Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 4, 2026

AI-DRIVEN SELF-REFLECTIVE MECHANISMS FOR GENERATIVE AGENTS: AUTONOMOUS PROMPT REVISION AND OPTIMIZATION

Saba Yousha[*1], Engr. Maroof Ahmed[2], Muhammad Amin Gilal[3], Abdullah Maitlo[4]

[*1]*Lecturer, Department of Information and Communication Technologies, The Begum Nusrat Bhutto Women University Sukkur, Sindh, Pakistan*
[2]*Faculty of Physical Science, Institute of Computer Science, Shah Abdul Latif University Khairpur Sindh Pakistan*
[3]*Faculty of Physical Science, Institute of Computer Science, Shah Abdul Latif University Khairpur*
[4]*Professor, Institute of Computer Science, Shah Abdul Latif University, Khairpur Mir's*

[*1]saba.yousha@bnvwu.edu.pk, [2]maroofahmednomi@gmail.com, [3]mameen.gilal@salu.edu.pk, [4]abdullah.maitlo@salu.edu.pk

DOI: https://doi.org/10.5281/zenodo.19482129

Keywords
Self-Reflection, Prompt Optimization, Generative Agents, Agentic Context Engineering, LLM Adaptation

Article History
Received: 12 February 2026
Accepted: 22 March 2026
Published: 09 April 2026

Corresponding Author: *
Saba Yousha

Abstract
The speedy development of large language models (LLMs) has changed the paradigm of AI optimization to focus on fine-tuning based on weights to one based on context. This paper discusses self-reflective processes, driven by AI, to allow generative agents to automatically revise and optimize prompts without human intervention or any adjustment of parameters. We combine current developments in Agentic Context Engineering (ACE), self-reflective systems like Reflexion and new prompt optimization systems like ZERA, GreenTEA, and IROTE. Through our analysis, we have found that self-reflective architectures (including generator, reflector and curator components) always perform better than the static prompting techniques in reasoning, code generation and domain specific tasks. There is empirical evidence of 1017 percent improvement in performance and a reduction in latency of adaptation up to 87 percent. We determine three central design concepts of successful implementations namely structured feedback generation, incremental context evolution and multi-criteria evaluation. This paper ends by mentioning limitations such as computational overhead, hallucination risks, and verification errors that explain about 70 percent of reasoning failures and giving future directions of robust and verifiable self-reflection by autonomous AI systems.

1. INTRODUCTION

Their application to large language models (LLMs) in complex, real-world settings has exposed a cardinal weakness in the form of static prompts, no matter how high quality they initially are, to adapt to the emergent edge cases, evolving task demands or subtle user intent. Conventional solutions to this issue have been based on the supervised fine-tuning (SFT) or reinforcement learning based on human feedback (RLHF)-solutions, which need to use large amounts of computational resources, manually curated data, and human supervision. They are not only costly but also fragile; the retraining cycle is required every time a change takes place and enhancements in one activity tend to hamper performance in another.

To counter these constraints, the new paradigm has seen the development of AI-based self-reflective systems which allow generative agents to update and optimize their own prompts independently. These systems are based on

metacognitive processes in human cognition, namely the capability to think about thinking, so that an LLM produces an initial output, criticizes the performance, and rewrites its instructions according to the weaknesses detected. The consequence is a closed loop system which gets continuously enhanced without updating the weights or having human labelling or supervision. This paper gives a detailed commentary on modern self-reflective processes of autonomous prompt revision and maximization. We answer the following three main research questions:

1. **RQ1:** Which architectural designs and structures can provide successful self-reflection of generative agents?
2. **RQ2:** What are these mechanisms which allow performance gains without updating the parameters?
3. **RQ3:** What does the existing empirical evidence tell us about the effectiveness of self-reflective optimization in various areas of tasks?

We make three contributions: first, we frame the existing fragmented body of literature concerning self-reflective AI into a conceptual framework; second, we provide quantitative data on the recent benchmark studies showing significant improvement in performance; and third, we determine the principles of design and limitation that future research and practical application can draw.

2. Theoretical Foundations

2.1 Self-Reflection as Metacognitive Analogy

Self-reflection in LLMs is the ability of a model to analyze, criticize, and enhance its outputs in a structured and repeated way. This property is not an emergent feature of the standard autoregressive models but needs to be carefully designed by prompting strategies or architectural components. Its theoretical basis is based on metacognitive theory that draws the difference between two cognitive levels, namely, the object-level (generation of responses) and the meta-level (evaluation and control of the processes of generation).

