https://doi.org/10.1007/s11704-026-51673-0

REVIEW ARTICLE

Fast, slow, and tool-augmented thinking for LLMs: a review

Xinda JIA[1], Jinpeng LI[2], Zezhong WANG[3], Jingjing LI[4], Xingshan ZENG[5], Yasheng WANG[6], Weinan ZHANG[7], Yong YU[7], Weiwen LIU[7]✉

1. School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
2. Huawei Technologies Co., Ltd., Beijing 100084, China
3. Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong 999077, China
4. Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong 999077, China
5. Huawei Hong Kong Research Center, Hong Kong 999077, China
6. College of Control Science and Engineering, Zhejiang University, Hangzhou 310000, China
7. School of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China

Received October 14, 2025; accepted July 6, 2026

E-mail: wwliu@sjtu.edu.cn

© The Author(s) 2026. This article is published with open access at link.springer.com and journal.hep.com.cn

Abstract

Large Language Models (LLMs) have demonstrated significant progress in reasoning across diverse domains. However, effective reasoning in real-world tasks requires adapting both the computational depth and the knowledge source to the demands of the problem, ranging from fast, intuitive responses to deliberate, step-by-step reasoning and from purely internal reasoning to externally tool-augmented processes. Drawing inspiration from cognitive psychology, we propose a novel taxonomy of LLM reasoning strategies along two orthogonal knowledge boundaries: (1) a fast/slow boundary separating intuitive from deliberative processes, and (2) an internal/external boundary distinguishing reasoning grounded in the model's parameters from reasoning augmented by external tools. We systematically survey recent work on adaptive reasoning in LLMs and categorize methods based on key decision factors. We conclude by highlighting open challenges and future directions toward more adaptive, efficient, and reliable LLMs.

Keywords

natural language processing; artificial intelligence; machine learning

1 Introduction

The mind is not a single, undifferentiated entity, but a collection of specialized mechanisms shaped by evolution to solve different categories of problems. — Steven Pinker, *How the Mind Works* (1997)

Large Language Models (LLMs) have been advancing rapidly, steadily pushing the boundaries toward Artificial General Intelligence (AGI) [1,2]. Significant breakthroughs have been made in LLM reasoning capabilities, including mathematics, coding, and logical reasoning, as seen in models like OpenAI o3 [3], DeepSeek-R1 [2], and Grok 4 [4]. Reasoning—the ability to draw inferences, make decisions, and solve problems based on available information—is central to human intelligence [5]. Recent research has identified several distinct reasoning strategies in LLMs, loosely aligning with theories of human cognition [6].

In particular, these strategies are often discussed in terms of fast, slow, and tool-augmented thinking. From a structural perspective, these reasoning strategies can be characterized with two orthogonal dimensions rather than as three mutually exclusive modes. The fast/slow knowledge boundary reflects differences in computation depth, distinguishing intuitive, rapid responses from deliberate, step-by-step reasoning. The internal/external knowledge boundary captures the differences in the knowledge sources, distinguishing reasoning grounded solely in the internal parameters of a model from reasoning augmented by external tools or information sources. In this view, tool-augmented thinking is not a standalone alternative to fast or slow thinking, but an external knowledge dimension that can intersect with both.

- **Fast thinking.** Analogous to human System 1 cognition, fast thinking involves intuitive responses generated directly from the model without explicit intermediate steps [7]. Fast thinking offers low latency and high throughput, making it particularly suitable for simple or routine tasks such as casual conversation, sentiment classification, surface-level summarization, and factual questions with high model confidence. However, it is prone to errors when the input is ambiguous, unfamiliar, or requires multi-step reasoning.

● **Slow thinking.** Similar to human System 2 cognition, slow thinking engages in step-by-step, deliberative reasoning. Techniques such as chain-of-thought (CoT) prompting [8] and its extensions like the tree-of-thoughts (ToT) [9], as well as other techniques including self-reflection [10] and intermediate verification [11], enable models to decompose problems, explore alternative reasoning paths, and validate intermediate results. This mode is particularly effective for complex tasks such as mathematical problem solving, logical deduction, planning, and scientific explanation, where intermediate reasoning steps are crucial for correctness, although at the cost of increased latency and computation.

● **Tool-augmented thinking.** Tool-augmented thinking extends beyond the model's innate capabilities by incorporating external tools (e.g., calculators, code interpreters, search engines or knowledge databases). This mode is especially useful when tasks require precise numerical computation, access to up-to-date or domain-specific knowledge, or interaction with external environments, such as web search, code execution, and retrieval-augmented generation (RAG). Tools can support both fast thinking, by providing quick access to relevant information, and slow thinking, by adding depth through detailed computations, intermediate verification, and exploration of alternative reasoning paths. This mirrors humans' reliance on external aids to compensate for memory limitations and knowledge gaps, enabling both faster responses and more thorough reasoning, depending on the task at hand.

However, the complexity and diversity of real-world applications demand that LLMs flexibly adapt their reasoning strategy to each task's accuracy and latency constraints. Hence, selecting and shifting between the appropriate reasoning strategies of LLMs becomes an active and growing area of research [12,13]. Different tasks call for different reasoning modes: fast, intuitive responses may suffice for casual conversation or surface-level summarization, while complex problem solving, multi-step logical deduction, or tasks requiring factual precision often necessitate slower, more deliberate reasoning or external tool integration. A mismatch between task demands and the model's reasoning strategy can lead to inefficiencies or errors [14].

For example, underthinking can lead to superficial or incorrect responses when a deeper analysis is needed, such as mathematical problem solving, legal argumentation, or scientific explanation [8]. On the other hand, overthinking can also be detrimental [15]. Applying complex reasoning strategies to simple tasks can introduce unnecessary latency and compute, and intermediate traces may contain hallucinated or spurious steps that mislead the final answer [14]. Moreover, language models that rely solely on internal knowledge are prone to hallucination, especially on factual or long-tail queries [16]. Yet blindly incorporating retrieved knowledge can destabilize model behavior and propagate errors [17].

Developing a deeper understanding of how LLMs reason, and how these reasoning processes can be guided or improved, is essential for both advancing model capabilities and enhancing their reliability in real-world applications. To address this need, we propose a taxonomy structured along two knowledge boundaries, as shown in Fig. 1. The fast/slow knowledge boundary reflects differences in reasoning depth, distinguishing intuitive, rapid responses from deliberate, step-by-step reasoning. The internal/external knowledge boundary captures differences in knowledge sources, distinguishing reasoning grounded solely in a model's internal parameters from reasoning augmented by external tools or information sources.

Building on these two knowledge boundaries, we further survey and categorize existing research on adaptive reasoning strategy selection. To the best of our knowledge, this is the first comprehensive review that systematically examines the adaptive selection among all three major reasoning strategies—fast thinking, slow thinking, and tool-augmented thinking. Our main contributions are summarized as follows:

- We present the first comprehensive taxonomy of reasoning strategies in LLMs, organized along two knowledge boundaries: the fast/slow knowledge boundary and the internal/external knowledge boundary.
- We conduct a systematic analysis of existing research on

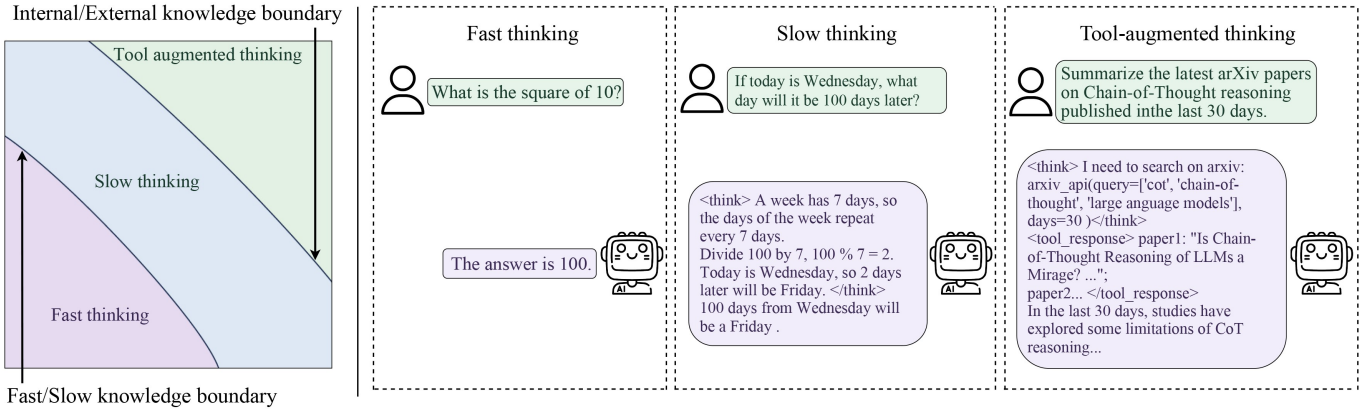


Fig. 1 Comparison of fast, slow, and tool-augmented thinking. The left part shows the two knowledge boundaries that define the 2×2 structure of reasoning strategies: the fast/slow knowledge boundary and the internal/external knowledge boundary. The right part showcases representative tasks for each reasoning strategy

adaptive reasoning strategy selection, categorizing methods based on key decision factors such as model confidence, task complexity, and utility gain.

