INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 09 ISSUE: 01 | JAN - 2025 SJIF RATING: 8.448 ISSN: 2582-3930

Expanding Horizons in Prompt Engineering: Techniques, Frameworks, and Challenges

Nan Wu
Stamford, CT, USA
Chris.wunan88@gmail.com

***Abstract*—Prompt engineering is a critical method for guiding large language models (LLMs) to perform diverse tasks effectively. This paper reviews prompt engineering techniques, including example-driven approaches, logic steps, and modular frameworks, highlighting their adaptability and limitations. It emphasizes the need for robust evaluation metrics, addressing gaps in reasoning quality and task adherence. By proposing novel evaluation approaches such as token-level entropy, this study advances understanding of prompt effectiveness, paving the way for more reliable, ethical, and domain-specific LLM applications.**

Keywords—prompt engineering, language models, chain of thought, AI benchmark

Funding Declaration

The author declares that no funding was received for conducting this study or for the preparation of this manuscript.

I. INTRODUCTION

Prompt engineering has emerged as a transformative approach for adapting large language models (LLMs) to perform a wide array of tasks without requiring additional training or fine-tuning. By crafting tailored instructions or examples within the input, prompt engineering enables LLMs to solve problems, generate creative content, and adapt to domain-specific challenges. Its versatility and simplicity have positioned it as a critical technique in leveraging the capabilities of state-of-the-art models, such as GPT, Gemini, and Claude.[1] The process is shown in Fig. 1

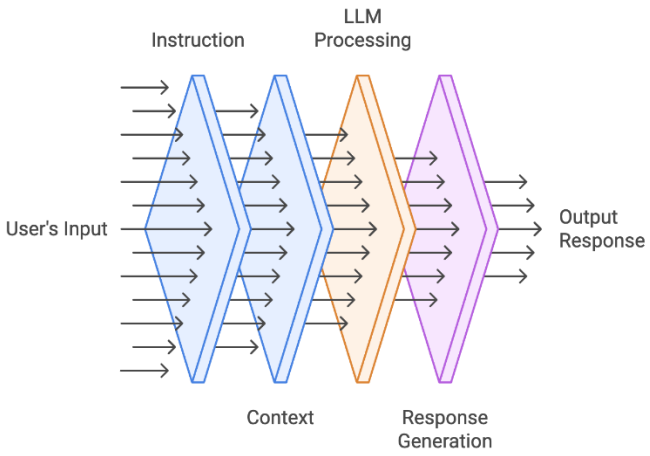

Fig. 1. Flow of Data in Prompt Engineering

Despite its widespread adoption and success, prompt engineering is far from being a solved problem. The field has grown rapidly, but its foundations remain primarily empirical. Most advancements rely on trial-and-error methods or ad hoc best practices, which often lack a rigorous theoretical or evaluative framework. While surveys and taxonomies of prompt engineering techniques exist, they predominantly focus on classifying methods

rather than addressing key questions about prompt effectiveness, evaluation metrics, and scalability.[2]

A particularly critical and underexplored area is the evaluation of prompts themselves. The effectiveness of a prompt often varies significantly based on task type, domain, or even minor changes in wording. Current evaluation methods primarily rely on task-specific accuracy or output correctness, which fail to capture nuances such as the logical coherence of reasoning, the robustness of responses, or the computational efficiency of different approaches. Furthermore, as chain-of-thought (CoT) prompting and other reasoning-driven methods gain traction, there is growing interest in understanding whether LLMs genuinely engage in step-by-step reasoning or merely mimic reasoning paths seen during pretraining—a phenomenon sometimes referred to as post-hoc reasoning.[3]

This paper aims to bridge these gaps by providing a focused review of prompt engineering techniques and their evaluation. Unlike existing surveys, which primarily emphasize the taxonomy of methods, we delve into the challenges and limitations associated with evaluating prompt effectiveness. We explore emerging approaches, such as token-level probability analysis and entropy metrics, to assess the reasoning quality and robustness of prompts. Additionally, we examine the broader implications of prompt engineering for real-world applications, from domain adaptation to ethical considerations, and highlight open questions that demand further exploration.

