Augmented Language Models: a Survey

Grégoire Mialon* *gmialon@meta.com*
Roberto Dessì*† *rdessi@meta.com*
Maria Lomeli* *marialomeli@meta.com*
Christoforos Nalmpantis* *christoforos@meta.com*
Ram Pasunuru* *rpasunuru@meta.com*
Roberta Raileanu* *raileanu@meta.com*
Baptiste Rozière* *broz@meta.com*
Timo Schick* *schick@meta.com*
Jane Dwivedi-Yu* *janeyu@meta.com*
Asli Celikyilmaz* *aslic@meta.com*
Edouard Grave* *egrave@meta.com*
Yann LeCun* *yann@meta.com*
Thomas Scialom* *tscialom@meta.com*

*Meta AI †Universitat Pompeu Fabra

Abstract

This survey reviews works in which language models (LMs) are augmented with reasoning skills and the ability to use tools. The former is defined as decomposing a potentially complex task into simpler subtasks while the latter consists in calling external modules such as a code interpreter. LMs can leverage these augmentations separately or in combination via heuristics, or learn to do so from demonstrations. While adhering to a standard missing tokens prediction objective, such augmented LMs can use various, possibly non-parametric external modules to expand their context processing ability, thus departing from the pure language modeling paradigm. We therefore refer to them as Augmented Language Models (ALMs). The missing token objective allows ALMs to learn to reason, use tools, and even act, while still performing standard natural language tasks and even outperforming most regular LMs on several benchmarks. In this work, after reviewing current advance in ALMs, we conclude that this new research direction has the potential to address common limitations of traditional LMs such as interpretability, consistency, and scalability issues.

Contents

1 Introduction: motivation for the survey and definitions **2**
1.1 Motivation . . . 2
1.2 Our classification . . . 4

2 Reasoning **4**
2.1 Eliciting reasoning with prompting . . . 4
2.2 Recursive prompting . . . 6
2.3 Explicitly teaching language models to reason . . . 6
2.4 Comparison and limitations of abstract reasoning . . . 8

3 Using Tools and Act **9**
3.1 Calling another model . . . 9
3.2 Information retrieval . . . 11
3.2.1 Retrieval-augmented language models . . . 11
3.2.2 Querying search engines . . . 12
3.2.3 Searching and navigating the web . . . 12
3.3 Computing via Symbolic Modules and Code Interpreters . . . 13
3.4 Acting on the virtual and physical world . . . 13

4 Learning to reason, use tools, and act **15**
4.1 Supervision . . . 15
4.2 Reinforcement learning . . . 16
4.3 Limitations and future directions . . . 18

5 Discussion **19**

6 Conclusion **21**

1 Introduction: motivation for the survey and definitions

1.1 Motivation

Large Language Models (LLMs) (Devlin et al., 2019; Brown et al., 2020; Chowdhery et al., 2022) have fueled dramatic progress in Natural Language Processing (NLP) and are already core in several products with millions of users, such as the coding assistant Copilot (Chen et al., 2021), Google search engine[1] or more recently ChatGPT[2]. Memorization (Tirumala et al., 2022) combined with compositionality (Zhou et al., 2022) capabilities made LLMs able to execute various tasks such as language understanding or conditional and unconditional text generation at an unprecedented level of performance, thus opening a realistic path towards higher-bandwidth human-computer interactions.

However, LLMs suffer from important limitations hindering a broader deployment. LLMs often provide non-factual but seemingly plausible predictions, often referred to as hallucinations (Welleck et al., 2020). This leads to many avoidable mistakes, for example in the context of arithmetics (Qian et al., 2022) or within a reasoning chain (Wei et al., 2022c). Moreover, many LLMs groundbreaking capabilities seem to emerge with size, measured by the number of trainable parameters: for example, Wei et al. (2022b) demonstrate that LLMs become able to perform some BIG-bench tasks[3] via few-shot prompting once a certain scale is attained. Although a recent line of work yielded smaller LMs that retain some capabilities from their largest counterpart (Hoffmann et al., 2022), the size and need for data of LLMs can be impractical for training but also maintenance: continual learning for large models remains an open research question (Scialom et al., 2022). Other limitations of LLMs are discussed by Goldberg (2023) in the context of *ChatGPT*, a chatbot built upon *GPT3*.

We argue these issues stem from a fundamental defect of LLMs: they are generally trained to perform statistical language modeling given (i) a single parametric model and (ii) a limited context, typically the n previous or surrounding tokens. While n has been growing in recent years thanks to software and hardware

[1] See *e.g.* https://blog.google/products/search/search-language-understanding-bert/
[2] https://openai.com/blog/chatgpt/
[3] https://github.com/google/BIG-bench

innovations, most models still use a relatively small context size compared to the potentially large context needed to always correctly perform language modeling. Hence, massive scale is required to store knowledge that is not present in the context but necessary to perform the task at hand.

