal prompt for a specific task, a specific model, and a specific test question, to completely elicit the model's reasoning ability is still an important question to be explored.

The recently proposed ChatGPT (https://openai.com/blog/chatgpt), a chatbot system based on the LLM, also exhibits a degree of reasoning ability. Unlike existing prompting-based approaches, ChatGPT is able to understand tasks in the form of interactive dialogues, which eliminates the need for given demonstrative examples. Furthermore, ChatGPT is capable of providing reasoning processes flexibly and allows users to modify the errors present in previous reasoning through interaction.

Reasoning with ChatGPT in the form of dialogues would be an important research direction in the future.

4.2 Theoretical analysis

Although previous works have demonstrated the outstanding performance of LLMs on various reasoning tasks through experimentation, there remains a dearth of sufficient evidence regarding whether the high performance of LLMs is primarily attributed to their true reasoning ability or other factors such as heuristics [24] or shortcuts [115]. In order to answer this question, a theoretical analysis of the reasoning may be the way to go. Xie et al. [116] analyzed the role of prompting theoretically by viewing the in-context learning as a Bayesian inference process. More recently, Dziri et al. [117] explored the boundaries of LLMs on some compositional tasks that require breaking problems down into sub-steps and synthesizing these steps into the answer. They find that LLMs solve compositional tasks by simplifying multi-step compositional reasoning into linearized subgraph matching, but do not necessarily involve the development of systematic problem-solving abilities. Furthermore, they provide theoretical arguments that illustrate how the performance of LLMs will degrade quickly as the task complexity increases. Turpin et al. [118] demonstrated that chain-of-thought prompting could be significantly influenced by adding biasing features to model inputs. For instance, rearranging the multiple-choice options within a few-shot prompt to consistently make the answer "(A)" can heavily influence the faithfulness of the explanations generated by LLMs. Interestingly, models tend to omit this influence in their explanations. When intentionally guiding models toward incorrect answers, they often produce Chain-of-Thought explanations that align with those incorrect answers. Another line of works [119–121] attempts to analyze the limitations of the transformer from a structural perspective. Overall, the advantage of the prompt-based method is that it is experimentally effective in improving the accuracy of the LLMs' answers and provides readable explanations. However, the potential drawbacks lie in that the explanations are not always faithful to the answers or to the real reasoning process of LLMs, and can be intentionally misleading. Additionally, the investigation into the development of the reasoning abilities of LLMs and the methods to forecast these abilities in future models remains underexplored. We believe that a more comprehensive theoretical analysis can inform method design and enable the model with real reasoning ability.

4.3 Lightweight training and inference

Since the LLM contains a large number of parameters, performing both training and inference processes involves large computational costs. Therefore, how to perform lightweight deployment of LLM in real applications is also an important challenge. Research in parameter-efficient fine-tuning aims to reduce the number of trainable parameters while maintaining as good a performance as possible. The two current representative approaches are adapter tuning [122] and low-rank adaptation (LoRA) [123]. Adapter tuning introduces small neural network modules as adapters in the transformer model. Adapters are usually inserted serially after the attention and feedforward layers. Throughout the fine-tuning procedure, the adapter modules will be optimized, while the parameters of the original language model will remain unchanged. This approach enables a significant reduction in the count of trainable parameters throughout the fine-tuning process. Unlike adapter tuning, LoRA approximates the parameter update matrices by a low-rank decomposition matrix, which leads to significant reductions in memory and storage usage. LoRA and its improved versions [124] have been used in several LLMs. During the inference phase, lightweight deployment is also crucial. In real-world applications, deploying a large language model as it is could be impractical due to memory and latency constraints. One of the effective strategies is model quantization [125–128], which involves converting the model's parameters to lower precision representations (e.g., from 32-bit floating point numbers to 16-bit or even 8-bit integers). This reduces the memory footprint and speeds up computations, making the model more suitable for deployment on resource-constrained devices. Additionally, specialized inference techniques, such as model distillation [129, 130], model pruning [131–134], and dynamic computation allocation, contribute to making the deployment of LLMs more efficient. Overall, lightweight training and inference of LLM remains a key challenge and future direction.