Prompt engineering has developed into a highly dynamic field, characterized by a paradoxical mix of case-specific applications and universal techniques. While the tasks for which prompts are crafted often vary widely—ranging from creative writing and logic puzzles to legal document analysis—the underlying methodologies are largely adaptable across domains. Techniques such as chain-of-thought prompting, few-shot learning, and self-consistency sampling have proven effective for diverse tasks, demonstrating the potential for a generalizable framework in prompt engineering. [4]

However, the popularity of specific approaches often owes more to viral products and social media trends than to rigorous benchmarking. For instance, techniques associated with tools that roast users' Twitter content or produce humorous AI-driven commentary have gained traction primarily due to their entertainment value rather than their objective performance metrics.[5] This raises questions about how the field evaluates success, as the most visible frameworks may not necessarily be the most effective. The reliance on viral appeal rather than systematic validation highlights the need for robust evaluation frameworks to ensure that prompt engineering progresses on the basis of substance rather than hype.

By providing a fresh perspective on prompt engineering frameworks and their current status, this paper aims to move beyond existing surveys and highlight the interplay between universal techniques and case-specific applications. Rather than proposing new methodologies, it seeks to offer readers a nuanced understanding of the field's evolution, emphasizing the factors that shape popular frameworks and the challenges that remain unaddressed, thereby enriching the discourse around the potential and limitations of LLM-driven prompt engineering.

II. THE LANDSCAPE OF PROMPT ENGINEERING

Prompt engineering has rapidly evolved from simple instructions to sophisticated frameworks designed to adapt large language models (LLMs) for specific tasks. At its core, prompt engineering focuses on crafting input that directs a model to produce desirable outputs, allowing for fine-tuned performance without retraining [6].The diversity of tasks and applications has led to the emergence of various techniques, categorized by their use of examples, logic-driven instructions, regulatory constraints, conversational dynamics, or integration with external data. Many popular frameworks, such as CRISPE[7], build upon these fundamental techniques, combining them in modular ways to create adaptable solutions.

A. *Categorizing Prompt Engineering Techniques*

Prompt engineering techniques can be classified based on the key attributes of the instructions provided to the model. These categories serve as a conceptual index to help readers understand the modular building blocks of prompts, highlighting their adaptability and combinatory potential.

1) *Example-Driven Techniques*

Providing examples within prompts is one of the most intuitive and effective ways to guide LLM behavior. These techniques vary in the number and nature of examples included:

Zero-Shot Prompting: No examples are provided. Instead, a direct instruction defines the task.[8] Zero-shot prompts rely entirely on the model's generalization capabilities, making them simple but less effective for complex tasks.

One-Shot/Few-Shot Prompting: Includes a single or multiple example(s) to provide context or format guidance.[9] The single example helps the model infer the task's requirements. Few-shot prompts are particularly effective in tasks requiring adherence to specific patterns, significantly improving accuracy over zero-shot approaches.

2) *Logic-Step Techniques*

Logic-step techniques explicitly guide the model through reasoning processes, helping it break down complex tasks into smaller, manageable steps.

Chain-of-Thought (CoT) Prompting: Guides the model to reason step-by-step before arriving at a conclusion.[10] CoT prompting is widely recognized for its ability to improve performance on multi-step reasoning tasks.

Tree-of-Thoughts (ToT) Prompting[11]**:** Expands CoT by exploring multiple reasoning paths. This allows the model to evaluate alternatives, correct mistakes, and improve the robustness of the output. ToT is particularly useful in tasks requiring creativity or strategic decision-making, as it mimics human-like backtracking and evaluation.

Chain of Code (CoC): A technique that solves problems by combining logical reasoning with a code-like structure, explicitly tracking intermediate states after each step. CoC is particularly suited for tasks involving computational processes or iterative logic.[12]

3) *Regulatory Instruction Techniques*

These techniques regulate the model's outputs by defining roles, specifying formats, or incorporating self-check mechanisms.

