Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-26)
Survey Track

A Review on Test-Time Scaling for Agentic Large Language Models

Jiayu An[1], **Zheng Chen**[1], **Yongcheng Jing**[2], **Dacheng Tao**[3] and **Bo Li**[1*]
[1]Hong Kong University of Science and Technology
[2]Wuhan University
[3]Nanyang Technological University
{anjiayu0618, zbellachen, yongchengjing, dacheng.tao}@gmail.com, bli@cse.ust.hk

Abstract

The field of *Large Language Models (LLMs)* is moving beyond simply scaling the parameters during pre-training. Instead, researchers are focusing on *Test-Time Scaling (TTS)*, which is a mechanism to allocate computational resources dynamically during the inference process. In the rapidly emerging field of *Agentic LLMs*, TTS has also shown great value in enabling agents to handle complex tasks that static models cannot handle. Despite all the substantial work in this field, it still lacks a comprehensive survey summarizing these advances. To the best of our knowledge, this paper presents the first systematic review of *TTS tailored for Agentic LLMs*. To this end, we propose a novel RAIE taxonomy along four scaling dimensions: (i) *Reasoning* optimizes the entire thought process through search algorithms and self-verification; (ii) *Acting* involves collecting information through external tools and environmental feedback; (iii) *Interaction* copes with complex tasks by forming various multi-agent networks; (iv) *Evolution* enables agents to continuously update their memory and strategies. Furthermore, we also introduce a task-oriented guideline for choosing the best TTS strategy. Finally, we sum up challenges and future directions to inspire further studies in this field. A curated list of papers and associated code will be made publicly available.

1 Introduction

Test-Time Scaling (TTS) improves reasoning performance by allocating additional computational resources during the inference stage [Shen *et al.*, 2025]. This approach allows models to generate more accurate and thoughtful responses than standard inference [Paglieri *et al.*, 2025]. *Agentic Large Language Models (LLMs)* represent a novel and vital research domain. These models exhibit higher levels of autonomy and decision-making capabilities than traditional agents [Team, 2025]. In this context, TTS provides immense value, enabling agents to handle complex long-range tasks [Wang *et al.*, 2025a]. For instance, *rStar-Math* [Guan *et al.*, 2025] shows how scaling internal reasoning deals with difficult mathematical problems. Additionally, the *Thinking vs. Doing* framework [Shen *et al.*, 2025] proves that scaling environmental interactions leads to sustained performance gains.

Despite the recent encouraging progress on TTS for agentic LLMs, current surveys exclusively focus on general TTS for non-agentic LLMs or neglect dynamic compute allocation as a primary scaling factor for agentic LLMs. For instance, *Deep Research Survey* [Zhang *et al.*, 2025f] investigates specialized autonomous agents, but does not address the broader principles of agentic TTS. Meanwhile, general surveys of LLM agents largely overlook dynamic test-time compute as a key dimension of scaling [Team, 2025]. To bridge this gap, we present, to the best of our knowledge, the first systematic survey of TTS for agentic LLMs. To achieve this goal, we propose a novel taxonomy organized around the RAIE framework. The taxonomy is illustrated in Figure 1. It categorizes methodologies into four critical dimensions: (i) **Internal Reasoning (R)** which uses algorithms like *Monte Carlo Tree Search* to scale cognitive depth [Yu *et al.*, 2025]; (ii) **Environmental Acting (A)** which explores how agents scale interaction with the world [Shen *et al.*, 2025]; (iii) **Collective Interaction (I)** where frameworks like *G-Designer* [Zhang *et al.*, 2025c] and *MAE* [Chen *et al.*, 2025c] optimize multi-agent collaboration; and (iv) **Continuous Evolution (E)** where methods like *EvoTest* [He *et al.*, 2025b] and *FLEX* [Cai *et al.*, 2025] allow agents to learn on-the-fly.

Building on our proposed taxonomy, we further introduce a task-oriented guideline that maps specific scaling mechanisms to diverse problem domains, aiming to facilitate both research development and practical adoption in the community. Finally, we summarize major challenges and outline several future research directions, highlighting system-level efficiency.

In sum, this paper for the first time provides a dedicated and comprehensive overview of agentic TTS via the proposed RAIE taxonomy. Leveraging this framework, we derive a roadmap in selecting appropriate scaling strategies. We further identify key challenges and promising research directions to spur continued progress in agentic TTS. Together, these contributions will consolidate the emerging landscape of agentic TTS, meanwhile serving as a practical toolkit for advancing future agentic TTS in real-world settings.

*Corresponding author

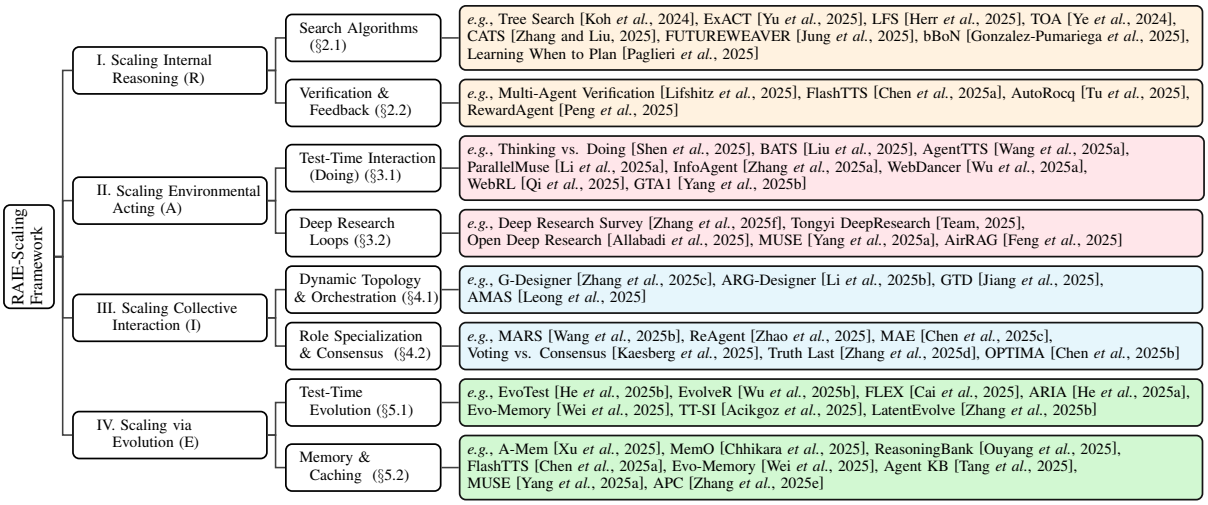

Figure 1: Proposed ***RAIE*** framework specifically tailored for *Agentic Test-Time Scaling*, comprising four key dimensions: *Reasoning*, *Acting*, *Interacting*, and *Evolving*.

