Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Active Abstract Finding for Generalization in LLM-Based AI Systems

Hong Su

suguest@126.com

Chengdu University of Information Technology https://orcid.org/0000-0003-2750-7808

Research Article

Keywords: Large Language Models, Abstraction Learning, Active Observation, Method Generalization

Posted Date: March 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9078366/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Active Abstract Finding for Generalization in LLM-Based AI Systems

Hong Su

***Abstract*—Identifying key factors that determine whether a method can be applied across different scenarios is important for improving the generalization ability of large language model (LLM)-based agents. However, existing approaches typically rely on passive observation or static datasets, making it difficult to discover the underlying abstract structures that enable methods to transfer across diverse contexts.**

In this paper, we propose an *active abstract finding* framework that enables AI systems to discover common factors across different observations and derive generalized reasoning structures. The proposed framework models abstraction as a process of identifying shared factors among observations while removing instance-specific factors. To improve the probability of discovering such structures, the AI system actively collects observations from different times, places, and contexts, and performs similarity comparison to extract common factors. The discovered abstractions are then represented symbolically and can be reused to derive reasoning methods applicable to new scenarios.

Experimental results demonstrate that abstraction-oriented observation strategies improve the ability of LLMs to identify method applicability. In our experiments, the proposed strategy achieves an average accuracy of 0.6417**, outperforming the random observation baseline** (0.5083) **and the place-diverse strategy** (0.6167)**, confirming the effectiveness of active abstraction in identifying key factors for generalization.**

***Index Terms*—Large Language Models, Abstraction Learning, Active Observation, Method Generalization**

I. Introduction

Large language models (LLMs) have demonstrated strong capabilities in reasoning, problem solving, and knowledge generation [1] [2]. When deployed as AI systems in dynamic environments, LLM-based systems often need to determine whether a previously learned method can be applied to a new object or scenario. This requires the ability to identify the key factors that determine the applicability of a method. Such a capability is essential for generalization, since real-world environments typically contain large variations across objects, contexts, and situations.

Humans often address this challenge through abstraction. By observing multiple instances of similar phenomena, people identify common structures while ignoring instance-specific details. The resulting abstractions allow a method discovered in one context to be applied to other related situations. In contrast, many current LLM-based systems rely primarily on passive observations or static datasets, which limits their ability to discover such shared structures. When observations lack diversity or when key factors are hidden among many irrelevant features, it becomes difficult for the system to correctly determine whether a method should be applied.

To address this challenge, we propose an *active abstract finding* framework for LLM-based agents. The core idea is to model abstraction as the process of identifying shared factors across multiple observations while removing instance-specific factors. Instead of relying on randomly available observations, the proposed framework enables the AI system to actively collect observations across different places, times, and contexts. These diverse observations increase the likelihood that irrelevant factors vary while the essential factors remain stable, making it easier to identify the underlying structure that determines method applicability.

The proposed framework also introduces a symbolic representation mechanism for abstractions. Once common factors are identified, they can be represented as symbolic structures that simplify reasoning and allow higher-level abstractions to be constructed from lower-level ones. This hierarchical abstraction process resembles human reasoning, where complex concepts are gradually built from simpler patterns.

The main contributions of this paper are summarized as follows:

- We propose an *active abstract finding* framework that enables LLM-based AI systems to identify shared factors across observations and generalize methods, while allowing the AI system to autonomously collect diverse information from the environment.
- We introduce an abstraction mechanism that separates instance-specific factors from shared factors and represents the resulting structures symbolically, enabling hierarchical abstraction and reusable reasoning structures.
- We design observation collection strategies that actively increase contextual and informational diversity by continuously collecting observations across different places and time periods, thereby facilitating abstraction discovery.

The remainder of this paper is organized as follows. Section II reviews related work on abstraction and reasoning in AI systems. Section IV presents the experimental setup and verification results. Finally, Section V concludes the paper and discusses future directions.

II. Related Work

Research related to this work can be broadly categorized into three areas: reasoning and decision-making in large lan-

H. Su is with the School of Computer Science, Chengdu University of Information Technology, Chengdu, China.
E-mail: suguest@126.com.

guage model (LLM) based agents, abstraction and representation learning in artificial intelligence, and active data collection for improving learning and generalization.