4.4 Learning paradigm

Fine-tuning and prompting are the two main paradigms for training models to reason. Fine-tuning involves adjusting the parameters of a pre-trained model to fit a specific task and requires a certain amount of labeled data and computational cost, particularly for large models. On the other hand, prompting needs only a few examples and is less computationally intensive compared to fine-tuning. But it is usually effective only when the model reaches a certain size. Based on recent studies, it has been observed that the use of prompting in large models can result in performance that is comparable to or even surpasses that achieved through fine-tuning with smaller models. How to effectively combine fine-tuning and prompting could be an interesting direction to explore. For LLMs, some researchers have explored ways to enhance the performance of LLM by fine-tuning using the reasoning chains generated through prompting itself [66, 79]. For models with smaller scales, an increasing number of works propose to distill the reasoning ability of LLMs to smaller models through fune-tinning [135–140].

4.5 Evaluation of reasoning

As discussed by Valmeekam et al. [141], current benchmarks could be insufficient to make substantive claims about LLMs' ability to reason, since many of the reasoning tasks used in these benchmarks are often straightforward and may only require shallow reasoning, which does not adequately reflect the complexity of real-world reasoning tasks. In order to address this challenge, it is necessary to develop benchmarks that incorporate more diverse and complex reasoning tasks, which require advanced reasoning skills and a deeper understanding of language.

In addition, the LLMs are primarily generative models that are capable of generating output sequences without explicit constraints. Evaluating and regulating their outputs could be a critical challenge that needs to be addressed. The current approach for evaluating the reasoning ability of LLMs is typically based on the answer classification accuracy or exact answer text match. It could be insufficient given the possibility of multiple valid answers for the real-world questions. Furthermore, the present methods occasionally produce anomalous errors, highlighting the necessity of the solutions for regulating and controlling the model's outputs.

4.6 Multimodal and multilingual reasoning

The current reasoning methods primarily focus on a single modality (i.e., natural language) and a single language (e.g., English). However, in real-world applications, reasoning often requires handling multiple modalities and languages. This

highlights the importance of extending reasoning methods to be capable of multimodal and multilingual reasoning. Lu et al. [142] and Zhang et al. [143] explored the extension of CoT prompts to reasoning tasks that involve both language and images. Chen [75] investigated the reasoning abilities of large language models over tabular modalities. Shi et al. [144] evaluated the reasoning abilities of LLMs in multilingual settings and found that models have strikingly strong multilingual reasoning abilities, even in underrepresented languages such as Bengali and Swahili.

4.7 Social impact of LLMs

The use of LLMs and reasoning methods in real-life scenarios could lead to various potential social risks. Although we focus on how to reason with LLMs, there may be risks in the LLMs themselves, such as privacy disclosure and toxic answers, that could be unintentionally elicited by reasoning methods. For instance, Shaikh et al. [145] demonstrated that zero-shot CoT consistently produces undesirable biases and toxicity. LLMs are trained on a large corpus that may contain sensitive information, such as personal details like names, addresses, email addresses, and phone numbers. Without proper safeguards in place, there is a risk that these models may inadvertently disclose such information. Additionally, malicious people could potentially use the knowledge contained within LLMs for harmful purposes, such as promoting hate speech and terrorism. It is crucial to carefully consider the social implications of these models before their deployment and usage.

5 Conclusions

In this paper, we provide a review of the current developments in reasoning by prompting LLMs. We discuss the benchmarks for evaluating reasoning ability, introduce the large language models, and categorize the methods to reason with LLMs. Although language models currently show excellent performance and have a tendency to continue to evolve rapidly, there are still many challenges that need to be addressed in future work. We hope this review would help researchers and foster further advancements in the field.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] McCarthy, J. Programs with common sense. 1960. Available at http://jmc.stanford.edu/articles/mcc59/mcc59.pdf

[2] Wason, P. C. Reasoning about a rule. *Quarterly Journal of Experimental Psychology* Vol. 20, No. 3, 273–281, 1968.

[3] Wason, P. C.; Johnson-Laird, P. N. *Psychology of Reasoning: Structure and Content*. Cambridge, USA: Harvard University Press, 1972.

[4] Khardon, R.; Roth, D. Learning to reason. *Journal of the ACM* Vol. 44, No. 5, 697–725, 1997.

[5] Fagin, R.; Halpern, J. Y.; Moses, Y.; Vardi, M. *Reasoning about Knowledge*. Cambridge, USA: MIT Press, 2004.

[6] Bottou, L. From machine learning to machine reasoning. *Machine Learning* Vol. 94, No. 2, 133–149, 2014.