In reality, self-reflection takes the form of the three-step cycle, which is evaluation, generation of feedback, and revision. The model generates a candidate output and then compares the output with explicit or implicit criteria then it generates a new version that has its own critique. This cycle can be repeated several times and each repetition can provide some changes that can be improved each time.

2.2 Distinction from Traditional Optimization

Self-reflective optimization differs fundamentally from both fine-tuning and static prompt Optimization Self-reflective optimization has several key differences with fine-tuning and a static prompt engineering. As Table 1 demonstrates, self-reflective optimization takes a special place in the design space: it can adapt to the task without changing the parameters, but it is still explainable, which is an important feature to be deployed in controlled environments, such as in finance and healthcare.

Table 1: Comparison of Optimization Paradigms

Dimension	Fine-Tuning	Static Prompting	Self-Reflective Optimization
Parameter updates	Yes	No	No
Human supervision required	High	Moderate	Minimal
Adaptation speed	Slow (hours/days)	Instant	Moderate (seconds/minutes)
Task specificity	Fixed after training	Fixed per prompt	Dynamic per interaction

Dimension	Fine-Tuning	Static Prompting	Self-Reflective Optimization
Computational cost	High	Low	Moderate
Explainability	Low	High	High

2.3 The Brevity Bias and Context Collapse Problems

The latest studies found two long-standing issues of the context-based adaptation that are specifically addressed by self-reflective mechanisms. The brevity bias is the propensity of prompt optimizers to give preference to the concise and generic over the task-specific knowledge (Zhang et al., 2025). Although brevity leads to better human readability, empirical data indicates that models gain more information density; there tends to be a positive correlation between performance and richness of context to the point of maximum context window of the model in question.

The context collapse issue is even more unobtrusive: on the condition that the user requested the LLM to rewrite their prompt holistically, the responses become increasingly shorter and less informative with each succession of prompts. In a single generation step, Zhang et al. (2025) reported a case where the size of a prompt containing 18,282 tokens with 66.7% accuracy collapsed to 122 tokens with 57.1% accuracy, which is even worse than when no adaptation was done. This process requires systematic methods that would retain knowledge accrued but allow it to be narrowed down.

2.4 Verification Errors as a Primary Bottleneck

The crucial observation made by recent empirical research is that the main constraint of the self-reflective systems is verification errors. Maity, Potamitis and Arora (2025) examined the iterative reasoning pipeline in a cross-method analysis that breaks down the pipeline into Generation, Verification, Reflection and Recalibration steps. Their results indicate that the percentage of verification errors contributes to about 70 percent of the reasoning errors in all question answering, sequential and programming tasks. This is an implication that it might be more effective to enhance the accuracy of verification than to enhance the generation or reflection aspects.

3. Architectural Frameworks for Self-Reflection

3.1 The Generator-Reflector-Curator Triad

The dominant architectural pattern emerging from recent literature is the three-component

The prevailing architectural trend that has emerged as a result of new literature is the three-part structure as the case of Agentic Context Engineering (ACE) (Zhang et al., 2025). The process of self-reflection is broken down into specialized roles by this design:

Generator: The generator is charged with the task of generating first reasoning paths or candidate solutions to a task prompt based on available context. The generator is an object-level generator which is more concerned with task accomplishment than assessment.

Reflector: Processes generator outputs to derive actionable information, pattern of errors and strategies that have worked. The reflector is meta-level based, where the natural language critiques are generated to point out the strengths and weaknesses.

Curator: Takes the outputs of reflector and packages them into structured information into the persistent context, either system prompts, or memory entries, or instruction sets. The curator is the person who makes sure that the insights are stored in a form that will be readable by the invocations of the generator at a later stage.

This three-party structure has a number of benefits. To begin with, role specialization enables every component to be optimized separately by fast engineering or in more advanced applications, by specific fine-tuned

models. Second, there is the avoidance of confirmation bias by the separation of generation and evaluation because the critic and the generator are not one. Third, structured curation stops context collapse by using a deterministic merging operation instead of using holistic rewriting that is mediated by LLMs.