2 Scaling Internal Reasoning

Scaling Internal Reasoning refers to the allocation of additional test-time compute to cognitive processes. It occurs especially within the agent and is independent of external environmental feedback. The essence of this paradigm is to dynamically scale the depth and breadth of thought during inference. In this framework, the agent builds a latent thought space to simulate and refine strategies, breaking complex problems into sub-goals and correcting errors before acting. As illustrated in the *RAIE framework*, internal reasoning acts as the commander, guiding the agent's "Acting" and "Interacting" processes. We categorize the methodologies for scaling internal reasoning into two complementary mechanisms. A summary of internal representative agentic TTS methods is provided in Table 1.

2.1 Search Algorithms

Search algorithms enable agents to explicitly explore the *thought space* or *action space* before execution. These methods can simulate potential trajectories and evaluate intermediate states. Current research in this domain can be categorized into the following three main streams.

Tree Search in Action Space. Traditional planning algorithms like *Monte Carlo Tree Search (MCTS)* have been successfully adapted, which perceives the reasoning process as a tree. Early attempts to scale inference compute focused on applying best-first search directly to environmental interactions. Koh *et al.* illustrate this by introducing a search algorithm that operates directly in a browser's state space [Koh *et al.*, 2024]. The agent uses a learned value function to evaluate action trajectories, backtracks from invalid paths, and trades increased inference time for higher success rates. Furthermore, recent works introduces reflection and cost constraints to standard search. *ExACT* [Yu *et al.*, 2025] integrates MCTS with a self-learning mechanism known as *Reflective-MCTS (R-MCTS)*. R-MCTS uses contrastive reflections to guide the search policy which compares successful trajectories with failed ones dynamically.

Addressing the budget aspect, *Cost-Augmented Monte Carlo Tree Search (CATS)* [Zhang and Liu, 2025] explicitly incorporates cost factors into tree expansion. It implements early pruning of infeasible solutions and prioritizes low-cost paths under resource constraints.

Adaptive and Self-Guided Exploration. Although structured search is powerful, fixed search heuristics may not align with the semantic capabilities of LLMs. A new methods focuses on empowering agentic LLMs to autonomously control the search process. Herr *et al.* proposed *LLM-First Search (LFS)*, which enables self-guided exploration [Herr *et al.*, 2025]. In LFS, the model acts as a controller, using internal scoring to dynamically decide between node expansion and backtracking. This removes the need for tuning and allows the search strategy to adapt to task difficulty. Moreover, The decision of *when* to search is as critical as *how* to search. Paglieri *et al.* investigated this trade-off in their work *Learning When to Plan* [Paglieri *et al.*, 2025]. They point out that always planning is computationally wasteful and can harm performance due to error accumulation. They propose a policy that dynamically allocates test-time compute. Activating planning only when task complexity requires it balances efficiency and performance.

Ensemble Search in Multi-Agent Systems. The search is not limited to single-agent thought process, but is increasingly applied to orchestrate multi-agent systems and select optimal trajectories from different rollouts. Scaling through generating multiple samples is a proven strategy, though selecting the best outcome remains challenging. *Behavior Best-of-N (bBoN)* [Gonzalez-Pumariega *et al.*, 2025] generates multiple rollouts and selects the best one based on abstracted behavior narratives. bBoN enables principled selection and substantially improves robustness. Moving beyond the selection of final trajectories, Ye *et al.* propose *Tree Search-based Orchestrated Agents (TOA)* [Ye *et al.*, 2024] to show interaction patterns can be optimized through search. TOA treats

the coordination of multiple agents as a search problem. It uses MCTS to dynamically optimize the collaboration graph for each input query. Similarly, *FUTUREWEAVER* [Jung *et al.*, 2025] introduces a framework for planning test-time compute under strict budget constraints. *FUTUREWEAVER* maximizes utility using a dual-level planning architecture to select and execute modularized collaboration patterns.

2.2 Verification & Logic Alignment

Although search algorithms provide the breadth to explore solution spaces, they cannot guarantee the correctness of the reasoning path. In long-horizon agentic tasks, a single logical fallacy may lead to failure of the entire system. Therefore, *Internal Reasoning Scaling* must also address the depth of processing. Specifically, how an agent assesses, verifies and corrects its own decisions. We categorize these advancements into *Verification & Logic Alignment*.

Collaborative Verification via Multi-Agent Systems. A defining characteristic of Agentic AI is the ability to distribute cognitive load across specialized roles. *Multi-Agent Verification (MAV)* [Lifshitz *et al.*, 2025] introduces a *Divide-and-Conquer* architecture for reliability. MAV deploys a suite of specialized verifier agents rather than a single scalar reward. That includes an *Edge-Case Inspector*, a *Domain Expert* and a *Logic Checker*. A central Meta-Verifier aggregates these diverse perspectives to form a final judgment. Similarly, *Agentic Reward Modeling (REWARDAGENT)* [Peng *et al.*, 2025] transforms the reward function itself into an agentic workflow and actively integrates heterogeneous signals. It combines human preference data with verifiable factuality checks and instruction-following constraints.

Neuro-Symbolic Integration for Formal Guarantees. To combat hallucinations in high-stakes domains, Agentic AI increasingly integrates with deterministic tools. The *Generate-Verify-Refine* closed loop is established to meet this requirement. *AUTOROCQ* [Tu *et al.*, 2025] illustrates this by coupling an LLM agent with the Rocq theorem prover. When the prover rejects a proof, the agent analyzes the formal error message and adjusts its strategy against ground-truth logic.

Runtime Verification & Efficiency. In open-ended environments, agent behaviors are unpredictable, which requires verification mechanisms that scale across the system lifecycle. Crucially, to support these intensive verification loops on resource-constrained edge devices, *FlashTTS* [Chen *et al.*, 2025a] introduces an *Asymmetric Multi-Model Memory Allocation* mechanism. It optimizes KV-cache management for the verification phase and deploys memory-hungry verifiers alongside generation models.

3 Scaling Environmental Acting

Internal Reasoning especially focuses on allocating compute to latent cognitive processes before acting. In contrast, Scaling Environmental Acting investigates how Agentic AI can enhance performance by dynamically scaling its interaction with external environments. In this paradigm, we unravel a new dimension of scaling where intelligence emerges from the feedback loop between the agent and the world.