[7] Forbus, K. D.; De Kleer, J. *Building Problem Solvers*. Cambridge, MA, USA: MIT Press, 1993.

[8] Newell, A.; Simon, H. The logic theory machine: A complex information processing system. *IEEE Transactions on Information Theory* Vol. 2, No. 3, 61–79, 1956.

[9] Bell, M. Z. Why expert systems fail. *Journal of the Operational Research Society* Vol. 36, No. 7, 613–619, 1985.

[10] Clark, P.; Tafjord, O.; Richardson, K. Transformers as soft reasoners over language. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence, 3882–3890, 2020.

[11] Tafjord, O.; Dalvi, B.; Clark, P. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In: Proceedings of the Findings of the Association for Computational Linguistics, 3621–3634, 2021.

[12] Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; Neubig, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys* Vol. 55, No. 9, Article No. 195, 2023.

[13] Hosseini, M. J.; Hajishirzi, H.; Etzioni, O.; Kushman, N. Learning to solve arithmetic word problems with verb categorization. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 523–533, 2014.

[14] Roy, S.; Roth, D. Solving general arithmetic word problems. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 1743–1752, 2015.

[15] Roy, S.; Vieira, T.; Roth, D. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics* Vol. 3, 1–13, 2015.

[16] Koncel-Kedziorski, R.; Hajishirzi, H.; Sabharwal, A.; Etzioni, O.; Ang, S. D. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics* Vol. 3, 585–597, 2015.

[17] Koncel-Kedziorski, R.; Roy, S.; Amini, A.; Kushman, N.; Hajishirzi, H. MAWPS: A math word problem repository. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 1152–1157, 2016.

[18] Ling, W.; Yogatama, D.; Dyer, C.; Blunsom, P. 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, 158–167, 2017.

[19] Dua, D.; Wang, Y.; Dasigi, P.; Stanovsky, G.; Singh, S.; Gardner, M. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2368–2378, 2019.

[20] Li, W.; Yu, L.; Wu, Y.; Paulson, L. C. Isarstep: A benchmark for high-level mathematical reasoning. In: Proceedings of the 9th International Conference on Learning Representations, 2021.

[21] Miao, S. Y.; Liang, C. C.; Su, K. Y. A diverse corpus for evaluating and developing English math word problem solvers. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 975–984, 2020.

[22] Cobbe, K.; Kosaraju, V.; Bavarian, M.; Hilton, J.; Nakano, R.; Hesse, C.; Schulman, J. Training verifiers to solve math word problems. *arXiv preprint* arXiv:2110.14168, 2021.

[23] Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; Steinhardt, J. Measuring mathematical problem solving with the MATH dataset. In: Proceedings of the 35th Conference on Neural Information Processing Systems Track on Datasets and Benchmarks, 2021.

[24] Patel, A.; Bhattamishra, S.; Goyal, N. Are NLP Models really able to Solve Simple Math Word Problems?. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2080–2094, 2021.

[25] Mishra, S.; Finlayson, M.; Lu, P.; Tang, L.; Welleck, S.; Baral, C.; Rajpurohit, T.; Tafjord, O.; Sabharwal, A.; Clark, P.; et al. LILA: A unified benchmark for mathematical reasoning. In: Proceedings of the Conference on Empirical Methods in Natural Language