- We identify and discuss open challenges and promising future directions for improving reasoning strategy selection in LLMs.

2 Formulation

Inspired by the decision-making principles widely adopted in adaptive reasoning systems, we view reasoning strategy selection in LLMs as a process of evaluating multiple decision signals and crossing the boundaries. Accordingly, we introduce a unified framework that abstracts existing methods into decision factor extraction and score-based strategy selection, without assuming a specific implementation.

2.1 Decision factor extraction

Given an input query x and model parameters θ, a reasoning strategy selection framework for LLMs first extracts some decision factors.

These are represented as

$$\mathbf{g}^{(d)}(x,\theta,c) = [f_1^d, f_2^d, \ldots, f_{n_d}^d], \tag{1}$$

$$\mathbf{g}^{(s)}(x,\theta,c) = [f_1^s, f_2^s, \ldots, f_{n_s}^s], \tag{2}$$

where $\mathbf{g}^{(d)}$ and $\mathbf{g}^{(s)}$ denote the lists of features relevant to reasoning depth and knowledge source selection, respectively. The optional control signal c captures explicit guidance or constraints—such as user prompts, predefined rules, compute budgets, or external routing directives. In some scenarios, c may be omitted or implicitly encoded within the model architecture or prompt.

The extracted decision factors can capture a range of behavioral signals from the model. These may include internal signals such as model confidence or uncertainty, as well as external contextual cues like task type or user intent. For example, the model's confidence distribution [15] measured via output logits serves as a proxy for uncertainty and can guide reasoning depth. Self-Route [18] incorporates query complexity as a model-conditioned decision factor. By training on datasets with a continuous gradient of difficulty, its routing mechanism learns to adaptively determine whether a given query can be addressed with shallow reasoning or requires deeper processing.

2.2 Strategy selection

Based on the extracted decision factors, the system applies a selection procedure to compute decision scores at one or more decision units:

$$\mathbf{\Phi}_d(x,\theta,c) = \mu_d\left(\mathbf{g}^{(d)}(x,\theta,c)\right), \tag{3}$$

$$\mathbf{\Phi}_s(x,\theta,c) = \mu_s\left(\mathbf{g}^{(s)}(x,\theta,c)\right), \tag{4}$$

where $\mathbf{\Phi}_d$ and $\mathbf{\Phi}_s$ denote vectors of decision scores. They may contain one or multiple entries, depending on whether the strategy selection procedure μ_d or μ_s is performed once globally [19,20] or dynamically during generation, such as at the token, sentence or step level [21,22]. Specifically, $\mathbf{g}^{(d)}$ or $\mathbf{g}^{(s)}$ may contain logit-derived confidence or uncertainty signals, and the corresponding selection procedure μ_d or μ_s can be applied in a sentence-wise manner. In this case, each sentence-level decision unit produces one decision score, and these scores together form the vector $\mathbf{\Phi}_d$ or $\mathbf{\Phi}_s$. For example, in FLARE [21], $\mathbf{g}^{(s)}$ is instantiated by token probabilities derived from the model logits, and μ_s operates as a sentence-wise selection procedure. For each temporarily generated sentence, it records the generation probability of each token and computes a sentence level selection score based on the lowest token probability. The collection of these sentence-level scores forms the vector $\mathbf{\Phi}_s$.

Importantly, each entry in these vectors corresponds to a scalar decision criterion, denoted by $\Phi_{d,t}$ or $\Phi_{s,t}$. In some methods, this score is explicitly exposed and can take various forms, including linear aggregation, minimum over a set of values, confidence-based scores, or other rule-based decision criteria [20]. In other methods, especially end-to-end learned methods, it may not be directly reported but can be interpreted as an implicit preference induced by the learned policy over alternative reasoning actions [23].

Such selection procedures can be manually designed, specified heuristically, or learned via supervised training or reinforcement learning [24]. For example, the CGR framework [25] instantiates an adaptive reasoning depth selection framework in a step wise manner. It uses token probabilities derived from the model logits as confidence decision factors. The selection procedure evaluates these factors at each predefined reasoning step by probing the current reasoning trace and produces a scalar uncertainty score, derived from the minimum token probability among the selected answer tokens, which is then compared against a predefined threshold to determine whether deeper reasoning should be triggered.

Specifically, $\mathbf{\Phi}_d$ and $\mathbf{\Phi}_s$ are vectors of selection scores used as routing variables, representing the expected benefit of allocating additional reasoning or external resources. The corresponding threshold vectors are denoted by $\boldsymbol{\delta}_d$ and $\boldsymbol{\delta}_s$, and the resulting selection outcomes are denoted by the vectors $\boldsymbol{k}_d$ and $\boldsymbol{k}_s$. For each decision unit t, when the corresponding score entry exceeds its associated threshold entry $\delta_{d,t}$ for reasoning depth or $\delta_{s,t}$ for knowledge source, the corresponding knowledge boundary is crossed, yielding the following decision functions:

$$k_{d,t}(x,\theta,c) = \begin{cases} \textit{fast thinking}, & \text{if } \Phi_{d,t} < \delta_{d,t}, \\ \textit{slow thinking}, & \text{if } \Phi_{d,t} \geqslant \delta_{d,t}, \end{cases} \tag{5}$$

$$k_{s,t}(x,\theta,c) = \begin{cases} \textit{internal knowledge}, & \text{if } \Phi_{s,t} < \delta_{s,t}, \\ \textit{external knowledge}, & \text{if } \Phi_{s,t} \geqslant \delta_{s,t}, \end{cases} \tag{6}$$

where the threshold vectors $\boldsymbol{\delta}_d$ and $\boldsymbol{\delta}_s$ are typically model-dependent. The implementation of this threshold mechanism varies significantly across the surveyed literature. In some approaches, each entry of the corresponding threshold vector is instantiated as an explicit scalar value that is either manually specified [26,27] or empirically calibrated on a validation set [28]. In these cases, overall system performance is highly sensitive to the chosen threshold, as it directly controls the frequency of strategy switching and governs the critical trade-off between computational efficiency and reasoning accuracy. Conversely, other methods realize this switching criterion implicitly

through training, such as through post-training alignment or learned gating networks [19,29]. Within our framework, these learned routing mechanisms function as implicit decision boundaries that serve the same role as the threshold vectors $\boldsymbol{\delta}_d$ and $\boldsymbol{\delta}_s$.

It is worth noting that most existing studies that decide whether to use a particular reasoning mode usually treat the fast/slow and internal/external dimensions separately. However, these two dimensions may still interact in practical reasoning systems. For example, when a model decides to use external knowledge or tools, this result can further serve as a condition for deciding how much reasoning should be performed around the tool call. Such explicit coupling, however, remains relatively underexplored in existing work. Therefore, this interaction can be regarded as a potential direction for future extensions of our framework. Specifically, Eqs. (5) and (6) can be extended by taking the internal/external selection result as an additional input to the fast/slow decision, thereby allowing the two dimensions to be modeled in a more coupled manner.

Finally, the model generates the response conditioned on the selected strategy:

$$y = f_\theta(x, \boldsymbol{k}_d(x,\theta,c), \boldsymbol{k}_s(x,\theta,c)), \tag{7}$$

where the vectors $\boldsymbol{k}_d$ and $\boldsymbol{k}_s$ denote the reasoning depth and knowledge source selection results across the decision units, respectively, and each can reduce to a single selection when only one decision unit is used for the corresponding dimension.