3.1 Test-Time Interaction

TTI assumes that for information-sparse or dynamic tasks, the bottleneck is information acquisition rather than reasoning depth. Therefore, scaling the number of parallel explorations and length of interaction trajectories becomes critical. We categorize TTI methodologies into the following four distinct parts.

Foundations: Interaction as Compute. Traditional scaling laws focus on training compute or inference token counts. However, *Thinking vs. Doing* [Shen *et al.*, 2025] establishes the theoretical basis for Interaction Scaling. Their analysis shows that simply increasing reasoning time yields diminishing returns in complex environments. In contrast, scaling the number of interaction steps and allowing the agent to gather missing observations leads to performance gains. Formally, let C_{total} be the total test-time compute budget. It is made up of internal reasoning compute C_{int} and external interaction compute C_{ext}. The agent's performance P on a task $\mathcal{T}$ can be modeled as:

$$P(\mathcal{T}) \propto f(C_{int}, C_{ext}) \quad \text{s.t.} \quad C_{int} + C_{ext} \leq B, \tag{1}$$

where C_{ext} is defined by the number of environment steps and cost per action. Their work shows that shifting the budget from C_{int} to C_{ext} is often optimal for tasks requiring a grounding in external knowledge.

Strategic & Resource-Aware Acting. Agentic AI differentiates itself from standard agents by possessing *Meta-Cognition* with respect to its resource constraints. These systems dynamically optimize their interaction strategy based on the remaining budgets. *BATS (Budget-Aware Tool-Use)* [Liu *et al.*, 2025] introduces a dynamic planning mechanism. The agent monitors a continuous budget variable b_t. The policy $\pi(a_t|s_t, b_t)$ adapts to dig deeper only when b_t is high. *AgentTTS* [Wang *et al.*, 2025a] extends this to multi-stage complex tasks. It formulates interaction scaling as a global optimization problem. Given a task decomposed into subtasks $\{s_1, \ldots, s_k\}$, AgentTTS searches for the optimal model size M_i and interaction steps K_i. For each subtask, its goal is to maximize the total success rate $\mathcal{R}$:

$$\max_{\{M_i, K_i\}} \sum_{j=1}^{k} \mathcal{R}(s_j \mid M_j, K_j) \text{ s.t.} \sum_{j=1}^{k} \text{Cost}(M_j, K_j) \leq B_{total}. \tag{2}$$

In the domain of Graphical User Interfaces (GUI), *GTA1* [Yang *et al.*, 2025b] applies scaling to the action space itself. Recognizing that GUI actions are often irreversible, GTA1 performs test-time Action scaling by parallel sampling multiple action proposals and selecting the optimal trajectory.

Deep Information Orchestration. For tasks requiring deep research, Agentic AI acts as an orchestrator of information flow. *PARALLELMUSE* [Li *et al.*, 2025a] proposes *Agentic Parallel Thinking* which performs uncertainty-guided branching by partitioning the reasoning space into functional regions. Multiple agent instances could explore disparate information sources simultaneously. Complementing this, *INFOAGENT* [Zhang *et al.*, 2025a] introduces a self-hosted search infrastructure. An entity tree is constructed to decom-

Method	Core Mechanism	TTC	MA	Key Capabilities
	Scaling Internal Reasoning			
ExACT [Yu *et al.*, 2025]	Reflective-MCTS with contrastive self-learning	High	Yes	Logic Search
LFS [Herr *et al.*, 2025]	Self-guided exploration via internal scoring (LLM-First Search)	Medium	Yes	Adaptive Exploration
ReAgent [Zhao *et al.*, 2025]	Controller-Verifier architecture with recursive backtracking	High	Yes	Error Correction
AutoRocq [Tu *et al.*, 2025]	Integration with Rocq theorem prover for formal verification	High	Yes	Neuro-Symbolic Verify
	Scaling Environmental Acting			
Thinking vs. Doing [Shen *et al.*, 2025]	Scaling interaction steps (C_{ext}) vs. reasoning steps (C_{int})	Medium	Yes	Grounded QA
BATS [Liu *et al.*, 2025]	Budget-aware tool-use planning based on remaining resources	Low	Yes	Resource Management
WebRL [Qi *et al.*, 2025]	Self-evolving online curriculum via outcome-supervised RL	High	Yes	Web Navigation
MUSE [Yang *et al.*, 2025a]	Hierarchical memory for long-horizon deep research	Medium	Yes	Research Synthesis
	Scaling Collective Interaction			
G-Designer [Zhang *et al.*, 2025c]	GNN-based dynamic topology generation	Medium	No	Adaptive Collaboration
GTD [Jiang *et al.*, 2025]	Guided Topology Diffusion for structure denoising	High	Yes	Generative Topology
MARS [Wang *et al.*, 2025b]	Hierarchical role specialization (Author-Reviewer-MetaReviewer)	Medium	Yes	Academic/Logic QA
MAE [Chen *et al.*, 2025c]	Proposer-Solver-Judge co-evolution via RL	High	Yes	Self-Improving Roles
Truth Last [Zhang *et al.*, 2025d]	Positional weighting strategy for consensus	Low	Yes	Bias Mitigation
OPTIMA [Chen *et al.*, 2025b]	Training-based communication optimization	High	No	Efficient Consensus
	Scaling via Evolution			
EvoTest [He *et al.*, 2025b]	Evolutionary prompt rewriting by Actor-Evolver	Medium	Yes	Strategy Optimization
FLEX [Cai *et al.*, 2025]	Experience library with forward learning reflection	Medium	Yes	Experience Reuse
Evo-Memory [Wei *et al.*, 2025]	Streaming memory with active forgetting & updating	Medium	Yes	Long-horizon Memory
TT-SI [Acikgoz *et al.*, 2025]	Test-time gradient updates via self-data augmentation	Very High	No	Parameter Learning
APC [Zhang *et al.*, 2025e]	Caching & reusing structured plan templates (System Optim)	Low	Yes	Latency Reduction
FlashTTS [Chen *et al.*, 2025a]	Speculative beam extension & KV-cache reuse (System Optim)	Low	No	Efficient Serving

Table 1: A comprehensive summary of representative agentic TTS methods. We highlight core mechanisms in the RAIE framework with their computational overhead at test time, model agnosticism, and primary capabilities. Abbreviations: *TTC* (Test-Time Cost), *MA* (Model-Agnostic).

pose high-level queries into granular sub-tree sampling operations. Similarly, *WebDancer* [Wu *et al.*, 2025a] optimizes the end-to-end information seeking process by learning from browsing trajectories.