A. LLM-Based Agents and Reasoning

Large language models have recently been extended from text generation systems to interactive agents capable of reasoning, planning, and tool usage. Frameworks such as ReAct [3] integrate reasoning and action by allowing LLMs to interact with external environments through iterative reasoning steps. Similarly, tool-augmented LLM systems enable models to perform complex tasks by invoking external functions or tools during reasoning [4]. Other works explore multi-step reasoning through chain-of-thought prompting [5] and self-reflection mechanisms [6].

These approaches primarily focus on improving reasoning processes during task execution. However, they typically assume that relevant observations are already available and do not explicitly address how agents should actively collect diverse observations to discover the key factors that determine whether a method can be applied to new scenarios. In contrast, the framework proposed in this paper emphasizes the role of observation diversity and active exploration in enabling abstraction discovery.

B. Abstraction and Representation Learning

Abstraction has long been studied in artificial intelligence as a mechanism for reducing complexity and enabling generalization. Classical symbolic AI research explored abstraction through hierarchical planning and symbolic representation of knowledge. In machine learning, representation learning methods aim to extract latent features that capture essential structures in data [7]. More recently, several studies have explored abstraction capabilities in neural networks and LLMs, including concept learning and structured reasoning [8].

While these approaches focus on learning abstract representations from data, they often rely on large datasets and offline training processes. The proposed abstract-finding framework differs in that it models abstraction as an interactive process performed by an AI agent. Instead of passively learning representations from static datasets, the agent actively collects observations and identifies common factors across them.

C. Active Learning and Exploration

Active learning and exploration strategies have been widely studied as methods for improving learning efficiency. In active learning, algorithms select informative samples to reduce uncertainty during model training [9]. In reinforcement learning, exploration strategies enable agents to discover useful information about the environment [10]. Recent works have also explored curiosity-driven exploration mechanisms for improving representation learning [11].

Although these methods emphasize the importance of exploration, their primary goal is usually to improve prediction accuracy or policy learning. In contrast, the proposed framework focuses on exploration for abstraction discovery. The system actively collects observations across different contexts to identify shared factors that determine method applicability. This perspective treats abstraction as the central objective of exploration rather than as a byproduct of model training.

In summary, existing research has explored reasoning in LLM-based agents, abstraction in representation learning, and exploration in active learning. However, the interaction between observation diversity, abstraction discovery, and method applicability has received limited attention. The proposed active abstract-finding framework aims to bridge this gap by enabling AI systems to actively collect diverse observations and discover the key factors that support generalization across scenarios.

III. Active Abstract Finding for Generalization in LLM-Based Agents

Humans often extend the applicability of a method through abstraction. Instead of relying solely on concrete solutions for individual problems, people actively collect observations from different situations, times, and locations, and then identify common relationships or patterns among these observations. By comparing objects obtained from diverse contexts, humans can remove instance-specific details and extract higher-level structures. Such abstraction enables a method discovered in one context to be applied across multiple related scenarios.

Large language model (LLM) based systems can adopt a similar capability. Rather than treating each task independently, an AI system can identify common structures across different observations and generalize them into reusable reasoning patterns. This capability is particularly important in complex environments where knowledge must be transferred across different objects, contexts, or situations.

In this paper, we propose a framework that enables AI systems to discover such common structures and apply them across broader scenarios. We refer to this process as *abstract finding*. An AI system capable of performing this process is referred to as an *abstract-finding AI system*. Through abstract finding, the system actively collects objects from different times, places, and environments, compares their characteristics, and derives generalized structures that support wider applicability of methods or relationships.

a) Definition of Abstract Finding: **Definition 1 (Abstract Finding).** Let $\{A_1, A_2, ..., A_n\}$ denote a set of observed objects or scenarios collected from different contexts (e.g., different times, locations, or environments). Each observation is represented as a set of factors

$$A_i = \{F_i, C\}, \quad i = 1, ..., n \tag{1}$$

where F_i denotes instance-specific factors and C denotes factors shared across observations.

Abstract finding is the process of identifying the shared factors C while removing the instance-specific components F_i. The resulting abstraction

$$A = C \tag{2}$$

represents a generalized structure that enables a method, relationship, or reasoning pattern to be applied across multiple scenarios.

An abstract-finding AI system performs this process by actively collecting observations, comparing similarities across objects obtained from different times and places, extracting common factors, and verifying the resulting abstractions through interaction and feedback.