We next instantiate this framework with three representative methods, covering both reasoning depth selection and knowledge source selection. For example, MUR [22] provides a concrete instantiation of adaptive selection between fast thinking and slow thinking. In MUR, fast thinking means directly proceeding with the current reasoning step, while slow thinking is implemented through test-time scaling (TTS). It first extracts step-level uncertainty m_t, derived from token probabilities, as the decision factor $\mathbf{g}^{(d)}$ in Eq. (1). The selection procedure μ_d in Eq. (3) aggregates the previous momentum uncertainty M_{t-1} and the current uncertainty m_t into $M_t = \alpha M_{t-1} + (1-\alpha)m_t$, which serves as an entry of decision score vector $\boldsymbol{\Phi}_d$ in Eq. (3). MUR then compares this score with a manually designed threshold each turn, such as $\exp(M_{t-1})/\gamma$, to determine the reasoning depth decision in Eq. (5): if the uncertainty exceeds the threshold, the model switches to slow thinking by applying additional TTS computation; otherwise, it remains in fast thinking. Finally, the outputs of all reasoning steps are concatenated into the final response, corresponding to Eq. (7). As an example of knowledge source selection, FLARE [21] instantiates adaptive selection between internal and external knowledge through confidence based active retrieval. Specifically, FLARE first generates a temporary next sentence $\hat{s}_t$ without retrieval and extracts token-level confidence scores as the decision factors $\mathbf{g}^{(s)}$ in Eq. (2). The selection procedure μ_s in Eq. (4) is instantiated as a sentence-wise confidence based algorithm: for each temporarily generated sentence, it computes the minimum token probability $\min_{w_i \in \hat{s}_t} p(w_i)$. In the original formulation, retrieval is triggered if any token probability falls below a manually specified confidence threshold θ. Equivalently, by defining an uncertainty-based decision score $\Phi_{s,t} = 1 - \min_{w_i \in \hat{s}_t} p(w_i)$ and setting $\delta_{s,t} = 1 - \theta$, the decision can be expressed in the form of Eq. (6): if $\Phi_{s,t} \leqslant \delta_{s,t}$, the model accepts the internally generated sentence without retrieval; otherwise, it invokes external retrieval and regenerates the sentence conditioned on the retrieved documents. Through this sentence-level selection process, the final response is generated as a trajectory of internal and external knowledge-use decisions, corresponding to Eq. (7).

Furthermore, AdaptThink [23] illustrates a learned implicit form of reasoning depth selection, which corresponds to the scalar case of our vectorized formulation. The decision factors $\mathbf{g}_d$ in Eq. (1) correspond to signals indicating the necessity of explicit reasoning, such as problem difficulty and the relative utility of *Thinking* over *NoThinking*. The selection procedure μ_d in Eq. (3) is instantiated by an RL-trained policy that learns to choose between the two modes. In its PPO-style objective, AdaptThink uses the advantage term $A(x,y) = \mathbb{I}(y_1 = \texttt{</think>}) \cdot \delta + R(x,y) - \bar{R}_{\text{ref}}(x)$, where δ serves as a threshold-like control parameter encouraging *NoThinking* when performance is not degraded. After training, the learned policy induces an implicit scalar preference score Φ_d in Eq. (3), and determines the reasoning-depth result k_d in Eq. (5): generating `</think>` at the beginning corresponds to fast thinking, whereas continuing explicit reasoning corresponds to slow thinking. Therefore, AdaptThink realizes Eq. (5) through an implicit decision boundary learned from its RL objective, and the resulting strategy selection conditions the final response generation in Eq. (7).

This unified two-stage framework, comprising decision factor extraction and strategy selection, offers a principled and modular approach to adaptive reasoning in LLMs. It separates the *what to consider* (captured by decision factors) from the *how to decide* (captured by selection functions), enabling more interpretable and controllable model behavior.

Prior work has explored different aspects of this framework. Some focus on improving decision factor extraction, for example by designing richer behavioral signals or learning better input representations [7,21]. Others emphasize the design of strategy selection mechanisms [19,23], leveraging methods such as rule-based heuristics, supervised learning, or reinforcement learning to map decision factors to optimal reasoning modes.

Our two-stage framework successfully accommodates a broad spectrum of adaptive reasoning methodologies, specifically capturing approaches that emphasize decision factor extraction [30–32], strategy selection mechanisms [12,18,20], or a combination of both [7,22,23,25]. Conversely, a primary boundary condition of our formulation is that it does not encompass *reasoning strategy fusion* paradigms. Methods such as SMaRT [33], which concurrently employ multiple reasoning strategies and combine their outputs to reach a final conclusion, operate on an ensemble principle rather than a routing mechanism. Consequently, our framework should be interpreted as a unifying abstraction designed specifically for selection-based adaptive reasoning, rather than fusion- or aggregation-based approaches.

3 Perspective of cognitive psychology

Understanding the reasoning strategies of LLMs through the lens of cognitive psychology offers a powerful basis for both interpreting their behavior and guiding the design of more adaptive and effective reasoning systems.

One of the foundational concepts in cognitive psychology is metacognition—the ability to monitor and regulate one's own cognitive processes [34]. In humans, metacognitive skills enable strategic decision-making about when to rely on intuition versus when to engage in more effortful reasoning. Complementing this, dual-process theory [6] distinguishes between two modes of thinking: System 1 (fast, automatic, intuitive) and System 2 (slow, effortful, analytical). These two systems are not mutually exclusive but interact dynamically, with System 2 often regulating or overriding System 1 when deeper reflection is needed. Cognitive load theory [35] emphasizes that high-complexity tasks that exceed working memory capacity often necessitate the use of external aids or simplified representations to maintain performance.

Analogously, the reasoning process of LLMs benefits from switching among fast, slow, and tool-augmented thinking, balancing efficiency and accuracy depending on task demands. However, this analogy should be understood pragmatically rather than as a claim of cognitive equivalence. LLMs do not possess human working memory, phenomenological experience, or biologically grounded automaticity. Therefore, their fast, slow, and tool-augmented thinking should not be interpreted as direct counterparts of human System 1 and System 2 processes. Nevertheless, since LLMs can rely on internal parametric knowledge, engage in deliberate reasoning, and use external resources, cognitive science continues to inspire architectural and algorithmic design of LLMs, shaping how they reason, adapt, and interact in increasingly intelligent ways.

4 Overall selection paradigm

In this section, we review common paradigms for selecting reasoning strategies in LLMs, as illustrated in Fig. 2. We group existing approaches into two categories: *implicit* and *explicit* selection. Implicit selection refers to models that internally determine reasoning strategies, typically learned end-to-end during post-training. In contrast, explicit selection refers to approaches that rely on externally specified mechanisms to guide reasoning strategy choice, such as hand-crafted rules and learned router networks.

From the perspective of our framework in Section 2, the two paradigms differ primarily in how selection procedures are instantiated. Specifically, implicit selection obtains the procedure μ in our framework (Eqs. (3) and (4)) from the model parameters θ, where this capability is acquired through SFT or RL-based post-training. On the contrary, explicit selection obtains the procedure from explicit rules or external routers. Based on the decision scores produced by the procedures, these scores are compared with the thresholds defined by the decision functions in Eqs. (5) and (6), resulting in the final depth of reasoning and the source of knowledge.

There are several practical differences between the two selection paradigms. *Explicit* selection provides fine-grained external control, as predefined rules or external routers determine when to invoke different reasoning strategies. In contrast, *implicit* selection is based on internally learned policies, offering smoother integration into the model but less direct controllability. Training cost also differs by design: *implicit* selection requires the model to acquire adaptive behavior through SFT or RL during post-training, whereas *explicit* selection typically trains a lightweight routing module.

4.1 Implicit selection

For implicit selection, the adaptive selection ability is typically acquired during post-training, where the model learns to associate certain input distributions or task features with distinct reasoning behaviors [23,36]. In particular, two types of post-training methods are adopted: *Supervised Fine-Tuning* (*SFT*) and *Reinforcement Learning* (*RL*).

4.1.1 Supervised Fine-Tuning (SFT)

SFT trains LLMs on curated datasets where distinct reasoning behaviors are embedded within the training samples [12,37–40].

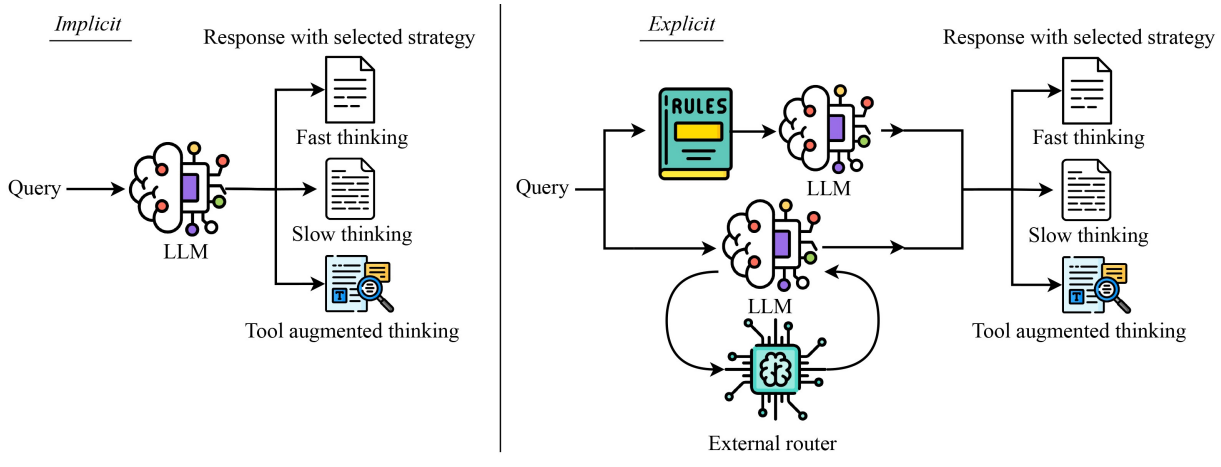


Fig. 2 The comparison between implicit and explicit reasoning strategy selection. In implicit selection (left), the reasoning strategy is determined internally by the LLM. In contrast, explicit selection (right) relies on additional rules or external routers to choose the appropriate reasoning strategy

Through exposure to diverse task types that span different knowledge boundaries and solution formats, the model implicitly learns to associate input patterns with appropriate reasoning strategies.

