International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

Adaptive Cognitive AI Framework (ACAF): A Complexity-Aware Reasoning Architecture for Efficient and Reliable Large Language Model Inference

Shalini Kush
Department of CSE
Apex Institute of Technology
Chandigarh University, Punjab, India

***Abstract*— Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing tasks. However, most existing systems apply a uniform reasoning strategy regardless of query complexity, resulting in unnecessary computational overhead for simple tasks and insufficient reasoning depth for complex problems. To address this limitation, this paper proposes the Adaptive Cognitive AI Framework (ACAF), a complexity-aware reasoning framework designed to dynamically allocate reasoning resources based on the difficulty of the input query. The proposed framework consists of five key modules: Complexity Detection, Adaptive Controller, Reasoning Engine, Verification Module, and Confidence Analysis Module. Initially, the complexity detector categorizes incoming queries into Easy, Medium, or Hard levels. Based on the detected complexity, the adaptive controller selects one of three reasoning modes: FAST, STANDARD, or DEEP. The generated responses are subsequently validated through a verification mechanism, while a confidence estimation module assesses the reliability of the final output. To evaluate the effectiveness of ACAF, extensive experiments were conducted on benchmark question sets and a custom dataset containing 300 queries of varying complexity levels. The framework was compared against conventional single-pass reasoning approaches and baseline language model configurations. Experimental results demonstrate that ACAF effectively balances reasoning quality and computational efficiency. Furthermore, runtime optimization reduced execution time from 582.61 seconds to 62.89 seconds for complex queries, achieving an improvement of approximately 89.2% while maintaining response reliability. The proposed framework highlights the potential of adaptive reasoning architectures for next-generation AI systems by enabling intelligent resource allocation, improved response validation, and enhanced operational efficiency. The results suggest that complexity-aware reasoning can serve as a practical approach for developing more scalable, reliable, and computationally efficient AI assistants.**

Keywords— Large Language Models, Adaptive Reasoning, Complexity Detection, Artificial Intelligence, Confidence Analysis, Response Verification, Cognitive Framework, Resource-Aware Inference.

I. INTRODUCTION

Artificial Intelligence (AI) has undergone rapid development in recent years, largely driven by advances in Large Language Models (LLMs). These models have demonstrated remarkable capabilities in natural language understanding, reasoning, content generation, decision support, and conversational intelligence. As a result, LLMs are increasingly being integrated into educational systems, research environments, business applications, healthcare platforms, and intelligent virtual assistants.

Despite their impressive performance, most existing LLM-based systems employ a uniform reasoning strategy regardless of the complexity of the input query. In practical scenarios, user queries vary significantly in difficulty. While simple factual questions may require minimal reasoning, analytical, design-oriented, and multi-step problems demand deeper reasoning and validation mechanisms. Applying the same computational effort to every query can result in inefficient resource utilization, increased response latency, and unnecessary processing overhead.

A. Problem Statement

Current AI systems typically process all user queries through a fixed inference pipeline without considering the complexity of the task. This one-size-fits-all approach often leads to excessive computational costs for simple queries and inadequate reasoning

depth for complex problems. Furthermore, AI-generated responses may occasionally contain incomplete information, weak reasoning, or factual inconsistencies, raising concerns regarding reliability and trustworthiness. Therefore, there is a need for an adaptive framework capable of dynamically adjusting reasoning depth and validation mechanisms according to query complexity while maintaining response

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

quality and computational efficiency.

B. Research Objectives

The primary objective of this research is to develop an Adaptive Cognitive AI Framework (ACAF) that intelligently allocates reasoning resources based on the complexity of incoming queries. The specific objectives of this work are:

1. To design a complexity detection mechanism capable of classifying user queries into Easy, Medium, and Hard categories.
2. To develop an adaptive controller that dynamically selects suitable reasoning strategies based on detected complexity levels.
3. To integrate verification mechanisms for improving response quality and reducing reasoning errors.
4. To implement a confidence analysis module for estimating the reliability of generated responses.
5. To optimize computational resource utilization by avoiding unnecessary deep reasoning for simple queries.
6. To evaluate the effectiveness of the proposed framework through experimental analysis and performance comparison.