Processing, 5807–5832, 2022.
[26] Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; Tafjord, O. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint* arXiv:1803.05457, 2018.
[27] Mihaylov, T.; Clark, P.; Khot, T.; Sabharwal, A. Can a suit of armor conduct electricity? A new dataset for open book question answering. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2381–2391, 2018.
[28] Talmor, A.; Herzig, J.; Lourie, N.; Berant, J. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 4149–4158, 2019.
[29] Geva, M.; Khashabi, D.; Segal, E.; Khot, T.; Roth, D.; Berant, J. Did Aristotle use a Laptop? A question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics* Vol. 9, 346–361, 2021.
[30] Sinha, K.; Sodhani, S.; Dong, J.; Pineau, J.; Hamilton, W. L. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, 4505–4514, 2019.
[31] Yu, W.; Jiang, Z.; Dong, Y.; Feng, J. Reclor: A reading comprehension dataset requiring logical reasoning. In: Proceedings of the 8th International Conference on Learning Representations, 2020.
[32] Liu, J.; Cui, L.; Liu, H.; Huang, D.; Wang, Y.; Zhang, Y. LogiQA: A challenge dataset for machine reading comprehension with logical reasoning. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence, 3622–3628, 2020.
[33] Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint* arXiv:2201.11903, 2022.
[34] Peirce, C. S.; Ketner, K. L. *Reasoning and the Logic of Things: The Cambridge Conferences Lectures of 1898*. Cambridge, USA: Harvard University Press, 1992.
[35] Harman, G. H. The inference to the best explanation. *The Philosophical Review* Vol. 74, No. 1, 88, 1965.
[36] Badreddine, S.; Garcez, A. d'Avila; Serafini, L.; Spranger, M. Logic tensor networks. *Artificial Intelligence* Vol. 303, 103649, 2022.
[37] Dong, H.; Mao, J.; Lin, T.; Wang, C.; Li, L.; Zhou, D. Neural logic machines. In: Proceedings of the 7th International Conference on Learning Representations, 2019.
[38] Manhaeve, R.; Dumančić, S.; Kimmig, A.; Demeester, T.; De Raedt, L. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence* Vol. 298, 103504, 2021.
[39] Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. Emergent abilities of large language models. *arXiv preprint* arXiv:2206.07682, 2022.
[40] Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. In: Proceedings of the 34th Conference on Neural Information Processing Systems, 2020.
[41] Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. 2019. Available at https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
[42] Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. A survey of large language models. *arXiv preprint* arXiv:2303.18223, 2023.
[43] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In: Proceedings of the 31st Conference on Neural Information Processing Systems, 5998–6008, 2017.
[44] Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 4171–4186, 2019.
[45] Radford, A.; Narasimhan, K. Improving language understanding by generative pre-training. 2018. Available at https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf
[46] Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint* arXiv:1907.11692, 2019.
[47] Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; Liu, Q. ERNIE: enhanced language representation with informative entities. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 1441–1451, 2019.
[48] Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R.; Le, Q. V. XLNet: Generalized autoregressive pretraining for language understanding. In: Proceedings of the 33rd Conference on Neural Information Processing Systems, 5754–5764, 2019.
[49] Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2227–2237, 2018.
[50] Shoeybi, M.; Patwary, M.; Puri, R.; LeGresley, P.; Casper, J.; Catanzaro, B. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint* arXiv:1909.08053, 2019.
[51] Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* Vol. 21, No. 140, 1–67, 2020.
[52] Rosset, C. Turing-NLG: A 17-billion-parameter language model by Microsoft. 2020. Available at https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/
[53] Smith, S.; Patwary, M.; Norick, B.; LeGresley, P.; Rajbhandari, S.; Casper, J.; Liu, Z.; Prabhumoye, S.; Zerveas, G.; Korthikanti, V.; et al. PaLM: Scaling language modeling with pathways. *arXiv preprint* arXiv:2204.02311, 2022.
[54] Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. PaLM: Scaling language modeling with pathways. *arXiv preprint* arXiv:2204.02311, 2022.
[55] Scao, T. L.; Fan, A.; Akiki, C.; Pavlick, E.; Ilić, S.; Hesslow, D.; Castagné, R.; Luccioni, A. S.; Yvon, F.; Gallé, M. Bloom: A 176B-parameter open-access multilingual language model. *arXiv preprint* arXiv:2211.05100, 2022.
[56] Zeng, A.; Liu, X.; Du, Z.; Wang, Z.; Lai, H.; Ding, M.; Yang, Z.; Xu, Y.; Zheng, W.; Xia, X. GLB-130B: An open bilingual pre-trained model. *arXiv preprint* arXiv:2210.02414, 2022.
[57] Peng, B.; Li, C.; He, P.; Galley, M.; Gao, J. Instruction tuning with GPT-4. *arXiv preprint* arXiv:2304.03277, 2023.
[58] OpenAI; Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; et al. GPT-4 technical report. *arXiv preprint* arXiv:2303.08774, 2023.
[59] Liu, J.; Liu, A.; Lu, X.; Welleck, S.; West, P.; Le Bras, R.; Choi, Y.; Hajishirzi, H. Generated knowledge prompting for commonsense

reasoning. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 3154–3169, 2022.

[60] Wiegreffe, S.; Hessel, J.; Swayamdipta, S.; Riedl, M.; Choi, Y. Reframing human-AI collaboration for generating free-text explanations. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 632–658, 2022.