For instance, [36] construct training data that includes both short and long Chain-of-Thought (CoT) reasoning paths, accompanied by a special indicator token signaling the preferred path. Similar ideas are also explored in Self-Braking Tuning (SBT) [12] and SynapseRoute [41]. Additionally, [38] introduces the use of reflection tokens, enabling the model to dynamically choose whether to invoke tool-augmented reasoning and retrieve the external knowledge.

4.1.2 Reinforcement Learning (RL)

Reinforcement Learning (RL) frameworks further enable models to optimize reasoning strategies based on reward signals.

For example, KAT-V1 [42] introduces a step-SRPO reinforcement learning framework, enabling the model to adaptively switch between reasoning and non-reasoning modes based on the input query. AdaCoT [19], trained using the Proximal Policy Optimization (PPO) algorithm, implements adaptive reasoning by determining the necessity of CoT reasoning based on query complexity. AdaptThink [23] formulates this as a constrained RL problem, introducing an importance sampling strategy to balance the use of different reasoning strategies. LHRMs [43] further explore hybrid reasoning by combining hybrid fine-tuning with online RL using Hybrid Group Policy Optimization (HGPO). AutoThink [24] adopts a multi-stage RL framework with reward shaping to learn policies that jointly optimize reasoning quality and computational cost.

4.2 Explicit selection

Explicit selection introduces external mechanisms that explicitly guide the model in choosing an appropriate reasoning strategy. We further categorize existing methods into *rule-based* and *model-based* selecting approaches.

4.2.1 Rule-based selection

Rule-based selection refers to approaches that rely on manually defined heuristics or fixed criteria to determine the appropriate reasoning strategy, typically according to model confidence, task complexity, or input characteristics.

For example, [13] point out that a simple prompt like “Let’s think step by step” can activate CoT reasoning, improving performance on complex tasks. Later, [22] propose a test-time control mechanism that uses a momentum-based uncertainty threshold to determine whether to engage in deeper reasoning strategies. Rule-based Confidence Reasoning (RCR) [44] explicitly compares the model’s internal-only and external context-based answers using a confidence scoring function, and selects the answer with higher confidence as the final output.

The rule-based methods are typically lightweight, interpretable, and easy to integrate into existing inference pipelines. However, their effectiveness relies heavily on the quality and generality of the handcrafted rules.

4.2.2 Model-based selection

Model-based selection refers to approaches that employ a learned external router to automatically determine the most appropriate reasoning strategy based on the input query [18,20,45].

ThinkSwitcher [20] enables a single LLM to adaptively switch between short chain-of-thought (SC) and long chain-of-thought (LC) reasoning modes. It trains a lightweight switching module that, given a query embedding, predicts the empirical pass rates of both modes and selects LC if its predicted success rate exceeds that of SC by a predefined threshold. Similarly, Self-Route [18] introduces a lightweight classifier trained on the model’s internal hidden representations to serve as a router. This classifier dynamically selects between reasoning modes during inference, allowing the model to adjust its strategy based on input complexity or uncertainty.

Compared to the rule-based methods, learned external routers can capture complex, data-driven knowledge boundaries that may be difficult to encode manually.

4.3 Discussion

Adaptive reasoning strategy selection generally aims at allocating reasoning effort which yields the highest marginal value.

Implicit selection enables LLMs to autonomously infer and adopt appropriate reasoning strategies without relying on external controllers or routing mechanisms. A central challenge for implicit selection methods lies in: how to effectively construct and compose the training data across different reasoning strategies. Improper data composition can bias the LLMs toward overusing certain strategies or failing to generalize across diverse tasks.

Explicit selection often adds an external router that turns this problem into a system-level policy over strategies, tools, and even models. In practice, many production systems employ learned routers to decide when to invoke heavier reasoning or specialized components. For example, GPT-5 series [46] represents a production-scale explicit selection framework. Its interface includes Instant, Thinking, and Auto modes, allowing users to manually control reasoning depth. Instant is typically used for routine or lightweight tasks, while Thinking is intended for more complex problems, corresponding to fast and slow thinking, respectively. In addition, an Auto mode employs a real-time router that selects among reasoning modes based on conversation type, complexity, tool requirements, and user intent. However, explicit selection can introduce orchestration overhead, and the router’s estimates may be misaligned with base-model capabilities—both underestimation and overestimation can add errors or costs.

5 Decision factors in strategy selection

In this section, we present a detailed review of existing literature through the lens of the key decision factors governing the choice of reasoning strategies. To this end, we develop a systematic taxonomy organized around two knowledge boundaries, as illustrated in Fig. 3.

The intersection of computation depth and knowledge source yields four distinct methodological categories: Fast-Internal, Slow-Internal, Fast-External, and Slow-External. The Fast-Internal category corresponds to standard direct inference without explicit deliberation. Because reasoning is fundamentally designed to surpass the limitations of direct inference, the majority of current research concentrates on Slow-Internal and Slow-External approaches. Within

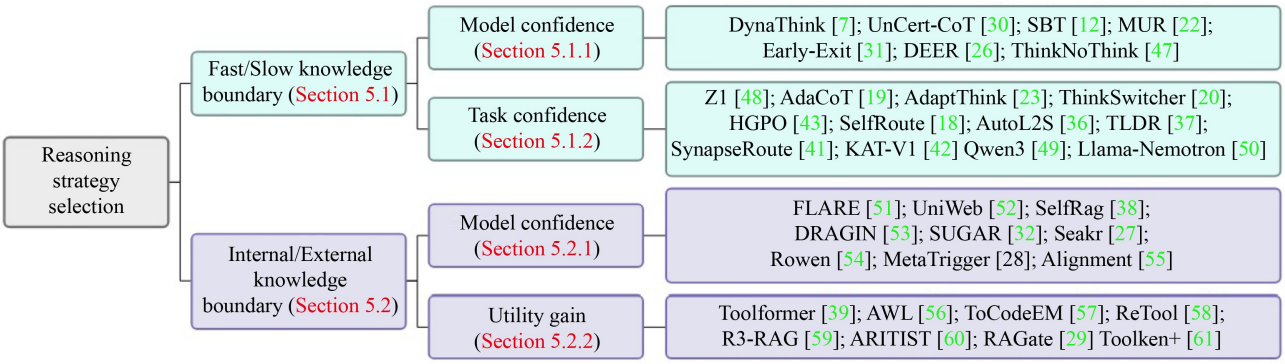


Fig. 3 Decision factors for adaptive reasoning strategy selection

these paradigms, models allocate additional computational resources or consult external tools to resolve complex queries. By contrast, the Fast-External category, characterized by immediate access to external resources without deliberate reasoning, has been less systematically studied as a dedicated research direction. Nevertheless, it is commonly reflected in production style systems such as function calling frameworks, where an LLM invokes predefined functions or APIs and incorporates the returned results, and simple RAG pipelines, where top ranked external passages are retrieved and directly provided as context without explicit reasoning chains. Acknowledging these cases makes the four quadrant taxonomy more comprehensive.

5.1 Fast/slow knowledge boundary

The fast/slow knowledge boundary determines when a model should rely on direct responses versus engaging in longer, more computationally intensive reasoning. Two primary decision factors at this knowledge boundary are *model confidence* and *task complexity*. In general, high confidence and low complexity are processed by the fast reasoning strategy, whereas uncertainty or greater complexity are handled by the slow reasoning strategy.

5.1.1 Model confidence

Model confidence captures the LLM's estimated likelihood that the current output is correct. Confidence guided selection dynamically adjusts reasoning strategies based on the model's internal confidence estimates, thereby optimizing the trade-off between performance and computational cost.

CRL [61] provides one of the first in-depth analyses of LLMs' self-reported and behavioral confidence on reasoning tasks, showing a positive but imperfect correlation between confidence and accuracy.