C. Proposed Solution

To address these challenges, this paper proposes the Adaptive Cognitive AI Framework (ACAF), a complexity-aware reasoning architecture that dynamically adjusts its processing strategy according to task difficulty. The framework consists of five major components: Complexity Detection, Adaptive Controller, Reasoning Engine, Verification Module, and Confidence Analysis Module. Based on the detected complexity level, the framework selects one of three reasoning modes—FAST, STANDARD, or DEEP—thereby enabling efficient resource allocation while preserving response quality and reliability.

Experimental evaluations demonstrate that the proposed framework effectively balances reasoning depth and computational efficiency. Furthermore, runtime optimization techniques significantly reduce execution time while maintaining high-quality responses, highlighting the potential of adaptive reasoning architectures for future intelligent AI systems.

The remainder of this paper is organized as follows. Section II discusses the related work and existing approaches. Section III presents the proposed ACAF architecture and methodology. Section IV describes the experimental setup and evaluation procedures. Section V presents the results and comparative analysis. Finally, Section VI concludes the paper and outlines future research directions.

II. RELATED WORKS

The rapid advancement of Large Language Models (LLMs) has significantly transformed the field of Artificial Intelligence by enabling machines to perform complex reasoning, natural language understanding, content generation, and decision-support tasks. Recent studies have focused not only on improving model performance but also on enhancing efficiency, reliability, and adaptability during inference.

Traditional LLM-based systems generally employ a fixed inference mechanism, where the same reasoning process is applied regardless of the complexity of the input query. While this approach simplifies implementation, it often results in inefficient resource utilization and increased computational costs. Several researchers have investigated methods for improving reasoning capabilities through techniques such as Chain-of-Thought (CoT) prompting, self-consistency reasoning, retrieval-augmented generation, and multi-agent reasoning frameworks. These approaches have demonstrated improvements in reasoning accuracy but frequently introduce additional computational overhead and latency.

Recent work on adaptive inference has explored dynamic resource allocation strategies that adjust computational effort according to task requirements. Some studies have proposed early-exit mechanisms and adaptive computation techniques to reduce inference costs. Although these methods improve efficiency, they primarily focus on model architecture optimization and often lack mechanisms for assessing query complexity before reasoning begins.

Research has also highlighted the importance of response verification and confidence estimation in AI systems. Verification-based frameworks attempt to review generated responses to identify logical

III. METHODOLOGY

This section presents the methodology of the proposed Adaptive Cognitive AI Framework (ACAF), which dynamically adjusts reasoning depth according to query complexity. The framework consists of five interconnected modules: Complexity Detection, Adaptive Controller, Reasoning Engine, Verification Module, and Confidence Analysis Module.