Role Assignment: Assigns the model a specific persona or context to guide its behavior.[13]

Formatting Constraints: Specifies how the output should be structured.[14]

Self-Check Instructions: Prompts the model to validate or refine its response.[15] This technique enhances accuracy by embedding quality control into the response process.

4) *Multi-Round Conversation Techniques*

Multi-round techniques leverage iterative interactions, allowing users to guide the model toward the desired outcome through dialogue.[16]

Interactive Prompting: Tasks are broken into smaller parts across multiple exchanges.

5) *External Data Integration Techniques*

These techniques integrate external resources such as databases, internet searches, or APIs, enabling the model to augment its responses with up-to-date or domain-specific information.[17]

Retrieval-Augmented Generation (RAG): Combines the model's generative abilities with external knowledge retrieval.

Internet-Connected Models: Enables prompts to direct the model to fetch real-time information via APIs or search engines.

Fig.2 provides a summary of these basic techniques.

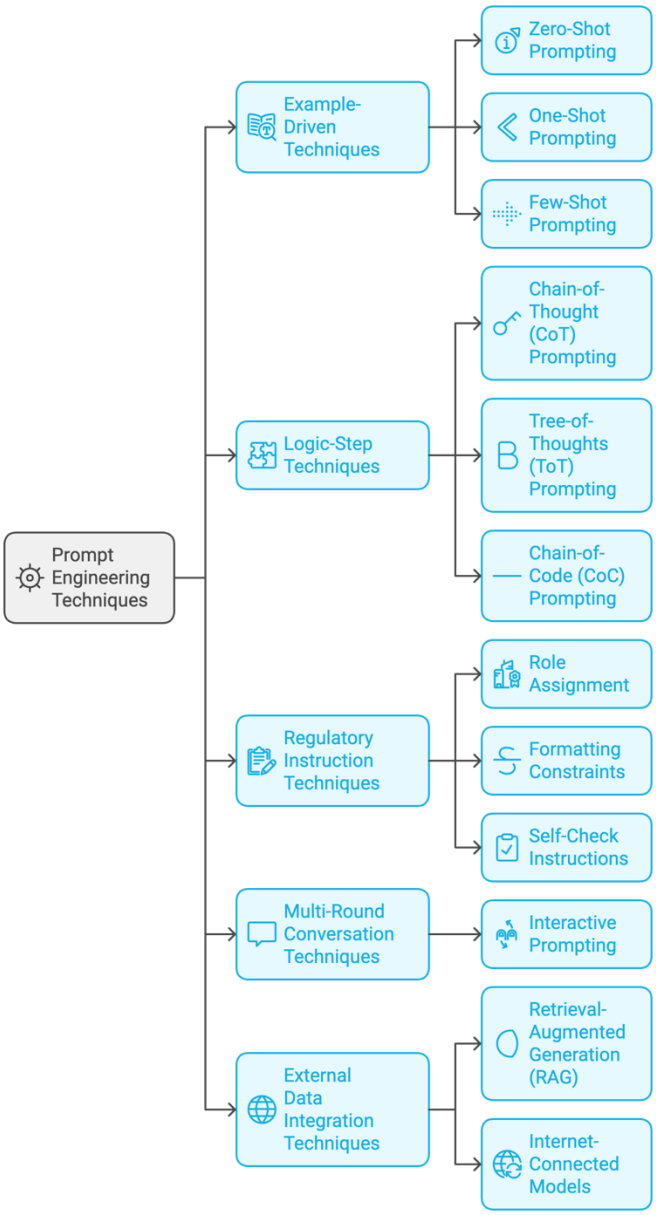

Fig. 2. Basic Prompt Technics

6) Modular Combinations and Frameworks

Prompt engineering techniques are not mutually exclusive and can be combined to create custom frameworks tailored to specific needs. However these frameworks are simplify the process of organizing and formatting prompts in specific scenarios, their value lies primarily in their practical utility rather than their academic significance. For example, frameworks like CRISPE or Co-STAR provide structured approaches by combining elements such as role assignment, formatting constraints, and logic-driven techniques, but these components are already well-documented as individual strategies. This paper highlights two examples to illustrate the modularity and adaptability of these frameworks, without implying that their academic exploration offers significant new insights.

a) CRISPE Framework

The CRISPE framework organizes prompts into six key elements that provide structure and clarity for generating outputs. These elements can be combined flexibly to design prompts that suit a variety of tasks.

