https://doi.org/10.1007/s11704-025-50058-z

REVIEW ARTICLE

A comprehensive taxonomy of prompt engineering techniques for large language models

Yao-Yang LIU[1], Zhen ZHENG[2], Feng ZHANG[1]✉, Jin-Cheng FENG[1], Yi-Yang FU[1], Ji-Dong ZHAI[3], Bing-Sheng HE[4], Xiao ZHANG[1], Xiao-Yong DU[1]

1. Key Laboratory of Data Engineering and Knowledge Engineering (MOE), School of Information, Renmin University of China, Beijing 100872, China
2. Microsoft AI, Microsoft Asia Pacific Research and Development Group, Beijing 100080, China
3. Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
4. School of Computing, National University of Singapore, Singapore 119077, Singapore

Received January 21, 2025; accepted March 19, 2025

E-mail: fengzhang@ruc.edu.cn

Abstract

Large Language Models (LLMs) have demonstrated remarkable performance across various downstream tasks, as evidenced by numerous studies. Since 2022, generative AI has shown significant potential in diverse application domains, including gaming, film and television, media, and finance. By 2023, the global AI-generated content (AIGC) industry had attracted over fanxiexian_myfh26 billion in investment. As LLMs become increasingly prevalent, prompt engineering has emerged as a key research area to enhance user-AI interactions and improve LLM performance. The prompt, which serves as the input instruction for the LLM, is closely linked to the model's responses. Prompt engineering refines the content and structure of prompts, thereby enhancing the performance of LLMs without changing the underlying model parameters. Despite significant advancements in prompt engineering, a comprehensive and systematic summary of existing techniques and their practical applications remains absent. To fill this gap, we investigate existing techniques and applications of prompt engineering. We conduct a thorough review and propose a novel taxonomy that provides a foundational framework for prompt construction. This taxonomy categorizes prompt engineering into four distinct aspects: profile and instruction, knowledge, reasoning and planning, and reliability. By providing a structured framework for understanding its various dimensions, we aim to facilitate the systematic design of prompts. Furthermore, we summarize existing prompt engineering techniques and explore the applications of LLMs across various domains, highlighting their interrelation with prompt engineering strategies. This survey underscores the progress of prompt engineering and its critical role in advancing AI applications, ultimately aiming to provide a systematic reference for future research and applications.

Keywords

prompt engineering; large language models; AI agents; survey; taxonomy

1 Introduction

Large Language Models (LLMs), such as GPT-4, have attracted considerable attention due to their advanced capabilities in language comprehension and generation [1–12]. Users can effectively leverage the diverse competencies of LLMs by employing task-specific instructions, or prompts [13–16]. However, despite their impressive abilities, LLMs encounter several challenges in practical applications. For instance, while Reinforcement Learning from Human Feedback (RLHF) training enhances LLM conversational skills, making them more human-like, this anthropomorphic assumption does not universally apply to all LLMs [17]. As a result, informal prompts often yield responses from LLMs that lack the professionalism and precision expected in various contexts. The generalization capabilities of LLMs can lead to highly variable outputs when presented with unstructured prompts. Additionally, LLMs are susceptible to the "hallucination" problem, where generated content may include logical fallacies, fabricated facts, and data-driven biases. Consequently, designing prompts that yield more professional, stable, and reliable responses remains a significant challenge.

To address these challenges, extensive research has been dedicated to developing and refining prompt engineering techniques, which are essential for enhancing the performance of LLMs across various tasks and application domains. Researchers have explored diverse methodologies, including role-play prompting [18–20], task instruction [21], knowledge augmentation [22–26], and recursive

prompting for reasoning [27–29]. For instance, Wei et al. [30] introduced a linear method for recursive prompting known as the "Chain-of-Thought" (CoT) approach. Although the CoT prompting technique has shown promising results, its inherent greedy strategy and linear structure introduce significant limitations, especially in scenarios that demand nuanced reasoning and greater flexibility. In response, Long [31] developed the Tree-of-Thought approach, which utilizes a tree-based structure. Similarly, Yao et al. [32] proposed the Graph-of-Thought approach, leveraging graph structures to enhance reasoning capabilities. As the application of system-level techniques in prompt engineering expands, LangChain[1)], a comprehensive framework integrates prompt templates with various system components to optimize prompts for LLMs. Although considerable research has been conducted on prompt engineering, a comprehensive review of the topic remains lacking.

1.1 Related surveys

The closest to our work are surveys on prompt learning and prompt engineering. Liu et al. [33] proposed the concept of prompt-based learning and provided an overview of fundamental principles, presenting a comprehensive range of prompt forms, including soft and hard prompts. Wei et al. [34] explored LLM performance under different prompt techniques, emphasizing its emergent abilities while reviewing challenges and outlining future directions for enhancing model performance through prompt engineering. With the advancement of LLMs, researchers have increasingly focused on hard prompt-based engineering techniques for these models. Recently, Sahoo et al. [35] has provided a description of the techniques and applications of prompt engineering, dividing prompt engineering techniques and applications into 12 categories within fixed scenarios. Li et al. [36] summarized goal-oriented prompt engineering techniques based on human reasoning principles. Some surveys concentrate on specific aspects, such as the study [28], which explores techniques aimed at enhancing reasoning performance, while Yu et al. [29] discussed these techniques from a philosophical reasoning perspective. Meanwhile, Mialon et al. [37] centered on prompt augmentation.

Synergy. While existing surveys focus on summarizing prompt engineering techniques within fixed scenarios, much less has been done to provide a general framework for prompt engineering. Prompt engineering is applied in numerous scenarios. Hence, classification fixed to a specific scenario complicates its migration to new application tasks. This calls for a taxonomy that encompasses the entire pipeline for LLM applications across various use cases. In this paper, we propose a universal prompt engineering taxonomy that outlines the pipeline for designing effective prompts. This taxonomy categorizes prompt engineering techniques from the perspective of their underlying principles. Building on this taxonomy, we correlate various applications of LLMs with prompt engineering techniques. Furthermore, this taxonomy encompasses both fundamental and advanced prompt techniques, offering detailed guidance on prompt design. With the continuous advancement of prompt engineering techniques, this taxonomy can be further extended.

1.2 Our contribution

We restrict our focus to discrete prefix prompts [38] rather than cloze prompts [39,40], as modern LLM architectures (specifically decoder-only models) demonstrate superior performance in various fields. Furthermore, our study concentrates on hard (discrete) prompts [33,41] rather than soft (continuous) prompts [33,42]. Finally, we refine the scope of prompt engineering as the technology to modify prompt content, allowing us to cover the full range of user and system-level technologies. To the best of our knowledge, no existing survey reviews prompt engineering techniques from a principled perspective and summarizes their applications. In contrast to previous surveys, the primary contributions of this paper are as follows:

- This survey represents the first comprehensive analysis of prompt engineering, approached from the perspective of human problem-solving principles. It includes an in-depth exploration of the field's background, taxonomy, applications, and a summary of prompt engineering.
- We present a comprehensive taxonomy of prompt engineering across four aspects: profile and instruction, knowledge, reasoning and planning, and reliability. This taxonomy provides a foundational framework encompassing essential building blocks and methodological abstractions crucial for prompt engineering.
- We summarize existing typical and state-of-the-art studies according to their domains, providing a convenient reference for researchers and developers.
- We generalize the taxonomy of methodologies as design factors for successful prompt engineering and propose a reasonable process flow for designing prompts.

2 Background

Prompt engineering is the technique of guiding model output through the strategic design of task-specific instructions (prompts) without altering model parameters. This approach has gained traction as a means to enhance the performance of LLMs across various tasks and fields. Research [43,44] indicates that the text input to LLMs significantly influences their effectiveness in downstream tasks. Therefore, prompt engineering serves as a strategic tool to guide model outputs. Given the intrinsic link between prompts and LLM functions, it is essential to design prompts with a nuanced understanding of LLM capabilities.

2.1 LLMs' Basic capabilities

Understanding the diverse capabilities of LLMs is crucial for creating effective and task-specific prompts. Existing literature [45] highlights that LLMs exhibit a broad range of capabilities, including reasoning, instruction-following, and in-context learning (ICL).

Reasoning is a core ability of LLMs, enabling them to perform complex multi-step reasoning through specific prompt design methods. This fundamental problem-solving ability supports various practical applications, such as medical diagnosis, legal decisions, and virtual assistants. Instruction-following refers to the LLM completing

[1)] LangChain-ai. See www.langchain.com/ website, 2024.

a new task according to task instructions without using an example. The generalization ability of LLMs allows them to generate, summarize, extract, classify, and rewrite text based on designed instructions.

In-context learning (ICL) is a paradigm that allows LLMs to perform tasks by learning from a few examples provided within the context [46]. This approach diverges from conventional supervised learning, which requires extensive training datasets and model parameter updates. The essence of ICL lies in its ability to leverage analogy and pattern recognition. When faced with a new query, LLMs refer to demonstration examples in the context to identify and apply relevant patterns without altering their underlying parameters. This process is akin to how humans learn from a few instances and generalize knowledge to new situations. The model's predictions are made by concatenating the query with contextual examples and using a scoring function to determine the most likely outcome. Prompt engineering primarily leverages ICL capabilities to steer the output of LLMs by carefully designing the content and structure of the input context.

2.2 Classification

To systematically categorize and generalize various prompt engineering techniques, we propose a novel taxonomy based on the functional divisions within agent systems. This approach aims to ensure both comprehensiveness and orthogonality in our taxonomy. Current research in agent systems divides agent functions into four main categories: profile function, memory function, planning function, and action function [47–49]. This comprehensive division ensures that any workflow based on LLMs can be covered by these four aspects.

Building on this functional division, we align prompt engineering workflows with these categories. We classify prompt engineering into four aspects:

- Profile and instruction. This foundational aspect defines the basic attributes and scenarios for LLMs. It standardizes LLM responses based on the information provided through natural language prompts. Profile and instruction examines prompt design from the perspective of textual content, proposing a foundational framework for basic prompt construction. Advanced prompt engineering techniques are developed based on this foundational framework.
- Knowledge. This aspect involves incorporating information from a local database into the prompt to mitigate the "hallucination" problem and improve the professionalism of the LLM. Although Knowledge techniques generally enhance LLMs in most cases, it is still necessary to assess whether knowledge augmentation is required based on the specific task.
- Reasoning and planning. This approach enhances the reasoning ability of LLMs. It includes decomposing goals, such as Chain of Thought [30], and utilizing tools and feedback to facilitate reasoning. This is crucial for improving LLM performance when handling complex tasks.
- Reliability. This aspect refers to the process of reducing bias in LLM responses. It involves ensuring both the stability of content generated multiple times (content bias) and the generation of content that is free of bias, stereotypes, or cultural impairment (value bias). Reliability is a crucial step in prompt design, and it can be combined with other prompt engineering techniques to enhance the performance of LLMs in real-world tasks.

Synergy. Our taxonomy provides a universal method for prompt design. Within this framework, profile and instruction represent the initial step in prompt design, establishing a foundational prompt suitable for daily use. Reliability serves as the final step, ensuring the stability and safety of the model's responses, which is essential for the practical application of prompt engineering. Both knowledge and reasoning and planning are techniques that enhance foundational prompts, improving the performance of LLMs through knowledge augment, target decomposition, and self-feedback. The application of these enhancement techniques depends on the specific use case. To date, our four categories encompass the vast majority of prompt engineering techniques. It is anticipated that future advancements in prompt engineering will focus on enhancing prompts. Due to the generality of our taxonomy, it can be expanded accordingly.

2.3 Application

In Subsection 2.2, we introduce a comprehensive taxonomy for prompt engineering, categorizing the process of prompt construction in practical applications into four distinct components. Prompt engineering facilitates the practical deployment of LLMs across various applications. In Section 4, we aim to provide a thorough classification of the application areas of LLMs. To align with our proposed taxonomy, we categorize the application fields of LLMs into two main areas: Cognitive Applications of LLMs and Transformative Applications of LLMs.

1) Cognitive applications of LLMs

Cognitive applications of LLMs refer to instances where users leverage LLMs to acquire or process information. The extensive number of parameters in LLMs endows them with vast knowledge and robust capabilities for knowledge processing. Based on the types of results generated by LLMs, cognitive applications can be classified into two categories.

- Information acquisition. This process involves extracting pertinent knowledge from LLMs through structured dialogues. The LLM outputs knowledge that meets task requirements. Common applications include chatbots and professional knowledge Q&A systems, where LLMs assist users in obtaining necessary information. In computer science, LLMs enhance system performance in search engines and training data augmentation.
- In-depth information analysis. This application leverages LLMs to analyze and reason over user input data, generating conclusions derived from the underlying information. In fields such as financial markets and chemical analysis, LLMs perform comprehensive analyses of complex data from multiple dimensions, deriving valuable insights and summaries.

2) Transformative applications of LLMs

Transformative Applications of LLMs refer to scenarios where LLMs autonomously execute task processes, reducing the need for human intervention. Based on the types of tasks executed by the LLM and the structure of prompt organization, transformative tasks can be categorized into two categories.

- Physical world applications. By employing prompt engineering techniques, such as Task Information (Subsection 3.1) and RAG (Subsection 3.2), LLMs can autonomously execute tasks in fields such as software engineering [50,51] and robotics [52]. With advancements in prompt engineering methodologies, LLMs have the potential to undertake specialized professional roles, such as AI-driven doctors [53] and AI legal advisors [54,55].
- Creative applications. By leveraging prompt engineering techniques, LLMs can generate literature and music [56–58], effectively replacing humans in the creative workflow. With the advent of increasingly multimodal LLMs, the application of these models in video creation has emerged as a significant area of development [59].

3 Taxonomy

This section provides an overview of current prompt engineering techniques in the context of generative model prompting. As illustrated in Fig. 1, we refine these techniques based on the distinctive features of different stages of the processing task.

3.1 Profile and instruction

Insight: Profile and Instruction summarizes prompt engineering techniques based on textual content. It defines the LLM's fundamental attributes and task specifications by designing prompts that incorporate personality information, task descriptions, and few-shot examples. As the first step in constructing prompts, it establishes a framework for more advanced prompt engineering approaches.

The primary purpose of this line of work is to determine a reasonable knowledge location for the LLM [60], concretely embodied in personality information (Seubsction 3.1), task information (Subsection 3.1), and demonstration information (Subsection 3.1), as shown in Fig. 2.

1) Personality information

LLMs can perform tasks by adopting specific roles, such as lawyers, teachers, and domain experts [18,19]. Personality information is a crucial part of the profile, defining the attributes of the LLM's role. These features are often included at the start of prompts to shape the LLM's response. Typically, personality information covers key characteristics like age, gender, and profession [20], along with tone and psychological traits, thereby reflecting the personality of the LLM's role. Assigning a specific role to an LLM establishes a suitable starting point for its responses. For instance, texts containing information about law are more likely to appear in the context of the word "lawyer". As a result, law-related information receives more attention along with the word "lawyer" during the inference stage, leading to responses pertinent to legal matters.

2) Task information

Task information plays a pivotal role in prompt design. Providing clear and well-defined task instructions enables LLMs to produce more specialized and contextually relevant outputs.