In practice, observations collected from different places or time points tend to contain different instance-specific factors with high probability. Let p denote the probability that two observations share the same instance-specific factor. When observations are collected from different contexts, the probability of obtaining different instance-specific factors increases:

$$P(F_i \neq F_j) = 1 - p \tag{3}$$

As the number of diverse observations increases, the intersection of common factors becomes easier to identify:

$$C = \bigcap_{i=1}^{n} A_i \tag{4}$$

Therefore, collecting observations from different times and locations improves the probability of correctly identifying the shared structure C. This observation motivates the need for an *active abstraction process*, where the system intentionally *performs actions to gather diverse observations* and then does the abstraction instead of relying only on randomly available data.

A. Active Abstraction

In many real-world situations, only a single instance of an object or scenario may be observed at a given time. However, abstraction generally requires multiple observations in order to identify common structures. Therefore, an AI system needs to simulate the way humans perform abstraction by actively collecting observations across different situations.

Unlike passive learning systems that rely on externally provided datasets, an abstract-finding AI system can actively interact with its environment. Through its own actions, the system may intentionally explore different situations, observe the resulting changes, and collect new objects or scenarios that contribute to the abstraction process. This active interaction enables the system to discover relationships that may not appear in static datasets.

Let s_t denote the system state at time t, and a_t denote an action performed by the AI system. Interaction with the environment produces a new observation

$$o_{t+1} = g(s_t, a_t, e_t) \tag{5}$$

where e_t represents environmental conditions. The resulting observation can be represented as

$$A_{t+1} = \{F_{t+1}, C_{t+1}\} \tag{6}$$

where F_{t+1} contains instance-specific factors and C_{t+1} contains candidate shared factors.

Algorithm 1: Abstract Finding Process

1: Collect observations $\{A_1, A_2, ..., A_n\}$ from different times, spaces, or objects
2: Represent each observation as factor set $A_i = \{F_i, C_i\}$
3: Compare observations to detect shared factors C
4: Remove instance-specific factors F_i to derive abstraction $A = C$
5: Use the LLM to infer reasoning method M from abstraction A
6: Apply M to new scenarios and update abstraction using feedback

Active abstraction can therefore be interpreted as selecting actions that increase the diversity of collected observations. One simple objective is to maximize observation diversity

$$\max_{a_t} D(\{A_1, A_2, ..., A_t\}) \tag{7}$$

where $D(\cdot)$ measures diversity among observed factor sets. Higher diversity increases the likelihood that instance-specific factors differ while shared factors remain consistent.

Through repeated interactions the system collects a sequence of observations

$$\{A_1, A_2, ..., A_n\} \tag{8}$$

from which the shared factors can be estimated

$$C = \bigcap_{i=1}^{n} C_i \tag{9}$$

The resulting abstraction C represents a generalized structure that supports broader applicability of methods or reasoning patterns.

This self-driven process allows the system to gradually accumulate observations across different times, spaces, and object instances:

$$\{T, S, O\} \xrightarrow{a} C \tag{10}$$

where T, S, and O denote variations across time, space, and objects, respectively, and a represents the active exploration process used to identify common factors C.

When a method or relationship is first observed in a scenario, abstraction may not yet be possible because only one instance exists. The system therefore schedules future observations or exploration actions to collect additional comparable cases.

To summarize the overall process, the abstract-finding mechanism can be described as an iterative procedure that combines observation, abstraction discovery, and verification.

Algorithm 1 summarizes the workflow of the proposed abstract-finding process.

1) Active Feedback Acquisition: After an abstracted relationship or method is applied, the outcomes provide feedback for evaluating its validity. If the abstraction performs well across different scenarios, the extracted common factors are likely meaningful. Otherwise, the system may refine the abstraction by reconsidering the factors involved.

Feedback may originate from internal system sensors or observed environmental changes. Both sources help improve the abstraction process over time.

2) Factor Removal for Abstraction Discovery: Iterative Testing and Verification: When similar objects are not immediately available, abstraction can be explored by iteratively removing potential factors and testing whether the method still applies.

$$issue = \{i_1, i_2, ..., i_n\} \tag{11}$$

A method is first applied to the issue as a whole. The system then removes individual factors and evaluates whether the method remains effective. Factors that can be removed without affecting the method are considered instance-specific, while factors that cannot be removed represent essential components of the abstraction.

3) Active Collection: Continuous Collection and Similarity Comparison: The abstraction process is inherently iterative. In many situations, a single observation does not provide enough information to distinguish instance-specific factors from shared structures. Therefore, the AI system must continuously collect observations from different contexts and compare them to identify common patterns.