Context: The background or information needed for the task (e.g., "I want to promote my new product Beta, a high-speed fan.").

Role: Specifies the persona or role the model should adopt (e.g., "You are a marketing expert writing for Facebook audiences.").

Instruction: Describes the specific task or action the model needs to perform (e.g., "Create a compelling Facebook post to attract clicks.").

Subject: Defines the main focus or theme of the task (e.g., "The benefits of Beta over traditional fans.").

Preset: Provides specific requirements or stylistic constraints (e.g., "Use Dyson's advertising tone for inspiration.").

Exception: Outlines what to avoid or exclude in the response (e.g., "Do not reference any unrelated brands.").

b) Co-STAR Framework

This is the framework that Shiela Tao used in her win at the Singapore Prompt Engineering Competition.[18]

Context: Provides the background or scenario for the task.

Objective: Specifies the goal or outcome the task should achieve.

Style: Defines the stylistic reference or inspiration for the output.
Example: "Follow Dyson's promotional tone and style."

Tone: Specifies the emotional or persuasive tone of the output.

Audience: Identifies the target audience for the response.

Response: Sets the desired qualities of the response, such as format or impact.

The key to effective prompt design is not memorizing these frameworks but understanding how to combine these basic building blocks to create tailored solutions for specific tasks.

By focusing on the modularity of these techniques, practitioners can adapt and innovate based on task requirements, blending elements like Chain-of-Thought reasoning with role assignments or formatting instructions. This flexibility highlights the balance between universal principles and task-specific customizations, which we explore further in the next section. Fig 3. Demonstrates how to build a customized framework from basic prompt techniques.

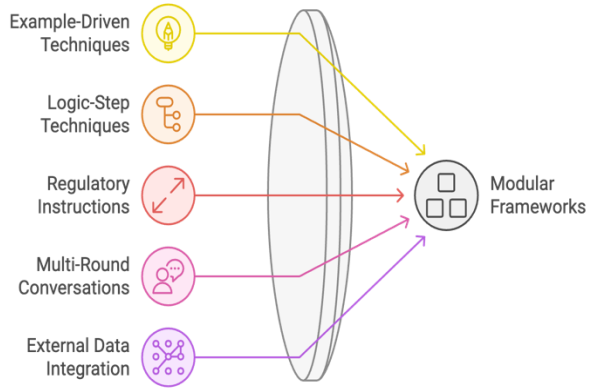

Fig. 3. Build Csutomized Prompt Framework

B. *Intersection of Universality and Specificity*

1) *Techniques for Universal Tasks*

Universal tasks are those where the model operates in broad, general domains requiring minimal task-specific customization. Techniques that excel here leverage the LLM's pre-trained general knowledge and flexibility.

Zero-Shot Prompting: This technique is ideal for universal tasks such as fact-checking, simple queries, or summarizations where no task-specific examples are required. It allows the model to rely on its training to generalize effectively without additional input. Studies highlight that zero-shot prompting achieves notable performance in open-domain QA and summarization tasks by leveraging pre-trained models' inherent capabilities [19]

CoT and CoC Prompting: Useful for universal logical reasoning tasks such as solving math problems or answering multi-step questions. CoT/CoC enhances reasoning transparency, enabling models to provide structured solutions to otherwise ambiguous tasks.

2) *Techniques for Specific Tasks*

Specific tasks often involve domain-specific knowledge or custom requirements. Techniques here provide targeted guidance or integrate external resources.

Role-Assigned Prompts: Useful for domain-specific tasks such as legal document analysis or medical diagnosis. Assigning a role (e.g., "You are a medical expert") helps focus the model's output to match specialized expectations.

Formatting constraints are particularly useful when the output must adhere to a specific structure to meet task requirements or integrate with downstream systems. For instance, if the output is intended for use in an API, formatting it as JSON ensures compatibility and seamless processing.