Task instruction. Task instruction is a prompt technique used to standardize the output of LLMs by incorporating clear task objectives and requirements into the prompt. This approach can be summarized into three key aspects: intent, domain, and demand. As shown in Fig. 3, intent defines the task's goal, demand specifies the detailed requirements, and domain identifies the information source relevant to the task. Research [61] has shown that when fundamental instructions lack clarity, LLM responses tend to be overly general. If

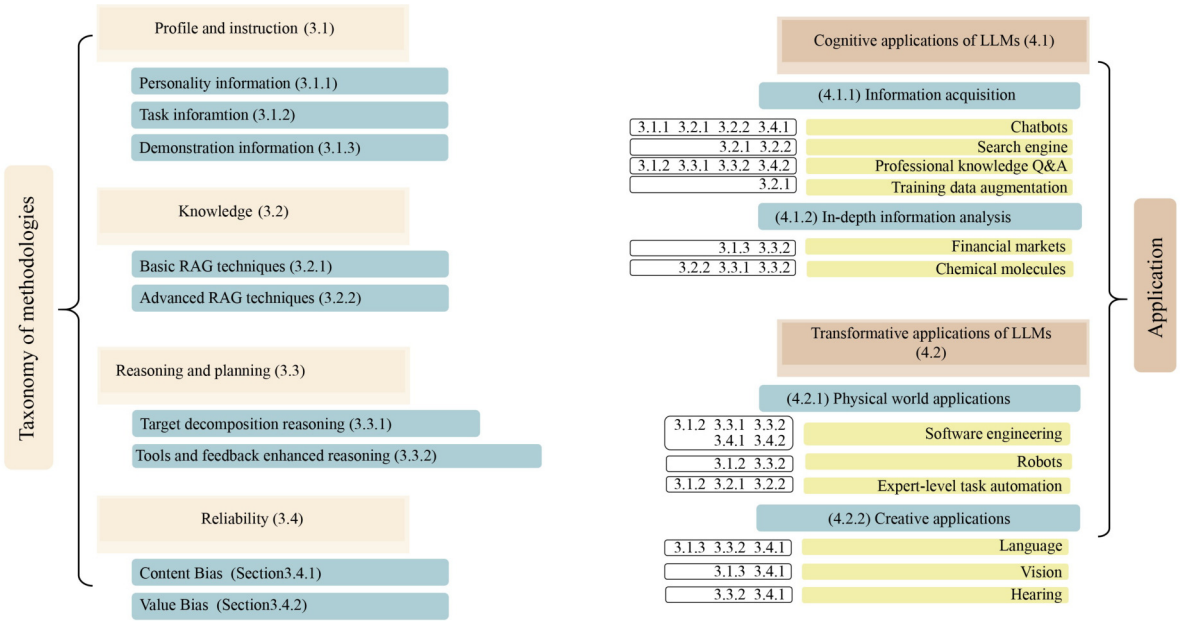


Fig. 1 The taxonomy tree of prompt engineering methodologies. The application tree on the right reveals the relationship between the practical application of LLMs and our categorized prompt engineering techniques (the numbers in parentheses stand for the corresponding subsections)

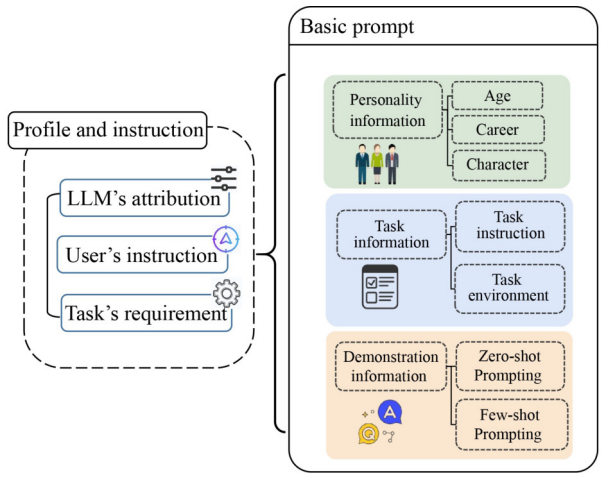


Fig. 2 Profile and instruction categorizes basic prompts into three components: personality information, task information, and demonstration information. This framework provides a foundational structure for prompt design in general-purpose applications

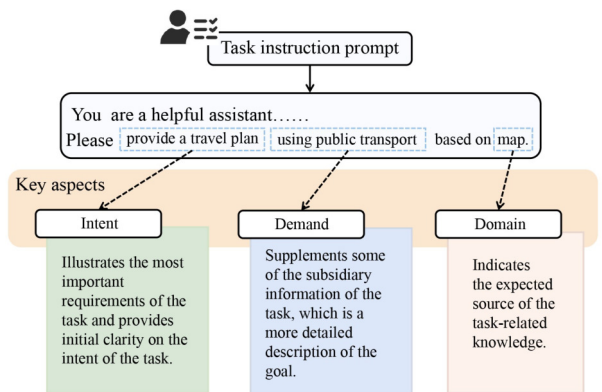


Fig. 3 An example prompt of the task instruction paradigm. The prompt consists of intent, demand, and domain, providing the basic information elements of the task

the prompt's structure is ambiguous and the content is too broad, the LLM faces numerous options, making it difficult to focus on the critical parts of the prompt. Consequently, this can lead to extensive but unfocused results. Studies such as [21] suggest that instruction understanding is a promising alternative paradigm for few-shot learning. Compared to examples, instructions provide stronger expressiveness and more stringent constraint capabilities [21].

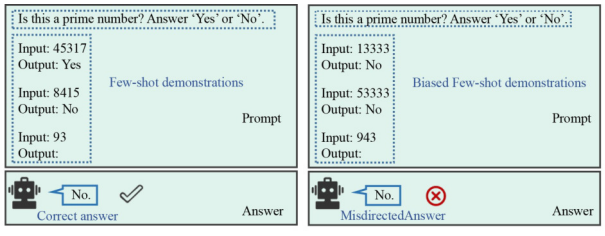
Task environment. Environment Information incorporates task environment details into the prompt depending on the type of task. Providing information regarding the virtual task environment of the LLM can facilitate response generation, ensuring alignment with task requirements. For instance, if the LLM is tasked with "Fetch a bottle from the kitchen", the LLM+P model [43] can incorporate information about the task environment and the cost of the action, such as the distance from the current location to the kitchen and the location of the objects that the task needs to interact with in the prompt. This facilitates the generation of a more feasible plan, represented using a PDDL (Planning Domain Definition Language) framework [43]. However, environmental information alone is not sufficient. The LLM should also learn the anticipated consequences of its actions and verify if the current environment meets the conditions specified in the prompt. DEPS [62] addresses this by describing the objects accessible to the LLM and defining an action format that outlines both the conditions and potential outcomes of those actions. This approach enables the LLM to understand the relationships between elements in the task environment, comprehend the environmental information, and generate responses consistent with the task requirements. These methods have shown significant improvements over traditional deep learning techniques in open-world scenarios, such as Minecraft [63].

3) Demonstration information

Demonstration information is a technique that involves adding specific input-output mappings to the prompt. Research [64] indicates that LLMs trained on sufficiently large and diverse datasets provide responses to zero-shot prompts that are comparable to those generated after supervised learning. Based on the number of labeled demonstrations, it can be categorized into zero-shot prompting with no examples and few-shot prompting with few examples.

Zero-shot prompting. Zero-shot prompting does not involve adding a labeled example to the prompt. It is composed of task and profile information [65]. This approach leverages the pre-existing knowledge of LLMs to generate responses based on the instructions of the task. Previous research [60] has shown that zero-shot prompting enables the model to access its existing knowledge by identifying already learned tasks. Furthermore, Reynolds et al. [60] suggested that there is significant potential for developing automated methods to generate task-appropriate zero-shot prompts.

Few-shot prompting. Unlike zero-shot prompting, few-shot prompting equips LLMs with a limited set of input-output examples, as illustrated in Fig. 4(a). This approach aids the model in comprehending both the task intent and the required output format. Providing several high-quality examples can improve the model's performance on complex tasks and standardize the form of outputs [66]. Carefully designed demonstrations within the prompt can achieve results comparable to fine-tuning, with the performance gap narrowing as the number of model parameters increases [13]. Several approaches are proposed for selecting and augmenting these demonstrations. For example, Liu et al. [67] and Su et al. [68] enhanced performance by choosing examples similar to the query input. Specifically, Liu et al. [67] employed a *K*-nearest neighbor



(a) (b)

Fig. 4 The biased demonstrations direct the LLM to the wrong location of the knowledge. This causes LLM to over-reference incomplete examples. (a) Few-shot prompting; (b) misdirected few-shot

(*K*NN) approach, while Su et al. [68] introduced Vote-*K* to incorporate diverse and representative examples by artificially labeling useful, previously unlabeled examples. Additionally, Jiang et al. [69] optimized the structure of examples, moving beyond the conventional "Q&A" format to identify the most suitable prompt template for each query through large corpus analysis.

However, few-shot prompting has some limitations, as shown in Fig. 4(b). Few-shot prompting requires additional tokens to incorporate the demonstrations, which presents a limitation for processing long text inputs. Recent research [43,70,71] demonstrates that the selection and variation of samples can significantly impact model performance. In response to these findings, Fei et al. [72] introduced a systematic method for measuring label biases, identifying three distinct types of label biases in in-context learning (ICL) for text classification. To address these biases, several approaches [71,73] are proposed to calibrate the model's output probabilities. These methods typically utilize output probabilities generated from a set of inputs either sourced from the task domain [73,74] or from standard task inputs [75]. By adjusting the samples in few-shot learning, these techniques aim to produce unbiased outputs.

3.2 Knowledge

Insight: Knowledge outlines prompt engineering techniques based on Retrieval-Augmented Generation (RAG).RAG functions by retrieving relevant informationfrom a knowledge database based on an initial prompt and integrating it with the original prompt. This approach enhances the timeliness and professionalism of LLMs, effectively mitigating hallucination issues and improving the model's performance in specialized tasks.

Knowledge techniques involve augmenting prompts with external documents or knowledge databases. This process integrates content from local knowledge databases into the original prompt, enhancing alignment with specific information and addressing challenges such as hallucination and timeliness in LLM training [76–78]. As the scale of parameters of LLMs grows, the computing resources required for fine-tuning also increase. Retrieval-Augmented Generation (RAG) is a key technology for supplementing LLMs with real-world information, enabling them to fully utilize their reasoning capabilities. Although RAG techniques encompass both retrieval and knowledge graph methodologies [76,79], this paper primarily focuses on summarizing RAG techniques that are directly related to prompt design, while excluding those unrelated to prompt engineering. According to the complexity of the RAG framework, we classify it into basic RAG techniques and advanced RAG techniques.

1) Basic RAG techniques

Basic Retrieval-Augmented Generation (RAG) techniques dynamically retrieve information from external knowledge sources, organize the final prompt based on the original query, and use the retrieved data as a reference. The RAG workflow typically consists of three main steps [26], as illustrated in Fig. 5:

- Indexing. Documents are divided into smaller chunks, encoded into vector representations, and stored in a database, such as an inverted index or a vector database.
- Retrieval. The top k chunks most relevant to the query are retrieved based on semantic similarity.
- Generation. The original query and the retrieved chunks are fed into a large language model (LLM) to generate the final response.

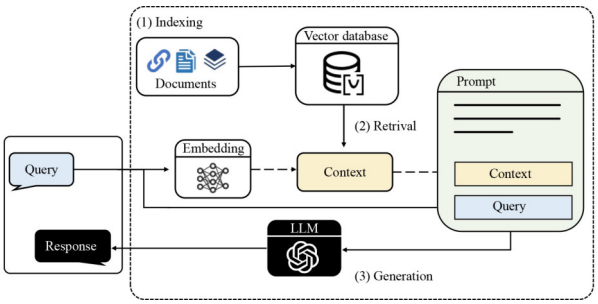


Fig. 5 A representative instance of the RAG process applied to question answering

A simple RAG prompt might look like: "Please answer the above question based on: Segment 1: Segment 2:". The segment chunks are then populated with retrieved results from the external knowledge source. Basic RAG techniques provide a foundational approach to augmenting LLMs with external knowledge. This can be implemented using functions and other relevant tools. There are several potential areas for optimization within basic RAG techniques [80]. From a prompt engineering perspective, three key challenges are: 1) retrieving the most relevant document, 2) effectively combining the retrieved content to achieve optimal results, and 3) iteratively refining the entire process [80].

2) Advanced RAG techniques

In this part, we begin with an overview of the representative RAG framework and then explore optimisation methods from three distinct perspectives: pre-retrieval optimization, post-retrieval optimization, and memory management. As illustrated in Fig. 6, advanced RAG enhances the alignment between retrieved knowledge and the prompt, enabling the construction of more coherent and effective prompts. The advanced RAG prompt engineering enhances the workflow of basic RAG by incorporating several key optimizations: In the pre-retrieval stage, it addresses semantic discrepancies between the query and document chunks by rewriting and decomposing the query. In the post-retrieval stage, it refines results by modifying the content and structure of the "query+document". Additionally, advanced RAG leverages retrieval history and conversation history to generate memory for LLMs, improving the accuracy of responses in long conversations.

- Pre-retrieval optimization. The pre-retrieval optimization focuses on augmenting the original query in order to retrieve documents that are more relevant to the task. MultiHop-RAG [23] developed a dataset designed to strengthen RAG's capabilities. This dataset includes a knowledge base and multi-hop queries. The queries are categorized into inference, comparison, temporal, and null queries, each tailored to specific types of reasoning tasks. Query standardization is performed based on their unique characteristics, ensuring consistency and effectiveness across all four query types.

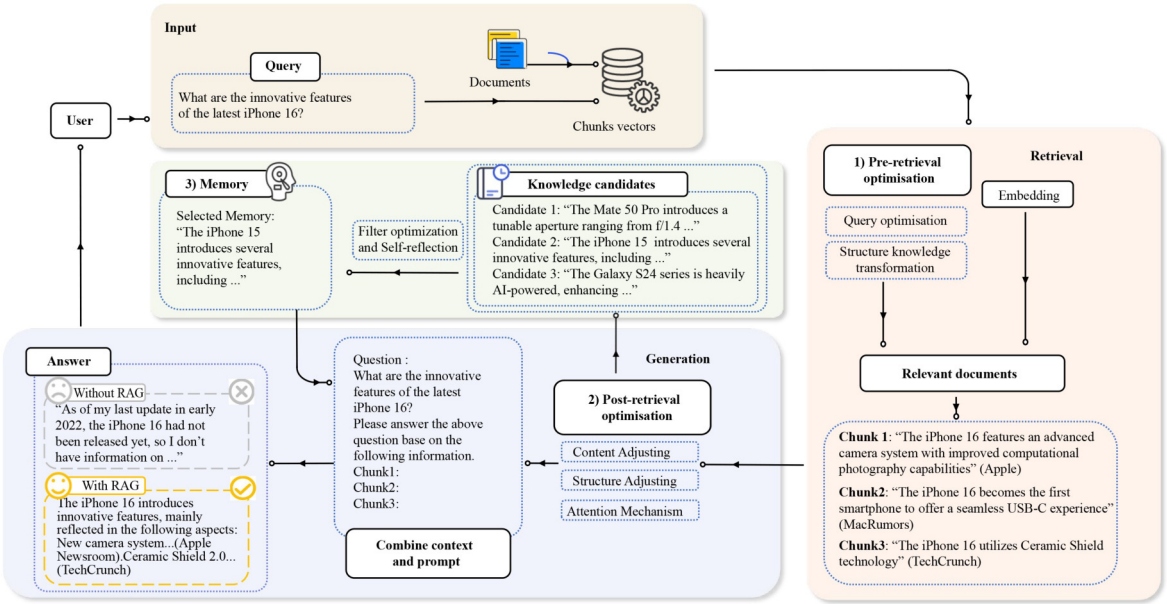


Fig. 6 A workflow of advanced retrieval-augmented generation (RAG) techniques. 1) Pre-retrieval optimisation phase. Before embedding the query, techniques such as query rewriting are employed to improve both the structure and content of the original query, facilitating the retrieval of more contextually relevant information. 2) Post-retrieval optimisation phase. Before inputting the retrieved documents into the LLM, the documents are refined through content filtering and structural adjustments to better align with the original query. 3) Memory module. This component preserves essential information and retrieval results in long-context dialogues, which enables more relevant retrieval when handling queries related to previous dialogue history

Query rewriting is another technique employed in pre-retrieval optimization. Query2doc [24] and ITER-RETGEN [81] both illustrate the effectiveness of query rewriting. These approaches leverage the capabilities of LLMs to generate pseudo-documents, which are then combined with the original query to form a revised version. This process effectively integrates corpus semantics into the user query. Query2doc has demonstrated significant improvements in BM25 [82] performance, achieving a 3% relative gain on MSMARCO [83] and a 15% relative gain on TREC DL [84], without requiring model fine-tuning. Similarly, ITER-RETGEN has shown enhanced performance across various question-answering tasks, including Natural Questions, TriviaQA [85], 2WikiMultiHopQA [86], and HotpotQA [87], surpassing previous baseline results.