Building on this, recent work has started to explore confidence-guided decision frameworks that dynamically switch between fast and slow reasoning. DynaThink [7] proposes two selection criteria to guide the choice between fast and slow reasoning: consistency verification and reasoning depth. In the formulation presented in Section 2, these criteria can be interpreted as the decision features used in the selection function. The framework selects fast thinking when a single answer achieves high consistency and low reasoning depth, which in our formulation corresponds to the decision score crossing the selection threshold. Problems that fail either condition default to slow thinking, involving more resource-intensive and deliberate analysis.

For code generation, UnCert-CoT [30] proposes entropy-based and probability-differential-based measures of confidence. When confidence is low, UnCert-CoT triggers long CoT decoding to sample multiple reasoning paths and selects the final code with the highest estimated likelihood of correctness. SBT [12] adopts a reasoning efficiency ratio and an overthinking label ratio to detect overthinking, enabling early termination when thresholds are met. Similarly, MUR [22] aggregates historical step-level uncertainties to assess the confidence of a reasoning path and selectively allocates computation to the most critical steps.

Another line of work focuses on self-truncation based on model confidence during reasoning. Early-Exit [31] finds models' hidden states encode the correctness of future answers, enabling early prediction of the correctness and deciding whether to exit reasoning. DEER [26] monitors transitions in reasoning, and dynamically exits when the confidence of the trial answer is sufficiently high. More recently, ThinkNoThink [47] jointly generates fast and slow reasoning steps and stops reasoning when two responses are consistent.

Collectively, these studies demonstrate that confidence estimation—whether derived from token probabilities, entropy, or the self-consistency [62] of reasoning paths—offers a powerful control signal for adaptive reasoning. Nonetheless, while higher confidence often correlates with better performance, LLMs can be overconfident, which can misguide strategy selection [61].

5.1.2 Task complexity

Task complexity captures the intrinsic difficulty of an input query, typically estimated by how much reasoning depth, exploration, or verification is likely required to reach a right answer. Unlike confidence, which is an internal belief about an answer, task complexity is an ex-ante property of the problem.

To build supervision for complexity-aware selection, Z1 [48] curates a dataset that integrates both short reasoning trajectories for simple problems and strong reasoning trajectories for complex problems, where complexity is determined by the Evol-Instruct method [63] to span the difficulty spectrum. Methods like AutoL2S [36], SynapseRoute [41], and KAT-V1 [42] seek to explore

automated annotation methods to label queries as "thinking" or "non-thinking" by applying a rule-based evaluation of answer accuracy, inference time, and token usage. TLDR [37] further introduces a dynamic re-weighting method that continuously adjusts the training data ratio between fast and slow thinking.

Recent work uses RL to endow models with internal control over reasoning depth. AdaCoT [19] formulates adaptive reasoning as a Pareto optimization problem, seeking to balance model performance with the computational costs. It employs an RL policy to dynamically assess the complexity of incoming queries and determine the necessity and extent of the reasoning steps. In the meanwhile, AdaptThink [23] employs a constrained optimization objective within an on-policy RL framework, teaching the model to adaptively switch to a fast thinking mode for simpler problems to reduce inference costs without sacrificing accuracy. The Hybrid Group Policy Optimization (HGPO) algorithm [43] is an online RL method designed to enable the model to implicitly learn when to select the appropriate reasoning strategy. HGPO is guided by comparing the relative quality of fast and slow thinking responses for a given query, using inter-group and intra-group reward signals.

A complementary approach externalizes the decision via lightweight routers. ThinkSwitcher [20] deploys a lightweight external router that dynamically selects between fast and slow thinking modes based on task complexity. The router is trained with supervision signals derived from the empirical pass rates of the backbone model executing both reasoning modes. Similarly, Self-Route [18] also trains a lightweight external router to estimate the model's own capability for different levels of task complexity by analyzing its hidden layer representations. The router then dynamically decides whether to use slow thinking for the current problem at inference time. The levels of task complexity are computed by the accuracy of a proxy LLM.

While some recent LLMs, such as Qwen3 [49] and Llama-Nemotron [50], support hybrid reasoning modes, they still rely on user-specified reasoning strategies rather than enabling automatic reasoning-mode selection. Overall, complexity-based selection offers a robust and scalable framework for adaptive reasoning by explicitly tailoring model behavior to input difficulty.

5.2 Internal/external knowledge boundary

The internal/external knowledge boundary concerns the extent to which a model can rely on its internal parametric knowledge to solve a given task, versus when it must retrieve external knowledge sources with tool-augmented thinking. Key decision factors at this boundary are the *model confidence* and the *utility gain* from using an external tool.

5.2.1 Model Confidence

Using model confidence to navigate between the internal/external knowledge boundary leverages an LLM's own uncertainty to select the appropriate knowledge source. The core intuition is that an LLM can monitor its certainty about its own outputs: high confidence suggests the model can answer correctly using its internal knowledge, whereas low confidence triggers a switch to retrieval, computation, or another external resource.

One direction focuses on uncertainty modeling via output probabilities. The confidence is derived directly from the model's output probabilities, such as token-level likelihoods [21,32,38,52], or output entropy [51], then compared with a threshold to decide on tool usage. Notably, SUGAR [32] addresses the challenge that in natural language, the same idea can be expressed in various syntactic and lexical forms. It proposes using semantic entropy to determine the knowledge boundary, measuring uncertainty over meaning rather than specific token sequences. In the formulation presented in Section 2, semantic entropy serves as a confidence-related decision feature in the selection function, where higher semantic entropy indicates lower confidence and triggers RAG. Although these methods are simple and efficient, their performance depends heavily on calibration: overconfident models may skip necessary retrieval, while underconfident models may trigger excessive tool use [15,64].

A recent direction is to exploit consistency across multiple generated samples or within the model's latent space [54]. Instead of relying solely on output probabilities, these methods examine the model's hidden representations or the variability across different reasoning paths to better gauge its true certainty. Seakr [27], for instance, measures the consistency of an LLM's internal hidden states across multiple generations for the same prompt to decide when to retrieve external knowledge. Rowen [53] takes a similar approach by generating semantically equivalent perturbations of a query and measuring the consistency of answers across different languages and models. A low consistency score, indicating a potential hallucination, triggers the retrieval of external knowledge for correction. MeCo [28] instead derives a meta-cognitive score of consistency with a lightweight representation probe and applies a dual-threshold policy to decide whether to invoke external tools. While these methods can better capture a model's knowledge gaps, they often require access to model internals or multiple inference passes.

5.2.2 Utility gain

Selection methods that use utility gain as a decision factor decide whether to retrieve external information or invoke tools by estimating the expected benefit of doing so—e.g., reductions in perplexity, or improvements in accuracy.

Toolformer [39] formulates external tool use as utility-driven modeling by executing candidate API calls and retaining only those that reduce the perplexity of subsequent tokens. This self-supervised filtering implicitly learns a policy that adaptively triggers external tools without sacrificing the general abilities of LLMs. For scientific problem solving, AWL [55] leverages the solvability of a sample without tools as the decision factor, guiding the model to answer easy queries internally while invoking tools on hard ones.

More recently, RL-based approaches have been used to endow LLMs with adaptive tool selection. For mathematical problem-solving, ToCodeEM [56] employs an Expectation-Maximization (EM) framework where the E-step performs guided exploration, estimating the expected success of code-integrated versus pure reasoning paths through rollouts. The M-step then uses these utility

estimates in an off-policy RL objective to jointly optimize tool usage and reasoning.

In addition, ReTool [57] treats tool invocation as an action within a continuous reasoning process and optimizes the policy with a simple final-correctness reward. The model autonomously discovers how to interleave textual reasoning with code execution to maximize expected task accuracy. R3-RAG [58] combines a fine-grained process reward for the relevance of each retrieval step, and a final outcome reward for the correctness of the answer with an RL framework. This enables the model to learn an adaptive policy for interleaving its internal reasoning with external knowledge retrieval. Meanwhile, ARTIST [59] introduces a unified framework where models learn to dynamically interleave internal reasoning with external tool use as a coherent thinking process, optimized by GRPO.

Another line of work uses lightweight external routers to endow models with adaptive tool-use capabilities. For example, RAGate [29] employs a lightweight router that adaptively selects only internal thinking or tool-augmented thinking, trained with supervision labels derived from human preference. Toolken+ [60] further improves adaptive tool selection by learning a lightweight reranker that considers the top-k tool candidates alongside a special "Reject" option.

5.3 Discussion

Different decision factors capture distinct but complementary aspects of the reasoning strategy selection problem. These factors can be integrated to enable more robust and adaptive decision-making. Specifically, confidence reflects the model's certainty in its current response; complexity estimates the inherent difficulty of the input problem; and utility gain evaluates the expected payoff of allocating additional computational resources. However, relying on these decision factors introduces specific failure modes if they are inaccurately estimated. Most notably, when confidence estimates are poorly calibrated, overconfident models tend to skip necessary external retrieval or internal deliberation, leading to premature and potentially hallucinated responses [65–66]. On the other hand, underconfident models may invoke excessive reasoning steps or redundant tool use, introducing severe latency and unnecessary computational overhead. Furthermore, poorly estimated complexity or utility gain can result in a fundamental mismatch between the chosen reasoning strategy and the problem's actual requirements, compromising both accuracy and efficiency.