III. EVALUATION OF PROMPT PERFORMANCE

1) *Evaluating the Performance of Language Models*

Evaluation is essential for understanding the effectiveness of prompts in guiding language models (LMs) toward desired outputs. By measuring task execution, reasoning capabilities, and adaptability, benchmarks provide a critical lens for assessing LMs' performance. However, there are currently no specific benchmarks tailored exclusively to evaluate prompts themselves. As a result, researchers rely on existing LM benchmarks to indirectly validate the impact of prompt engineering techniques. While this approach offers

valuable insights, it also highlights the need for more targeted benchmarks in the future.

2) 3.2 Commonly Used Benchmarks

a) Standardized Test-Inspired Benchmarks

Standardized test-inspired benchmarks, such as those modeled on SAT or GRE-style tasks, evaluate a model's reasoning and problem-solving capabilities. These benchmarks include tasks like multistep math problems, logical reasoning, and verbal comprehension, providing insights into how models handle complex cognitive challenges. They also allow for direct comparisons between LM performance and human proficiency in these areas. For example, Chain-of-Thought (CoT) prompting is often tested on SAT-style math problems, showcasing its ability to guide models through structured reasoning processes.

b) Natural Language Understanding Benchmarks

Natural language understanding benchmarks like GLUE and SuperGLUE are foundational for evaluating general NLP capabilities.[20] GLUE assesses tasks such as sentiment analysis, sentence similarity, and textual entailment, offering a baseline for general language understanding. SuperGLUE builds on this by introducing more complex tasks and evaluation metrics, catering to models that surpass the performance levels measured by GLUE.

c) Domain Benchmarks

Domain benchmarks, such as the Massive Multitask Language Understanding (MMLU) benchmark[21], assess the breadth and depth of knowledge across 57 academic subjects, including mathematics, medicine, and law. The newer MMLU-Pro focuses more on reasoning rather than rote knowledge, expanding answer choices from 4 to 10 and improving prompt stability. Similarly, BIG-Bench provides a collaborative evaluation framework with 207 tasks covering a wide array of topics, including linguistics, biology, and social bias. Its subset, BIG-Bench Hard (BBH)[22], narrows the focus to 23 especially challenging tasks designed to test the limits of LM reasoning and commonsense understanding.

d) Holistic Benchmarks

Benchmarks like HELM (Holistic Evaluation of Language Models) [23]go beyond traditional evaluations by examining LMs across multiple dimensions, including accuracy, robustness, fairness, bias, toxicity, and efficiency. HELM evaluates 30 prominent models across 42 scenarios, offering a comprehensive and evolving benchmark that adapts to new challenges. Its multi-dimensional evaluation approach makes it particularly relevant for assessing how regulatory prompts or specific techniques affect LM behavior in complex or sensitive applications.

These benchmarks evaluate language models across diverse tasks, offering insights into the effectiveness of prompt techniques. While not designed specifically for prompts, these benchmarks indirectly validate strategies like Chain-of-Thought reasoning, especially in complex domains. A summary of these benchmarks is shown in Table I

TABLE I. TYPES OF LLM BENCHMARKS

Type of Benchmarks	**What is being measured**	**Example**
Standardized Test	Reasoning and problem-solving capabilities, multistep logic, verbal comprehension, and performance vs. human.	SAT, GRE, GMAT
NLP Benchmarks	General natural language understanding capabilities like sentiment analysis, and textual entailment.	GLUE, SuperGLUE
Domain Performance	Coding, mathematical, breadth and depth of domain knowledge across academic	MMLU, BIG-Bench

	subjects, reasoning over facts.	
Hilistic Benchmarks	Multi-dimensional aspects such as accuracy, robustness, fairness, bias, and efficiency.	HELM

3) 3.3 Gaps and Future Works

While benchmarks discussed in 3.2 provide valuable tools for evaluating language models, they leave gaps in assessing the nuanced impact of prompt engineering. Below, we discuss three key areas requiring further attention.

a) Missing Dimensions in Benchmarking

Alignment is one dimension requires more attention in these benchmarks. It ensures that models produce outputs consistent with human values, ethical principles, and safe practices. Misaligned models risk generating harmful, biased, or otherwise inappropriate content, particularly in sensitive applications such as healthcare or law.