- Post-retrieval optimization. The post-retrieval optimization focuses on the combination of retrieved documents and the user's original input. This approach refines the retrieved documents by adjusting their content, structure, and attention mechanisms as required.

(i) Content adjusting. RALM with CoN [88] reconstructs document content to enhance alignment with the original query. This method strategically emphasizes critical sections and modifies the retrieved document accordingly. Prior to inputting into the LLM, potential responses are generated from the retrieved content and subsequently evaluated to identify the most relevant context. ARM-RAG [89] takes a different approach, utilizing neural information retrieval to trace reasoning chains, particularly in solving mathematical problems. During their experiments, they found that accuracy could be improved by replacing words that might cause significant shifts in the model's reasoning. This technique involves blurring words that could disrupt the reasoning process, thus improving the model's overall accuracy.

(ii) Structure adjusting. Adjusting the structure of retrieved documents can enhance the effectiveness of the model's responses [90]. Articles can be classified into four categories based on their relevance to the query: Gold Documents, Relevant Documents, Related Documents, and Irrelevant Documents, in decreasing order of relevance [90]. Their study demonstrated that incorporating Irrelevant Documents into the document reconstruction process improves performance accuracy by more than 30%.

(iii) Attention mechanism. System2Attention (S2A) [91] enhances the soft attention mechanism in LLMs within the Transformer [92] architecture by refining and refocusing the attention process. Their approach leverages the LLM itself to build stronger attention mechanisms. Specifically, it uses prompts to adjust the LLM, enabling it to reconstruct the retrieved document into a new one by removing irrelevant text. Additionally, S2A introduces further techniques to refine attention. It generates the final response based on the reconstructed document, essentially refocusing attention one more time. S2A demonstrates promising performance, particularly on the TriviQA [85] dataset, where it outperforms LLaMA 2-70B-chat [93] in factuality with scores of 80.3% compared with 62.8%. On GSM-IC [94], S2A improves accuracy from 51.7% to 61.3%.

- Memory. In addition to constructing an effective external knowledge base, RAG can also store retrieved results and conversational histories to build memories. These memories are classified into two types: external and internal memories.

(i) External memories. The Selfmem framework [25] employs RAG in an iterative manner to create both a memory pool and a memory selector. This selector identifies an output to serve as memory for subsequent generation rounds. The core idea of this framework is based on prompt engineering. In this approach, the prompts presented to the LLM are crafted to be more similar to the model's outputs stored in the memory pool rather than the original training data. The memory pool undergoes multiple rounds of search optimization and retention to ensure that the most representative results are identified and preserved.

(ii) Internal memories. Internal Memories leverage the reasoning capabilities of LLMs to evaluate and provide feedback on the retrieved information, forming the internal memory of the LLM [26]. To implement this, the MetaRAG [95] framework is introduced, which incorporates three core processes: monitoring, evaluation, and planning during inference. MetaRAG has demonstrated significant performance improvements, achieving a 34.6% accuracy increase on the 2WikiMultihopQA dataset and a 26% accuracy improvement on the HotpotQA dataset.

3.3 Reasoning and planning

Insight: Reasoning and planning summarizes prompt engineering techniques aimed at enhancing the reasoning capabilities of LLMs, such as Chain-of-Thought (CoT) prompting. By decomposing tasks, leveraging external tools for reasoning, and incorporating feedback, it improves the model's ability to solve complex problems, as shown in Fig. 7. It represents a crucial step in prompt engineering and is a core component in the design of effective prompts.

1) Target decomposition reasoning

Target decomposition is the key prompt technology that enhances the reasoning ability of LLMs. It mirrors the core of human reasoning, where individuals, through experience, learn to tackle complex goals by breaking them down into more manageable sub-goals [96]. This section introduces the essential prompt strategies: plan-execute decomposition and iterative decomposition, as shown in Fig. 7.

- Plan-execute decomposition. The approach involves decomposing a complex query Q into a series of simpler subproblems that can be addressed sequentially. This method emphasizes the relationship between problem-solving steps, enabling the inheritance and promotion of previous solutions. The Least-To-Most Prompting method [97] consists of two main phases: Decomposition and Sub-problem Solving. In the Decomposition phase, the prompt includes examples and specific instructions to illustrate the breakdown process. Subsequently, in the Sub-problem Solving phase, attention shifts to demonstrating how each sub-problem is resolved using the provided examples, which are recorded in a list. This list stores previously answered sub-questions along with their corresponding answers, and it helps identify the next question to address in the sequence.

- Iterative decomposition. Chain of Thought (CoT) is the core technique of iterative decomposition. CoT decomposes task goals during the reasoning process. It iteratively decomposes the problem to identify and address sub-goals. This process is then repeated until the target is fully completed. Finally, the reasoning results are generated. The origin of the CoT within LLMs can be linked to the pioneering ideas proposed by Jason et al. [30]. Their ideas revolve around generating thought chains to enhance LLMs' ability to perform complex reasoning tasks. When LLMs are provided with limited samples during inference, the prompt follows a structured triplet format: <input, chain of thought, output>. This structured framework equips LLMs with the capacity to produce similar chains of reasoning, offering valuable insights into their computational capabilities and reasoning processes.

(i) User-level CoT prompt. The work by Wei et al. [30] is regarded as the pioneering study on CoT prompting. By providing the LLM with a sequence of intermediate reasoning steps within the prompt, the model can emulate the human problem-solving process, ultimately arriving at an accurate solution. This process can be further simplified through the use of zero-shot and few-shot prompting techniques, as outlined in Subsection 3.1. Additionally, Kojima et al. [64] demonstrated that the inclusion of the phrase "Let's think step by step" in the prompt enhances the LLM's ability

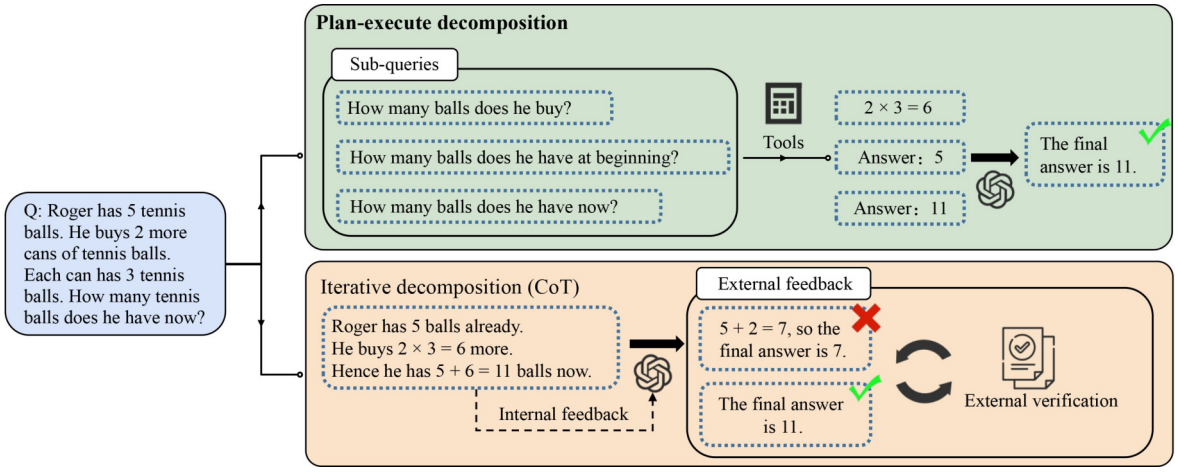


Fig. 7 An example of utilizing reasoning techniques to solve problems. Task decomposition methods complement LLMs by structuring the reasoning process, thereby enhancing their ability to tackle complex problems. The integration of tool utilization within prompts enables LLMs to concentrate on reasoning tasks, while feedback mechanisms further strengthen their reasoning capabilities

to perform CoT decomposition. Leveraging few-shot examples, LLMs can autonomously decompose and solve complex problems, leading to reductions in both energy consumption and processing time. The scope of the CoT is extensive, encompassing a wide range of problems encountered in daily human life, including mathematical calculations, common-sense reasoning, and more. Notably, CoT, based on few-shot examples, demands less effort and time compared to directly training an LLM.

(ii) Technique enhanced CoT. Following the introduction of CoT, researchers have focused on improving its efficiency and adaptability.

Auto-CoT automates problem sampling and inference chain generation using Question Clustering and Demonstration Sampling [98]. In clustering, Sentence-BERT [99] encodes queries as vectors, grouped via *K*-Means to form clusters of related problems. Each cluster's central problem is selected for Zero-Shot-CoT [64], generating structured inference chains. Active Prompt [100] addresses the challenge of adapting CoT manual annotation examples to diverse tasks. It enables LLMs to adjust to task-specific requirements through example prompts annotated with manually designed CoT inferences. To improve CoT interpretability, Faithful CoT [101] integrates Natural Language (NL) and Symbolic Logic (SL) programs for structured reasoning. The NL program decomposes queries into subproblems, while the SL program (e.g., Python and Datalog) solves them iteratively, enhancing accuracy and explainability.

Researchers have extended Chain-of-Thought reasoning by developing more flexible frameworks, among which XoT [102] represents a significant advancement. This approach enables adaptive switching between reasoning modes when encountering obstacles, guided by two distinct validation methods. Passive validation ensures fundamental correctness through basic error detection, while active validation assesses whether generated responses align with the original query. XoT also introduces diverse structural formats to enhance reasoning capabilities. Tree-of-Thought [102] structures reasoning hierarchically, enabling step-by-step decomposition, whereas Graph-of-Thought [32] models reasoning as an interconnected network, allowing for greater flexibility and adaptability in complex problem-solving.

2) Tools and feedback enhanced reasoning

Advanced reasoning techniques utilize systematic methodologies to optimize prompt design. These techniques include delegating specific computational tasks to external tools and simulating human feedback learning processes. The following discussion is organized into two sections: external support and feedback.

● External support. LLMs are expected to perform both semantic understanding tasks (e.g., task intent understanding) and complex reasoning tasks (e.g., numerical computation) when used for reasoning. However, current LLMs often face challenges in executing both tasks simultaneously. For instance, in complex code generation tasks, the generated code frequently contains errors or bugs. External support reduces the workload of LLMs by utilizing tools, allowing them to focus on reasoning and planning. According to the external support they utilize, external support is categorized into experimental simulators, code interpreters, and integration tools.

(i) Experimental simulator. Current LLMs face challenges when dealing with real-world problems that require a deep understanding of physical principles [103]. Due to the limitations of LLMs, LLMs' responses are often based on the semantic interpretation of the text rather than the rules of physics, making it difficult for them to reason logically through existing knowledge [104,105]. As a result, relying solely on LLMs for solving physical problems is problematic. To address this limitation, it is crucial to conduct simulation experiments on physical problems and incorporate the results into LLM-generated responses.

Mind's Eye [103] leverages a computational physics engine [106] to simulate real-world physical processes. A Text-to-Code language model is employed to generate rendering code for the physics engine, allowing the simulation of physical experiments relevant to the posed question. The simulation outcomes are then incorporated into the LLM prompt in natural language, compensating for the model's lack of physical understanding. Mind's Eye demonstrates a significant improvement in inference accuracy.

(ii) Code interpreter. When confronted with complex reasoning problems, offloading the precise computation task to external modules can significantly improve the model's solution accuracy. Researchers [107,108] prompt Codex [50] to generate executable, code-based solutions. These solutions address a variety of tasks, from university-level exercises to mathematical word problems and financial question-answering (QA). This idea of cooperation between LLMs and code has also been applied to solve more specific problems [109,110]. One possible interpretation of these findings is that code-based approaches benefit from well-defined structural consistency, offering advantages in robustness and logical reasoning compared to natural language. This form of code-driven reasoning allows LLMs to focus more on problem-solving logic rather than the intricacies of textual representation.

The PAL [111] method employs a CoT prompting strategy to decompose complex symbolic reasoning, mathematical problem-solving, and algorithmic tasks into intermediate steps, represented through Python code and natural language annotations. In this setup, computational tasks are delegated to the Python interpreter. Similarly, Chen et al. [112] introduced "Program of Thoughts" (PoT) prompting, which separates computational and reasoning processes through specific prompt designs. Unlike PAL, PoT follows a zero-shot approach for prompt generation. Luo et al. [113] proposed a framework named MultiPoT to select the optimal external programming language based on task types to overcome the Python language's extension limitations. MultiPoT creates custom hints for each programming language to ensure semantic consistency and structural diversity. When addressing issues, it integrates multiple programming languages and selects the final answer generated from each PL by self-consistency. The results obtained incorporate the benefits and diversity of multiple programming languages.

(iii) Integration tools. In addition to code interpreters and simulators, more tools have been collected in the form of APIs. Integration tools are the frameworks that incorporate diverse tools

into prompts. The strong generalization ability of LLM allows it to effectively utilize natural language as an intermediary to manipulate external tools [114]. Additionally, prompt engineering techniques serve as the guides for these tools, providing instructions and guidance on how to effectively utilize them. Based on this insight, Parisi et al. [115] built a text-to-text_api prompt enhancement framework called TALM. The TALM bootstrap LLM generates "|tool-call" and "tool input text" to construct API call requests, and then appends the results returned by external APIs to the text sequence, significantly expanding the model's capabilities. Similar to this, Toolformer [116] uses the same approach of including API requests in the prompts to leave the sub-operations to external tools, while Toolformer also suggests ways to build datasets and fine-tune them. Galactica [117] adopts a special token to initiate a request to call an external tool. This work delivers the concept of "work memory", which generates a special segment in which algorithms and problem-solving code are generated when the model needs to call an external tool. These methods require a strategy to determine what tasks should be offloaded.

To overcome the limitations of manually writing API requests, some works provide strategies to make LLM use tools automatically. MRKL [118] provides a modular neural-symbolic architecture that divides tasks into corresponding task APIs through a router and integrates the returned results. ART [119] organizes its task library, retrieves tasks related to the original prompt in the task library, then decomposes a series of tasks in turn into corresponding sub-tool sequences, generates a specific cue word to be processed, and finally integrates the results of the tool API into the original cue word to realize the automated tool usage of LLM. More large-scale architectures such as Chameleon [120], Gorilla [121], HuggingGPT [122], and ToolAlpaca [123] take advantage of richer API tools and more systematic API requests, greatly enhancing the ability of LLM to use tools to handle tasks. Based on these architectures, several "LLM + Tool" agents such as TPTU [124], TPTUv2 [125], and TaskMatrix. AI [126] have also demonstrated a strong ability to handle complex tasks in different scenarios.

- Feedback. Similar to human learning processes, feedback enables individuals to identify areas for improvement and resolve issues more effectively. The provision of feedback enables the LLM to ascertain the correct and incorrect responses through prompting, thereby enhancing its capacity for reasoning. In accordance with the subject of the judgment, feedback can be classified into two distinct categories: internal feedback and external feedback.

(i) Internal feedback. LLMs possess the inherent capability to engage in self-feedback. Numerous scholars in this field have demonstrated the self-feedback ability of LLMs and have proposed corresponding methods to enhance and leverage this mechanism effectively. By harnessing self-feedback, LLMs can iteratively refine their outputs, contributing to their continual improvement and better performance.

Internal feedback techniques. Some scholars [127] demonstrated that LLMs can undergo iterative self-refinement without requiring additional training, introducing the SELF-REFINE approach. This method refines the output of LLMs through alternating iterations of feedback and refinement to enhance the output quality. The feedback generation process is guided by a few-shot prompt [13]. SELF-REFINE outperforms models such as GPT-3.5 and GPT-4, directly yielding absolute improvements ranging from 5% to 40%. In tasks involving code generation, when applied to CODEX, SELF-REFINE improves initial generation by up to 13%.