Active Exploration & Evolution. The most advanced form of TTI involves agents that evolve through interaction. The environment serves not just as a task space, but as a source of training signal. *WebRL* [Qi *et al.*, 2025] introduces a self-evolving online curriculum. WebRL employs a self-reflection mechanism to convert failed long-horizon attempts into simpler and solvable curriculum tasks. It is combined with an *Outcome-Supervised Reward Model (ORM)* to establish a closed loop system. The interaction directly promotes policy improvement internally:

$$\mathcal{T}_{new} \leftarrow \text{Reflect}(\tau_{fail}), \quad \pi_{t+1} \leftarrow \text{RL}(\pi_t, \mathcal{T}_{new}, \text{ORM}). \tag{3}$$

3.2 Deep Research Loops

While TTI scales the *breadth* of simple actions, *Deep Research Loops* scale the *temporal depth* and *complexity* of information acquisition. A deep research agent autonomously constructs a long-horizon investigation process until a certain information density is achieved.

The Deep Research Pipeline. The foundational structure of this paradigm is defined by the *Deep Research Survey* [Zhang *et al.*, 2025f]. This work formalizes the research lifecycle into four distinct stages: (1) *Planning*, (2) *Question Development*, (3) *Web Exploration*, and (4) *Report Generation*. Building on this pipeline, *Tongyi DeepResearch* [Team, 2025] demonstrates a full-stack implementation of an agentic model. The model features a recursively expanding planning mechanism that allows the agent to autonomously determine when to stop searching and start writing. Similarly, *Open Deep Research (ODR)* [Allabadi *et al.*, 2025] explores the strategic improvements required for open-source models to compete in this domain. By optimizing the *Search-and-Utilize* loop, ODR enables agents to navigate the open web dynamically.

Reasoning & Evolution within the Loop. To maximize the efficiency of the research loop, advanced agents employ reasoning-guided navigation and long-term memory. *AirRAG* [Feng *et al.*, 2025] integrates MCTS into the retrieval process. Reasoning actions could be expanded to dynamically explore the solution space. We can model the agent's objective as maximizing the cumulative Information Gain (IG) over a research horizon T:

$$\max_{\pi} \sum_{t=1}^{T} \mathbb{E}_{a_t \sim \pi} \left[IG(s_t, a_t) - \lambda \cdot \text{Cost}(a_t) \right], \tag{4}$$

where the policy π is optimized via MCTS to balance the value of new information against retrieval costs. Finally, *MUSE* [Yang *et al.*, 2025a] addresses the stateless limitation of traditional agents. It introduces a self-evolving framework with *Hierarchical Memory*. The agent abstracts successful trajectories into high-level skills stored in long-term memory to learn on the job.

4 Scaling Collective Interaction

For the aspect of single-agent scaling, researchers focus on internal depth or external breadth. *Scaling Collective Inter-*

action explores how intelligence emerges from the collaboration of multiple specialized agents. In this paradigm, the compute is scaled by increasing the number of active agents and optimizing their communication structures. Agentic AI is required to match the specific requirements of each query which is different from static multi-agent systems. Thus, *Dynamic Topology* is used to autonomously design and restructure the collaboration graph at test time.

4.1 Dynamic Topology & Orchestration

The core premise of dynamic topology is that the ideal collaboration structure is not invariant but query-dependent. A linear chain of commands is sufficient for simple queries. However, it requires a dense cyclic graph of experts for problems with more complexity. Therefore, the topology is treated as a learnable variable during the inference.

Generative Topology Design. Instead of selecting from predefined templates, advanced frameworks now generate collaboration graphs from scratch. *G-Designer* [Zhang *et al.*, 2025c] formulates topology construction as a graph generation problem. It employs a Graph Neural Network (GNN) and a Variational Graph Auto-Encoder (VGAE) to refine the communication structure. It dynamically adds or prunes edges to optimize information flow. *ARG-Designer* [Li *et al.*, 2025b] approaches this using an autoregressive generation paradigm. Given a complex task, it has the ability to sequentially select the necessary agent roles and construct a customized network. Taking a generative modeling approach, *GTD (Guided Topology Diffusion)* [Jiang *et al.*, 2025] models the search for an optimal topology as a denoising process. It utilizes a Graph Diffusion Model to sample collaboration structures from a continuous space. The goal is to optimize a multi-objective function that balances accuracy, cost, and robustness. Formally, given a query q, the system seeks the optimal graph $\mathcal{G}^*$ that maximizes the expected utility U:

$$\mathcal{G}^* = \underset{\mathcal{G}\in\Omega}{\operatorname{argmax}}\, \mathbb{E}\left[\mathcal{P}(\text{Solve}(q, \mathcal{G})) - \lambda \cdot \text{Cost}(\mathcal{G})\right] \quad (5)$$

where $\mathcal{P}$ represents the performance of the agent team executing under topology $\mathcal{G}$. Meanwhile, Cost penalizes excessive communication overhead.

Difficulty-Aware Orchestration. One of the key characteristics of Agentic AI is its Meta-Cognition. It stands for the ability to assess task difficulty and allocate resources accordingly. *MoMA (Mixture-of-Models-and-Agents)* [Guo *et al.*, 2025] proposes a flexible routing framework which orchestrates diverse computational units within. It decides whether to dispatch a query to a standalone LLM or to dynamically instantiate a specialized agent team.

Context-Adaptive Communication. Beyond structure and routing, the information flow within the graph must also adapt as well. *AMAS* [Leong *et al.*, 2025] proposes a mechanism to adaptively determine the communication topology based on the evolving context. At each step of the problem-solving process, agents determine which peers to query instead of broadcasting. It ends up with a real-time evolving and non-stationary communication graph.

4.2 Role Specialization & Consensus

While Dynamic Topology addresses *how* agents connect, *Role Specialization & Consensus* addresses *what* they do and *how* they agree. The mere duplication of homogeneous agents often leads to echo chambers where errors are amplified. In order to effectively scale reasoning, Agentic AI systems employ heterogeneous role definitions and robust consensus protocols.