Existing benchmarks should also add a focus point on evaluating the security aspects of language models, assessing their resilience against adversarial prompts that aim for sensitive information extraction or other malice intentions. This could include safeguarding against the disclosure of system instructions, user details, or other technical specifications that should not be provided to users.

b) Evaluating Prompt Effectiveness and Context Management

It's also importtant to evaluate how well language models adhere to user instructions and the factors influencing prompt effectiveness. Models can deviate from instructions, producing outputs that fail to align with user expectations. This deviation is particularly problematic in tasks necessitating precision, such as writing formatted documents, where consistent adherence to instructions is crucial.

Additionally, prompt length also impacts usability, especially in conversational applications. Longer prompts consume more of the model's context window, reducing the space available for user interactions and potentially pushing earlier chat history out of memory. Future evaluations should also address efficiency of prompt designs, balancing complexity and context window usage to ensure optimal performance across diverse applications.

c) Token-Level Entropy as a New Metric

Token-level entropy offers a novel way to evaluate the internal mechanics of model reasoning. [24]By analyzing the information density and variability of outputs at the token level, this metric can provide insights into how well a prompt structures the model's response. For instance, a high-quality Chain-of-Thought prompt should result in a uniform distribution of information across tokens, ensuring that each step contributes meaningfully to the final answer. This concept is demonstrated in Fig. 4. Integrating token-level entropy into benchmarks could bridge the gap between high-level task evaluation and the fine-grained assessment of model reasoning processes.

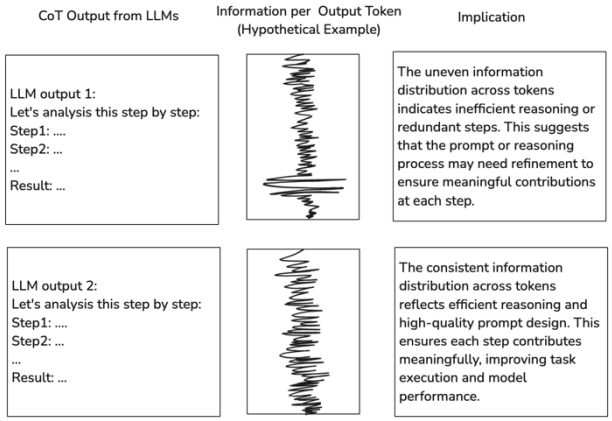

Fig. 4. The Implication of Token-level Entropy on Prompt Effectiveness

Addressing these gaps requires expanding existing benchmarks and developing new frameworks that complement them. By incorporating cross-domain testing, prompt adherence metrics, and token-level analysis, future evaluations can provide a more holistic

and nuanced understanding of both language models and the prompts that guide them.

IV. CONCLUSION

Prompt engineering is a transformative technique that bridges the gap between the generic capabilities of large language models (LLMs) and the specific needs of diverse tasks. This paper has explored the evolving landscape of prompt engineering, highlighting its modular techniques, the intersection of universality and specificity, and its critical role in maximizing LLM potential. While existing benchmarks do not directly evaluate prompts, they can still provide valuable insights into prompt evaluation with some limitations.

Addressing these limitations requires future works on evaluation metrics tailored to prompt engineering, such as token-level entropy and prompt effectiveness and adherence testing. By advancing evaluation methodologies and refining prompt techniques, the field can move beyond empirical practices, ensuring LLMs deliver not only high performance but also ethical and reliable outputs for real-world applications.

REFERENCES

[1] "Getting started with LLM prompt engineering." Jul. 2024. Accessed: Sep. 30, 2024. [Online]. Available: https://learn.microsoft.com/en-us/ai/playbook/technology-guidance/generative-ai/working-with-llms/prompt-engineering

[2] P. Sahoo, A. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, "A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications," Feb. 05, 2024, Cornell University. doi: 10.48550/arXiv.2402.