[61] Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large language models are zero-shot reasoners. *arXiv preprint* arXiv:2205.11916, 2022.

[62] Creswell, A.; Shanahan, M.; Higgins, I. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint* arXiv:2205.09712, 2022.

[63] Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q.; et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint* arXiv:2205.10625, 2022.

[64] Jung, J.; Qin, L.; Welleck, S.; Brahman, F.; Bhagavatula, C.; Le Bras, R.; Choi, Y. Maieutic prompting: Logically consistent reasoning with recursive explanations. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 1266–1279, 2022.

[65] Ye, X.; Durrett, G. The unreliability of explanations in few-shot prompting for textual reasoning. In: Proceedings of the 36th Conference on Neural Information Processing Systems, 2022.

[66] Zelikman, E.; Wu, Y.; Goodman, N. D. Star: Bootstrapping reasoning with reasoning. *arXiv preprint* arXiv:2203.14465, 2022.

[67] Li, Y.; Lin, Z.; Zhang, S.; Fu, Q.; Chen, B.; Lou, J.; Chen, W. On the advance of making language models better reasoners. *arXiv preprint* arXiv:2206.02336, 2022.

[68] Creswell, A.; Shanahan, M. Faithful reasoning using large language models. *arXiv preprint* arXiv:2208.14271, 2022.

[69] Prystawski, B.; Thibodeau, P. H.; Goodman, N. D. Psychologically-informed chain-of-thought prompts for metaphor understanding in large language models. *arXiv preprint* arXiv:2209.08141, 2022.

[70] Drozdov, A.; Schärli, N.; Akyürek, E.; Scales, N.; Song, X.; Chen, X.; Bousquet, O.; Zhou, D. Compositional semantic parsing with large language models. *arXiv preprint* arXiv:2209.15003, 2022.

[71] Lu, P.; Qiu, L.; Chang, K.; Wu, Y. N.; Zhu, S. C.; Rajpurohit, T.; Clark, P.; Kalyan, A. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint* arXiv:2209.14610, 2022.

[72] Madaan, A.; Yazdanbakhsh, A. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint* arXiv:2209.07686, 2022.

[73] Fu, Y.; Peng, H.; Sabharwal, A.; Clark, P.; Khot, T. Complexity-based prompting for multi-step reasoning. *arXiv preprint* arXiv:2210.00720, 2022.

[74] Zhang, Z.; Zhang, A.; Li, M.; Smola, A. Automatic chain of thought prompting in large language models. *arXiv preprint* arXiv:2210.03493, 2022.

[75] Chen, W. Large language models are few(1)-shot table reasoners. *arXiv preprint* arXiv:2210.06710, 2022.

[76] Khot, T.; Trivedi, H.; Finlayson, M.; Fu, Y.; Richardson, K.; Clark, P.; Sabharwal, A. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint* arXiv:2210.02406, 2022.

[77] Press, O.; Zhang, M.; Min, S.; Schmidt, L.; Smith, N. A.; Lewis, M. Measuring and narrowing the compositionality gap in language models. *arXiv preprint* arXiv:2210.03350, 2022.

[78] Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; Zhou, D. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint* arXiv:2203.11171, 2022.

[79] Huang, J.; Gu, S. S.; Hou, L.; Wu, Y.; Wang, X.; Yu, H.; Han, J. Large language models can self-improve. *arXiv preprint* arXiv:2210.11610, 2022.

[80] Madaan, A.; Zhou, S.; Alon, U.; Yang, Y.; Neubig, G. Language models of code are few-shot commonsense learners. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 1384–1403, 2022.

[81] Zhou, H.; Nova, A.; Larochelle, H.; Courville, A. C.; Neyshabur, B.; Sedghi, H. Teaching algorithmic reasoning via In-context learning. *arXiv preprint* arXiv:2211.09066, 2022.

[82] Gao, L.; Madaan, A.; Zhou, S.; Alon, U.; Liu, P.; Yang, Y.; Callan, J.; Neubig, G. PAL: Program-aided language models. *arXiv preprint* arXiv:2211.10435, 2022.

[83] Chen, W.; Ma, X.; Wang, X.; Cohen, W. W. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint* arXiv:2211.12588, 2022.