As a consequence, a growing research trend emerged with the goal to solve these issues, slightly moving away from the pure statistical language modeling paradigm described above. For example, a line of work circumvents the limited context size of LLMs by increasing its relevance: this is done by adding information extracted from relevant external documents. Through equipping LMs with a module that retrieves such documents from a database given a context, it is possible to match certain capabilities of some of the largest LMs while having less parameters (Borgeaud et al., 2022; Izacard et al., 2022). Note that the resulting model is now non-parametric since it can query external data sources. More generally, LMs can also improve their context via reasoning strategies (Wei et al. (2022c); Taylor et al. (2022); Yang et al. (2022c) *inter alia*) so that a more relevant context is produced in exchange for more computation before generating an answer. Another strategy is to allow LMs to leverage external tools (Press et al. (2022); Gao et al. (2022); Liu et al. (2022b) *inter alia*) to augment the current context with important missing information that was not contained in the LM's weights. Although most of these works aim to alleviate the downfalls of LMs mentioned above separately, it is straightforward to think that more systematically augmenting LMs with both reasoning and tools may lead to significantly more powerful agents. We will refer to these models as **Augmented Language Models (ALMs)**. As this trend is accelerating, keeping track and understanding the scope of the numerous results becomes arduous. This calls for a taxonomy of ALMs works and definitions of technical terms that are used with sometimes different intents.

Definitions. We now provide definitions for terms that will be used throughout the survey.

- **Reasoning.** In the context of ALMs, reasoning is decomposing a potentially complex task into simpler subtasks the LM can solve more easily by itself or using tools. There exist various ways to decompose into subtasks, such as recursion or iteration. In that sense, reasoning is akin to planning as defined for example in LeCun (2022). In this survey, reasoning will very often refer to the various strategies to improve reasoning skills in LMs, such as step-by-step reasoning using few-shot examples. It is not yet fully understood whether the LM is really reasoning, or simply producing a larger context that increases the likelihood of correctly predicting the missing tokens. We refer to Huang and Chang (2022) for a discussion on this topic: although reasoning may currently be an abuse of language given the current state of the art, the term is already in use within the community. A more pragmatic definition of reasoning in the context in ALMs is giving more computation steps to the model before yielding the answer to a prompt.

- **Tool.** For ALMs, a tool is an external module that is typically called using a rule or a special token and whose output is included in the ALM's context. The tool can gather external information, or have an effect on the virtual or physical world (generally perceived by the ALM). An example of a tool fetching external information is a document retriever, while a tool having an external effect is a robotic arm. A tool can be called at training or at inference time. More generally, learning to interact with a tool may consist in learning to call its API.

- **Act.** For ALMs, calling a tool having an effect on the virtual or physical world and observing the result, typically by including it in the ALM's current context. For example, some works from the survey discuss searching the web, or robotic arm manipulation via LMs. With a slight abuse of term, we will sometimes denote the call of a tool by an ALM as an action, even if it does not have an external effect.

Why jointly discussing reasoning and tools? The combination of reasoning and tools within LMs should allow to solve a broad range of complex tasks without heuristics, hence with better generalization capabilities. Typically, reasoning would foster the LM to decompose a given problem into potentially simpler subtasks while tools would help getting each step right, for example obtaining the result from a mathematical operation. Put it differently, reasoning is a way for LMs to combine different tools in order to solve complex tasks, and tools are a way to not fail a reasoning with valid decomposition. Both should benefit from the

other. Moreover, reasoning and tools can be put under the same hood, as both augment the context of the LM so that it better predicts the missing tokens, albeit in a different way.

Why jointly discussing tools and actions? Tools that gather additional information and tools that have an effect on the virtual or physical world can be called in the same fashion by the LM. For example, there is seemingly no difference between a LM outputting python code for solving a mathematical operation, and a LM outputting python code to manipulate a robotic arm. A few works discussed in the survey are already using LMs that have effects on the virtual or physical world: under this view, we can say that the LM have the potential to act, and expect important advances in the direction of LMs as autonomous agents.

1.2 Our classification

We decompose the works included in the survey under three axes. Section 2 studies works which augment LM's reasoning capabilities as defined above. Section 3 focuses on works allowing LMs to interact with external tools and act. Finally, Section 4 explores whether reasoning and tools usage are implemented via heuristics or learned, *e.g.* via supervision or reinforcement. Other axes could naturally have been chosen for this survey and are discussed in Section 5. For conciseness, the survey focuses on works that combine reasoning or tools with LMs. However, the reader should keep in mind that many of these techniques were originally introduced in another context than LMs, and consult the introduction and related work section of the papers we mention if needed. Finally, although we focus on LLMs, not all works we consider employ large models, hence we stick to LMs for correctness in the remainder of the survey.