Role-Based Architectures. By assigning distinct cognitive functions to different agents, a system can achieve the separation of concerns that enhances global performance. *MARS (Multi-Agent Review System)* [Wang *et al.*, 2025b] emulates the academic peer-review process and designates specific roles to carry out the mechanism. The structure involves an *Author* to generate solutions, *Reviewers* to provide independent critiques, and a *Meta-Reviewer* to make final decisions. Furthermore, *ReAgent* [Zhao *et al.*, 2025] introduces a *Controller-Verifier* architecture capable of backtracking in the system-level. When the Verifier detects a logical inconsistency, the Controller will initiate a rollback to a previous valid state. In general, a non-linear error-correction loop can be implemented during inference. Finally, *MAE (Multi-Agent Evolve)* [Chen *et al.*, 2025c] proposes a *Proposer-Solver-Judge* triangle where roles are co-evolved rather than static. Through the process of iterative interaction and reinforcement learning, each role gradually improves its specific ability. Thus, the system scales through both interaction and evolution.

Consensus Protocols & Optimization. Once diverse perspectives are generated, the system must aggregate them into a single coherent output. *Voting vs. Consensus* [Kaesberg *et al.*, 2025] provides a foundational analysis of decision-making protocols. It reveals that unanimity minimizes false positives in high-stakes knowledge tasks. However, it is more robust using *Majority Voting* for reasoning tasks. The paper suggests that the ideal aggregation rule is actually task-dependent. However, not all votes should count equally. *Key Decision-Makers* [Zhang *et al.*, 2025d] uncovers a positional bias in multi-agent debates. It proposes a *Truth Last* strategy. The system flexibly weights agents based on their consistency and speaking order to reduce the echo chamber effect.

Beyond fixed protocols, recent work has focused on optimizing the interaction itself through training. *OPTIMA* [Chen *et al.*, 2025b] introduces a *Communication Optimization* framework. OPTIMA could explicitly optimize the communication protocol itself by training on a *Generate-Rank-Select-Train* pipeline. This workflow teaches agents how to exchange information more efficiently and reach consensus with fewer communication turns. Formally, it maximizes a reward R that balances task success S and communication cost C:

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[S(\tau) - \lambda \cdot C(\tau)\right] \quad (6)$$

where τ represents the multi-agent interaction trajectory.

5 Scaling via Evolution

Previous scaling dimensions focus on expanding the search space, external actions, or agent quantity. *Scaling via Evolution* explores the temporal dimension instead. It addresses the

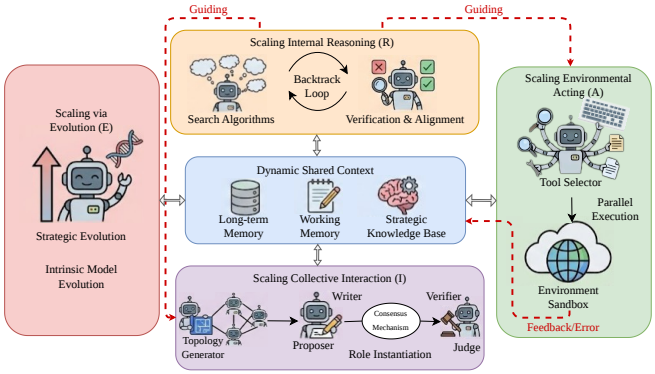

Figure 2: The cognitive architecture of agentic test-time compute.

fundamental limitation of static agents which is the inability to learn from experience after deployment. Agentic AI overcomes this by implementing a *Test-Time Training* loop. The system updates its strategies, memory, or even internal parameters on-the-fly.

5.1 Test-Time Evolution

Test-Time Evolution indicates that an agent's performance should increase with the number of tasks it processes. Agentic AI systems continuously learn during test phase to adapt to distribution shifts and personalize behavior without retraining. We categorize these approaches into three levels.

Strategic Evolution (Prompt & Principle Optimization). The most accessible way of evolution is refining the agent's high-level strategies or system prompts based on feedback. *EvoTest* [He *et al.*, 2025b] introduces a framework which includes an *Actor* and an *Evolver*. Instead of updating weights, the evolver analyzes the actor's failure trajectories and rewrites the system prompt. Similarly, *EvolveR* [Wu *et al.*, 2025b] formalizes an *Experience-Driven Lifecycle*. It distills successful interaction trajectories into abstract principles during an offline phase. These rules are retrieved to guide online decision-making. *FLEX* [Cai *et al.*, 2025] extends this by building a structured *Experience Library*. The agent reflects on both successes and failures to update the library, which prevents repeated mistakes.

Memory & Knowledge Adaptation. For agents operating in dynamic environments, evolution requires maintaining an up-to-date state of the world. *Evo-Memory* [Wei *et al.*, 2025] proposes a self-evolving memory system. The process of *Forgetting, Updating, and Consolidating* ensures that the context remains relevant over long interaction streams. Addressing the challenge of concept drift, *ARIA* [He *et al.*, 2025a] introduces an *Adaptive Reflective* mechanism. When ARIA detects inconsistencies between internal knowledge and external observations, it will update a timestamped knowledge repository to renovate its knowledge base.

Intrinsic Model Evolution (Parameter & Latent Scaling). The deepest form of evolution involves modifying the model's internal representations during inference. The authors of *LatentEvolve* [Zhang *et al.*, 2025b] implement a dual-process evolution. "Daytime" scaling retrieves historical latent states in reasoning, while "Nighttime" memory consolidation optimizes latent representations. The model can scale its capability in the continuous latent space rather than discrete token space. Finally, *TT-SI (Test-Time Self-Improvement)* [Acikgoz *et al.*, 2025] employs an uncertainty-based mechanism to identify weak spots in the agent's knowledge. The model generates synthetic training data via *Self-Data Augmentation*, and performs immediate fine-tuning. Formally, the agent updates its parameters θ at test step t to minimize the uncertainty $\mathcal{U}$ on the current task distribution $\mathcal{D}_t$:

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta \mathbb{E}_{x \sim \mathcal{D}_t} \left[\mathcal{L}_{self}(x, \theta_t) + \lambda \mathcal{U}(x, \theta_t) \right], \quad (7)$$

where $\mathcal{L}_{self}$ represents the self-supervised loss on generated data.

5.2 Memory & Caching

Effective TTS also requires a mechanism to persist and efficiently retrieve the accrued intelligence. Without structured memory, an agent will be forced to reason from scratch for every query. *Memory & Caching* establishes the infrastructure for long-term state management. It is the key for Agentic AI to maintain continuity and trade storage for compute by reusing past reasoning.