Notably, these decision factors closely align with human metacognitive processes, which involve monitoring and regulating one's own cognitive activity [6]. Just as humans adjust their reasoning strategies based on perceived confidence, task difficulty, and the potential payoff of further effort, language models can benefit from the similar mechanisms to dynamically select between fast, slow, or tool-augmented reasoning. Table 1 provides a summary of existing methods and their application across different knowledge boundaries.

6 Summary and future prospects

Adaptively selecting LLM reasoning strategies has continued to be an inspiring domain. Recent studies have begun to explore mechanisms that enable dynamic alignment of reasoning modes with the demands of specific tasks. As LLM capabilities continue to scale, their applications are rapidly expanding across critical domains such as education, healthcare, law, science, and engineering [67]. Developing efficient reasoning strategies can greatly improve their usability and trustworthiness, helping to reduce unnecessary computational costs and minimize hallucinations.

Looking ahead, we envision several promising directions for future research.

- Pre-training/mid-training foundation

While most existing work focuses on post-training mechanisms for reasoning strategy selection, it is during pre-training that LLMs acquire most of their internal knowledge and foundational cognitive capacities. This means that adaptive reasoning capabilities are primarily instilled during post-training, which can lead to superficial boundary awareness or catastrophic forgetting of general knowledge. To address this, future work should explore how knowledge boundaries can be explicitly modeled much earlier, during pre-training or mid-training. The field must determine whether this boundary awareness can be deeply integrated into large-scale pre-training without harming the model's general capabilities or downstream adaptability.

- Unified selection

Existing strategy selection methods often lack the capability to integrate multiple reasoning strategies in a seamless, human-like manner. More specifically, current approaches typically treat internal reasoning depth and external tool invocation as isolated, disjointed decisions, preventing models from fully optimizing their computational resources. A promising direction is to develop models that autonomously assess task complexity and dynamically select the optimal combination of strategies at each step. How can we jointly model these orthogonal dimensions to enable seamless, unified transitions across reasoning strategies?

- Reasoning orchestration systems

As multi-agent frameworks become more prevalent [68], current orchestration systems often struggle to resolve inter-agent conflicts and suffer from compounding inefficiencies as the system scales. Future frameworks must determine how to effectively delegate subtasks and ensure coherence in collective reasoning. A major challenge moving forward will be designing orchestration policies that maintain efficiency, interpretability, and robustness even as the number of agents and the complexity of their interactions increase.

- Multimodal selection

Human reasoning often integrates multiple modalities such as language, vision, action, and spatial awareness [69]. However, current adaptive reasoning research focuses heavily on text-based tasks, and models lack the flexibility to dynamically shift between modalities when solving complex, real-world problems. Future research must investigate how to effectively fuse and select among diverse reasoning modalities, including text, images, audio, video,

Table 1 Summary of representative reasoning strategy selection methods

Work	Boundary type	Decision factor	Base model	Open source	Application	Selection paradigm
DynaThink [7]	Fast/Slow	Model Confidence	GPT-3.5 Turbo; GPT-4; Gemini; Mixtral-8x7B	Yes	Math reasoning; Commonsense QA	Explicit
UnCert-CoT [30]	Fast/Slow	Model Confidence	Qwen2.5-Coder-7B-Base; deepSeek-coder-6.7B-base	No	Code generation	Explicit
SBT [12]	Fast/Slow	Model Confidence	Qwen2.5-Math-1.5B/7B-Instruct; Llama-3.2-1B-Instruct; Llama-3.1-8B-Instruct	Yes	Math reasoning	Explicit
MUR [22]	Fast/Slow	Model Confidence	Qwen3-1.7B/4B/8B	Yes	Math reasoning; Science QA	Explicit
Early-Exit [31]	Fast/Slow	Model Confidence	R1-Distill-Llama-8B/70B; R1-Distill-Qwen-1.5B/7B/32B; QwQ-32B	Yes	Math reasoning; Logic QA	Explicit
Z1 [48]	Fast/Slow	Task Complexity	Qwen2.5-Coder-7B-Instruct	Yes	Code reasoning; Math; Science QA	Implicit
AdaCoT [19]	Fast/Slow	Task Complexity	Internal MoE 15B/150B (ByteDance)	No	Math reasoning; General QA; Exams	Implicit
ThinkSwitcher [20]	Fast/Slow	Task Complexity	DeepSeek-R1-Distill-Qwen-1.5B/7B/14B	No	Math reasoning	Explicit
TLDR [37]	Fast/Slow	Task Complexity	DeepSeek-R1-Distill-7B/14B	No	Math reasoning	Implicit
SynapseRoute [41]	Fast/Slow	Task Complexity	Qwen3-30B-a3b	No	Medical QA	Implicit
FLARE [21]	Internal/External	Model Confidence	text-davinci-003	Yes	Multihop QA; Commonsense reasoning	Explicit
SelfRag [38]	Internal/External	Model Confidence	Llama-2-7B/13B	Yes	Open-domain QA; Fact verification	Implicit
SUGAR [32]	Internal/External	Model Confidence	Llama-2-chat-7B; contriever-msmarco	No	Open-domain QA	Explicit
Seakr [27]	Internal/External	Model Confidence	Llama-2-7B-Base/Chat; Llama-3-8B-Base/Instruct	Yes	Multi-hop QA; Open-domain QA	Explicit
Alignment [54]	Internal/External	Model Confidence	Llama-3.1-8B-Instruct; Qwen-2.5-7B-Instruct; DeepSeek-R1-671B	No	Arithmetic; Open-domain QA	Explicit
Toolformer [39]	Internal/External	Utility Gain	GPT-J 6.7B	No	Open-domain QA; Math reasoning	Implicit
AWL [55]	Internal/External	Utility Gain	Llama-3.1-8B-Instruct / Qwen2.5-14B-Instruct	Yes	Science QA; Math Solving	Implicit
ToCodeEM [56]	Internal/External	Utility Gain	Qwen2-Math-7B; Qwen-2.5-Base-7B; DeepSeekMath-Instruct-7B	Yes	Math reasoning	Implicit
ReTool [57]	Internal/External	Utility Gain	Qwen2.5-32B-Instruct / DeepSeek-R1-Distill-Qwen-32B	Yes	Math reasoning	Implicit
RAGate [29]	Internal/External	Utility Gain	Llama-2-7B / 13B / 70B; GPT-2; BERT-ranker; TF-IDF	Yes	Conversational systems	Explicit

and physical execution. Researchers have yet to answer how a model can dynamically evaluate and coordinate the most appropriate reasoning strategies across these different modalities when confronted with interleaved multimodal tasks.

● Personalized reasoning

Different users may require different styles or levels of reasoning depending on their goals, expertise, or preferences. In practice, most existing LLMs apply a one-size-fits-all reasoning approach, failing to adjust the depth, formality, or transparency of their reasoning chains based on specific user profiles or contexts. Developing models capable of personalizing their reasoning behavior is a highly promising avenue. However, it remains an open question how to reliably evaluate and support this personalization in highly specialized domains—such as sports coaching or legal advice—where task-specific training data and gold-standard evaluation criteria are severely limited.

● Robustness and safety:

As reasoning systems become more adaptive and autonomous, ensuring their robustness and safety becomes increasingly critical. With greater autonomy over their reasoning strategies, models often become vulnerable to blindly trusting unreliable external tools or failing to recognize when their adaptive strategies are compromised by distribution shifts. Ensuring robustness involves handling ambiguous inputs, avoiding unethical conclusions, and maintaining human alignment. What mechanisms can be developed to allow adaptive reasoning systems to autonomously detect when strategy switching or external tool invocation becomes unsafe?

Acknowledgements

The work was supported by the National Natural Science Foundation of China (Grant Nos. 62502310 and 62322603).

Competing interests

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

Open Access

This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made.

The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

To view a copy of this licence, visit website at creativecommons.org/licenses/by/4.0/.