SELF (Self-Evolution with Language Feedback) [128] enables LLMs to engage in self-feedback and self-refinement. This is achieved by providing LLMs with feedback in natural language, empowering them to autonomously evolve. The evolution process involves generating responses to unlabeled instructions and iteratively refining these responses through interactions. SELF teaches LLMs fundamental meta-skills using a limited set of examples in natural language, fostering a continuous cycle of self-evolution for LLMs. Self-Contrast [129] enhances the self-feedback capacity of LLMs. This approach prompts LLMs to generate various perspectives to address the same problem, subsequently facilitating comparison and reflection on the disparities among these perspectives.

Some scholars [130] proposed that LLMs can autonomously provide self-feedback regarding hyperparameter perception. Through hyperparameter-aware instruction tuning, LLMs ascertain the optimal decoding strategy and configuration based on input samples, thereby achieving self-regulation. Their proposed HAG (Hyperparameter Aware Generation) consists of two components. Firstly, by inputting a query Q to the LLM, HAG generates appropriate hyperparameters. Subsequently, HAG instructs the LLM to adjust the model's decoding strategies and hyperparameters according to the generated parameters, culminating in the generation of the final result post-adjustment.

Evaluation of internal feedback. For the evaluation of internal feedback, some scholars [131] proposed that glass-box features should be taken into account in self-assessment of LLMs. They proposed that the softmax distribution serves as a dependable indicator for quality evaluation. Self-feedback can enhance performance on certain tasks for LLMs, while potentially worsening performance on others [132]. In light of this, some scholars [133] introduced the concept of Self-Bias to evaluate the bias of LLM output accuracy across different tasks. Additionally, their research reveals that models with larger parameters and access to external feedback possess the ability to more accurately assess and mitigate Self-Bias.

(ii) External feedback. Certain external tools can verify the extrapolation outputs of LLMs. CRITIC framework [134] enables appropriate tools to assess the output of LLMs, which in turn allows LLMs to adjust and refine their output based on this feedback. These external evaluation tools encompass search engines, code interpreters, text APIs, and more. The feedback process primarily involves validating the initial output expectations generated by the LLMs and then modifying the output based on critiques from the verification process. Through this iterative process, external tools offer feedback to LLMs from an external perspective, thereby enhancing the performance of LLMs.

3.4 Reliability

Insight: Reliability, as the final step in the prompt design process, focuses on ensuring more consistent responses from LLMs while reducing biased outputs. It provides a foundation for the practical deployment of LLMs in realworld applications.

In the first three parts, we introduce how to design an effective prompt for LLMs using the Profile-Knowledge-Reasoning process. Due to the uncertainty of the LLM's response [135], the same prompt can lead to diversified responses from the LLM. At the same time, due to the LLM's context learning ability, it is particularly sensitive to the order, type, and historical bias of the prompts [135–137]. The bias of the prompt may worsen the performance of the LLM in downstream tasks. Furthermore, the reasoning strategy used by the LLM is opaque, which means that the responses of the LLM are not always trustworthy [138], and there are a series of risks associated with the responses of the LLM [139,140]. In practical applications, users often demand stable and reliable responses from the LLM. The process of reducing the bias of the LLM's response through prompt integration or by generating auxiliary knowledge from the LLM is known as improving the reliability of the LLM. In this part, we will explain from both perspectives of content bias and value bias.

1) Content bias

We define content bias as the deviation between the completion results of the LLM and the task requirements. For most LLM tasks, designing prompts according to the profile-knowledge-reasoning process can help the LLM perform well in response to task requirements. However, when the task is more complex, it is difficult to obtain stable and perfect output through a single round of prompt input [137,139]. An intuitive idea is to adjust the order of the prompts or change the method of prompting to generate multiple rounds for the same question, which actually uses the concept of ensemble. The prompt method based on this idea is called prompt ensembling.

Prompt ensembling refers to the use of multiple different prompts to complete the same task, enhancing the reliability of the results through multiple responses. Prompt ensembling borrows from the pattern of ensemble learning and includes two sub-processes: first generating multiple prompts as input, and then combining the responses of multiple prompts through a specific strategy to obtain the final result [141,142]. Bagging and boosting are two typical ensemble methods widely used in many classic tasks and have unique applications in LLM.

● Bagging prompt. The application of Bagging Prompt in LLM mainly falls into two categories: the majority vote method based on Self-Consistency [143] and the beam search method based on Step-Verifier [144]. BPE [145] focuses on constructing few-shot CoT prompts based on Self-Consistency, which outperforms a single prompt. However, since Self-Consistency is a method based on a greedy approach, it cannot guarantee that the inference chain is entirely correct. Additionally, its voting mechanism is atomic, meaning it lacks the ability to differentiate the quality among various responses.

In response to these challenges, DiVeRSe [146] was developed to enhance answer reliability through a three-stage process: "generate-verify-check". First, it generates diverse completions using multiple prompts. Next, a "step-aware voting verifier" model distinguishes good answers from poor ones and verifies the correctness of inference steps. DiVeRSe extends traditional methods by extracting step-level labels from intermediate results, ensuring accuracy in the inference flow. While it integrates the benefits of bagging Prompt methods, DiVeRSe still relies on manual sample selection, and sample bias can affect final results.

Motivated by these issues, the AMA (Ask Me Anything) method was introduced [147]. AMA consists of two stages: the multiple prompt step and the answer aggregation step. In the multiple prompt step, AMA employs a functional prompt chain where the question() function transforms the input into open-ended questions, providing diverse perspectives for LLMs to address different aspects of the problem. The answer() function then generates intermediate answers. AMA highlights the limitations of simple voting due to equal weighting and question similarity, which can distort results. To overcome this, AMA uses an answer aggregation method based on weakly supervised learning with an information entropy penalty, ensuring that results better reflect different perspectives. Essentially, AMA optimizes the Bagging random sampling approach.

● Boosting prompt. Boosting prompt methods often adopt a two-stage paradigm. PromptBoosting [148] applies the AdaBoost algorithm on the prompt set, achieving good results in text classification. However, the PromptBoosting method requires a prepared high-quality prompt set and cannot optimize for specific prompts.

To overcome these limitations, Prefer [149] establishes a feedback mechanism to reflect on the shortcomings of the weak learners in the current iteration. Based on feedback, Prefer also implements the automatic synthesis and selection of prompts, avoiding the bias problem brought by the prompts. PromptBoosting's ensemble method for weak learners refers to the traditional ensemble method of weighted summation. However, many works have pointed out that LLMs have a serious optimistic estimation problem [150,151]. This makes the weighted calculation of answer accuracy unable to eliminate content bias.

Bilateral Prompt Bagging [149] assesses the confidence of the generated results in each iteration. When the assessment result of the answer is not trustworthy, a reverse confidence evaluation is performed, calculating the confidence that this answer is incorrect. The final correct probability of a round of generated results is evaluated by combining forward and reverse confidence. This design of positive and negative confidence effectively avoids the optimistic estimation problem of LLMs [150]. By leveraging the feedback reasoning capabilities of LLMs (as discussed in Subsection 2.3), it enhances the quality of single-round generated results, strengthens weak learners, and ensures a more reliable and efficient final outcome.

2) Value bias

We define value bias as the deviation between the content generated

by LLMs and human societal values. The content generated by LLMs often touches upon socially sensitive areas, such as issues of social harm and discrimination [152,153]. Shaikh et al. [139] showed that Chain of Thought (CoT) can continually enhance the performance of LLMs across a variety of NLP tasks. However, this also increases the likelihood of the model producing harmful or inappropriate results. Research [154] indicates that CoT prompts should be used carefully when dealing with socially relevant issues. Due to the principles of autoregressive decoding inherent in LLMs, harmful text might still be generated during the beam-search phase of the inference stage if the volume of harmful text in the training corpus is large enough. This underlines the importance of developing mechanisms to manage and minimize harmful or socially unacceptable outputs for the safe deployment of AI systems.

Tang et al. [155] proposed the Detox-Chain method, which links detoxification sub-steps together to achieve rapid detoxification of prompts. The Detox-Chain method reduces the possibility of LLM generating toxic texts by substituting toxic textual representations. However, due to the detoxification technology for prompts altering the distribution or content of the prompt to varying extents, it affects the quality of the output produced by the LLM. At the current stage, the detoxification methods for LLMs are still primarily based on reinforcement learning or knowledge editing techniques during the training phase.

4 Applications

This section offers practitioners a comprehensive overview of current application domains for LLMs and highlights the prompt engineering techniques utilized across diverse use cases. While existing studies [156] have introduced taxonomies for LLM applications, these classifications tend to focus on specific scenarios, limiting their relevance for prompt engineering design. To address this gap, we propose a novel taxonomy that categorizes applications by task characteristics into two primary types: cognitive applications and transformative applications of LLMs. Cognitive applications highlight LLMs' roles in providing and processing information, whereas transformative applications involve tasks where LLMs operate autonomously, potentially substituting human involvement. This taxonomy provides detailed insights into prompt engineering techniques tailored for each application category, as illustrated in Fig. 1.

4.1 Cognitive applications of LLMs

Cognitive applications of LLMs involve utilizing these models to acquire and analyze information. The process of leveraging LLMs to extract and present knowledge is termed Information Acquisition. This category encompasses applications such as chatbots, professional knowledge Q&A, search engines, and training data augmentation. Conversely, when LLMs are employed to interpret and analyze input data, the process is referred to as In-depth Information Analysis. Key applications in this analytical domain include the simulation of financial market dynamics and the exploration of chemical molecular structures, showcasing the model's ability to provide insightful analyses and predictive insights across specialized fields.

1) Information acquisition

Information acquisition is a fundamental application of LLMs, leveraging their extensive knowledge base to efficiently deliver relevant information to users.

> **Finding:** Information acquisition is supported by various prompt engineering techniques, which deliver the required information to users. *Profile* techniques equip the LLM with detailed, task-specific prompt, enabling a more precise context. *Knowledge* techniques provide the model with up-to-date, task-relevant information, helping prevent overconfident responses. When addressing specific queries, reasoning techniques enhance the model's problem-solving capacity, significantly improving its ability to generate accurate and reliable answers.

● Chatbots. General chatbots, also known as dialogue agents, integrate tasks such as information retrieval, multi-turn interaction, and text generation (including code). LLMs encapsulate extensive knowledge within their parameters during training. Users can obtain information from LLMs through role-playing dialogue, exemplified by the chatbot [157].

For instance, Glaese et al. [158] introduced Sparrow, a dialogue agent based on a 70B parameter Chinchilla LLM. It utilizes prompt engineering techniques, including Basic RAG Techniques (Subsection 3.2) and Personality Information (Subsection 3.1), to address hallucination issues by integrating external knowledge from Google search queries.

Similarly, OpenAI [159] utilizes supervised fine-tuning with high-quality data, along with reinforcement learning from human feedback (RLHF), to develop the GPT-3.5 LLM, which powers the ChatGPT chatbot. The subsequent GPT-4 model [160] underpins the ChatGPT Plus chatbot. ChatGPT employs prompt techniques such as Personality Information (Subsection 3.1) for role-playing conversations and Memory (Subsection 3.2) to maintain conversation history, thereby excelling in extended discussions. Microsoft Copilot [161] is an AI-powered productivity tool that integrates an LLM with Microsoft 365. Supported by an OpenAI model, Copilot utilizes additional integration tools (Subsection 3.3), accesses up-to-date information via Bing, and incorporates Retrieval-Augmented Generation (RAG) technology (Subsection 3.2) to enhance user prompts.

Anthropic [162] introduces the Claude series of chatbots. These bots are further refined via fine-tuning with high-quality data and guided by RLHF to generate responses that are beneficial, harmless, and honest. The Claude 3.5 Sonnet model has outperformed competitor models in various AI system evaluation benchmarks, including undergraduate-level expert knowledge (MMLU), graduate-level expert reasoning (GPQA), and basic mathematics (GSM8K).

● Search engines. Information Retrieval (IR) systems are integral to dialogues, question answering, and recommendation systems, serving as primary means of obtaining information. LLMs can improve traditional IR components, such as query rewriters, and can also function as the engine for generative retrieval.

(i) LLM-based query rewriter. The query rewriter enhances user queries by adding synonyms or related terms to address vocabulary mismatches and clarify ambiguities, thus aligning more accurately with user intent. In conversational retrieval, the query rewriter comprehends the context of the entire conversation, clarifying ambiguous content and generating a more effective new query based

on the user's dialogue history.

HyDE [163] represents pioneering work on LLM-based query rewriting, guiding LLM generation using various prompt engineering techniques in Profile and Perception (Subsection 3.1). HyDE generates detailed hypothetical documents based on the given query, which are then retrieved from the corpus using a dense retriever. Query2doc [24,164] exemplifies another innovative approach to query rewriting, generating pseudo-documents by prompting LLMs with a few demonstrations, reflecting the use of the Demonstration Information approach (Subsection 3.1). These pseudo-documents are subsequently expanded with generated documents. Based on prompt engineering optimization, Jagerman et al. [165] studied the impact of different prompting methods and model sizes on query rewriting [165].

GFF [166] adopts a "generate, filter, and fuse" method for query expansion, utilizing LLMs to extract related keywords from the original query through a reasoning chain. GFF employs prompt ensembling strategies (Subsection 3.4) to filter generated keywords using techniques such as Self-Consistency [143], ensuring the quality and relevance of keywords, which are then integrated with the original query for downstream reranking tasks.

(ii) LLM-based generative retrieval. Generative retrieval methods utilize a unified model to directly generate document identifiers (DocIDs) related to user queries, integrating prompt engineering techniques. Traditional IR systems typically follow the "index-retrieve-rerank" paradigm, which has proven effective in practice [163,167]. However, the constituent modules —indexing, retrieval, and reranking —operate independently. LLM-based generative retrieval unifies these modules, initiating a new paradigm for IR systems.

Researchers have demonstrated that LLMs, such as the GPT series, can directly generate URLs related to user queries [168]. This capability allows the LLM to function as a generative retriever, producing document identifiers from the original input to obtain relevant documents. Ziems et al. [168] introduced the LLM-URL model, which employs the GPT-3 text-davinci-003 model to generate candidate URLs. It utilizes the Demonstration Information approach (Section 3.1) to standardize model output and incorporates a URL filtering mechanism that extracts valid URLs from candidates using regular expressions.

- Professional knowledge Q&A. The expansive capabilities of LLMs enable them to possess specialized knowledge across various fields, effectively organizing and presenting information to address specific inquiries. This ability to provide accurate responses to user queries is commonly referred to as professional question and answer (QA).

LawGPT [169] is a robust language model designed to address legal knowledge inquiries with high accuracy. It employs the LLM internal feedback method (Subsection 3.3) to continually enhance its precision in legal question answering. Through this approach, LawGPT generates legal queries pertinent to specific legal texts and subsequently provides responses in the form of "text segment-question" pairs, ensuring that its answers are rich in legal information.

MultiMedQA [170] is a benchmark that integrates clinical expertise with medical knowledge, employing a Few-shot prompting technique (Subsection 3.1). Collaborating closely with a panel of seasoned clinicians, MultiMedQA produces exemplary few-shot demonstrations and sample scenarios. Moreover, it adeptly demonstrates Reasoning and Planning capabilities, exhibiting a robust chain-of-thought process (Subsection 3.3) for a myriad of medical issues by leveraging insights from various medical professionals. Furthermore, MultiMedQA excels in internal feedback (Subsection 3.3) verification, a crucial aspect in refining its accuracy. Given the multifaceted nature of medical queries, it employs the self-consistency strategy (Subsection 3.4), allowing the model to compare and evaluate diverse perspectives to ensure coherence and reliability in its responses.

- Training data augmentation. Due to the high cost of manually annotating labels, a common challenge in training neural retrieval models is the lack of training data. LLMs can learn patterns from manually annotated data and generate additional data consistent with the existing dataset.