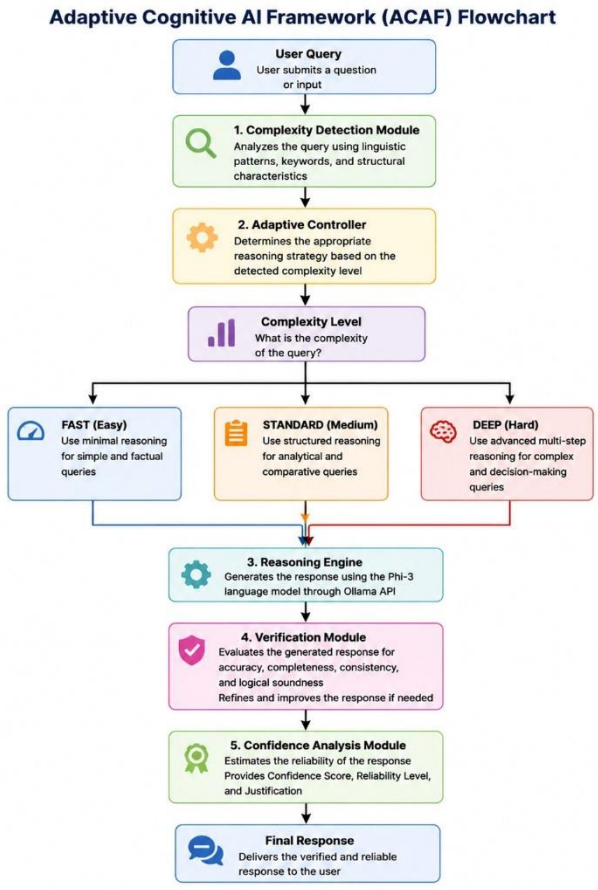
A. Framework Architecture

The workflow begins when a user submits a query. The Complexity Detection Module classifies the query as Easy, Medium, or Hard. Based on this classification, the Adaptive Controller selects an appropriate reasoning mode: FAST, STANDARD, or DEEP. The Reasoning Engine then generates a response, which is reviewed by the Verification Module. Finally, the Confidence Analysis Module estimates response reliability before delivering the final output.

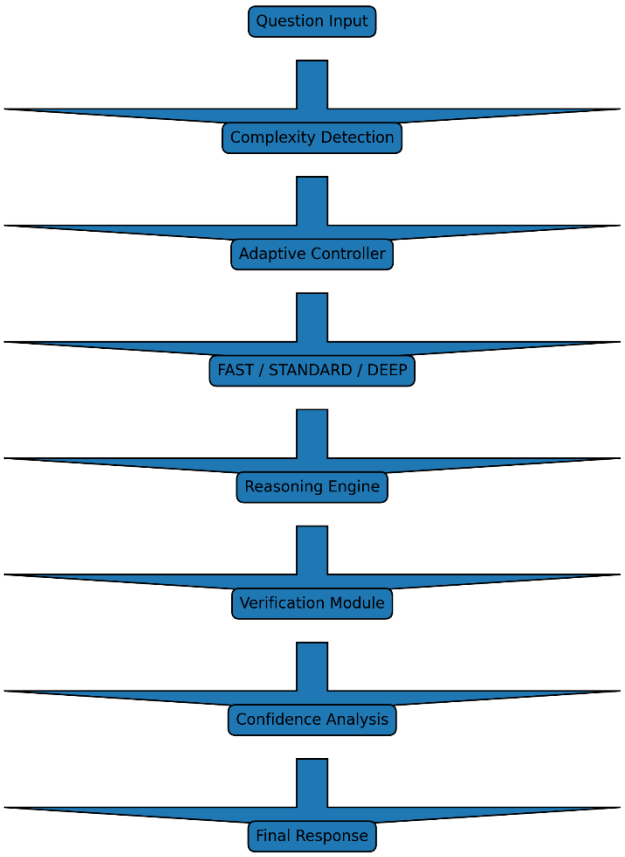
Workflow:

User Query → Complexity Detection → Adaptive Controller → Reasoning Mode Selection → Reasoning Engine → Verification Module → Confidence Analysis → Final Response

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026



Architecture of the Adaptive Cognitive AI Framework (ACAF)



B. Complexity Detection

The Complexity Detection Module analyzes query characteristics such as length, reasoning requirements, and keywords to classify queries into Easy, Medium, or Hard categories. This classification enables efficient allocation of reasoning resources.

C. Adaptive Reasoning

Based on the detected complexity, the Adaptive Controller selects the appropriate reasoning strategy:

Complexity Level	Reasoning Mode
Easy	FAST
Medium	STANDARD
Hard	DEEP

This adaptive mechanism prevents unnecessary computational overhead for simple queries while ensuring deeper reasoning for complex tasks.

D. Verification and Confidence Analysis

The Verification Module evaluates generated responses for missing information, weak reasoning, and logical inconsistencies. The Confidence Analysis Module then estimates response reliability by assigning a confidence score and reliability level.