References

[1] Anthropic. Claude sonnet 4. See anthropic.com/claude/sonnet website, 2025

[2] DeepSeek-AI. DeepSeek-R1: incentivizing reasoning capability in LLMs via reinforcement learning. 2025, arXiv preprint arXiv: 2501.12948

[3] OpenAI. OpenAI o3. See en.wikipedia.org/wiki/OpenAI_o3 website, 2025

[4] X.AI. Grok 4. See x.ai/news/grok-4 website, 2025

[5] Zhang D, Li Z Z, Zhang M L, Zhang J, Liu Z, Yao Y, Xu H, Zheng J, Chen X, Zhang Y, Yin F, Dong J, Guo Z, Song L, Liu C L. From system 1 to system 2: a survey of reasoning large language models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2026, 48(3): 3335−3354

[6] Kahneman D. Thinking, Fast and Slow. New York: Farrar, Straus and Giroux, 2011

[7] Pan J, Zhang Y, Zhang C, Liu Z, Wang H, Li H. DynaThink: fast or slow? A dynamic decision-making framework for large language models. In: Proceedings of 2024 Conference on Empirical Methods in Natural Language Processing. 2024, 14686−14695

[8] Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E H, 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 Systems. 2022, 1800

[9] Yao S, Yu D, Zhao J, Shafran I, Griffiths T L, Cao Y, Narasimhan K. Tree of thoughts: deliberate problem solving with large language models. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. 2023, 517

[10] Ji Z, Yu T, Xu Y, Lee N, Ishii E, Fung P. Towards mitigating LLM hallucination via self reflection. In: Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023. 2023, 1827−1843

[11] Liang Z, Liu Y, Niu T, Zhang X, Zhou Y, Yavuz S. Improving LLM reasoning through scaling inference computation with collaborative verification. 2024, arXiv preprint arXiv: 2410.05318

[12] Zhao H, Yan Y, Shen Y, Xu H, Zhang W, Song K, Shao J, Lu W, Xiao J, Zhuang Y. Let LRMs break free from overthinking via self-braking tuning. 2025, arXiv preprint arXiv: 2505.14604

[13] 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

[14] Sun Z, Wang Q, Wang H, Zhang X, Xu J. Detection and mitigation of hallucination in large reasoning models: a mechanistic perspective. 2025, arXiv preprint arXiv: 2505.12886

[15] Schuster T, Fisch A, Gupta J, Dehghani M, Bahri D, Tran V Q, Tay Y, Metzler D. Confident adaptive language modeling. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 1269

[16] Roberts A, Raffel C, Shazeer N. How much knowledge can you pack into the parameters of a language model? In: Proceedings of 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020, 5418−5426

[17] Moskvoretskii V, Marina M, Salnikov M, Ivanov N, Pletenev S, Galimzianova D, Krayko N, Konovalov V, Nikishina I, Panchenko A. Adaptive retrieval without self-knowledge? Bringing uncertainty back home. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2025, 6355−6384

[18] He Y, Ding X, Cai B, Zhang Y, Xiong K, Sun Z, Qin B, Liu T. Self-route: automatic mode switching via capability estimation for efficient reasoning. 2025, arXiv preprint arXiv: 2505.20664

[19] Lou C, Sun Z, Liang X, Qu M, Shen W, Wang W, Li Y, Yang Q, Wu S. AdaCoT: pareto-optimal adaptive chain-of-thought triggering via reinforcement learning. 2025, arXiv preprint arXiv: 2505.11896

[20] Liang G, Zhong L, Yang Z, Quan X. ThinkSwitcher: when to think hard, when to think fast. In: Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2025. 2025, 5185−5201

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

[22] Yan H, Xu F, Xu R, Li Y, Zhang J, Luo H, Wu X, Luu A T, Zhao H, Lin Q, Liu J. MUR: momentum uncertainty guided reasoning for large language models. In: Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2026, 23078−23103

[23] Zhang J, Lin N, Hou L, Feng L, Li J. AdaptThink: reasoning models can learn when to think. In: Proceedings of 2025 Conference on Empirical Methods in Natural Language Processing. 2025, 3716−3730

[24] Tu S, Lin J, Zhang Q, Tian X, Li L, Lan X, Zhao D. Learning when to think: shaping adaptive reasoning in R1-style models via multi-stage RL. 2025, arXiv preprint arXiv: 2505.10832

[25] Nogueira J P, Sun W, Silva A, Zumot L. Certainty-guided reasoning in large language models: a dynamic thinking budget approach.

2025, arXiv preprint arXiv: 2509.07820

[26] Yang C, Si Q, Duan Y, Zhu Z, Zhu C, Li Q, Lin Z, Cao L, Wang W. Dynamic early exit in reasoning models. 2025, arXiv preprint arXiv: 2504.15895

[27] Yao Z, Qi W, Pan L, Cao S, Hu L, Liu W, Hou L, Li J. SeaKR: self-aware knowledge retrieval for adaptive retrieval augmented generation. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2025, 27022−27043

[28] Li W, Li D, Dong K, Zhang C, Zhang H, Liu W, Wang Y, Tang R, Liu Y. Adaptive tool use in large language models with meta-cognition trigger. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2025, 13346−13370

[29] Wang X, Sen P, Li R, Yilmaz E. Adaptive retrieval-augmented generation for conversational systems. In: Proceedings of the Findings of the Association for Computational Linguistics: NAACL 2025. 2025, 491−503

[30] Zhu Y, Li G, Jiang X, Li J, Mei H, Jin Z, Dong Y. Uncertainty-guided chain-of-thought for code generation with LLMs. 2025, arXiv preprint arXiv: 2503.15341

[31] Zhang A, Chen Y, Pan J, Zhao C, Panda A, Li J, He H. Reasoning models know when they're right: probing hidden states for self-verification. In: Proceedings of the 2nd Conference on Language Modeling. 2025

[32] Zubkova H, Park J H, Lee S W. SUGAR: leveraging contextual confidence for smarter retrieval. In: Proceedings of 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2025, 1−5

[33] Verma N, Bharadwaj M, Jang W, Singh H, Wang Y, Fashandi H, Lee C. SMaRT: select, mix, and ReinvenT–A strategy fusion framework for LLM-driven reasoning and planning. 2025, arXiv preprint arXiv: 2510.18095

[34] Flavell J H. Metacognition and cognitive monitoring: a new area of cognitive–developmental inquiry. American Psychologist, 1979, 34(10): 906−911

[35] Sweller J. Cognitive load during problem solving: effects on learning. Cognitive Science, 1988, 12(2): 257−285

[36] Luo F, Chuang Y N, Wang G, Le H A D, Zhong S, Liu H, Yuan J, Sui Y, Braverman V, Chaudhary V, Hu X. AutoL2S: auto long-short reasoning for efficient large language models. In: Proceedings of the Findings of the Association for Computational Linguistics: ACL 2026. 2026, 16836−16858

[37] Li Z Z, Liang X, Tang Z, Ji L, Wang P, Xu H, Xing W, Huang H, Deng W, Wu Y N, Gong Y, Guo Z, Liu X, Yin F, Liu C L. *TL;DR*: too long, do re-weighting for efficient LLM reasoning compression. 2025, arXiv preprint arXiv: 2506.02678

[38] 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

[39] Schick T, Dwivedi-Yu J, Dessì 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

[40] Nakano R, Hilton J, Balaji S, Wu J, Ouyang L, Kim C, Hesse C, Jain S, Kosaraju V, Saunders W, Jiang X, Cobbe K, Eloundou T, Krueger G, Button K, Knight M, Chess B, Schulman J. WebGPT: browser-assisted question-answering with human feedback. 2021, arXiv preprint arXiv: 2112.09332

[41] Zhang W, Qiao S, Luo L, Li Y, Zheng C, Xu Q, Li M, Gui Y, He Y, Qiu J, Hong J, Sun J. SynapseRoute: an auto-route switching framework on dual-state large language model. 2025, arXiv preprint arXiv: 2507.02822

[42] Zhan Z, Deng K, Tang H, Xiang W, Wu K, Li W, Zhu W, Xu J, Huang L, Feng Z, Wang S, Yan S, Chen X, Liu J, Peng Z, Gao Z, Huang H, Zhang X, Wang J, Lin Z, Li M, Wang H, Zhan Z, Wu Y, Zhang Y, Yang J, Chen G, Zhang H, Chen B, Yu B. KAT-V1: kwai-AutoThink technical report. 2025, arXiv preprint arXiv: 2507.08297

[43] Jiang L, Wu X, Huang S, Dong Q, Chi Z, Dong L, Zhang X, Lv T, Cui L, Wei F. Think only when you need with large hybrid-reasoning models. In: Proceedings of the 39th Conference on Neural Information Processing Systems. 2025

[44] Huang Y, Chen S, Cai H, Dhingra B. To trust or not to trust? Enhancing large language models' situated faithfulness to external contexts. In: Proceedings of the 13th International Conference on Learning Representations. 2025

[45] LG AI Research. EXAONE 4.0: unified large language models integrating non-reasoning and reasoning modes. 2025, arXiv preprint arXiv: 2507.11407

[46] OpenAI. Introducing GPT-5. See startupdefense.io/learn/introducing-gpt-5 website, 2025