3.2 Reflexion: Verbal Reinforcement Learning

Reflexion framework, introduced by Shinn et al. (2023) at NeurIPS, is an extension of self-reflection, which uses episodic memory and trajectory analysis. Having been originally created in interactive agent settings, Reflexion has history of previous efforts, their results, and understandings formed out of the failure. In the case of an instance of a new task, the agent would consult this episodic memory to retrieve useful lessons, and then come up with a response.
Reflexion has made key innovations of:

Avoiding extreme binary success/failure signals: Conversion of sparse reward signals into rich natural language feedback.
Long-term memory integration: Long-term memory incorporation is the continuity of reflections between episodes so as to allow cumulative learning.
Strategy adaptation: Changing not only individual outputs but also the strategies of reasoning.
Reflexion has shown better performance in sequential decision-making as opposed to conventional prompting methods especially in situations where tasks have longer horizons of planning.

3.3 ZERA: Principle-Based Prompt Optimization

The framework proposed by Yi, Khang and Park (2025) is called ZERA (Zero-init Instruction Evolving Refinement Agent) and it optimizes system and user prompts together by applying principled low-overhead refinement. In contrast to the older approaches that make use of unstructured feedback ZERA analyses prompts based on eight generalizable criteria with weights that are automatically inferred. The framework rewrites prompts according to structured critiques, which allows prompt convergence to high-quality prompts to refine with minimal examples and shorter iteration cycles.
The results of experimental analysis of five LLMs and nine various datasets, including reasoning, summarization, and code generation tasks, showed the steady improvement over the powerful baselines. The ablation research attested that each element of an architectural work had its role to play in efficient timely construction.

3.4 GreenTEA: Evolutionary Prompt Optimization

The article by Dong et al. (2025) suggested the agentic LLM workflow of GreenTEA, which is an automatic prompt optimization tool that balances between candidate exploration and knowledge exploration. The framework uses an agent collaborative team to refine prompts (iteratively) on the basis of error sample feedbacks. A generation agent corrects the prompt to directly counteract the most prominent gaps that are caused by common error patterns that are recognized by an analyzing agent through topic modeling.
This refinement procedure is informed by a genetic algorithm structure, which models natural selection by developing candidate prompts by crossover and mutation, to successively enhance model performance. Numerous numerical experiments on the publicly available benchmark datasets implied better performance when compared to human-written prompts and the current state of the art prompt optimization algorithms, inclusive of logical reasoning, quantitative reasoning, commonsense reasoning, as well as ethical decision-making.

3.5 IROTE: Trait Elicitation Through Self-Reflection

The issue of eliciting the stable human-like traits of the LLM was tackled by Bai et al. (2025) in IROTE (Human-like Traits Elicitation through In-Context Self-Reflective Optimization). Based on psychological theories that propose that traits are developed in the identity-related reflection, their approach naturally produces and optimizes

textual self-reflection in prompts, which includes self-perceived experience, to trigger the trait-driven behavior of the LLM.

The optimization is done through the repetition of an information-theoretic objective that makes the connections between the behavior of the LLMs and the target trait maximized, and noisy redundancy in the reflection minimized without any fine-tuning. Massive experiments with three human trait systems showed that just one self-reflection created by IROTE can cause stable impersonation of target characteristics in various downstream tasks not limited to simple questionnaire response, which the strong baselines always failed to achieve.

3.6 AgentFactory: Executable Subagent Accumulation

A radically new self-evolution paradigm called AgentFactory suggested by Zhang et al. (2026) retains successful task solutions in executable subagent code and not textual experience. More importantly, these subagents are constantly optimized in accordance to the feedback of execution, and they gain more strength and efficiency as they come across more tasks. The saved subagents are pure Python code and have standardized documentation, and can be used anywhere with any system that supports Python.

The Agent Factory uses a three stage lifecycle: (1) Install - Build subagents on the fly to address initial problems; (2) Self-Evolve - When faced with related tasks, identify weaknesses in saved subagents and automatically evolve them into standalone Python modules to be used in other AI systems; (3) Deploy - Export mature subagents as standalone Python modules to be used in other AI systems. This method facilitates the unending ability amassing in which the library of executable subagents is increased and enhanced with time.