The overall process can be summarized as

$$O \rightarrow S \rightarrow A \rightarrow L \tag{12}$$

where O denotes observation collection, S denotes similarity comparison among observations, A denotes abstraction discovery, and L denotes the learning update stage.

Continuous collection is necessary because observations obtained at different times, places, or situations are likely to contain different instance-specific factors F_i. By accumulating observations across diverse contexts, the system increases the probability that irrelevant factors vary while the shared factors remain stable. Similarity comparison then enables the system to identify the stable components that appear consistently across multiple observations, which correspond to candidate common factors used for abstraction.

Learning is required to ensure that the discovered abstractions can be reused and improved over time. After an abstraction C is identified and a reasoning method M is derived from it, the system applies the method to new scenarios and evaluates the outcome. The resulting feedback is used to refine the abstraction and update the system's internal knowledge:

$$(C, M) \xrightarrow{\text{feedback}} L \tag{13}$$

Without this learning stage, the system would repeatedly rediscover similar abstractions and would not accumulate generalized knowledge across tasks.

Through repeated cycles of observation, comparison, abstraction discovery, and learning, the system gradually accumulates generalized structures that apply to broader situations.

B. Symbolic Representation of Common Factors for Simplifying Complex Relationships

After common factors are identified through abstraction, they can be represented symbolically to simplify reasoning about complex relationships. Removing instance-specific factors leaves only the shared structure among observations, which can be represented by a symbolic abstraction. For example, if two observations contain different specific factors but share a common structure, the abstraction can be expressed as

$$A_1 = \{F_1, C\}, \quad A_2 = \{F_2, C\} \ \rightarrow \ A = C \tag{14}$$

where F_1 and F_2 denote instance-specific factors and C denotes the shared structure. The abstraction A therefore represents the common factors across the observations.

Once a symbolic representation is introduced, the abstraction can be reused in more complex reasoning processes. Instead of repeatedly describing the detailed factors that form the abstraction, the system can operate directly on the symbol representing the shared structure. This greatly simplifies reasoning in complex scenarios.

Symbolic abstraction also allows the formation of hierarchical abstraction layers. A newly discovered abstraction may itself become a component of a higher-level abstraction. Formally, if multiple abstractions share a higher-level structure, they can be further abstracted as

$$\{A_1, A_2, ..., A_k\} \rightarrow A' \tag{15}$$

where A' represents a higher-level abstraction constructed from lower-level abstractions. Through repeated abstraction, the system gradually builds multiple abstraction layers.

This hierarchical abstraction process resembles human reasoning. Humans often summarize repeated patterns into symbolic concepts and then reuse these concepts to reason about more complex situations. For example, multiplication can be viewed as a symbolic abstraction of repeated addition. Instead of repeatedly writing a sequence of addition operations, a single multiplication symbol represents the entire process.

Similarly, complex methods or processes observed across multiple scenarios can be represented by symbols and treated as single reasoning units. This allows the system to perform reasoning at a higher level of abstraction while hiding lower-level implementation details.

Symbolic representation can also capture temporal relationships. When a sequence of events consistently follows a pattern, the sequence can be represented as a symbolic causal relationship. For example,

$$A \rightarrow B \tag{16}$$

may represent a cause–effect relationship discovered across multiple observations.

Through symbolic abstraction, LLM-based agents can organize discovered knowledge into hierarchical structures, enabling more efficient reasoning and supporting generalization across broader scenarios.

IV. Verification

A. Experimental Setup

This section evaluates whether identifying key factors through abstraction improves the ability of an AI system to determine whether a method is applicable to new objects.

The experiment has two objectives. First, we evaluate whether observations with higher diversity help an LLM identify key factors required for method applicability. Second, we examine whether the active data collection process, corresponding to the proposed *active abstraction* mechanism, improves abstraction quality compared with purely random observations.

Three observation strategies are compared, denoted as m_0, m_1, and m_2. These strategies differ in how observations used for abstraction are collected. Each strategy follows the same overall procedure but applies different constraints during data collection.

The system first collects n groups of observations ($n = 6$) from a local server named `InfoMethodRandomServer`. Each observation contains a set of information factors, a method label, and a place identifier.

The three strategies are defined as follows.

m_0 **(random collection).** The system randomly collects n groups of observations without any constraints. This strategy serves as a baseline where no abstraction-oriented exploration is performed.

m_1 **(place-diverse collection).** The system attempts to collect n groups of observations with different places. The server is queried up to 20 times and observations that maximize place diversity are selected.

m_2 **(place-and-information diverse collection).** The system attempts to collect n groups of observations that maximize both place diversity and information diversity. The server is queried up to 20 times and observations with the most distinct places and information factors are selected.