[47] Wu H, Xu B, Shu Y, Yang M, Qin C. Thinking with nothinking calibration: a new in-context learning paradigm in reasoning large language models. 2025, arXiv preprint arXiv: 2508.03363

[48] Yu Z, Wu Y, Zhao Y, Cohan A, Zhang X P. Z1: efficient test-time scaling with code. In: Proceedings of 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track. 2025, 2688−2712

[49] Qwen Team. Qwen3 technical report. 2025, arXiv preprint arXiv: 2505.09388

[50] NVIDIA. Llama-Nemotron: efficient reasoning models. 2025, arXiv preprint arXiv: 2505.00949

[51] Li J, Tang T, Zhao W X, Wang J, Nie J Y, Wen J R. The web can be your oyster for improving large language models. 2023, arXiv preprint arXiv: 2305.10998

[52] Su W, Tang Y, Ai Q, Wu Z, Liu Y. DRAGIN: dynamic retrieval augmented generation based on the real-time information needs of large language models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024, 12991−13013

[53] Ding H, Pang L, Wei Z, Shen H, Cheng X. Retrieve only when it needs: adaptive retrieval augmentation for hallucination mitigation in large language models. 2024, arXiv preprint arXiv: 2402.10612

[54] Xu H, Wang Z, Zhu Z, Pan L, Chen X, Fan S, Chen L, Yu K. Alignment for efficient tool calling of large language models. In: Proceedings of 2025 Conference on Empirical Methods in Natural Language Processing. 2025, 17776−17792

[55] Lyu B, Cao Y, Watson-Parris D, Bergen L, Berg-Kirkpatrick T, Yu R. Adapting while learning: grounding LLMs for scientific problems with intelligent tool usage adaptation. 2024, arXiv preprint arXiv: 2411.00412

[56] Wang H, Li L, Qu C, Xu W, Zhu F, Chu W, Lin F. To code or not to code? Adaptive tool integration for math language models via expectation-maximization. In: Proceedings of the Findings of the Association for Computational Linguistics: ACL 2025. 2025, 3060−3075

[57] Feng J, Huang S, Qu X, Zhang G, Qin Y, Zhong B, Jiang C, Chi J, Zhong W. ReTool: reinforcement learning for strategic tool use in LLMs. 2025, arXiv preprint arXiv: 2504.11536

[58] Li Y, Luo Q, Li X, Li B, Cheng Q, Wang B, Zheng Y, Wang Y, Yin Z, Qiu X. R3-RAG: learning step-by-step reasoning and retrieval for LLMs via reinforcement learning. In: Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2025. 2025, 10491−10507

[59] Singh J, Magazine R, Pandya Y, Nambi A. Agentic reasoning and tool integration for LLMs via reinforcement learning. 2025, arXiv preprint arXiv: 2505.01441

[60] Yakovlev K, Nikolenko S, Bout A. Toolken+: improving LLM tool usage with reranking and a reject option. In: Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2024. 2024, 5967−5974

[61] Pawitan Y, Holmes C. Confidence in the reasoning of large language models. Harvard Data Science Review, 2025, 7(1): 1−30

[62] 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

[63] Xu C, Sun Q, Zheng K, Geng X, Zhao P, Feng J, Tao C, Lin Q, Jiang D. WizardLM: empowering large pre-trained language models to follow complex instructions. In: Proceedings of the 12th International Conference on Learning Representations. 2024

[64] Mialon G, Fourrier C, Swift C, Wolf T, LeCun Y, Scialom T. GAIA: a benchmark for general AI assistants. In: Proceedings of the 12th International Conference on Learning Representations. 2024

[65] 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: ACL 2024. 2024, 11375−11388

[66] Huang L, Yu W, Ma W, Zhong W, Feng Z, Wang H, Chen Q, Peng W, Feng X, Qin B, Liu T. A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions. ACM Transactions on Information Systems, 2025, 43(2): 42

[67] Liu W, Qin J, Huang X, Zeng X, Xi Y, Lin J, Wu C, Wang Y, Shang L, Tang R, Lian D, Yu Y, Zhang W. The real barrier to LLM agent usability is agentic ROI. 2025, arXiv preprint arXiv: 2505.17767

[68] Tran K T, Dao D, Nguyen M D, Pham Q V, O'Sullivan B, Nguyen H D. Multi-agent collaboration mechanisms: a survey of LLMs. 2025, arXiv preprint arXiv: 2501.06322

[69] Lehmann J, Seufert T. The interaction between text modality and the learner's modality preference influences comprehension and cognitive load. Frontiers in Psychology, 2020, 10: 2820



Xinda JIA is currently a PhD student in the School of Mechanical Engineering at Shanghai Jiao Tong University, China. He received his BS degree in information and computational science from Jilin University, China in 2024. His research interests include large language models, complex reasoning, and embodied AI.



Jinpeng LI is currently a researcher at Huawei 2012 Labs. He received his PhD degree in computer science from Peking University, China in 2024. His research interests include large language models, post-training, reinforcement learning, and agentic AI. He has published more than 20 papers in top-tier international conferences and journals, and his research has resulted in 10 patents. He has also served as an Area Chair and reviewer for leading conferences, including NeurIPS, ACL, WWW, EMNLP, and IJCAI.



Zezhong WANG received the BS degree in network engineering from Hainan University, China in 2018, the MS degree in information engineering from The Chinese University of Hong Kong (CUHK), China in 2019, and the PhD degree in systems engineering and engineering management from CUHK in 2025. His research interests include LLM agents, dialogue systems, AI explainability, and AI safety. He has published several papers in top-tier AI and NLP conferences, including ICML, EMNLP, and NAACL.



Jingjing LI is a senior machine learning engineer at Roblox Corporation, specializing in LLMs and NLP. She received her PhD in computer science and engineering from The Chinese University of Hong Kong, China in 2023. Prior to joining Roblox, she worked as an applied scientist at TikTok E-Commerce, focusing on LLM post-training and intelligent conversational systems. Her research interests lie at the intersection of NLP, text generation, and LLM safety. She actively serves the academic community as an Area Chair for major NLP venues, including ACL ARR and EMNLP.



Xingshan ZENG is currently a researcher specializing in speech and semantics at Huawei Noah's Ark Lab. He received his PhD degree from The Chinese University of Hong Kong, China. His research interests include large language model agents and complex reasoning. He has published over 40 papers in leading international conferences and journals, received the ACL 2024 Outstanding Paper Award, and led the open-source ToolACE series of tool-augmented large language models, which have accumulated over 700,000 downloads on Hugging Face.



Yasheng WANG is a senior researcher specializing in large language models. He received his PhD degree in Engineering from Zhejiang University, China and is currently with Tencent. His research interests include pretraining and post-training, reinforcement learning, model

alignment, knowledge distillation, reasoning, and AI agents. His work focuses on building capable, efficient, and trustworthy language models with enhanced abilities in tool use, retrieval, code execution, task planning, and complex real-world problem solving.



Weinan ZHANG is currently a professor, PhD pdvisor, and assistant to the Dean at the School of Computer Science, Shanghai Jiao Tong University, China. His research areas include reinforcement learning, agentic AI, and embodied AI. He has published over 200 research papers at prestigious international conferences and journals, accumulating over 30k citations on Google Scholar, been selected as Elsevier China Highly Cited Researcher. He has been serving as an area chair at ICML, NeurIPS, ICLR, KDD, etc. and an associate editor at TPAMI. Weinan earned his PhD deree from the Computer Science Department of University College London, UK in 2016 and his BE degree from the ACM Honored Class of Shanghai Jiao Tong University, China in 2011.



Yong YU received his MS degree from the CS Department, East China Normal University, China. He is currently a professor with the Department of Computer Science, Shanghai Jiao Tong University, China and the Director of the Apex Data & Knowledge Management Lab. As the principal investigator, he took charge of several National Natural Science Foundation of China and China National High Tech (863) Program projects. His research interests include Web search, semantic search, data mining, and machine learning. He has published over 200 papers and served as a PC Member of several conferences, including WWW, RecSys, and a dozen of other related conferences, such as NIPS, ICML, SIGIR, ISWC and so on.



Weiwen LIU is currently an associate professor and PhD advisor in the School of Computer Science at Shanghai Jiao Tong University, China. She received her PhD degree from The Chinese University of Hong Kong, China in 2020. Her research interests include AI agents, large language models, and information retrieval. She has published over 80 papers in leading conferences and journals, received the Best Paper Award at DLP-RecSys and the Top Ten Vertical-Domain Large Models Award, and holds 12 patents. Her ToolACE series ranked first among open-source models on BFCL and has exceeded 700,000 downloads on Hugging Face. She has served as an area chair or reviewer for ICLR, ICML, and KDD.