Table 2: Comparative Analysis of Self-Reflection Frameworks

Framework	Memory Type	Update Granularity	Primary Application	Key Limitation
ACE (Zhang et al., 2025)	Persistent context	Incremental (delta)	Multi-task agents	Context management overhead
Reflexion (Shinn et al., 2023)	Episodic	Trajectory-level	Sequential decisions	Memory retrieval costs
ZERA (Yi et al., 2025)	None	Principle-based	Instruction optimization	Task-specific criteria tuning
GreenTEA (Dong et al., 2025)	None	Population (genetic)	Prompt discovery	Evolutionary sampling cost
IROTE (Bai et al., 2025)	Self-reflection prompt	Trait-level	Personality elicitation	Trait-specific optimization
AgentFactory (Zhang et al., 2026)	Executable code	Subagent-level	Reusable capability accumulation	Code generation complexity

4. Empirical Evaluation

4.1 Benchmark Performance

Recent assessment gives strong grounds of the effectiveness of self-reflective processes. ACE had made 17.1% progress in the AppWorld agent benchmark (a test of API knowledge), code generation, and interaction with the environment compared to base LLM performance and routinely outperformed both In-Context Learning (ICL) and the recently developed GEPA a state-of-the-art prompt evolution algorithm (Zhang et al., 2025). Particularly, on average, ReAct+ACE surpassed ReAct+ICL by 12.3-percentage and ReAct+GEPA by 11.9 percentage points.

ACE also showed average positive progress of 10.9 percent over ICL, MIPROv2 as well as GEPA in offline transactions in financial reasoning with FiNER and Formula benchmarks requiring XBRL (eXtensible Business Reporting Language) knowledge (Zhang et al., 2025). The gains are especially impressive due to the fact that financial reasoning requires the level of domain-specific knowledge and numerical accuracy both of which are usually weak on the part of the static prompts.

4.2 Efficiency Gains

In addition to performance at the level of performance, there are substantial efficiency benefits associated with self reflexive mechanisms. ACE lowered the latency and trial attempts of the adaptation task in AppWorld offline adaptation tasks by 82.3 and 75.1 percent compared to GEPA, respectively (Zhang et al., 2025). In online adaptation in FiNER, ACE realized a 91.5 reduction in the adaptation latency and 83.6 saving on the cost in tokens input and generations compared to the baselines of Dynamic Cheatsheet.

These efficiency projects are based on two design choices. First, in comparison to holistic prompt rewriting, incremental delta updates do not involve the high computational cost. Second, deterministic based curation with structured curation operations avoids the context merging of the LLM which is otherwise a repeated generation step.

4.3 Generalization Across Models and Tasks

When self-reflective mechanisms work it is generalized both to model architecture and task domain. The study of ZERA showed that the metrics of performance were consistently enhanced when using five small open-source to big commercial LM systems (Yi et al., 2025). Likewise, IROTE was found to exhibit high performance in three systems of human traits, and experiments conducted by ablation demonstrated the value of every architectural component (Bai et al., 2025).

One of the findings worthy of a special mention is that the self-reflection has allowed smaller open-source schemes to close on the performance of bigger proprietary systems. ACE-enhanced open-source models were also found to match the highest rated business agents in the public leaderboard in AppWorld evaluations (Zhang et al., 2025). This implies that self-reflective processes can somewhat democratize access to high-performance AI by the reduction of the strength of a large number of parameters.

4.4 Critical Analysis: What to do When Self-Improvement Fails

Maity, Potamitis, and Arora (2025) have played an important negating role to the positive judgments of iterative self-improvement. Their study harmonized two opposite literatures: one side has papers indicating that iterative reasoning can be useful, and another side shows that the many feedback loops can be destructive instead of facilitating to LLMs in solving reasoning problems (e.g., Large Language Models Cannot Self-Correct Reasoning Yet).

Breaking down the iterative reasoning process into Generation, Verification, Reflection, and Recalibration phases, they observed that around 70 percent of reasoning errors happen in the V stage, i.e. at the point where there are mismatches between the generated results and ground truth. This was found in both Reflexion and Backprompting techniques, in three task areas (HotpotQA to question answering, ALFWorld to sequential decision-making, and HumanEval to Python program writing), and of 200 dissimilar problems.

In the case of ReAct agent that needs external context retrieval, the most frequent error was caused by underspecified search query that produced incorrect pages, and this would lead to ineffective Verification and Reflection latter. The Generation stage kept suggesting the same non-progressing step in sequence of decision making processes, whereas the Reflection stage did not have enough granularity (e.g., trying different action instead of specific causal injections) most of the time. This critical examination holds that accuracy in verification is a advantage to all fields but most strongly in the programming industry where better test cases bring out high reliability.