Cognitive Memory Architectures. To support complex reasoning, memory must be more than a flat list of text chunks. *A-Mem* [Xu *et al.*, 2025] draws inspiration from structuring memory as a network of interconnected notes. A-Mem employs dynamic indexing and linking to replace static vector databases. The relationships between memory nodes are reorganized when new information comes in. *MemO* [Chhikara *et al.*, 2025] advances this by adopting a *Graph-Based Memory* where entities and their relations are explicitly modeled. The model captures the subtle dependencies in long conversations and accomplishes multi-hop retrieval. Furthermore, *MUSE* [Yang *et al.*, 2025a] contributes a *Hierarchical Memory Architecture*. It explicitly separates storage into "Procedural Memory" (abstract high-level skills and plans) and "Episodic Memory" (specific execution trajectories). This allows the agent to retrieve generalizable strategies for novel tasks without the noise of historical details.

Experience & Knowledge Banks. Beyond internal state, Agentic AI can also scale by accessing shared or accumulated experience repositories. *Agent KB* [Tang *et al.*, 2025] proposes a universal memory infrastructure that enables *Cross-Framework Knowledge Transfer*. It allows agents built on different architectures to share solutions through standardizing experience storage. *ReasoningBank* [Ouyang *et al.*, 2025] focuses on storing *Generalizable Reasoning Strategies*. Instead of raw data, it distills all trajectories into a bank of reasoning patterns that serve as high-level cognitive templates. Addressing the temporal dimension, *Evo-Memory* [Wei *et al.*, 2025] introduces a *Streaming Memory Container*. Memory is structured as a continuous stream, while data is temporally ordered. This design forces the system to perform active memory management which emulates a working memory buffer.

Efficient Caching Strategies. Finally, caching at system-level is crucial for making TTS computationally feasible. *APC (Agentic Plan Caching)* [Zhang *et al.*, 2025e] introduces a specialized cache for *Structured Plan Templates*. High-level

planning is computationally expensive and often repetitive. APC caches and adapts abstract plans to redundant reasoning and reduce latency. At the serving layer, *FlashTTS* [Chen *et al.*, 2025a] optimizes the underlying inference engine. It implements *Speculative Beam Extension* and efficient *KV-Cache Reuse* strategies. This design achieves the theoretical benefits of memory scaling especially for irregular reasoning paths.

Taken together, as shown in Figure 2, the reviewed works aforementioned in the previous sections indicate that TTS is best viewed as a design space for adaptive inference rather than as a single technique. Despite differing implementations, these methods share a common principle: allocating computation dynamically by treating inference as iterative search and policy refinement over trajectories, with shared context and feedback enabling controllable trade-offs between compute and reliability.

6 Challenges and Future Directions

TTS has shown that more inference compute yields intelligence gains comparable to scaling parameters. However, the field is still in its early stages and faces many challenges. Before discussing future directions, we propose a practical scaling roadmap to contribute to addressing the challenges.

Figure 3 presents a roadmap for selecting TTS strategies. Users assess whether a task is self-contained at first. Branch A deals with tasks based on internal reasoning and provides paths for both logical precision and creative exploration. Branch B focuses on open environments and recommends strategies based on different type of information. Finally, evaluating system latency helps to choose between efficient caching and robust collective interaction. This framework aims to help researchers deploy agentic systems and advance this field in the following three directions for future research.

Unified Scaling Laws for Reasoning and Acting. Currently, scaling internal reasoning (*e.g.*, via latent search) and external acting (*e.g.*, via tool use) are often studied in isolation. However, optimal performance is highly likely to occur in a dynamic equilibrium between the two. As highlighted by Shen *et al.* [Shen *et al.*, 2025], there is a critical need to formulate unified scaling laws that can mathematically predict the marginal utility in allocating compute. Future systems should act as meta-controllers to arbitrate between deep internal verification and broad external information seeking based on the specific entropy of the query [Zhao *et al.*, 2025].

System-Level Efficiency and Caching. The prohibitive latency and cost of TTS remain major bottlenecks for real-world deployment. System-level optimization techniques such as *FlashTTS* [Chen *et al.*, 2025a] and *Agentic Plan Caching (APC)* [Zhang *et al.*, 2025e] have shown that caching structured plans and optimizing KV-cache reuse can significantly reduce overhead. Future work must bridge the gap between AI algorithms and system infrastructure. Specialized serving engines must handle irregular computation paths in dynamic topologies while without sacrificing responsiveness [Zhang *et al.*, 2025c] .

Reliability in Continuous Self-Evolution. Moving from static agents to self-evolving systems introduces the risks of instability and concept drift. Frameworks like *EvoTest* [He *et al.*, 2025b] and *FLEX* [Cai *et al.*, 2025] enable agents to optimize their strategies on-the-fly. Ensuring that such updates do not lead to catastrophic forgetting or performance degradation is significant. Future research should focus on robust mechanisms that allow agents to learn from test-time interactions (*e.g.*, through streaming memory in *Evo-Memory* [Wei *et al.*, 2025]). Meanwhile, they should also be able to guarantee the consistency and safety in their decision-making processes.

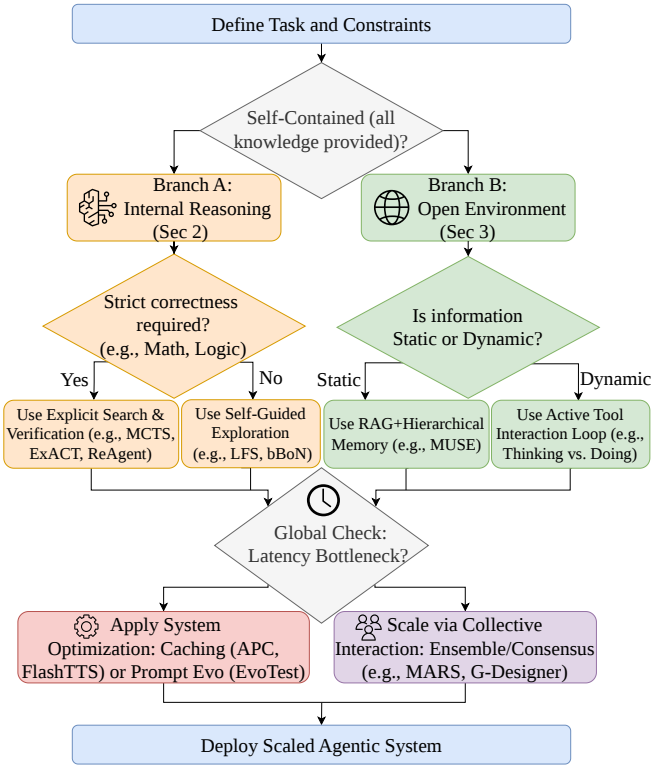

Figure 3: A practical scaling roadmap for agentic TTS.