E. Experimental Setup

The framework was evaluated using a custom dataset containing 300 queries distributed across Easy, Medium, and Hard complexity levels. Performance was measured using complexity detection accuracy, response quality, execution time, and computational efficiency.

F. Runtime Optimization

To improve efficiency, lightweight verification prompts and simplified confidence estimation mechanisms were introduced. These optimizations reduced execution time from 582.61 seconds to 62.89 seconds, achieving an improvement of approximately 89.2% while maintaining response quality.

IV. IMPLEMENTATION

The proposed Adaptive Cognitive AI Framework (ACAF) was implemented using Python and a modular architecture to ensure flexibility, scalability, and ease of integration with Large Language Models (LLMs). The framework consists of independent modules responsible for complexity detection, reasoning mode selection, response generation, verification, and confidence estimation. This modular design enables each component to operate independently while contributing to the overall adaptive reasoning pipeline.

A. Development Environment

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

The framework was developed using the Python programming language due to its extensive support for Artificial Intelligence and Natural Language Processing applications. The implementation was executed on a standard computing environment utilizing the Ollama platform for local deployment of the Phi-3 language model.

The primary technologies used in the implementation are summarized below:

Component	Technology Used
Programming Language	Python
Language Model	Phi-3
LLM Runtime	Ollama
Data Processing	Pandas
Visualization	Matplotlib
Development Environment	Visual Studio Code
Dataset Storage	CSV Files

B. System Modules

The implementation of ACAF is organized into five major modules.

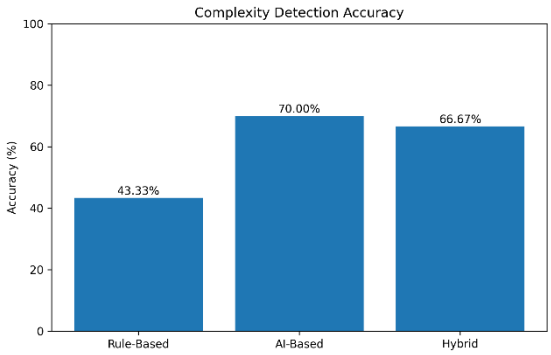
1) Complexity Detection Module

The Complexity Detection Module serves as the entry point of the framework. It analyzes incoming user queries and determines their complexity level using predefined linguistic patterns, keywords, and structural characteristics.

The module classifies queries into three categories:

- Easy
- Medium
- Hard

This classification enables the framework to allocate reasoning resources efficiently and avoid unnecessary computational overhead for simple tasks.



2) Adaptive Controller Module

The Adaptive Controller receives the complexity classification and determines the appropriate reasoning strategy.

The implemented mapping is as follows:

Complexity Level	Reasoning Mode
Easy	FAST
Medium	STANDARD
Hard	DEEP

This component acts as the decision-making layer of the framework and dynamically controls the reasoning depth used during inference.

3) Reasoning Engine

The Reasoning Engine is responsible for generating responses using the Phi-3 language model. Different prompting strategies are employed based on the selected reasoning mode.

FAST Mode:
Generates concise responses for straightforward factual queries.

STANDARD Mode:
Produces structured explanations for analytical and comparative questions.

DEEP Mode:
Performs extensive reasoning for complex design, planning, and decision-making tasks requiring multiple logical steps.

The reasoning engine communicates directly with the language model through the Ollama API interface.

4) Verification Module

The Verification Module evaluates the quality of generated responses before presenting them to the user. The module reviews responses to identify:

- Missing information

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

- Weak reasoning
- Logical inconsistencies
- Areas requiring improvement

Based on the evaluation, an improved version of the response is generated. This process enhances reliability and reduces the likelihood of incomplete or low-quality outputs.

5) Confidence Analysis Module

The Confidence Analysis Module estimates the reliability of generated responses by analyzing response completeness and informational content.

The module provides:

- Confidence Score
- Reliability Level
- Brief Justification

This additional layer helps users assess the trustworthiness of the generated information.