[3] Z. Chu et al., "A Survey of Chain of Thought Reasoning: Advances, Frontiers and Future," Jan. 01, 2023, Cornell University. doi: 10.48550/arXiv.2309.

[4] O. Fagbohun, R. M. Harrison, and A. Dereventsov, "An Empirical Categorization of Prompting Techniques for Large Language Models: A Practitioner's Guide," Feb. 18, 2024, Cornell University. doi: 10.48550/arxiv.2402.14837.

[5] Wordware, "Twitter Personality." [Online]. Available: https://twitter.wordware.ai/?ref=blog.wordware.ai

[6] J. White et al., "A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2302.11382.

[7] "Creating ChatGPT Prompts: A Framework." Sep. 28, 2024. [Online]. Available: https://github.com/mattnigh/ChatGPT3-Free-Prompt-List?tab=readme-ov-file#creating-chatgpt-prompts-a-framework

[8] S. Sivarajkumar, M. Kelley, A. Samolyk-Mazzanti, S. Visweswaran, and Y. Wang, "An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2309.08008.

[9] T. Schick and H. Schütze, "True Few-Shot Learning with Prompts -- A Real-World Perspective," Jan. 01, 2021, Cornell University. doi: 10.48550/arxiv.2111.13440.

[10] J. Lee et al., "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," Jan. 01, 2022, Cornell University. doi: 10.48550/arxiv.2201.11903.

[11] S. Yao et al., "Tree of Thoughts: Deliberate Problem Solving with Large Language Models," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2305.10601.

[12] C. Li et al., "Chain of Code: Reasoning with a Language Model-Augmented Code Emulator," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2312.04474.

[13] S. Avin, R. Gruetzemacher, and J. Fox, "Exploring AI Futures Through Role Play," Feb. 05, 2020. doi: 10.1145/3375627.3375817.

[14] X. Tang et al., "Struc-Bench: Are Large Language Models Really Good at Generating Complex

Structured Data?," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2309.08963.

[15] A. Madaan et al., "Self-Refine: Iterative Refinement with Self-Feedback," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2303.17651.

[16] H. Lautraite, N. Naji, L. Marceau, M. Queudot, and É. Charton, "Multi-stage Clarification in Conversational AI: The case of Question-Answering Dialogue Systems," Jan. 01, 2021, Cornell University. doi: 10.48550/arxiv.2110.15235.

[17] Y. Gao et al., "Retrieval-Augmented Generation for Large Language Models: A Survey," Jan. 01, 2023, Cornell University. doi: 10.48550/arxiv.2312.10997.

[18] S. Teo, "How I Won Singapore's GPT-4 Prompt Engineering Competition." Dec. 28, 2023. [Online]. Available: https://towardsdatascience.com/how-i-won-singapores-gpt-4-prompt-engineering-competition-34c195a93d41

[19] V. Sanh et al., "Multitask Prompted Training Enables Zero-Shot Task Generalization," Jan. 01, 2021, Cornell University. doi: 10.48550/arxiv.2110.08207.

[20] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding," Jan. 01, 2018, Cornell University. doi: 10.48550/arxiv.1804.07461.

[21] "cais/mmlu." Feb. 2024. Accessed: Nov. 30, 2024. [Online]. Available: https://huggingface.co/datasets/cais/mmlu

[22] "BIG-bench/bigbench/benchmark_tasks at main · google/BIG-bench." Jan. 2021. Accessed: Sep. 30, 2024. [Online]. Available: https://github.com/google/BIG-bench/tree/main/bigbench/benchmark_tasks

[23] R. Bommasani, P. Liang, and T. Lee, "Holistic Evaluation of Language Models," May 25, 2023, Wiley. doi: 10.1111/nyas.15007.

[24] A. Kumar, R. Morabito, S. Umbet, J. Kabbara, and A. Emami, "Confidence Under the Hood: An Investigation into the Confidence-Probability Alignment in Large Language Models," May 25, 2024, Cornell University. doi: 10.48550/arxiv.2405.16282..