7 Conclusion

In this paper, motivated by the paradigm shift in scaling from training-time parameter growth to test-time compute allocation, we for the first time systematically review emerging methodologies tailored for *agentic LLMs* that scale their test-time computation. To this end, we organize prior work along key dimensions, including deepening internal reasoning, expanding external action spaces, orchestrating collective interactions, and supporting continuous self-evolution. Collectively, these approaches reconceptualize inference as a generative and adaptive search process rather than a static model readout, revealing a new axis for improving capability and robustness. Future research should give priority to unified scaling laws and system efficiency to bridge the gap with actual deployment. These advances are crucial to achieving fully autonomous and self-evolving agents. We hope our work can provide a unifying perspective and serve as a foundation for future research on TTS in agentic LLMs.

Acknowledgements

Bo Li's research was supported in part by an NSFC grant 62432008, RGC RIF grant R6021-20, an RGC TRS grant T43-513/23N-2, RGC CRF grants C7004-22G, C1029-22G and C6015-23G, NSFC/RGC grant CRS_HKUST601/24 and RGC GRF grants 16207922, 16207423 and 16203824.

References

[Acikgoz *et al.*, 2025] Emre Can Acikgoz, Cheng Qian, Heng Ji, Dilek Hakkani-Tür, and Gokhan Tur. Self-improving llm agents at test-time. *arXiv preprint arXiv:2510.07841*, 2025.

[Allabadi *et al.*, 2025] Doaa Allabadi, Kyle Bradbury, and Jordan M Malof. Improving and evaluating open deep research agents. *arXiv preprint arXiv:2508.10152*, 2025.

[Cai *et al.*, 2025] Zhicheng Cai, Xinyuan Guo, Yu Pei, Jiangtao Feng, Jinsong Su, Jiangjie Chen, Ya-Qin Zhang, Wei-Ying Ma, Mingxuan Wang, and Hao Zhou. Flex: Continuous agent evolution via forward learning from experience. *arXiv preprint arXiv:2511.06449*, 2025.

[Chen *et al.*, 2025a] Hao Mark Chen, Zhiwen Mo, Guanxi Lu, Shuang Liang, Lingxiao Ma, Wayne Luk, and Hongxiang Fan. Democratizing agentic ai with fast test-time scaling on the edge. *arXiv preprint arXiv:2509.00195*, 2025.

[Chen *et al.*, 2025b] Weize Chen, Jiarui Yuan, Chen Qian, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Optima: Optimizing effectiveness and efficiency for llm-based multi-agent system. In *ACL (Findings)*, 2025.

[Chen *et al.*, 2025c] Yixing Chen, Yiding Wang, Siqi Zhu, Haofei Yu, Tao Feng, Muhan Zhang, Mostofa Patwary, and Jiaxuan You. Multi-agent evolve: Llm self-improve through co-evolution. *arXiv preprint arXiv:2510.23595*, 2025.

[Chhikara *et al.*, 2025] Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

[Feng *et al.*, 2025] Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Jingyi Song, and Hao Wang. AirRAG: Autonomous strategic planning and reasoning steer retrieval augmented generation. In *EMNLP (Findings)*, 2025.

[Gonzalez-Pumariega *et al.*, 2025] Gonzalo Gonzalez-Pumariega, Vincent Tu, Chih-Lun Lee, Jiachen Yang, Ang Li, and Xin Eric Wang. The unreasonable effectiveness of scaling agents for computer use. *arXiv preprint arXiv:2510.02250*, 2025.

[Guan *et al.*, 2025] Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rstar-math: Small LLMs can master math reasoning with self-evolved deep thinking. In *ICML*, 2025.

[Guo *et al.*, 2025] Xiyu Guo, Shan Wang, Chunfang Ji, Xuefeng Zhao, Wenhao Xi, Yaoyao Liu, Qinglan Li, Chao Deng, and Junlan Feng. Towards generalized routing: Model and agent orchestration for adaptive and efficient inference. *arXiv preprint arXiv:2509.07571*, 2025.

[He *et al.*, 2025a] Yufei He, Ruoyu Li, Alex Chen, Yue Liu, Yulin Chen, Yuan Sui, Cheng Chen, Yi Zhu, Luca Luo, Frank Yang, et al. Enabling self-improving agents to learn at test time with human-in-the-loop guidance. In *EMNLP*, 2025.

[He *et al.*, 2025b] Yufei He, Juncheng Liu, Yue Liu, Yibo Li, Tri Cao, Zhiyuan Hu, Xinxing Xu, and Bryan Hooi. Evotest: Evolutionary test-time learning for self-improving agentic systems. *arXiv preprint arXiv:2510.13220*, 2025.

[Herr *et al.*, 2025] Nathan Herr, Tim Rocktäschel, and Roberta Raileanu. Llm-first search: Self-guided exploration of the solution space. *arXiv preprint arXiv:2506.05213*, 2025.

[Jiang *et al.*, 2025] Eric Hanchen Jiang, Guancheng Wan, Sophia Yin, Mengting Li, Yuchen Wu, Xiao Liang, Xinfeng Li, Yizhou Sun, Wei Wang, Kai-Wei Chang, et al. Dynamic generation of multi-llm agents communication topologies with graph diffusion models. *arXiv preprint arXiv:2510.07799*, 2025.

[Jung *et al.*, 2025] Dongwon Jung, Peng Shi, and Yi Zhang. Futureweaver: Planning test-time compute for multi-agent systems with modularized collaboration. *arXiv preprint arXiv:2512.11213*, 2025.

[Kaesberg *et al.*, 2025] Lars Benedikt Kaesberg, Jonas Becker, Jan Philip Wahle, Terry Ruas, and Bela Gipp. Voting or consensus? decision-making in multi-agent debate. In *ACL (Findings)*, 2025.

[Koh *et al.*, 2024] Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents. *arXiv preprint arXiv:2407.01476*, 2024.

[Leong *et al.*, 2025] Hui Yi Leong, Yuheng Li, Yuqing Wu, Wenwen Ouyang, Wei Zhu, Jiechao Gao, and Wei Han. Amas: Adaptively determining communication topology for llm-based multi-agent system. In *EMNLP*, 2025.

[Li *et al.*, 2025a] Baixuan Li, Dingchu Zhang, Jialong Wu, et al. Parallelmuse: Agentic parallel thinking for deep information seeking. *arXiv preprint arXiv:2510.24698*, 2025.

[Li *et al.*, 2025b] Shiyuan Li, Yixin Liu, Qingsong Wen, Chengqi Zhang, and Shirui Pan. Assemble your crew: Automatic multi-agent communication topology design via autoregressive graph generation. *arXiv preprint arXiv:2507.18224*, 2025.