Yoo et al. [171] proposed GPT3Mix, which generates synthetic data from existing datasets based on the GPT3 LLM. GPT3Mix employs the Demonstration Information method (Subsection 3.1) and the Task Information method (Subsection 3.1). The prompts of GPT3Mix include two parts: real examples from the dataset and task specifications, which are used to create synthetic data and pseudo labels.

Yoo et al. [171] utilized this new augmented dataset to fine-tune BERT and DistilBERT models, achieving excellent performance in classification tasks. In the context of information retrieval, it is easy to collect many documents; however, the challenging and expensive task is to collect real user queries and label the relevant documents accordingly. Given the LLM's powerful natural language processing capabilities, many researchers [172,173] suggested using an LLM-driven process to create pseudo queries or relevance labels based on existing datasets.

In cutting-edge work, Dai et al. [174] introduced ChatAug, which transforms training samples from a small dataset into multiple samples that are conceptually similar but semantically distinct. ChatAug outperforms state-of-the-art text data augmentation methods in terms of test accuracy and augmented sample distribution.

2) In-depth information analysis

In addition to providing information, LLMs can leverage their reasoning capabilities to analyze patterns within specific data or information. Applications requiring the evaluation of complex systems are categorized under In-depth Information Analysis.

Finding: Unlike information acquisition, in-depth information analysis presents LLMs with more complex challenges, such as financial market analysis. Profile techniques provide task information and demonstrations, enabling the LLM to better align with specialized requirements. By retrieving more specific task data and memorizing essential information, knowledge techniques further enhance the model's domain expertise. Employing professional tools to assist with task handling can effectively support the model in generating more valuable analysis.

● Financial markets. By providing Internet-scale data for LLMs, these models can utilize hybrid training methods in financial tasks, driving open-source development in the financial field. BloombergGPT [175] is a formidable large language model with 5 billion parameters, meticulously trained to excel in the intricate realm of finance through vast datasets.

BloombergGPT combines prompt engineering technologies such as personality information (Subsection 3.1) and external financial tools (Subsection 3.3) to provide personalized, time-sensitive financial advice. Its versatility extends across various tasks within the financial sector, making it beneficial for professionals. Leveraging techniques such as few-shot learning and other sophisticated methodologies, scholars fine-tune BloombergGPT to suit specific task formats, thereby enhancing its efficacy in delivering best-in-class results.

FinGPT [176] stands out as an open-source, large language model tailored specifically for the financial sector, boasting versatile applications such as robot consultation, algorithmic trading, and low-code accessibility. Its remarkable capability lies in making quantitative inferences from vast financial datasets, effectively capturing the intricate dynamics of the financial markets. A key feature of FinGPT is its adept utilization of external feedback methods (Subsection 3.3) within prompt engineering. By leveraging stock prices as indicators, FinGPT engages in reinforcement learning, continuously refining its understanding and interpretation of financial texts. This process empowers FinGPT to anticipate and predict market responses to a wide array of financial events, thereby enhancing its overall performance and utility in financial analysis and decision-making.

● Chemical molecules. The study of chemical molecules encompasses the properties, composition, structure, and chemical reactions of matter. This field involves complex computational and predictive tasks, such as property prediction, chemical structure optimization, and reaction prediction. As data scales increase, traditional chemical computational methods can no longer meet the demand, making the application of LLMs in chemistry increasingly important.

BioinspiredLLM [177] is an autoregressive transformation large language model specifically tailored for biomaterials and biomimetic materials. This sophisticated model excels in accurately recalling vast amounts of information pertaining to biomaterials, effectively reflecting intricate patterns between various biomaterials, and facilitating research tasks within this field. Scientific researchers continuously uncover new patterns, and BioinspiredLLM accommodates this by offering additional contextual content when queried. Moreover, the model leverages the Retrieve and Generate (RAG) framework, based on Knowledge prompt engineering techniques (Subsection 3.2), to provide comprehensive and insightful answers.

ChatDrug [178] harnesses conversational and reasoning capabilities to facilitate AI-assisted drug discovery endeavors. ChatDrug employs Chain-of-Thought (CoT) technology (Subsection 3.3) to decompose complex concepts into more understandable attributes, thereby improving reasoning ability and problem-solving efficiency. Additionally, ChatDrug incorporates prompt design tailored for domain-specific modules, with the flexibility to implement prompt functions via few-shot learning methodologies (Subsection 3.1). Leveraging the Retrieve and Generate (RAG) method (Subsection 3.2) enriches ChatDrug's professional knowledge, thereby enhancing the quality of responses.

ChemCrow [179] enables reasoning across common chemistry tasks such as material design and synthesis, from reasoning about simple drug discovery cycles to planning the synthesis of substances across a wide range of molecular complexity. ChemCrow combines the inferential power of LLMs with the chemistry expertise of computational tools (Subsection 3.2), successfully planning and synthesizing an insect repellent, three organocatalysts, and guiding the screening and synthesis of a novel chromophore with target properties. Enhanced by the RAG method (Subsection 3.2) and search engine tools (Subsection 3.3), ChemCrow can obtain scientific information from the Internet and build relevant databases. It can also accept external feedback from humans and adjust itself accordingly.

4.2 Transformative applications of LLMs

Transformative applications of LLMs facilitate automation and enhance workflows traditionally dependent on human input. These applications can be broadly categorized into physical world applications and creative applications. Through prompt engineering, LLMs can impact the physical world across various application scenarios, including software engineering, robotics, and expert-level task automation. Conversely, the impact of LLMs on creative domains extends to areas traditionally reserved for human creativity and artistic expression, including language generation, visual content creation, and auditory processing. These enhancements contribute to expanding human creativity and comprehension.

1) Physical world applications

Applications of LLMs in the physical world encompass areas such as software engineering, robotics, and advanced automation.

> **Finding:** The powerful capabilities of LLMs demonstrate their significant potential to replace repetitive manual human tasks. *Profile* techniques provide task-specific requirements and real-world environment information. In specialized domains, such as AI-driven healthcare, *knowledge* techniques supplement the model with domainspecific expertise, while enhancing reliability by providing professional knowledge. *Reasoning* techniques can effectively enhance the capabilities of LLMs in addressing a wide range of complex real-world problems. *Reliability* techniques ensure that the model's actions remain stable and dependable.

● Software engineering. One of the most advanced and widely used applications of LLMs is generating and completing computer programs in various programming languages. This section discusses programming-specific LLMs, which are fine-tuned or pre-trained specifically for programming applications. However, it is important to note that general chatbots, which are partially trained on code datasets (such as ChatGPT), are increasingly utilized in programming tasks.

Code generation refers to the use of LLMs to output new code based on the requirements provided in the prompts. Several LLMs and methods have been proposed for computer programming. OpenAI first released CodeX [50], an LLM based on GPT-3 (up to

12B parameters) pre-trained on public datasets to generate independent Python functions from natural language strings. CodeX standardizes output using prompt techniques such as personality information (Subsection 3.1) and few-shot prompting (Subsection 3.1).

The Copilot [161] plugin based on CodeX has also become a benchmark for code generation assistance tools. The HumanEval [50] dataset has become one of the commonly used benchmark datasets for subsequent code generation. Nijkamp et al. [51] sequentially trained CodeGen series LLMs (up to 16B parameters) on three datasets: the natural language dataset (THEPILE), the multilingual programming source code dataset (BIGQUERY), and the single-language Python dataset (BIGPYTHON). CodeGen models have been trained in various programming languages, including C, C++, Go, Java, JavaScript, and Python. The results indicate that the largest CodeGen model outperforms the Codex-12B model. CodeGen can utilize CoT technology (Subsection 3.3) to enhance the reasoning ability of generated code.

Nijkamp et al. [51] also tested CodeGen for multi-step program synthesis, decomposing the program into multi-step natural language prompts, with the synthesis system completing the synthesis of subroutines at each step. They also created a multi-round programming benchmark (MTPB) for the multi-round program synthesis method. Due to its excellent performance across multiple programming languages, Zheng et al. [180] trained CodeGeex on three datasets: The Pile, CodeParrot, and public repository supplement data on GitHub. CodeGeex employs a standard Transformer architecture, supporting high-precision code generation while also enabling automatic translation and conversion of code snippets between different programming languages.

● Robotics. Multimodal large language models are increasingly applied in robotics, where robots interact with the physical world through various methods. These models leverage their reasoning and planning capabilities to guide robot actions.

PaLM-E [52], a general visual language model, incorporates embedded data into multimodal training, serving as an effective inference engine. Experimental results demonstrate that, in addition to general visual language tasks, PaLM-E excels in tasks such as entity capture, performing admirably in real desktop and mobile manipulation scenarios. Fine-tuning of PaLM-E involves scaling the language model size, enhancing its adaptability across different tasks and environments. This multimodal approach ingeniously employs prompt engineering across diverse tasks and scenarios. Task information (Subsection 3.1) enables PaLM-E to discern the attributes of various tasks, while task environment information provides tailored scenarios for specific tasks. Additionally, PaLM-E utilizes Target decomposition reasoning (Subsection 3.3) to break down input tasks in advance, enabling the multimodal model to provide more realistic and effective responses.

● Expert-level task automation. In contrast to applications aimed at information retrieval, expert-level task automation involves applications where LLMs serve as substitutes for human experts in specialized fields. Examples include LLM-based agents that perform the functions of professionals, such as legal advisors or medical practitioners.

WisdomInterrogatory [55] is adept at responding to new legal issues using existing legal documents and provisions within the field of law. Through task information (Subsection 3.1), it can utilize corresponding legal knowledge to answer consultations. WisdomInterrogatory systematically clarifies attribute information, information sources, task types, and answer plans for various legal scenarios. It employs RAG technology (Subsection 3.2) for retrieving relevant legal information and enhances relevance through pre-retrieval optimization (Subsection 3.2). By prompting the LLM with few-shots (Subsection 3.1), WisdomInterrogatory can generate tailored responses for specific legal tasks, ensuring effective adaptation to a wide range of legal scenarios while providing accurate and insightful answers.

ChatLaw [54] is a comprehensive legal language model crafted from a deep understanding of the legal domain. It serves as a robust tool for navigating complex legal issues, offering nuanced insights and practical guidance derived from its comprehensive knowledge base and advanced computational techniques. ChatLaw addresses real-world legal challenges by synthesizing legal awareness, relationships, behaviors, and other phenomena within the legal realm. Leveraging prompt engineering within the legal domain, ChatLaw significantly enhances the performance of the Chinese legal language model. During the construction of its training dataset, ChatLaw meticulously fine-tunes and supplements established datasets to ensure relevance and accuracy. It employs RAG technology (Subsection 3.2) to retrieve the latest legal information and utilizes post-retrieval optimization (Subsection 3.2) to further refine its responses and improve overall effectiveness.

ChatDoctor [53] is a pre-trained language model designed to address medical queries and offer advice in medical contexts. One of ChatDoctor's notable features is its external knowledge brain, akin to a Retrieval Augmented Generator (Subsection 3.2). This knowledge repository encompasses a vast array of information on diseases, symptoms, relevant medical tests, and more. Continuously updated and refined, the knowledge brain retrieves new insights from sources such as encyclopedias, medical literature, and other credible references. With its robust dataset and access to a rich knowledge base, ChatDoctor serves as a valuable resource for individuals seeking medical advice. Its ability to understand patient needs and provide tailored recommendations underscores its utility in guiding users toward appropriate medical treatment options.

2) Creative applications

LLMs can be applied in the field of artistic creation, enabling more efficient and intelligent art production.

> **Finding:** With the support of prompt engineering techniques, LLMs are able to generate artistic creations with exceptional efficiency. In both literary and visual arts, *profile* techniques are employed to define specific requirements and details for the creative process. *Knowledge* techniques facilitate the retrieval of multimodal knowledge,enhancing the model's creative capacity. While *reasoning* and *reliability* techniques are less frequently appliedin artistic creation, this is largely due to the inherently creative and less deterministic nature of artistic work.

● Language. CoPoet [56] is a collaborative iterative poetry creation model. Users can iteratively request suggestions from CoPoet through natural language instructions, and CoPoet can generate further content based on user input. Utilizing internal feedback (Subsection 3.3), CoPoet refines the creation process or aids users in their creations by assimilating user feedback.

Ippolito et al. [57] identified barriers to creative writing between AI-driven writing agents and experienced professional writers. They assessed the creative capabilities of the AI writing agent by engaging experts in professional creation from diverse countries, races, and backgrounds. These experts provided open-ended qualitative feedback on the content generated by the AI writing agent. The authors analyze the evaluation results and propose lessons that could enhance the creative abilities of LLMs. Brainstorming emerges as a crucial aspect of creation, and few-shot learning (Subsection 3.1) can be employed to provide prompts for creativity. Task environment information (Subsection 3.1) can help LLMs enhance imagination. By providing environmental information to the language model, detailed data can facilitate more effective brainstorming. Additionally, the authors suggest that this could also be achieved through methods like RAG (Subsection 3.2) and Automatic Post-Editing (APE) [181].

● Vision. As a multi-modal large language model, Sora possesses the capability to process and generate images and videos. Sora excels in a multitude of image and video editing tasks, including seamlessly creating looped videos, animating static images, and extending video duration forwards or backwards in time. Beyond merely inputting text to produce videos, Sora can also utilize pre-existing images or videos as few-shot inputs. This expanded functionality allows Sora to undertake a broader spectrum of editing tasks, such as extending videos and converting images into videos.

● Hearing. ChatMusician [58] adeptly integrates various musical elements, crafting well-structured compositions. By unifying symbolic music understanding and generation tasks, it generates coherent pieces across styles. Utilizing prompt engineering, ChatMusician explores musical creation dynamically. It formats tasks with few-shot prompts (Subsection 3.1) to inspire the creation of LLMs. It also adopts musician role-play prompts (Subsection 3.1) to enhance its perspective.

5 Possible research opportunities

In this section, we outline several promising future research directions that address key challenges in prompt engineering. Although some directions are already covered in existing studies, as discussed in Section 3, we believe these areas warrant further exploration and development.

1) Prompt attack and defense

Due to the unique structure of LLMs, prompts serve as both operational instructions and input data channels. This dual role introduces risks, as specific prompts can be crafted to induce harmful outputs from LLMs, a technique known as prompt attack [154]. Although recent research [182] has proposed defense mechanisms against prompt attacks, few studies focus on prompt engineering as a defense strategy. Adjusting the content or structure of prompts may help prevent generating risky responses. Profile and instruction, in Subsection 3.1, may be useful in such cases, as it provides a framework for organizing prompt content and structure.

2) Thinking powered RAG

Recent studies [183] indicate that applying Retrieval-Augmented Generation (RAG) in all contexts may not consistently enhance LLM capabilities. Integrating RAG with reasoning techniques, such as feedback mechanisms [184], could improve the cognitive efficacy of RAG, facilitating its application in a more contextually appropriate manner. Current methods [184,185] predominantly rely on pre-training and lack interpretability. Developing quantitative approaches to guide retrieval processes and enhance interpretability represents a compelling research direction.

3) Multi-Hop retrieval-augmented generation

The Multi-Hop Question Answering (MHQA) task [23,186] focuses on answering questions that require gathering information from multiple sources and performing multi-step reasoning to arrive at comprehensive answers. Enhancing both the relevance of retrieved documents and the accuracy of reasoning across multiple documents can significantly improve the performance of LLMs on these complex questions. As a straightforward solution, recent studies [31,32] have integrated retrieval into the chain-of-thought reasoning process, enabling LLMs to leverage retrieved documents for answer generation. However, inevitable noise in retrieved documents may mislead LLMs toward incorrect reasoning paths, resulting in erroneous answers [187]. Rather than solely enhancing reasoning capabilities, optimizing retrieved materials through prompt engineering presents a promising approach to improve prompt effectiveness.