[84] Sun, Y.; Zhang, Y.; Qi, L.; Shi, Q. TSGP: Two-stage generative prompting for unsupervised commonsense question answering. *arXiv preprint* arXiv:2211.13515, 2022.

[85] Dua, D.; Gupta, S.; Singh, S.; Gardner, M. Successive prompting for decomposing complex questions. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 1251–1265, 2022.

[86] Zhang, H.; Zhang, Y.; Li, L. E.; Xing, E. The impact of symbolic representations on in-context learning for few-shot reasoning. *arXiv preprint* arXiv:2212.08686, 2022.

[87] Kazemi, S. M.; Kim, N.; Bhatia, D.; Xu, X.; Ramachandran, D. LAMBADA: Backward chaining for automated reasoning in natural language. *arXiv preprint* arXiv:2212.13894, 2022.

[88] Weng, Y.; Zhu, M.; He, S.; Liu, K.; Zhao, J. Large language models are reasoners with self-verification. *arXiv preprint* arXiv:2212.09561, 2022.

[89] He, H.; Zhang, H.; Roth, D. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint* arXiv:2301.00303, 2023.

[90] Lyu, Q.; Havaldar, S.; Stein, A.; Zhang, L.; Rao, D.; Wong, E.; Apidianaki, M.; Callison-Burch, C. Faithful chain-of-thought reasoning. *arXiv preprint* arXiv:2301.13379, 2023.

[91] Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; Chen, W. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. *arXiv preprint* arXiv:2302.00618, 2023.

[92] Liang, Z.; Zhang, J.; Shao, J.; Zhang, X. MWP-BERT: A strong baseline for math word problems. *arXiv preprint* arXiv:2107.13435, 2021.

[93] Dalvi, B.; Jansen, P.; Tafjord, O.; Xie, Z.; Smith, H.; Pipatanangkura, L.; Clark, P. Explaining answers with entailment trees. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 7358–7370, 2021.

[94] Hong, R.; Zhang, H.; Yu, X.; Zhang, C. METGEN: A module-based entailment tree generation framework for answer explanation. In: Proceedings of the Findings of the Association for Computational Linguistics, 1887–1905, 2022.

[95] Hong, R.; Zhang, H.; Zhao, H.; Yu, D.; Zhang, C. Faithful question answering with monte-carlo planning. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, 3944–3965, 2023.

[96] Nye, M. I.; Andreassen, A. J.; Gur-Ari, G.; Michalewski, H.; Austin, J.; Bieber, D.; Dohan, D.; Lewkowycz, A.; Bosma, M.; Luan, D.; et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint* arXiv:2112.00114, 2021.

[97] Lewkowycz, A.; Andreassen, A.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; et al. Solving quantitative reasoning problems with language models. In: Proceedings of the 36th Conference on Neural Information Processing Systems, 2022.

[98] Choudhary, N.; Reddy, C. K. Complex logical reasoning over

knowledge graphs using large language models. *arXiv preprint* arXiv:2305.01157, 2023.

[99] Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. d. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. Evaluating large language models trained on code. *arXiv preprint* arXiv:2107.03374, 2021.

[100] Rae, J. W.; Borgeaud, S.; Cai, T.; Millican, K.; Hoffmann, J.; Song, H. F.; Aslanides, J.; Henderson, S.; Ring, R.; Young, S.; et al. Scaling language models: Methods, analysis & insights from training Gopher. *arXiv preprint* arXiv:2112.11446, 2021.

[101] Scao, T. L.; Fan, A.; Akiki, C.; Pavlick, E.; Ilic, S.; Hesslow, D.; Castagné, R.; Luccioni, A. S.; Yvon, F.; Gallé, M.; et al. BLOOM: A 176B-parameter open-access multilingual language model. *arXiv preprint* arXiv:2211.05100, 2022.

[102] Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; Casas, D. d. L.; Hendricks, L. A.; Welbl, J.; Clark, A.; et al. Training compute-optimal large language models. *arXiv preprint* arXiv:2203.15556, 2022.

[103] Christiano, P. F.; Leike, J.; Brown, T. B.; Martic, M.; Legg, S.; Amodei, D. Deep reinforcement learning from human preferences. In: Proceedings of the 31st Conference on Neural Information Processing Systems, 4299–4307, 2017.

[104] Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. Training language models to follow instructions with human feedback. In: Proceedings of the 36th Conference on Neural Information Processing Systems, 2022.