5. Principles and Implementation Strategy of a design

5.1 Structured Feedback Generation

One of the things that have been noted to be always pertinent throughout successful implementations is the aspect of a structural as well as criteria based feedback. Unstructured feedback - e.g. this output could be better - generates less improvement in the sense that it is not action specific. Effective feedback should:

1. **Identify explicit deficiencies** (e.g., "Step 3 has an off-by-one error in it)
2. **Give remedial advice** (e.g., "Change the loop bound to the len of the array (instead of the len-1)")
3. **Draws on applicable criteria** (e.g. "Breach of the requirement to operate on empty inputs")

The principle is operationalized in the ZERA framework by assessing prompts on eight generalizable criteria, which have weights given automatically, generating structured critique that is used to revise (Yi et al., 2025). One of the most promising methods presented by Chuang et al. (2025) is Self-REF (Self-Reflection with Error-based Feedback): a lightweight training strategy that can be used to motivate LLMs to express their confidence in their solutions with the help of confidence tokens, with significant enhancements in the downstream routing and rejection learning tasks in comparison to more traditional methods, like verbal spectrums and looking at the token

5.2 Incremental Context Evolution

One way to avoid context collapse is to avoid holistic rewritings in favor of incremental updates in a successful implementation. ACE delta-based methodology is based on keeping context in the form of a systematic set of atomic bullets, each of which has the metadata that helps to keep track of its utility and harmfulness (Zhang et al., 2025). Information is updated on a local basis: only the bullets that are considered problematic are updated, and the successfully performed bullets are not altered.

The design facilitates three properties, which are desired:

- **Physician-compositionality:** New knowledge can be added without interfering with existing knowledge.
- **Traceability:** The provenance of every element of the context is maintained.
- **Parallelizability:** Several delta updates can be deterministically combined.

5.3 Multi-Criteria Evaluation

Single-metric optimisation runs the risk that the results it offers are optimum in the intended measure but poor in all the other dimensions, which might not be measured. As a case in point, balancing factual accuracy can give too conservative results that can miss out pertinent information. Dominating models deal with this by the use of multi-criteria evaluation. The eight criteria of ZERA are widely covering prominence on quality, and the most appropriate set of the criteria is task-specific.

5.4 Verification-Aware Design

Since in error rate of reasoning verification errors are the leading cause of errors that happen (Maity et al., 2025), the effective self-reflecting systems should place a focus on verification accuracy. Empirical analysis recommendations involve: discouraging re-selection of the same actions in successive steps unless the environment state is not the same; ensuring that the final products are the same as the reasoning path to offer more confidence when applying these techniques in practice; and use of external knowledge base to base verification.

6. Limitations and Challenges

6.1 Verification as the Primary Bottleneck

This is the greatest limitation that has been found in the recent studies that is not a failure of reflection as such, but of the verification. Maity et al. (2025) have shown that despite the thought of reflection producing faithful and accurate thoughts, the error in verification may cause wrong final results. In ReAct agents, the retrieval causes of wrong pages when the queries are referred to as underspecified in the search policy are impractical in terms of verification and reflection of this search policy.

6.2 Simulated, Not Authentic, Understanding

The basic drawback of existing self-reflective apparatus is the fact that it makes the illusion of reflection by means of created prompting instead of actually adopting metacognition. LLMs lack beliefs regarding their states of knowledge; what is seen as self-critique is in reality the model that produces text which satisfies patterns seen in training data where human beings are self-critical of their work. Such distinction is important since simulated reflection may give plausible yet false critiques, which undergoes revision cycles that worsen and not increases the quality of the output produced.

6.3 Hallucination and Error Distortion

Self-reflection has the ability to magnify the mistakes instead of rectifying them when the critic aspect perceives some issues that are nonexistent or it is not successful in detecting the actual one. An early hallucination can be adopted in multi-iteration loops, as the hallucination is further reinforced as the model progresses to produce outputs that are more and more aggressive, but erroneous. Such mitigation measures are the use of extraneous knowledge basis, the restriction of the number of iterations, and guardrail that discards implausible critiques.

6.4 Computational Overhead

Self-reflection is also iterative and this makes it very expensive in terms of computing. This generating and generating process needs more and more generating cycles, and latency as well as the number of consumed tokens grows. Although visionary has many fewer overheads compared to holistic rewriting based on the incremental approach of ACE, the self-reflective systems are still considerably more costly than single-pass generation. In real-time applications that require the latency of less than a second, the existing techniques of self-reflection can be not feasible.