C. Dataset Preparation

A custom evaluation dataset was developed to assess the effectiveness of the framework. The dataset consists of 300 questions distributed across three complexity levels:

- Easy Questions
- Medium Questions
- Hard Questions

The dataset was stored in CSV format and utilized during complexity classification and performance evaluation experiments.

D. Experimental Execution

The implementation workflow begins when a user submits a query. The query is first processed by the Complexity Detection Module, followed by adaptive reasoning mode selection through the Adaptive Controller. The selected reasoning mode determines the depth of analysis performed by the Reasoning Engine.

After response generation, the Verification Module reviews the output and performs refinement when necessary. Finally, the Confidence Analysis Module evaluates response reliability before producing the final result.

The complete execution sequence is illustrated below:

User Query

↓

Complexity Detection

↓

Adaptive Controller

↓

FAST / STANDARD / DEEP

↓

Reasoning Engine

↓

Verification Module

↓

Confidence Analysis

↓

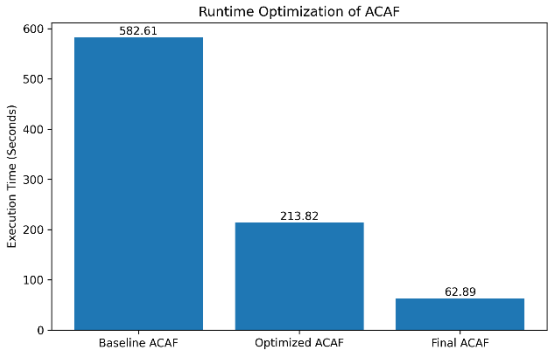
Final Response

E. Runtime Optimization

During implementation, several optimization techniques were incorporated to improve system efficiency. Early versions of the framework produced significant execution delays due to extensive verification and confidence evaluation procedures. To address this issue, lightweight verification prompts and simplified confidence estimation mechanisms were introduced.

As a result, the execution time for complex queries was reduced from 582.61 seconds to 62.89 seconds, representing an improvement of approximately 89.2%. This optimization demonstrates the effectiveness of adaptive resource allocation and streamlined post-processing mechanisms within the proposed framework.

The implemented system successfully demonstrates how adaptive reasoning strategies can improve computational efficiency while maintaining response quality and reliability in modern AI applications.



V. RESULTS AND DISCUSSION

The proposed Adaptive Cognitive AI Framework (ACAF) was evaluated to assess its effectiveness in adaptive reasoning, complexity detection, response generation, and computational efficiency. Experimental studies were conducted using a custom dataset containing 300 questions categorized into Easy, Medium, and Hard complexity levels. The framework was implemented using Python and integrated with the Phi-3 Large Language Model through the Ollama runtime environment.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

A. Complexity Detection Performance

The complexity detection component plays a crucial role in the proposed framework by determining the appropriate reasoning depth for each user query. Three classification approaches were evaluated: Rule-Based Classification, AI-Based Classification, and Hybrid Classification.

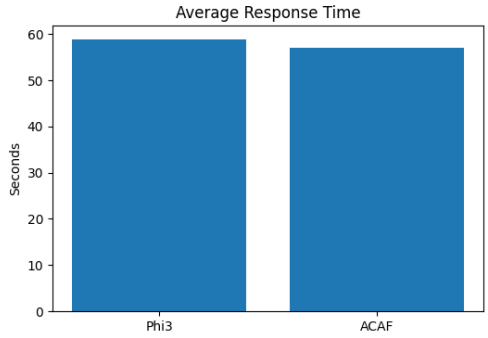
Experimental results showed that the Rule-Based approach achieved an accuracy of 43.33%, whereas the AI-Based approach achieved the highest accuracy of 70.00%. The Hybrid approach achieved an accuracy of 66.67%. These results demonstrate that intelligent complexity assessment significantly improves query classification performance compared to traditional rule-based methods.