4) Advanced reasoning with rationales

Techniques like Chain of Thought (CoT) [30] and Least-to-Most [97] have shown potential for improving LLM reasoning. However, these methods primarily structure reasoning pathways without genuinely enabling LLMs to internalize the underlying rationale. Techniques like CoT establish fixed reasoning pathways for LLMs, akin to rapid brainstorming. Research [188] indicates that slower, more deliberate thinking is often more effective for solving complex problems. While some research [188,189] has explored ways to strengthen the rationale capabilities of LLMs, numerous avenues for further exploration remain. This enhancement in the reasoning process closely aligns with findings in o1 [190], underscoring the substantial potential of these techniques.

5) Domain-specific prompt engineering frameworks

With the expanding application of LLMs across diverse fields, the development of domain-specific prompt engineering frameworks has become a prominent research focus. Optimizing prompts tailored to sectors such as healthcare, law, and finance enables LLMs to produce more accurate, dependable, and efficient outputs [53,169]. As domain-specific agents advance, there is a corresponding rise in specialized agent components and models trained for distinct fields.

Prompt engineering remains central to facilitating interactions between LLMs and various tools [47]. Substantial work is still required to design prompt engineering frameworks that align with the professional standards essential for domain-specific agents.

6) Automatic optimization of prompt

Prompt engineering is a crucial technique directly influencing the responses generated by LLMs. This approach fundamentally differs from programming techniques, as systems based on prompts are guided rather than explicitly programmed. The specific LLM and the details within the prompts both influence the model's output, a topic that will be further discussed in Section 6. Although some research has explored the interpretability of prompts, it typically identifies only the most significant factors that influence the consistency of generated responses. Thus, optimizing prompts remains a challenge. This includes providing explanations for prompts that lead to erroneous answers and automating the optimization of prompts. Exploring automatic optimization within the prompt space and integrating automation with existing prompt techniques represent a practical future research avenue.

6 Summary of findings and insights

● Principles of prompt engineering. Recent studies [33,191] position prompt engineering as an empirical approach. Due to the autoregressive structure of LLMs, prompts function not only as the primary interaction method but also as critical determinants of model outputs. Unlike traditional software engineering, where systems are explicitly programmed, prompt engineering for LLMs relies on guiding model behavior through input modifications. The model's performance is highly sensitive to specific prompt details without any obvious reason those details should matter. Most advanced techniques that leverage this insight, such as the reasoning methods described in Subsection 3.3, are grounded in experimental practices rather than theoretical constructs. Given the impressive capabilities of LLMs, empirically driven approaches to prompt engineering remain essential.

● Prompt design factors. In this paper, we introduce a comprehensive taxonomy for prompt engineering, which synthesizes effective empirical insights into distinct modules for prompt design. The profile and instruction establish the foundational structure of prompts, emphasizing that role information, task information, and example information are critical components in the prompt design. Prompt enhancement techniques are essential for the application of LLMs in real-world scenarios. Currently, knowledge and reasoning are effective strategies for enhancing prompts, as they improve LLM performance by incorporating specific details, such as additional task information and reasoning examples. Although future techniques for augmenting prompt details may emerge, significant research is still required in the domains of knowledge and reasoning. Prior study [192] indicates that the introduction of noise knowledge can enhance LLM performance, highlighting the necessity for more accurate and interpretable knowledge techniques. For instance, employing knowledge graphs can facilitate the acquisition of more precise external knowledge [193]. From an engineering perspective, reliability enhances LLM systems by integrating diverse prompt techniques. It is grounded in the principles of ensemble learning, where techniques such as voting are employed to retain the most stable model responses. Future advancements in prompt engineering can consider these factors comprehensively to design prompts that not only align closely with task requirements but also maximize model performance.

● Post-training impact on prompt engineering. Recently, OpenAI introduced the o1 family of models [190], which have demonstrated breakthrough performance in software engineering and scientific tasks. The o1 model integrates inference processes—such as Chain-of-Thought (CoT) reasoning —into its training framework. By allowing for extended reasoning time, the o1 model enhances its overall problem-solving capabilities. With advancements in such LLMs, reasoning, planning, and reliability techniques are likely to be integrated into future frameworks. At present, profile, instruction, and knowledge technologies remain indispensable due to their relevance to practical tasks.

● Limitations and challenges in prompt engineering. Despite significant advancements in prompt engineering for enhancing large language model (LLM) performance, several critical challenges persist. These challenges pertain to highly ambiguous tasks, adversarial tasks, and scalability in dynamic environments. First, LLMs frequently encounter ambiguous prompts that may lead to multiple plausible answers. Traditional methods often struggle to balance response relevance and diversity. To address this issue, Sun et al. [194] proposed AmbigPrompt —an iterative prompting framework that adaptively guides the LLM to generate distinct and pertinent responses, thereby mitigating ambiguity in user queries. Second, LLMs remain susceptible to adversarial attacks, such as jailbreaking prompts, which can trigger the generation of inappropriate or harmful content. In response, Paulus et al. [195] introduced AdvPrompter, a technique that employs a secondary LLM to rapidly generate human-readable adversarial prompts. This work highlights the urgent need for robust defense mechanisms against such vulnerabilities. Third, scaling prompt engineering techniques to more complex or dynamic environments presents substantial challenges. Kepel and Valogianni [196] addressed this issue with the development of the Automatic Prompt Engineering Toolbox (APET), which enables GPT-4 to autonomously apply prompt engineering techniques. This automation is pivotal for managing scalability concerns effectively. While these advancements offer promising solutions to some of these challenges, further research is necessary to fully overcome these limitations.

7 Conclusion

Prompt engineering is gaining increasing significance across various application domains as a crucial technique for optimizing the performance of large language models (LLMs). Drawing inspiration from the division of agent functions, this survey systematically categorizes prompt engineering techniques into four distinct aspects. This structured approach offers a clear and comprehensive review of the existing research in this area. With a new taxonomy, we examine the applications of LLMs that utilize prompt engineering, presenting

specific case studies and empirical results that illustrate the effectiveness of these techniques across different tasks. We aspire for this survey to serve as a comprehensive guide for readers, illuminating the advancements in prompt engineering and offering valuable insights into its practical applications.

Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2023YFB3002002), the National Natural Science Foundation of China (Grant Nos. 62461146205 and 62322213), Beijing Nova Program (Nos. 20230484397 and 20220484137), the State Key Laboratory of Computer Architecture (ICT, CAS) (CLQ202414), and CCF - ApsaraDB Research Fund (CCF-Aliyun2024003).

Competing interests

The authors declare that they have no competing interests or financial conflicts to disclose.

References

[1] Zhang R, Su Y, Trisedya B D, Zhao X, Yang M, Cheng H, Qi J. AutoAlign: fully automatic and effective knowledge graph alignment enabled by large language models. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(6): 2357−2371

[2] He J, Li Y, Zhai Z, Fang B, Thorne C, Druckenbrodt C, Akhondi S, Verspoor K. Focused contrastive loss for classification with pre-trained language models. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(7): 3047−3061

[3] Zhao Z, Fan W, Li J, Liu Y, Mei X, Wang Y, Wen Z, Wang F, Zhao X, Tang J, Li Q. Recommender systems in the era of large language models (LLMs). IEEE Transactions on Knowledge and Data Engineering, 2024, 36(11): 6889−6907

[4] Li J, Liu Y, Fan W, Wei X-Y, Liu H, Tang J, Li Q. Empowering molecule discovery for molecule-caption translation with large language models: a ChatGPT perspective. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(11): 6071−6083

[5] Cao Z, Cao C, Xu J, Xu J, Chen Z, Chen Z, Ma X. SCG-tree: shortcut enhanced graph hierarchy tree for efficient spatial queries on massive road networks. Frontiers of Computer Science, 2025, 19(9): 199610

[6] Liu P, Cai P, Zhong K, Li C, Chen H. LRP: learned robust data partitioning for efficient processing of large dynamic queries. Frontiers of Computer Science, 2025, 19(9): 199607

[7] Chen C, Ma W, Gao C, Zhang W, Zeng K, Ye T, Chen Y, Du X. GaussDB-AISQL: a composable cloud-native SQL system with AI capabilities. Frontiers of Computer Science, 2025, 19(9): 199608

[8] Yang J-Q, Dai C, Ou D, Li D, Huang J, Zhan D-C, Zeng X, Yang Y. COURIER: contrastive user intention reconstruction for large-scale visual recommendation. Frontiers of Computer Science, 2025, 19(7): 197602

[9] Zhang T, Zhao J, Yu C, Chen L, Gao Y, Cao B, Fan J, Yu G. Labeling-based centrality approaches for identifying critical edges on temporal graphs. Frontiers of Computer Science, 2025, 19(2): 192601

[10] Wang D, Cai P, Qian W, Zhou A. Efficient and stable quorum-based log replication and replay for modern cluster-databases. Frontiers of Computer Science, 2022, 16(5): 165612

[11] Li X-H, Cao C C, Shi Y, Bai W, Gao H, Qiu L, Wang C, Gao Y, Zhang S, Xue X, Chen L. A survey of data-driven and knowledge-aware explainable AI. IEEE Transactions on Knowledge and Data Engineering, 2022, 34(1): 29−49

[12] Chen L, Zhou X, Yang X, Sellis T. Guest editorial special issue on online recommendation using AI and big data techniques. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(10): 9809−9811

[13] Brown T B, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler D M, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D. Language models are few-shot learners. In: Proceedings of the 34th International Conference on Neural Information Processing System. 2020

[14] Zhu Y, Wang Y, Qiang J, Wu X. Prompt-learning for short text classification. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(10): 5328−5339

[15] Xue H, Salim F D. PromptCast: a new prompt-based learning paradigm for time series forecasting. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(11): 6851−6864

[16] Liu J, Fei H, Li F, Li J, Li B, Zhao L, Teng C, Ji D. *TKDP*: threefold knowledge-enriched deep prompt tuning for few-shot named entity recognition. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(11): 6397−6409

[17] Abercrombie G, Curry A C, Dinkar T, Rieser V, Talat Z. Mirages. On anthropomorphism in dialogue systems. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023

[18] Van Buren D. Guided scenarios with simulated expert personae: a remarkable strategy to perform cognitive work. 2023, arXiv preprint arXiv: 2306.03104

[19] Zheng M, Pei J, Logeswaran L, Lee M, Jurgens D. When "a helpful assistant" is not really helpful: Personas in system prompts do not improve performances of large language models. In: Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024. 2024, 15126–15154

[20] Park J S, O'Brien J, Cai C J, Morris M R, Liang P, Bernstein M S.

Generative agents: interactive simulacra of human behavior. In: Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology. 2023

[21] Efrat A, Levy O. The turking test: can language models understand instructions? 2020, arXiv preprint arXiv: 2020

[22] Chen J, Lin H, Han X, Sun L. Benchmarking large language models in retrieval-augmented generation. In: Proceedings of the 38th AAAI Conference on Artificial Intelligence. 2024

[23] Tang Y, Yang Y. MultiHop-RAG: benchmarking retrieval-augmented generation for multi-hop queries. 2024, arXiv preprint arXiv: 2401.15391

[24] Wang L, Yang N, Wei F. Query2doc: query expansion with large language models. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023

[25] Cheng X, Luo D, Chen X, Liu L, Zhao D, Yan R. Lift yourself up: retrieval-augmented text generation with self-memory. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 1899

[26] Jiang Z, Xu F F, Gao L, Sun Z, Liu Q, Dwivedi-Yu J, Yang Y, Callan J, Neubig G. Active retrieval augmented generation. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023. 2023, 7969–7992

[27] Qiao S, Ou Y, Zhang N, Chen X, Yao Y, Deng S, Tan C, Huang F, Chen H. Reasoning with language model prompting: a survey. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023

[28] Huang J, Chang K C C. Towards reasoning in large language models: A survey. In: Findings of the Association for Computational Linguistics: ACL 2023. July 2023, 1049–1065

[29] Yu F, Zhang H, Tiwari P, Wang B. Natural language reasoning, a survey. ACM Computing Surveys, 2024, 56(12): 304

[30] Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E D, Le Q V, Zhou D. Chain-of-thought prompting elicits reasoning in large language models. In: Proceedings of the 36th International Conference on Neural Information Processing System. 2022, 1800

[31] Long J. Large language model guided tree-of-thought. 2023, arXiv preprint arXiv: 2305.08291

[32] Yao Y, Li Z, Zhao H. Beyond chain-of-thought, effective graph-of-thought reasoning in language models. 2023, arXiv preprint arXiv: 2305.16582

[33] 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, 2023, 55(9): 195

[34] Wei J, Tay Y, Bommasani R, Raffel C, Zoph B, Borgeaud S, Yogatama D, Bosma M, Zhou D, Metzler D, Chi E H, Hashimoto T, Vinyals O, Liang P, Dean J, Fedus W. Emergent abilities of large language models. Transactions on Machine Learning Research, 2022. Survey Certification

[35] Sahoo P, Singh A K, Saha S, Jain V, Mondal S, Chadha A. A systematic survey of prompt engineering in large language models: techniques and applications. 2024, arXiv preprint arXiv: 2402.07927

[36] Li H, Leung J, Shen Z. Towards goal-oriented prompt engineering for large language models: a survey. 2024, arXiv preprint arXiv: 2401.14043

[37] Mialon G, Dessì R, Lomeli M, Nalmpantis C, Pasunuru R, Raileanu R, Rozière B, Schick T, Dwivedi-Yu J, Celikyilmaz A, Grave E, LeCun Y, Scialom T. Augmented language models: a survey. Transactions on Machine Learning Research, 2023, 2023

[38] Shin T, Razeghi Y, Logan IV R L, Wallace E, Singh S. AutoPrompt: eliciting knowledge from language models with automatically generated prompts. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. 2020

[39] Petroni F, Rocktäschel T, Riedel S, Lewis P, Bakhtin A, Wu Y, Miller A. Language models as knowledge bases? In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. 2019

[40] Cui L, Wu Y, Liu J, Yang S, Zhang Y. Template-based named entity recognition using BART. In: Proceedings of the Findings of the Association for Computational Linguistics. 2021

[41] Vu T, Lester B, Constant N, Al-Rfou' R, Cer D. SPoT: better frozen model adaptation through soft prompt transfer. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. 2022

[42] Wen Y, Jain N, Kirchenbauer J, Goldblum M, Geiping J, Goldstein T. Hard prompts made easy: gradient-based discrete optimization for prompt tuning and discovery. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 2019

[43] Liu B, Jiang Y, Zhang X, Liu Q, Zhang S, Biswas J, Stone P. LLM+P: empowering large language models with optimal planning proficiency. 2023, arXiv preprint arXiv: 2304.11477

[44] Zhou W, Zhang S, Poon H, Chen M. Context-faithful prompting for large language models. In: Proceedings of the Findings of the Association for Computational Linguistics. 2023

[45] Li S, Ning X, Wang L, Liu T, Shi X, Yan S, Dai G, Yang H, Wang Y. Evaluating quantized large language models. In: Proceedings of the 41st International Conference on Machine Learning. 2024

[46] Dong Q, Li L, Dai D, Zheng C, Ma J, Li R, Xia H, Xu J, Wu Z, Chang B, Sun X, Li L, Sui Z. A survey on in-context learning. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024

[47] Wang L, Ma C, Feng X, Zhang Z, Yang H, Zhang J, Chen Z, Tang J, Chen X, Lin Y, Zhao W X, Wei Z, Wen J. A survey on large language model based autonomous agents. Frontiers of Computer Science, 2024, 18(6): 186345

[48] Yang K, Liu J, Wu J, Yang C, Fung Y R, Li S, Huang Z, Cao X, Wang X, Wang Y, Ji H, Zhai C. If LLM is the wizard, then code is the wand: a survey on how code empowers large language models to serve as intelligent agents. 2024, arXiv preprint arXiv: 2401.00812

[49] Roy D, Zhang X, Bhave R, Bansal C, Las-Casas P, Fonseca R, Rajmohan S. Exploring LLM-based agents for root cause analysis. In: Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering. 2024