[Lifshitz *et al.*, 2025] Shalev Lifshitz, Sheila A. McIlraith, and Yilun Du. Multi-agent verification: Scaling test-time compute with multiple verifiers. In *ICLR Workshop*, 2025.

[Liu *et al.*, 2025] Tengxiao Liu, Zifeng Wang, Jin Miao, et al. Budget-aware tool-use enables effective agent scaling. *arXiv preprint arXiv:2511.17006*, 2025.

[Ouyang *et al.*, 2025] Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*, 2025.

[Paglieri *et al.*, 2025] Davide Paglieri, Bartłomiej Cupiał, Jonathan Cook, Ulyana Piterbarg, Jens Tuyls, Edward Grefenstette, Jakob Nicolaus Foerster, Jack Parker-Holder, and Tim Rocktäschel. Learning when to plan: Efficiently allocating test-time compute for llm agents. *arXiv preprint arXiv:2509.03581*, 2025.

[Peng *et al.*, 2025] Hao Peng, Yunjia Qi, Xiaozhi Wang, Zijun Yao, Bin Xu, Lei Hou, and Juanzi Li. Agentic reward modeling: Integrating human preferences with verifiable correctness signals for reliable reward systems. In *ACL*, 2025.

[Qi *et al.*, 2025] Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, et al. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *ICLR*, 2025.

[Shen *et al.*, 2025] Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amrith Setlur, Shengbang Tong, Diego Caples, Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. Thinking vs. doing: Improving agent reasoning by scaling test-time interaction. In *NeurIPS*, 2025.

[Tang *et al.*, 2025] Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, et al. Agent kb: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*, 2025.

[Team, 2025] Tongyi DeepResearch Team. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

[Tu *et al.*, 2025] Haoxin Tu, Huan Zhao, Yahui Song, Mehtab Zafar, Ruijie Meng, and Abhik Roychoudhury. Agentic program verification. *arXiv preprint arXiv:2511.17330*, 2025.

[Wang *et al.*, 2025a] Fali Wang, Hui Liu, Zhenwei DAI, Jingying Zeng, Zhiwei Zhang, Zongyu Wu, Chen Luo, Zhen Li, Xianfeng Tang, Qi He, and Suhang Wang. AgentTTS: Large language model agent for test-time compute-optimal scaling strategy in complex tasks. In *NeurIPS*, 2025.

[Wang *et al.*, 2025b] Xiao Wang, Jia Wang, Yijie Wang, Pengtao Dang, Sha Cao, and Chi Zhang. Mars: toward more efficient multi-agent collaboration for llm reasoning. *arXiv preprint arXiv:2509.20502*, 2025.

[Wei *et al.*, 2025] Tianxin Wei, Noveen Sachdeva, Benjamin Coleman, et al. Evo-memory: Benchmarking llm agent test-time learning with self-evolving memory. *arXiv preprint arXiv:2511.20857*, 2025.

[Wu *et al.*, 2025a] Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhenglin Wang, Zhengwei Tao, Ding-Chu Zhang, Zekun Xi, Xiangru Tang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Webdancer: Towards autonomous information seeking agency. In *NeurIPS*, 2025.

[Wu *et al.*, 2025b] Rong Wu, Xiaoman Wang, Jianbiao Mei, et al. Evolver: Self-evolving llm agents through an experience-driven lifecycle. *arXiv preprint arXiv:2510.16079*, 2025.

[Xu *et al.*, 2025] Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for LLM agents. In *NeurIPS*, 2025.

[Yang *et al.*, 2025a] Cheng Yang, Xuemeng Yang, Licheng Wen, et al. Learning on the job: An experience-driven self-evolving agent for long-horizon tasks. *arXiv preprint arXiv:2510.08002*, 2025.

[Yang *et al.*, 2025b] Yan Yang, Dongxu Li, Yutong Dai, et al. Gta1: Gui test-time scaling agent. *arXiv preprint arXiv:2507.05791*, 2025.

[Ye *et al.*, 2024] Hai Ye, Mingbao Lin, Hwee Tou Ng, and Shuicheng Yan. Multi-agent sampling: Scaling inference compute for data synthesis with tree search-based agentic collaboration. *arXiv preprint arXiv:2412.17061*, 2024.

[Yu *et al.*, 2025] Xiao Yu, Baolin Peng, Vineeth Vajipey, Hao Cheng, Michel Galley, Jianfeng Gao, and Zhou Yu. Exact: Teaching ai agents to explore with reflective-mcts and exploratory learning. In *ICLR*, 2025.

[Zhang and Liu, 2025] Zihao Zhang and Fei Liu. Cost-augmented monte carlo tree search for llm-assisted planning. *arXiv preprint arXiv:2505.14656*, 2025.

[Zhang *et al.*, 2025a] Gongrui Zhang, Jialiang Zhu, Ruiqi Yang, et al. Infoagent: Advancing autonomous information-seeking agents. *arXiv preprint arXiv:2509.25189*, 2025.

[Zhang *et al.*, 2025b] Guibin Zhang, Fanci Meng, Guancheng Wan, Zherui Li, Kun Wang, Zhenfei Yin, Lei Bai, and Shuicheng Yan. Latentevolve: Self-evolving test-time scaling in latent space. *arXiv preprint arXiv:2509.24771*, 2025.

[Zhang *et al.*, 2025c] Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. G-designer: Architecting multi-agent communication topologies via graph neural networks. In *ICML*, 2025.

[Zhang *et al.*, 2025d] Qian Zhang, Yan Zheng, Jinyi Liu, Hebin Liang, and Lanjun Wang. Key decision-makers in multi-agent debates: Who holds the power? *arXiv preprint arXiv:2511.11040*, 2025.

[Zhang *et al.*, 2025e] Qizheng Zhang, Michael Wornow, and Kunle Olukotun. Agentic plan caching: Test-time memory for fast and cost-efficient LLM agents. In *NeurIPS*, 2025.

[Zhang *et al.*, 2025f] Wenlin Zhang, Xiaopeng Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Huifeng Guo, Yong Liu, and Xiangyu Zhao. Deep research: A survey of autonomous research agents. *arXiv preprint arXiv:2508.12752*, 2025.

[Zhao *et al.*, 2025] Xinjie Zhao, Fan Gao, Xingyu Song, Yingjian Chen, Rui Yang, Yanran Fu, Yuyang Wang, Yusuke Iwasawa, Yutaka Matsuo, and Irene Li. Reagent: Reversible multi-agent reasoning for knowledge-enhanced multi-hop qa. In *EMNLP*, 2025.