2 Reasoning

In general, reasoning is the ability to make inferences using evidence and logic. Reasoning can be divided into multiple types of skills such as commonsense reasoning (McCarthy et al., 1960; Levesque et al., 2012), mathematical reasoning (Cobbe et al., 2021), symbolic reasoning (Wei et al., 2022c), etc. Often, reasoning involves deductions from inference chains, called as multi-step reasoning. In the context of LMs, we will use the definition of reasoning provided in Section 1. Previous work has shown that LLMs can solve simple reasoning problems but fail at complex reasoning (Creswell et al., 2022): hence, this section focuses on various strategies to augment LM's reasoning skills. One of the challenges with complex reasoning problems for LMs is to correctly obtain the solution by composing the correct answers predicted by it to the sub-problems. For example, a LM may correctly predict the dates of birth and death of a celebrity, but may not correctly predict the age. Press et al. (2022) call this discrepancy the compositionality gap for LMs. For the rest of this section, we discuss the works related to three popular paradigms for eliciting reasoning in LMs. Note that Huang and Chang (2022) propose a survey on reasoning in language models. Qiao et al. (2022) also propose a survey on reasoning albeit with a focus on prompting. Since our present work focuses on reasoning combined with tools, we refer the reader to Huang and Chang (2022); Qiao et al. (2022) for a more in-depth review of works on reasoning for LLMs.

2.1 Eliciting reasoning with prompting

In recent years, prompting LMs to solve various downstream tasks has become a dominant paradigm (Brown et al., 2020). In prompting, examples from a downstream task are transformed such that they are formulated as a language modeling problem. Prompting typically takes one of the two forms: zero-shot, where the model is directly prompted with a test example's input; and few-shot, where few examples of a task are prepended along with a test example's input. This few-shot prompting is also known as in-context learning or few-shot learning. As opposed to "naive" prompting that requires an input to be directly followed by the output/answer, elicitive prompts encourage LMs to solve tasks by following intermediate steps before predicting the output/answer. Wei et al. (2022c) showed that elicitive prompting enables LMs to be better reasoners in a few-shot setting. Later, Kojima et al. (2022) showed similar ability in a zero-shot setting. We discuss them in detail in the following paragraphs.

Question: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
Answer: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.
Question: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?
Answer:
<LM>

Figure 1: An example of few-shot Chain-of-Thought prompt. **<LM>** denotes call to the LM with the above prompt.

Few-shot setting. Wei et al. (2022c) introduced chain-of-thought (CoT), a few-shot prompting technique for LMs. The prompt consists of examples of a task, with inputs followed by intermediate reasoning steps leading to the final output, as depicted in Figure 1. Table 1 shows that CoT outperforms standard prompting methods. Wei et al. (2022b) observe that the success of the few-shot strategy emerges with scale, while Tay et al. (2022) add that without fine-tuning, successful use of CoT generally requires 100B+ parameters LMs such as *LaMDA* (Thoppilan et al., 2022), *PaLM* (Chowdhery et al., 2022) or *GPT3* (Brown et al., 2020; Ouyang et al., 2022), before proposing *UL2*, a 20B open source model that can perform CoT. Using few-shot CoT prompting, *Minerva* (Lewkowycz et al., 2022) achieves excellent performance on math benchmarks such as GSM8K (Cobbe et al., 2021). Wang et al. (2022c) further improve CoT with *Self-consistency*: diverse reasoning paths are sampled from a given language model using CoT, and the most consistent answer is selected as the final answer. Press et al. (2022) introduce *Self-ask*, a prompt in the spirit of CoT. Instead of providing the model with a continuous chain of thought as in Figure 1, *Self-ask* explicitly states the follow-up question before answering it and relies on a scaffold (e.g, *"Follow-up question:"* or *"So the final answer is:"*), so that the answers are more easily parseable. The authors demonstrate an improvement over CoT on their introduced datasets aiming at measuring the compositionality gap. They observe that this gap does not narrow when increasing the size of the model. Note that Press et al. (2022) focus on 2-hop questions, *i.e.*, questions for which the model only needs to compose two facts to obtain the answer. Interestingly, *Self-ask* can easily be augmented with a search engine (see Section 3). *ReAct* (Yao et al., 2022b) is another few-shot prompting approach eliciting reasoning that can query three tools throughout the reasoning steps: `search` and `lookup` in Wikipedia, and `finish` to return the answer. *ReAct* will be discussed in more detail in the next sections.

Zero-shot setting. Kojima et al. (2022) extend the idea of eliciting reasoning in LMs to zero-shot prompting. Whereas few-shot provides examples of the task at hand, zero-shot conditions the LM on a single prompt that is not an example. Here, Kojima et al. (2022) simply append *Let's think step by step* to the input question before querying the model (see Figure 2), and demonstrate that zero-shot-CoT for large LMs does well on reasoning tasks such as GSM8K although not as much as few-shot-CoT.

Question: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?
Answer: Let's think step by step
<LM>

Figure 2: An example of zero-shot Chain-of-Thought prompt. **<LM>** denotes call to the LM with the above prompt.