[105] Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Zhou, D. Rationale-augmented ensembles in language models. *arXiv preprint* arXiv:2207.00747, 2022.

[106] Zelikman, E.; Wu, Y.; Mu, J.; Goodman, N. D. STaR: Bootstrapping reasoning with reasoning. *arXiv preprint* arXiv:2203.14465, 2022.

[107] Zhou, Z. H.; Wu, J.; Tang, W. Ensembling neural networks: Many could be better than all. *Artificial Intelligence* Vol. 137, Nos. 1–2, 239–263, 2002.

[108] Zhou, Z. H. *Ensemble Methods: Foundations and Algorithms*. Boca Raton, USA: CRC Press, 2012.

[109] Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; Choi, Y. The curious case of neural text degeneration. In: Proceedings of the 8th International Conference on Learning Representations, 2020.

[110] Fan, A.; Lewis, M.; Dauphin, Y. Hierarchical neural story generation. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, 889–898, 2018.

[111] Battiti, R. Maximum satisfiability problem. In: *Encyclopedia of Optimization*. Floudas, C. A.; Pardalos, P. M. Eds. New York, NY, USA: Springer, 2035–2041, 2001.

[112] Qian, J.; Wang, H.; Li, Z.; Li, S.; Yan, X. Limitations of language models in arithmetic and symbolic induction. *arXiv preprint* arXiv: 2208.05051, 2022.

[113] Liang, P.; Bommasani, R.; Lee, T.; Tsipras, D.; Soylu, D.; Yasunaga, M.; Zhang, Y.; Narayanan, D.; Wu, Y.; Kumar, A. Holistic evaluation of language models. *arXiv preprint* arXiv:2211.09110, 2022.

[114] Lu, Y.; Bartolo, M.; Moore, A.; Riedel, S.; Stenetorp, P. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 8086–8098, 2022.

[115] Du, M.; He, F.; Zou, N.; Tao, D.; Hu, X. Shortcut learning of large language models in natural language understanding. *Communications of the ACM* Vol. 67, No. 1, 110–120, 2024.

[116] Xie, S. M.; Raghunathan, A.; Liang, P.; Ma, T. An explanation of in-context learning as implicit Bayesian inference. In: Proceedings of the 10th International Conference on Learning Representations, 2022.

[117] Dziri, N.; Lu, X.; Sclar, M.; Li, X. L.; Jiang, L.; Lin, B. Y.; West, P.; Bhagavatula, C.; Bras, R. L.; Hwang, J. D. Faith and fate: Limits of transformers on compositionality. *arXiv preprint* arXiv:2305.18654, 2023.

[118] Turpin, M.; Michael, J.; Perez, E.; Bowman, S. R. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *arXiv preprint* arXiv:2305.04388, 2023.

[119] Yao, S.; Peng, B.; Papadimitriou, C.; Narasimhan, K. Self-attention networks can process bounded hierarchical languages. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, 3770–3785, 2021.

[120] Hao, Y.; Angluin, D.; Frank, R. Formal language recognition by hard attention transformers: Perspectives from circuit complexity. *Transactions of the Association for Computational Linguistics* Vol. 10, 800–810, 2022.

[121] Hahn, M. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics* Vol. 8, 156–171, 2020.

[122] Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; Laroussilhe, Q. d.; Gesmundo, A.; Attariyan, M.; Gelly, S. Parameter-efficient transfer learning for NLP. In: Proceedings of the 36th International Conference on Machine Learning, 2790–2799, 2019.

[123] Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W. LoRA: Lowrank adaptation of large language models. In: Proceedings of the 10th International Conference on Learning Representations, 2022.

[124] Zhang, Q.; Chen, M.; Bukharin, A.; He, P.; Cheng, Y.; Chen, W.; Zhao, T. Adaptive budget allocation for parameter-efficient fine-tuning. In: Proceedings of the 11th International Conference on Learning Representations, 2023.

[125] Park, G.; Park, B.; Kwon, S. J.; Kim, B.; Lee, Y.; Lee, D. LUT-GEMM: Quantized matrix multiplication based on LUTs for efficient inference in large-scale generative language models. *arXiv preprint* arXiv:2206.09557, 2022.

[126] Tao, C.; Hou, L.; Zhang, W.; Shang, L.; Jiang, X.; Liu, Q.; Luo, P.; Wong, N. Compression of generative pre-trained language models via quantization. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 4821–4836, 2022.