After collecting the observation groups, they are provided to a large language model (LLM). The LLM is asked to determine whether *Method 1* can be applied to a set of test objects.

The applicability of the method depends on the presence of a key factor I_2. Objects containing I_2 should be classified as applicable to Method 1, whereas objects without I_2 should not.

The test objects are generated using DeepSeek[1] on March 3, 2026. The dataset contains objects with different combinations of information factors, where some contain the key factor I_2 and others do not.

The evaluation metric is classification accuracy. Let N denote the number of test objects, and let $\hat{y}_i$ and y_i denote the predicted and ground-truth applicability labels for object i, respectively. The accuracy is defined as

[1]https://chat.deepseek.com/

TABLE I
Accuracy results of the three strategies

Run	m_0	m_1	m_2
1	0.3333	0.5000	0.5000
2	0.3333	0.5000	0.5000
3	0.6667	0.8333	0.8333
4	0.6667	0.6667	0.6667
5	0.3333	0.6667	0.6667
6	0.6667	0.5000	0.5000
7	0.5000	0.5000	0.6667
8	0.6667	0.6667	0.5000
9	0.3333	0.5000	0.5000
10	0.3333	0.5000	0.8333
11	0.6667	0.6667	0.6667
12	0.5000	0.6667	0.8333
13	0.5000	0.6667	0.5000
14	0.6667	0.5000	0.6667
15	0.3333	0.5000	0.6667
16	0.5000	0.5000	0.6667
17	0.5000	0.6667	0.6667
18	0.6667	0.8333	0.8333
19	0.3333	0.8333	0.5000
20	0.6667	0.6667	0.6667

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\hat{y}_i = y_i)$$

where $\mathbf{1}(\cdot)$ denotes the indicator function.

B. Results and Analysis

The experiment was repeated 20 times for each strategy. The resulting accuracies for m_0, m_1, and m_2 are summarized in Table 1 and illustrated in Fig. 1.

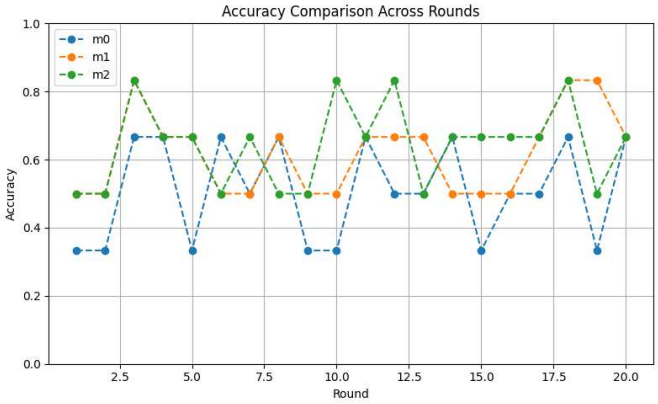


Fig. 1. Accuracy comparison of the three abstraction strategies

The average accuracies are

$$m_0 = 0.5083, \quad m_1 = 0.6167, \quad m_2 = 0.6417.$$

Table 1 shows that the random strategy m_0 frequently produces low accuracy values (e.g., 0.3333), indicating that the LLM often fails to correctly identify method applicability when observations are collected without abstraction-oriented guidance. In contrast, the abstraction-guided strategies m_1 and m_2 frequently achieve higher accuracies such as 0.6667 and

0.8333, suggesting that more informative observation sets help the LLM identify the key factor I_2.

Fig. 1 further illustrates this trend. Both m_1 and m_2 consistently outperform the random baseline m_0, and m_2 slightly outperforms m_1 in several runs. This indicates that increasing both place diversity and information diversity improves the identification of the underlying abstract relationship.

A two-sample t-test confirms that the abstraction-guided strategies (m_1 and m_2) significantly outperform the baseline strategy (m_0) ($p < 0.05$).