[50] Chen M, Tworek J, Jun H, Yuan Q, de Oliveira Pinto H P, Kaplan J, Edwards H, Burda Y, Joseph N, Brockman G, Ray A, Puri R, Krueger G, Petrov M, Khlaaf H, Sastry G, Mishkin P, Chan B, Gray S, Ryder N, Pavlov M, Power A, Kaiser L, Bavarian M, Winter C, Tillet P, Such F P, Cummings D, Plappert M, Chantzis F, Barnes E, Herbert-Voss A, Guss W H, Nichol A, Paino A, Tezak N, Tang J, Babuschkin I, Balaji S, Jain

S, Saunders W, Hesse C, Carr A N, Leike J, Achiam J, Misra V, Morikawa E, Radford A, Knight M, Brundage M, Murati M, Mayer K, Welinder P, McGrew B, Amodei D, McCandlish S, Sutskever I, Zaremba W. Evaluating large language models trained on code. 2021, arXiv preprint arXiv: 2107.03374

[51] Nijkamp E, Pang B, Hayashi H, Tu L, Wang H, Zhou Y, Savarese S, Xiong C. CodeGen: an open large language model for code with multi-turn program synthesis. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[52] Driess D, Xia F, Sajjadi M S M, Lynch C, Chowdhery A, Ichter B, Wahid A, Tompson J, Vuong Q, Yu T, Huang W, Chebotar Y, Sermanet P, Duckworth D, Levine S, Vanhoucke V, Hausman K, Toussaint M, Greff K, Zeng A, Mordatch I, Florence P. PaLM-E: an embodied multimodal language model. In: Proceedings of the 40th International Conference on Machine Learning. 2023

[53] Li Y, Li Z, Zhang K, Dan R, Jiang S, Zhang Y. ChatDoctor: a medical chat model fine-tuned on a large language model meta-AI (LLaMA) using medical domain knowledge. Cureus, 2023, 15(6): e40895

[54] Yue S, Chen W, Wang S, Li B, Shen C, Liu S, Zhou Y, Xiao Y, Yun S, Huang X, Wei Z. DISC-LawLLM: fine-tuning large language models for intelligent legal services. 2023, arXiv preprint arXiv: 2309.11325

[55] Yiquan W, Yuhang L, Yifei L, Ang L, Siying Z, Kun K. wisdominterrogatory. see https://github.com/zhihaiLLM/wisdomInterrogatory website

[56] Chakrabarty T, Padmakumar V, He H. *Help me write a poem*: instruction tuning as a vehicle for collaborative poetry writing. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. 2022

[57] Ippolito D, Yuan A, Coenen A, Burnam S. Creative writing with an AI-powered writing assistant: perspectives from professional writers. 2022, arXiv preprint arXiv: 2211.05030

[58] Yuan R, Lin H, Wang Y, Tian Z, Wu S, Shen T, Zhang G, Wu Y, Liu C, Zhou Z, Ma Z, Xue L, Wang Z, Liu Q, Zheng T, Li Y, Ma Y, Liang Y, Chi X, Liu R, Wang Z, Lin C, Liu Q, Jiang T, Huang W, Chen W, Chen W, Fu J, Benetos E, Xia G, Dannenberg R, Xue W, Kang S, Guo Y. ChatMusician: understanding and generating music intrinsically with LLM. In: Proceedings of the Findings of the Association for Computational Linguistics. 2024

[59] OpenAI. See openai.com/index/sora/ website, 2024

[60] Reynolds L, McDonell K. Prompt programming for large language models: beyond the few-shot paradigm. In: Proceedings of 2021 CHI Conference on Human Factors in Computing Systems. 2021, 314

[61] Xu X, Tao C, Shen T, Xu C, Xu H, Long G, Lou J-G, Ma S. Re-reading improves reasoning in large language models. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024

[62] Wang Z, Cai S, Chen G, Liu A, Ma X, Liang Y, Team CraftJarvis. Describe, explain, plan and select: interactive planning with large language models enables open-world multi-task agents. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 1480

[63] Zhu X, Chen Y, Tian H, Tao C, Su W, Yang C, Huang G, Li B, Lu L, Wang X, Qiao Y, Zhang Z, Dai J. Ghost in the minecraft: generally capable agents for open-world environments via large language models with text-based knowledge and memory. 2023, arXiv preprint arXiv: 2305.17144

[64] Kojima T, Gu S S, Reid M, Matsuo Y, Iwasawa Y. Large language models are zero-shot reasoners. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 1613

[65] Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I, others. Language models are unsupervised multitask learners. OpenAI blog, 2019, 1(8): 9

[66] Logan IV R, Balazevic I, Wallace E, Petroni F, Singh S, Riedel S. Cutting down on prompts and parameters: simple few-shot learning with language models. In: Proceedings of the Findings of the Association for Computational Linguistics. 2022

[67] Liu J, Shen D, Zhang Y, Dolan B, Carin L, Chen W. What makes good in-context examples for GPT-3? In: Proceedings of Deep Learning Inside Out (DeeLIO 2022): the 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures. 2022

[68] Su H, Kasai J, Wu C H, Shi W, Wang T, Xin J, Zhang R, Ostendorf M, Zettlemoyer L, Smith N A, Yu T. Selective annotation makes language models better few-shot learners. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[69] Jiang Z, Xu F F, Araki J, Neubig G. How can we know what language models know? Transactions of the Association for Computational Linguistics, 2020, 8: 423−438

[70] 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. 2022

[71] Nie F, Chen M, Zhang Z, Cheng X. Improving few-shot performance of language models via nearest neighbor calibration. 2022, arXiv preprint arXiv: 2212.02216

[72] Fei Y, Hou Y, Chen Z, Bosselut A. Mitigating label biases for in-context learning. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023

[73] Ma H, Zhang C, Bian Y, Liu L, Zhang Z, Zhao P, Zhang S, Fu H, Hu Q, Wu B. Fairness-guided few-shot prompting for large language models. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023

[74] Reif Y, Schwartz R. Beyond performance: quantifying and mitigating label bias in LLMs. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2024

[75] Han Z, Hao Y, Dong L, Sun Y, Wei F. Prototypical calibration for few-shot learning of language models. In: Proceedings of the 11th International Conference on Learning Representations. 2023.

[76] Hu L, Liu Z, Zhao Z, Hou L, Nie L, Li J. A survey of knowledge enhanced pre-trained language models. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(4): 1413−1430

[77] Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, Küttler H, Lewis M, Yih W T, Rocktäschel T, Riedel S, Kiela D. Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 793

[78] Pan S, Luo L, Wang Y, Chen C, Wang J, Wu X. Unifying large language models and knowledge graphs: a roadmap. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(7): 3580−3599

[79] Yang L, Chen H, Li Z, Ding X, Wu X. Give us the facts: enhancing large language models with knowledge graphs for fact-aware language modeling. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(7): 3091−3110

[80] Barnett S, Kurniawan S, Thudumu S, Brannelly Z, Abdelrazek M. Seven failure points when engineering a retrieval augmented generation system. In: Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI. 2024

[81] Shao Z, Gong Y, Shen Y, Huang M, Duan N, Chen W. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In: Proceedings of the Findings of the Association for Computational Linguistics. 2023

[82] Thakur N, Reimers N, RückléA, Srivastava A, Gurevych I. BEIR: a heterogenous benchmark for zero-shot evaluation of information retrieval models. 2021, arXiv preprint arXiv: 2104.08663

[83] Nguyen T, Rosenberg M, Song X, Gao J, Tiwary S, Majumder R, Deng L. MS MARCO: a human generated machine reading comprehension dataset. In: Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches 2016 Co-located with the 30th Annual Conference on Neural Information Processing Systems. 2016

[84] Craswell N, Mitra B, Yilmaz E, Campos D, Voorhees E M. Overview of the TREC 2019 deep learning track. 2020, arXiv preprint arXiv: 2003.07820

[85] Joshi M, Choi E, Weld D, Zettlemoyer L. TriviaQA: a large scale distantly supervised challenge dataset for reading comprehension. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. 2017

[86] Ho X, Duong Nguyen A K, Sugawara S, Aizawa A. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In: Proceedings of the 28th International Conference on Computational Linguistics. 2020

[87] Yang Z, Qi P, Zhang S, Bengio Y, Cohen W, Salakhutdinov R, Manning C D. HotpotQA: a dataset for diverse, explainable multi-hop question answering. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 2018

[88] Yu W, Zhang H, Pan X, Cao P, Ma K, Li J, Wang H, Yu D. Chain-of-note: enhancing robustness in retrieval-augmented language models. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024

[89] Melz E. Enhancing LLM intelligence with ARM-RAG: auxiliary rationale memory for retrieval augmented generation. 2023, arXiv preprint arXiv: 2311.04177

[90] Cuconasu F, Trappolini G, Siciliano F, Filice S, Campagnano C, Maarek Y, Tonellotto N, Silvestri F. The power of noise: redefining retrieval for RAG systems. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2024

[91] Weston J, Sukhbaatar S. System 2 Attention (is something you might need too). 2023, arXiv preprint arXiv: 2311.11829

[92] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017

[93] Touvron H, Lavril T, Izacard G, Martinet X, Lachaux M A, Lacroix T, Rozière B, Goyal N, Hambro E, Azhar F, Rodriguez A, Joulin A, Grave E, Lample G. LLaMA: open and efficient foundation language models. 2023, arXiv preprint arXiv: 2302.13971

[94] Shi F, Chen X, Misra K, Scales N, Dohan D, Chi E, Schärli N, Zhou D. Large language models can be easily distracted by irrelevant context. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 1291

[95] Zhou Y, Liu Z, Jin J, Nie J-Y, Dou Z. Metacognitive retrieval-augmented large language models. In: Proceedings of the ACM Web Conference 2024. 2024

[96] Austin J T, Vancouver J B. Goal constructs in psychology: structure, process, and content. Psychological Bulletin, 1996, 120(3): 338−375

[97] Zhou D, Schärli N, Hou L, Wei J, Scales N, Wang X, Schuurmans D, Cui C, Bousquet O, Le Q V, Chi E H. Least-to-most prompting enables complex reasoning in large language models. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[98] Zhang Z, Zhang A, Li M, Smola A. Automatic chain of thought prompting in large language models. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[99] Reimers N, Gurevych I. Sentence-BERT: sentence embeddings using siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. 2019

[100] Diao S, Wang P, Lin Y, Pan R, Liu X, Zhang T. Active prompting with chain-of-thought for large language models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. 2024

[101] Lyu Q, Havaldar S, Stein A, Zhang L, Rao D, Wong E, Apidianaki M, Callison-Burch C. Faithful chain-of-thought reasoning. In: Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics. 2023

[102] Liu T, Guo Q, Yang Y, Hu X, Zhang Y, Qiu X, Zhang Z. Plan, verify and switch: integrated reasoning with diverse X-of-thoughts. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023

[103] Liu R, Wei J, Gu S S, Wu T-Y, Vosoughi S, Cui C, Zhou D, Dai A M. Mind's eye: grounded language model reasoning through simulation. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[104] Li L, Xu J, Dong Q, Zheng C, Sun X, Kong L, Liu Q. Can language models understand physical concepts? In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023

[105] Wang Y, Duan J, Fox D, Srinivasa S. NEWTON: are large language models capable of physical reasoning? In: Proceedings of the Findings of the Association for Computational Linguistics. 2023

[106] Todorov E, Erez T, Tassa Y. MuJoCo: a physics engine for model-based control. In: Proceedings of 2012 IEEE/RSJ International

Conference on Intelligent Robots and Systems. 2012

[107] Drori I, Zhang S, Shuttleworth R, Tang L, Lu A, Ke E, Liu K, Cheng L, Tran S, Cheng N, Wang R, Singh N, Patti T L, Lynch J, Shporer A, Verma N, Wu E, Strang G. A neural network solves, explains, and generates university math problems by program synthesis and few-shot learning at human level. 2021, arXiv preprint arXiv: 2112.15594

[108] Fu Y, Peng H, Sabharwal A, Clark P, Khot T. Complexity-based prompting for multi-step reasoning. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[109] Cheng Z, Xie T, Shi P, Li C, Nadkarni R, Hu Y, Xiong C, Radev D, Ostendorf M, Zettlemoyer L, Smith N A, Yu T. Binding language models in symbolic languages. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[110] Wu Y, Jiang A Q, Li W, Rabe M N, Staats C, Jamnik M, Szegedy C. Autoformalization with large language models. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 2344

[111] Gao L, Madaan A, Zhou S, Alon U, Liu P, Yang Y, Callan J, Neubig G. PAL: program-aided language models. In: Proceedings of the 40th International Conference on Machine Learning. 2023

[112] Chen W, Ma X, Wang X, Cohen W W. Program of thoughts prompting: disentangling computation from reasoning for numerical reasoning tasks. Transactions on Machine Learning Research, 2023, 2023

[113] Luo X, Zhu Q, Zhang Z, Qin L, Wang X, Yang Q, Xu D, Che W. Multipot: Multilingual program of thoughts harnesses multiple programming languages. arXiv e-prints, 2024, arXiv–2402

[114] Zeng A, Attarian M, Ichter B, Choromanski K M, Wong A, Welker S, Tombari F, Purohit A, Ryoo M S, Sindhwani V, Lee J, Vanhoucke V, Florence P. Socratic models: composing zero-shot multimodal reasoning with language. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[115] Parisi A, Zhao Y, Fiedel N. TALM: tool augmented language models. 2022, arXiv preprint arXiv: 2205.12255

[116] Schick T, Dwivedi-Yu J, Dessí R, Raileanu R, Lomeli M, Hambro E, Zettlemoyer L, Cancedda N, Scialom T. Toolformer: language models can teach themselves to use tools. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 2997

[117] Taylor R, Kardas M, Cucurull G, Scialom T, Hartshorn A, Saravia E, Poulton A, Kerkez V, Stojnic R. Galactica: a large language model for science. 2022, arXiv preprint arXiv: 2211.09085

[118] Karpas E, Abend O, Belinkov Y, Lenz B, Lieber O, Ratner N, Shoham Y, Bata H, Levine Y, Leyton-Brown K, Muhlgay D, Rozen N, Schwartz E, Shachaf G, Shalev-Shwartz S, Shashua A, Tenenholtz M. MRKL systems: a modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. 2022, arXiv preprint arXiv: 2205.00445

[119] Paranjape B, Lundberg S, Singh S, Hajishirzi H, Zettlemoyer L, Ribeiro M T. ART: automatic multi-step reasoning and tool-use for large language models. 2023, arXiv preprint arXiv: 2303.09014

[120] Lu P, Peng B, Cheng H, Galley M, Chang K-W, Wu Y N, Zhu S-C, Gao J. Chameleon: plug-and-play compositional reasoning with large language models. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023

[121] Patil S G, Zhang T, Wang X, Gonzalez J E. Gorilla: large language model connected with massive APIs. In: Proceedings of the 38th International Conference on Neural Information Processing Systems. 2024

[122] Shen Y, Song K, Tan X, Li D, Lu W, Zhuang Y. HuggingGPT: solving AI tasks with ChatGPT and its friends in hugging face. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023

[123] Tang Q, Deng Z, Lin H, Han X, Liang Q, Cao B, Sun L. ToolAlpaca: generalized tool learning for language models with 3000 simulated cases. 2023, arXiv preprint arXiv: 2306.05301

[124] Ruan J, Chen Y, Zhang B, Xu Z, Bao T, Du G, Shi S, Mao H, Li Z, Zeng X, Zhao R. TPTU: large language model-based AI agents for task planning and tool usage. 2023, arXiv preprint arXiv: 2308.03427

[125] Kong Y, Ruan J, Chen Y, Zhang B, Bao T, Shi S, Qing D, Hu X, Mao H, Li Z, Zeng X, Zhao R, Wang X. TPTU-v2: boosting task planning and tool usage of large language model-based agents in real-world industry systems. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024