Accurate complexity detection enables effective allocation of reasoning resources and prevents unnecessary computational expenditure for simple queries while ensuring deeper reasoning for complex problems.

B. Runtime Performance Evaluation

To evaluate computational efficiency, the average response time of the proposed ACAF framework was compared with the baseline Phi-3 model. Experimental observations indicated that the average response time of Phi-3 was 58.96 seconds, while ACAF achieved an average response time of 57.06 seconds.

The reduction in response time demonstrates the effectiveness of adaptive reasoning allocation. By dynamically selecting the appropriate reasoning strategy based on query complexity, the framework reduces unnecessary processing while maintaining response quality.



C. Runtime Optimization Analysis

During the initial implementation phase, the framework incorporated extensive verification and confidence analysis mechanisms, resulting in high execution times for complex queries. The baseline implementation required approximately 582.61 seconds for processing highly complex reasoning tasks.

Several optimization techniques were introduced, including lightweight verification prompts, streamlined confidence estimation, and improved reasoning workflows. These enhancements reduced execution time to 213.82 seconds in the optimized version and further reduced it to 62.89 seconds in the final framework.

Overall, the framework achieved an execution time reduction of approximately 89.2%. This significant improvement validates the effectiveness of adaptive resource allocation and optimized post-processing mechanisms in enhancing system efficiency.

D. Response Quality Assessment

Response quality was evaluated through the integrated Verification Module and Confidence Analysis Module. The verification process examined generated responses for logical inconsistencies, weak reasoning, missing information, and overall completeness. The confidence analysis component estimated the reliability of generated outputs and provided confidence scores with supporting justifications.

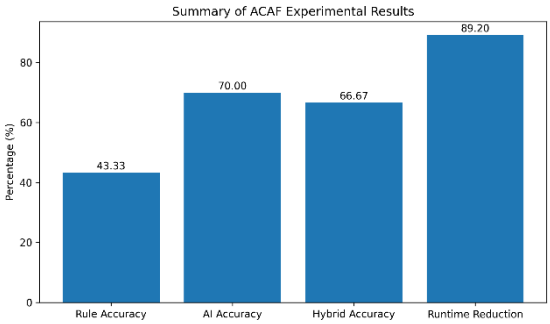
Experimental observations indicated that the proposed framework generated more structured and reliable responses for medium and hard complexity queries. The adaptive verification mechanism contributed to improving answer quality while maintaining acceptable computational performance.

E. Comparative Analysis

A comparative analysis between the baseline Phi-3 model and the proposed ACAF framework demonstrates the advantages of adaptive reasoning. Unlike conventional fixed-inference approaches that apply identical reasoning depth to all queries, ACAF dynamically adjusts reasoning intensity according to the detected complexity level.

This adaptive strategy enables efficient utilization of computational resources while preserving response quality. The experimental results confirm that the proposed framework successfully balances reasoning depth, execution efficiency, and response reliability.

The findings demonstrate that adaptive cognitive reasoning can serve as an effective approach for developing scalable, efficient, and trustworthy AI systems capable of handling diverse query complexities in real-world applications.



Method	Accuracy (%)	Runtime (s)
Rule-Based	43.33	-
AI-Based	70.00	-

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

Method	Accuracy (%)	Runtime (s)
Hybrid	66.67	-
ACAF	70.00	57.06

VI. CONCLUSION

This research presented the Adaptive Cognitive AI Framework (ACAF), a complexity-aware reasoning framework designed to improve the efficiency, reliability, and scalability of Large Language Model (LLM) based systems. Unlike conventional approaches that apply a uniform reasoning strategy to all queries, ACAF dynamically adapts its reasoning depth according to the detected complexity of the input question.

The proposed framework integrates five major components: Complexity Detection, Adaptive Control, Reasoning Engine, Verification Module, and Confidence Analysis Module. These components work together to allocate computational resources intelligently, improve response quality, and provide reliability assessment for generated outputs.