The performance difference can be explained by the diversity of observations used for abstraction. Let D_{place} denote the diversity of observation locations and D_{factor} denote the diversity of information factors. The probability that the LLM correctly identifies the key factor can be expressed as

$$P(\text{correct}) = f(D_{place}, D_{factor})$$

where $f(\cdot)$ increases with both diversity terms. The three strategies can therefore be interpreted as

$$m_0 : (D_{place}, D_{factor}) \approx \text{low}$$

$$m_1 : (D_{place}, D_{factor}) \approx (\text{high}, \text{low})$$

$$m_2 : (D_{place}, D_{factor}) \approx (\text{high}, \text{high})$$

which explains the empirical observation

$$P_{m_2}(\text{correct}) > P_{m_1}(\text{correct}) > P_{m_0}(\text{correct}).$$

C. Discussion

The experimental results indicate that abstraction-oriented observation strategies improve the ability of the LLM to determine method applicability for new objects. Compared with the random strategy m_0, both m_1 and m_2 guide the observation process to increase contextual diversity, which helps the LLM identify the key factor I_2.

The comparison between m_1 and m_2 further highlights the importance of factor diversity in abstraction. While m_1 increases only place diversity, m_2 increases both place and information diversity, providing stronger evidence for identifying common structures. Consequently, the LLM is more likely to correctly identify the factor that determines method applicability.

These results support the proposed abstraction framework. When an AI system actively collects observations that differ in contextual and informational factors, instance-specific components become easier to remove and the shared structure becomes more visible.

Furthermore, the experiment reflects the role of active abstraction in Human Simulation Computation (HSC) [2]. Instead of passively receiving observations, the AI system intentionally explores diverse scenarios to facilitate abstraction. Such action-driven abstraction enables the system to progressively identify key factors that determine method applicability across different contexts.

V. Conclusion

This paper proposed an active abstract-finding framework for improving the generalization ability of LLM-based agents. The framework models abstraction as the process of identifying shared factors across observations while removing instance-specific factors. To facilitate abstraction discovery, the AI system actively collects observations from diverse contexts and performs similarity comparison to extract common structures.

Experimental results show that abstraction-oriented observation strategies improve the ability of LLMs to determine method applicability. In particular, the proposed strategy that increases both place diversity and information diversity achieves higher accuracy than random observation strategies, demonstrating the effectiveness of active abstraction in identifying key factors.

Future work will investigate more complex environments where multiple interacting factors influence method applicability and explore how abstraction mechanisms can be integrated with autonomous exploration strategies in large-scale AI systems.

References

[1] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang *et al.*, "A survey on evaluation of large language models," *ACM transactions on intelligent systems and technology*, vol. 15, no. 3, pp. 1–45, 2024.

[2] H. Su, "Human simulation computation: A human-inspired framework for adaptive ai systems," *arXiv preprint arXiv:2601.13887*, 2026.

[3] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *The eleventh international conference on learning representations*, 2022.

[4] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," *Advances in neural information processing systems*, vol. 36, pp. 68 539–68 551, 2023.

[5] Y. Xia, R. Wang, X. Liu, M. Li, T. Yu, X. Chen, J. McAuley, and S. Li, "Beyond chain-of-thought: A survey of chain-of-x paradigms for llms," in *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 10 795–10 809.

[6] N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning, 2023," *URL https://arxiv. org/abs/2303.11366*, vol. 8, 2024.

[7] K. Sun, L. Wang, B. Xu, W. Zhao, S. W. Teng, and F. Xia, "Network representation learning: From traditional feature learning to deep learning," *IEEE Access*, vol. 8, pp. 205 600–205 617, 2020.

[8] M. Bober-Irizar and S. Banerjee, "Neural networks for abstraction and reasoning," *Scientific Reports*, vol. 14, no. 1, p. 27823, 2024.

[9] V.-L. Nguyen, M. H. Shaker, and E. Hüllermeier, "How to measure uncertainty in uncertainty sampling for active learning," *Machine Learning*, vol. 111, no. 1, pp. 89–122, 2022.
[10] N. Botteghi, B. Sirmacek, M. Poel, C. Brune, and R. Schulte, "Curiosity-driven reinforcement learning agent for mapping unknown indoor environments," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 5, no. 1, pp. 129–136, 2021.
[11] A. Ten, P.-Y. Oudeyer, and C. Moulin-Frier, "Curiosity-driven exploration: Diversity of mechanisms and functions," *The Drive for Knowledge: The Science of Human Information Seeking*, p. 53, 2022.

PLACE PHOTO HERE

Hong Su received the MS and PhD degrees, in 2006 and 2022, respectively, from Sichuan University, Chengdu, China. He is currently a researcher of Chengdu University of Information Technology Chengdu, China. His research interests include blockchain, large language model and human simulation computing.