[126] Liang Y, Wu C, Song T, Wu W, Xia Y, Liu Y, Ou Y, Lu S, Ji L, Mao S, Wang Y, Shou L, Gong M, Duan N. TaskMatrix.AI: completing tasks by connecting foundation models with millions of APIs. 2023, arXiv preprint arXiv: 2303.16434

[127] Madaan A, Tandon N, Gupta P, Hallinan S, Gao L, Wiegreffe S, Alon U, Dziri N, Prabhumoye S, Yang Y, Gupta S, Majumder B P, Hermann K, Welleck S, Yazdanbakhsh A, Clark P. Self-refine: iterative refinement with self-feedback. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023

[128] Lu J, Zhong W, Huang W, Wang Y, Zhu Q, Mi F, Wang B, Wang W, Zeng X, Shang L, Jiang X, Liu Q. SELF: self-evolution with language feedback. 2023, arXiv preprint arXiv: 2310.00533

[129] Zhang W, Shen Y, Wu L, Peng Q, Wang J, Zhuang Y, Lu W. Self-contrast: better reflection through inconsistent solving perspectives. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. 2024

[130] Wang S, Li S, Sun T, Fu J, Cheng Q, Ye J, Ye J, Qiu X, Huang X. LLM can achieve self-regulation via hyperparameter aware generation. In: Proceedings of the Findings of the Association for Computational Linguistics. 2024

[131] Huang H, Qu Y, Liu J, Yang M, Xu B, Zhao T, Lu W. Self-evaluation of large language model based on glass-box features. In: Proceedings of the Findings of the Association for Computational Linguistics. 2024

[132] Huang J, Chen X, Mishra S, Zheng H S, Yu A W, Song X, Zhou D. Large language models cannot self-correct reasoning yet. In: Proceedings of the 12th International Conference on Learning Representations. 2024

[133] Xu W, Zhu G, Zhao X, Pan L, Li L, Wang W Y. Perils of self-feedback: self-bias amplifies in large language models. 2024, arXiv preprint arXiv: 2402.11436v1

[134] Gou Z, Shao Z, Gong Y, Shen Y, Yang Y, Duan N, Chen W. CRITIC: large language models can self-correct with tool-interactive critiquing. In: Proceedings of the 12th International Conference on Learning Representations. 2024

[135] Zhao Z, Wallace E, Feng S, Klein D, Singh S. Calibrate before use: improving few-shot performance of language models. In: Proceedings of the 38th International Conference on Machine Learning. 2021

[136] Si C, Gan Z, Yang Z, Wang S, Wang J, Boyd-Graber J L, Wang L. Prompting GPT-3 to be reliable. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[137] Liang P, Bommasani R, Lee T, Tsipras D, Soylu D, et al. Holistic evaluation of language models. Transactions on Machine Learning Research, 2023, 2023

[138] Ye X, Durrett G. The unreliability of explanations in few-shot prompting for textual reasoning. 2022, arXiv preprint arXiv: 2205.03401

[139] 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. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023

[140] Lin S, Hilton J, Evans O. TruthfulQA: measuring how models mimic human falsehoods. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. 2022

[141] Schick T, Schütze H. Exploiting cloze-questions for few-shot text classification and natural language inference. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics. 2021

[142] Lester B, Al-Rfou R, Constant N. The power of scale for parameter-efficient prompt tuning. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021

[143] Wang X, Wei J, Schuurmans D, Le Q V, Chi E H, Narang S, Chowdhery A, Zhou D. Self-consistency improves chain of thought reasoning in language models. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[144] Li Y, Lin Z, Zhang S, Fu Q, Chen B, Lou J-G, Chen W. Making language models better reasoners with step-aware verifier. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023

[145] Pitis S, Zhang M R, Wang A, Ba J. Boosted prompt ensembles for large language models. 2023, arXiv preprint arXiv: 2304.05970

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

[147] Arora S, Narayan A, Chen M F, Orr L J, Guha N, Bhatia K, Chami I, Ré C. Ask me anything: a simple strategy for prompting language models. In: Proceedings of the 11th International Conference on Learning Representations. 2023

[148] Hou B, O'Connor J, Andreas J, Chang S, Zhang Y. PromptBoosting: black-box text classification with ten forward passes. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 542

[149] Zhang C, Liu L, Wang C, Sun X, Wang H, Wang J, Cai M. PREFER: prompt ensemble learning via feedback-reflect-refine. In: Proceedings of the 38th AAAI Conference on Artificial Intelligence. 2024

[150] Gao T, Fisch A, Chen D. Making pre-trained language models better few-shot learners. arXiv preprint arXiv:2012.15723, 2020

[151] Allingham J U, Ren J, Dusenberry M W, Gu X, Cui Y, Tran D, Liu J Z, Lakshminarayanan B. A simple zero-shot prompt weighting technique to improve prompt ensembling in text-image models. In: Proceedings of the 40th International Conference on Machine Learning. 2023

[152] Meade N, Poole-Dayan E, Reddy S. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. 2022

[153] Nadeem M, Bethke A, Reddy S. StereoSet: measuring stereotypical bias in pretrained language models. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing. 2021

[154] Liu Y, Jia Y, Geng R, Jia J, Gong N Z. Formalizing and benchmarking prompt injection attacks and defenses. In: Proceedings of the 33rd USENIX Security Symposium. 2024

[155] Tang Z, Zhou K, Li J, Ding Y, Wang P, Yan B, Hua R, Zhang M. CMD: a framework for context-aware model self-detoxification. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024

[156] Kaddour J, Harris J, Mozes M, Bradley H, Raileanu R, McHardy R. Challenges and applications of large language models. 2023, arXiv preprint arXiv: 2307.10169

[157] Bowman R, Cooney O, Newbold J W, Thieme A, Clark L, Doherty G, Cowan B. Exploring how politeness impacts the user experience of chatbots for mental health support. International Journal of Human-Computer Studies, 2024, 184: 103181

[158] Glaese A, McAleese N, Trębacz M, Aslanides J, Firoiu V, Ewalds T, Rauh M, Weidinger L, Chadwick M, Thacker P, Campbell-Gillingham L, Uesato J, Huang P S, Comanescu R, Yang F, See A, Dathathri S, Greig R, Chen C, Fritz D, Elias J S, Green R, Mokrá S, Fernando N, Wu B, Foley R, Young S, Gabriel I, Isaac W, Mellor J, Hassabis D, Kavukcuoglu K, Hendricks L A, Irving G. Improving alignment of dialogue agents via targeted human judgements. 2022, arXiv preprint arXiv: 2209.14375

[159] OpenAI. chatgpt. see the website of OpenAI 2024–5-10

[160] OpenAI, Achiam J, Adler S, Agarwal S, Ahmad L, et al. GPT-4 technical report. 2024, arXiv preprint arXiv: 2303.08774

[161] Microsoft. copilot. See the website of copilot.microsoft.com 2024-5-10

[162] Anthropic. The claude 3 model family: opus, sonnet, Haiku. See the website of api.semanticscholar.org/CorpusID:268232499 2024

[163] Gao L, Ma X, Lin J, Callan J. Precise zero-shot dense retrieval without relevance labels. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023

[164] Jagerman R, Zhuang H, Qin Z, Wang X, Bendersky M. Query expansion by prompting large language models. ArXiv, 2023, abs/2305.03653

[165] Jagerman R, Zhuang H, Qin Z, Wang X, Bendersky M. Query expansion by prompting large language models. 2023, arXiv preprint arXiv: 2305.03653

[166] Li M, Zhuang H, Hui K, Qin Z, Lin J, Jagerman R, Wang X, Bendersky M. Generate, filter, and fuse: query expansion via multi-step keyword generation for zero-shot neural rankers. 2023, arXiv preprint

arXiv: 2311.09175
[167] Wu Y, Wu W, Xing C, Zhou M, Li Z. Sequential matching network: a new architecture for multi-turn response selection in retrieval-based chatbots. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. 2017
[168] Ziems N, Yu W, Zhang Z, Jiang M. Large language models are built-in autoregressive search engines. In: Proceedings of the Findings of the Association for Computational Linguistics. 2023
[169] Hongcheng Liu Y M Y WY. L. Xiezhi chinese law large language model. see https://github.com/LiuHC0428/LAW_GPT website, 2023
[170] Singhal K, Azizi S, Tu T, Mahdavi S S, Wei J, et al. Large language models encode clinical knowledge. Nature, 2023, 620(7972): 172−180
[171] Yoo K M, Park D, Kang J, Lee S-W, Park W. GPT3Mix: leveraging large-scale language models for text augmentation. In: Proceedings of the Findings of the Association for Computational Linguistics. 2021, 2225−2239
[172] Bonifacio L, Abonizio H, Fadaee M, Nogueira R. InPars: unsupervised dataset generation for information retrieval. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2022
[173] Jeronymo V, Bonifacio L, Abonizio H, Fadaee M, Lotufo D R, Zavrel J, Nogueira R. InPars-v2: large language models as efficient dataset generators for information retrieval. 2023, arXiv preprint arXiv: 2301.01820
[174] Dai H, Liu Z, Liao W, Huang X, Wu Z, Zhao L, Liu W, Liu N, Li S, Zhu D, Cai H, Sun L, Li Q, Shen D, Liu T, Li X. AugGPT: leveraging ChatGPT for text data augmentation. 2023, arXiv preprint arXiv: 2302.13007
[175] Wu S, Irsoy O, Lu S, Dabravolski V, Dredze M, Gehrmann S, Kambadur P, Rosenberg D, Mann G. BloombergGPT: a large language model for finance. 2023, arXiv preprint arXiv: 2303.17564
[176] Yang H, Liu X-Y, Wang C D. FinGPT: open-source financial large language models. 2023, arXiv preprint arXiv: 2306.06031
[177] Luu R K, Buehler M J. BioinspiredLLM: conversational large language model for the mechanics of biological and bio-inspired materials. Advanced Science, 2024, 11(10): 2306724
[178] Liu S, Wang J, Yang Y, Wang C, Liu L, Guo H, Xiao C. ChatGPT-powered conversational drug editing using retrieval and domain feedback. 2023, arXiv preprint arXiv: 2305.18090
[179] Bran A M, Cox S, Schilter O, Baldassari C, White A D, Schwaller P. Augmenting large language models with chemistry tools. Nature Machine Intelligence, 2024, 6(5): 525−535
[180] Zheng Q, Xia X, Zou X, Dong Y, Wang S, Xue Y, Shen L, Wang Z, Wang A, Li Y, Su T, Yang Z, Tang J. CodeGeeX: a pre-trained model for code generation with multilingual benchmarking on HumanEval-X. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2023
[181] Raunak V, Sharaf A, Wang Y, Awadalla H, Menezes A. Leveraging GPT-4 for automatic translation post-editing. In: Proceedings of the Findings of the Association for Computational Linguistics. 2023
[182] Sharma R K, Gupta V, Grossman D. Defending language models against image-based prompt attacks via user-provided specifications. In: Proceedings of 2024 IEEE Security and Privacy Workshops. 2024
[183] Ni S, Bi K, Guo J, Cheng X. When do LLMs need retrieval augmentation? Mitigating LLMs' overconfidence helps retrieval augmentation. In: Proceedings of the Findings of the Association for Computational Linguistics. 2024
[184] Asai A, Wu Z, Wang Y, Sil A, Hajishirzi H. Self-RAG: learning to retrieve, generate, and critique through self-reflection. In: Proceedings of the 12th International Conference on Learning Representations. 2024
[185] Liu Y, Peng X, Zhang X, Liu W, Yin J, Cao J, Du T. RA-ISF: learning to answer and understand from retrieval augmentation via iterative self-feedback. In: Proceedings of the Findings of the Association for Computational Linguistics. 2024
[186] Shi Z, Zhang S, Sun W, Gao S, Ren P, Chen Z, Ren Z. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. 2024
[187] Xu S, Pang L, Shen H, Cheng X, Chua T-S. Search-in-the-chain: interactively enhancing large language models with search for knowledge-intensive tasks. In: Proceedings of the ACM Web Conference 2024. 2024
[188] Zelikman E, Wu Y, Mu J, Goodman N D. STaR: bootstrapping reasoning with reasoning. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022
[189] Zelikman E, Harik G, Shao Y, Jayasiri V, Haber N, Goodman N D. Quiet-STaR: language models can teach themselves to think before speaking. 2024, arXiv preprint arXiv: 2403.09629
[190] OpenAI. o1. see https://openai.com/o1/ website2024
[191] Schulhoff S, Ilie M, Balepur N, Kahadze K, Liu A, Si C, Li Y, Gupta A, Han H, Schulhoff S, Dulepet P S, Vidyadhara S, Ki D, Agrawal S, Pham C, Kroiz G C, Li F, Tao H, Srivastava A, Da Costa H, Gupta S, Rogers M L, Goncearenco I, Sarli G, Galynker I, Peskoff D, Carpuat M, White J, Anadkat S, Hoyle A M, Resnik P. The prompt report: a systematic survey of prompt engineering techniques. 2024, arXiv preprint arXiv: 2406.06608
[192] Yoran O, Wolfson T, Ram O, Berant J. Making retrieval-augmented language models robust to irrelevant context. arXiv preprint arXiv:2310.01558, 2023
[193] Microsoft. GraphRAG. See the website of www.microsoft.com/en-us/research/project/graphrag/, 2024
[194] Sun W, Cai H, Chen H, Ren P, Chen Z, de Rijke M, Ren Z. Answering ambiguous questions via iterative prompting. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023, 7669−7683
[195] Paulus A, Zharmagambetov A, Guo C, Amos B, Tian Y. AdvPrompter: fast adaptive adversarial prompting for LLMs. 2024, arXiv preprint arXiv: 2404.16873
[196] Kepel D, Valogianni K. Autonomous prompt engineering in large language models. 2024, arXiv preprint arXiv: 2407.11000

Yao-Yang LIU joined the Key Laboratory of Data Engineering and Knowledge Engineer (MOE), Renmin University of China, China in 2023. His major research interests include machine learning, database, and large language models.

Zhen ZHENG received his PhD degree from the Department of Computer Science and Technology, Tsinghua University, China in 2019. He joined Alibaba in August 2019. His research interests include AI compiler, large scale machine learning systems, and heterogeneous computing.

Feng ZHANG received his PhD degree in computer science from Tsinghua University, China in 2017. He is a professor with the Key Laboratory of Data Engineering and Knowledge Engineer (MOE), Renmin University of China, China. His major research interests include big data management systems and parallel and distributed systems.

Jin-Cheng FENG received her bachelor degree from the School of Information, Renmin University of China, China in 2025. She will begin her study for the master's degree at the Gaoling School of Artificial Intelligence, Renmin University of China, China from September 2025.

Yi-Yang FU joined Knowledge reasoning and cognition, Renmin University of China, China in 2023. She is now a junior student at School of Information, Renmin University of China, China.

Ji-Dong ZHAI received his BS degree in computer science from the University of Electronic Science and Technology of China, China in 2003, and PhD degree in computer science from Tsinghua University, China in 2010. He is an associate professor in Department of Computer Science and Technology, Tsinghua University, China. His research interests include performance evaluation for high performance computers, performance analysis, and modeling of parallel applications.

Bing-Sheng HE received his bachelor degree in computer science from Shanghai Jiao Tong University, China (1999−2003), and his PhD degree in computer science from Hong Kong University of Science and Technology, China (2003−2008). He is an associate professor in School of Computing, National University of Singapore, Singapore. His research interests are high performance computing, distributed and parallel systems, and database systems.

Xiao ZHANG received his master's degree from Renmin University and his PhD degree from the Institute of Computing Technology, Chinese Academy of Science, China in 1998 and 2001, respectively, both in computer science and technology. He is a professor at the School of Information, Renmin University of China, China. His research interests fouces on big data management systems.

Xiao-Yong DU obtained his BS degree from Hangzhou University, China in 1983, his ME degree from Renmin University of China, China in 1988, and his PhD degree from Nagoya Institute of Technology, Japan in 1997. He is currently a professor with the School of Information, Renmin University of China, China. His current research interests include databases and intelligent information retrieval.