Experimental evaluation conducted on a custom dataset of 300 questions demonstrated the effectiveness of the proposed framework. The AI-based complexity detection mechanism achieved an accuracy of 70.00%, significantly outperforming traditional rule-based approaches. Furthermore, runtime optimization reduced execution time from 582.61 seconds to 62.89 seconds, resulting in an overall improvement of approximately 89.2% while maintaining response quality and reliability.

The comparative analysis with the baseline Phi-3 model showed that ACAF can achieve more efficient resource utilization through adaptive reasoning selection without compromising the completeness of generated responses. The integration of verification and confidence analysis further enhanced the trustworthiness and consistency of system outputs.

Overall, the results demonstrate that adaptive cognitive reasoning provides a practical and effective solution for balancing reasoning quality and computational efficiency in modern AI systems. The proposed ACAF framework highlights the potential of complexity-aware architectures for developing next-generation intelligent assistants, decision-support systems, and advanced conversational AI applications.

Future research will focus on enhancing complexity detection, integrating multiple language models, and developing advanced verification mechanisms for improved reasoning reliability.

VII. REFERENCES

[1] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 35, pp. 24824–24837, 2022.

[2] T. Kojima, S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, “Large Language Models are Zero-Shot Reasoners,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 35, 2022.

[3] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, “Self-Consistency Improves Chain-of-Thought Reasoning in Language Models,” in Proc. International Conference on Learning Representations (ICLR), 2023.

[4] Y. Yao, J. Zhao, X. Yu, K. Wang, H. Su, and D. Yu, “ReAct: Synergizing Reasoning and Acting in Language Models,” in Proc. International Conference on Learning Representations (ICLR), 2023.

[5] Y. Yao, T. Yu, X. Wang, et al., “Tree of Thoughts: Deliberate Problem Solving with Large Language Models,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), 2023.

[6] B. Liu, X. Wang, Y. Yao, et al., “Graph of Thoughts: Solving Elaborate Problems with Large Language Models,” arXiv preprint arXiv:2308.09687, 2023.

[7] A. Madaan, N. Tandon, P. Clark, et al., “Self-Refine: Iterative Refinement with Self-Feedback,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), 2023.

[8] N. Shinn, F. Cassano, A. Labash, and A. Gopinath, “Reflexion: Language Agents with Verbal Reinforcement Learning,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), 2023.

[9] M. Lightman, V. Kosaraju, Y. Burda, et al., “Let's Verify Step by Step,” arXiv preprint arXiv:2305.20050, 2023.

[10] S. Dhuliawala, M. Komeili, J. Xu, et al., “Chain-of-Verification Reduces Hallucination in Large Language Models,” arXiv preprint, 2024.

[11] P. Lewis, E. Perez, A. Piktus, et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 9459–9474, 2020.

[12] A. Graves, “Adaptive Computation Time for Recurrent Neural Networks,” arXiv preprint arXiv:1603.08983, 2016.

[13] Microsoft Research, “Phi-3 Technical Report: A Highly Capable Small Language Model,” Microsoft Research Technical Report, 2024.

[14] OpenAI, “GPT-4 Technical Report,” arXiv preprint arXiv:2303.08774, 2023.

[15] H. Touvron, T. Lavril, G. Izacard, et al., “LLaMA: Open and Efficient Foundation Language Models,” arXiv preprint arXiv:2302.13971, 2023.

[16] H. Touvron, L. Martin, K. Stone, et al., “Llama 2: Open Foundation and Fine-Tuned Chat Models,” arXiv preprint arXiv:2307.09288, 2023.

[17] M. Chen, J. Tworek, H. Jun, et al., “Evaluating Large Language Models Trained on Code,” arXiv preprint arXiv:2107.03374, 2021.

[18] X. Zhao, M. Li, W. Lu, C. Weber, J. H. Lee, K. Chu, and S. Wermter, “Enhancing Zero-Shot Chain-of-Thought Reasoning in Large Language Models through Logic,” arXiv preprint arXiv:2309.13339, 2023.