[127] Yao, Z.; Aminabadi, R. Y.; Zhang, M.; Wu, X.; Li, C.; He, Y. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In: Proceedings of the 36th Conference on Neural Information Processing Systems, 2022.

[128] Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; Han, S. SmoothQuant: Accurate and efficient post-training quantization for large language models. In: Proceedings of the 40th International Conference on Machine Learning, 38087–38099, 2023.

[129] Hsieh, C. Y.; Li, C. L.; Yeh, C. K.; Nakhost, H.; Fujii, Y.; Ratner, A.; Krishna, R.; Lee, C. Y.; Pfister, T. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In: Proceedings of the Findings of the Association for Computational Linguistics, 8003–8017, 2023.

[130] Dasgupta, S.; Cohn, T.; Baldwin, T. Cost-effective distillation of large language models. In: Proceedings of the Findings of the Association for Computational Linguistics, 7346–7354, 2023.

[131] Kurtic, E.; Campos, D.; Nguyen, T.; Frantar, E.; Kurtz, M.; Fineran, B.; Goin, M.; Alistarh, D. The optimal BERT surgeon: Scalable and accurate second-order pruning for large language models. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 4163–4181, 2022.

[132] Kurtic, E.; Frantar, E.; Alistarh, D. ZipLM: Hardware-aware structured pruning of language models. *arXiv preprint* arXiv:2302.04089, 2023.

[133] Frantar, E.; Alistarh, D. Massive language models can be accurately pruned in one-shot. *arXiv preprint* arXiv:2301.00774, 2023.
[134] Ma, X.; Fang, G.; Wang, X. LLM-Pruner: On the structural pruning of large language models. *arXiv preprint* arXiv:2305.11627, 2023.
[135] Li, S.; Chen, J.; Shen, Y.; Chen, Z.; Zhang, X.; Li, Z.; Wang, H.; Qian, J.; Peng, B.; Mao, Y. Explanations from large language models make small reasoners better. *arXiv preprint* arXiv:2210.06726, 2022.
[136] Wang, P.; Chan, A.; Ilievski, F.; Chen, M.; Ren, X. PINTO: Faithful language reasoning using prompt-generated rationales. *arXiv preprint* arXiv:2211.01562, 2022.
[137] Shridhar, K.; Stolfo, A.; Sachan, M. Distilling multi-step reasoning capabilities of large language models into smaller models via semantic decompositions. *arXiv preprint* arXiv:2212.00193, 2022.
[138] Magister, L. C.; Mallinson, J.; Adámek, J.; Malmi, E.; Severyn, A. Teaching small language models to reason. *arXiv preprint* arXiv: 2212.08410, 2022.
[139] Ho, N.; Schmid, L.; Yun, S. Y. Large language models are reasoning teachers. *arXiv preprint* arXiv:2212.10071, 2022.
[140] Fu, Y.; Peng, H.; Ou, L.; Sabharwal, A.; Khot, T. Specializing smaller language models towards multi-step reasoning. *arXiv preprint* arXiv: 2301.12726, 2023.
[141] Valmeekam, K.; Marquez, M.; Olmo, A.; Sreedharan, S.; Kambhampati, S. PlanBench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *arXiv preprint* arXiv:2206.10498, 2022.
[142] Lu, P.; Mishra, S.; Xia, T.; Qiu, L.; Chang, K.; Zhu, S. C.; Tafjord, O.; Clark, P.; Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering. *arXiv preprint* arXiv: 2209.09513, 2022.
[143] Zhang, Z.; Zhang, A.; Li, M.; Zhao, H.; Karypis, G.; Smola, A. Multimodal chain-of-thought reasoning in language models. *arXiv preprint* arXiv:2302.00923, 2023.
[144] Shi, F.; Suzgun, M.; Freitag, M.; Wang, X.; Srivats, S.; Vosoughi, S.; Chung, H. W.; Tay, Y.; Ruder, S.; Zhou, D. Language models are multilingual chain-of-thought reasoners. *arXiv preprint* arXiv:2210. 03057, 2022.
[145] Shaikh, O.; Zhang, H.; Held, W.; Bernstein, M.; Yang, D. On second thought, let's not think step by step! bias and toxicity in zero-shot reasoning. *arXiv preprint* arXiv:2212.08061, 2022.