6.5 Prompt Bloat and Context Window Limits

The persistent context of self-reflective systems increases monotonically as the number of experience accumulation increases. Extensive deployments can eventually surpass the maximum context window that the model assumes, and truncate or summarize and, in doing so run the risk of losing information that is useful. Grow-and-refine mechanism used by ACE tries to solve this by using the periodical pruning method which depends on semantic similarity, yet the choice of the retention policies is an open research question.

6.6 Stability and Convergence

Set of loop self-reflection is not necessarily towards high-quality outputs that are stable. It can be able to produce successively different but not progressively better outputs on oscillations when there is a contradictory feedback to be employed alternately by the critic between iterations. This has been empirically determined to be the property of convergence that is highly sensitive to prompt design, nature of the task and model architecture parameters, which are commonly considered hyperparameters and not designed principles.

7. Future Directions

7.1 Verifiable Self-Reflection

One of the areas that the future research must take a very critical step is the creation of the self-reflective processes that can confirm what they produce with the external sources. Existing systems are solely based on internal consistency checks which fail to identify errors that are based on lack of knowledge and wrong knowledge that occur. Since the verification errors are approximately 70% of the sources of thinking

failures (Maity et al., 2025), the increase in the accuracy of verification can bring higher results than the enhancement of generation or reflection attributes. The combination of retrieval-augmented generation (RAG) and self-reflection can provide a new direction ahead as the models will have bases on which to base their arguments.

7.2 Meta-Learning for Reflection Strategies

Instead of the reflection prompts being designed by hand, in the future systems can learn the best reflection strategies with the help of meta-learning. Last but not least, the GreenTEA framework offers one of the first attempts, where genetic algorithms evolve instant optimization plans (Dong et al., 2025). By generalizing this method to the reflector (what change to make, how many attempts to perform) it may be possible to have significant increases in effectiveness and efficiency.

7.3 Multi-Agent Reflective Systems

The AgentFactory model is an indication of the speculative ability of building accumulating performable subagents as opposed to written cogitations (Zhang et al., 2026). This mechanism maintains successful solutions in the form of reusable code, which allows them to reliably re-run tasks, in comparison to natural language memories. The future of work may utilize both textual reflection (to be interpretable) and code synthesis (to be reliable) and come up with hybrid systems that learn what and how to do it.

7.4 Safety and Alignment Guarantees

In the process of attaining autonomy of self-reflective systems, it is crucial to make sure that reflecting can result in neither destructive nor inconsistent behavior of the system. Such open questions can be formulated as: How do we ensure self-improvement does not violate safety constraints? How can reflection be assured not to find and take advantage of loopholes in reward specifications? Early methods encompass adversarial reflection (when a critic conducts a search to identify policy violations) as well as constitutional reflection (when a feedback ought to be composed respectful of established concepts), but monitored frameworks have not been developed yet.

8. Conclusion

The introduction of self-reflective processes powered by AI is a component of a new development in the application of generative agents. These systems have made significant performance gains in a wide range of tasks fields, with sufficient performance and explainability without updating parameters, and be more autonomous to quick revision and optimization. To represent such a tripartite design the generator, reflector and curator design pattern has become a common design model that has manifested in frameworks including ACE, ZERA, GreenTEA and IROTE.

The empirical results of agent benchmarks and domain-specific tasks indicate the stable increase in task completion rates in the range of 10 17% and a decrease in adaptation latency in the range of 87 and token costs in the range of 84 (Zhang et al., 2025). These advantages can be explained by three design concepts: the structured generation of feedback, which enables the provision of actionable criticisms, the incremental evolution of context, which eliminates loss of knowledge and multi-criteria assessment, which optimizes conflicting goals.

But still there are severe limitations. As it is critically analyzed, the cause of reasoning errors is mostly that of verification error, so it can be assumed that the existing systems have an excessive focus on reflection, and verification does not occur to the extent at which it should (Maity et al., 2025). Also, these systems fabricate, instead of actually comprehending, reflection, and exhibit hallucination of risks and exaggeration of errors, as well as have a large computational cost, and can be unable to obtain stable solutions.

The direction of this study indicates the greater change in design of AI systems: the movement toward more dynamic, self-enhancing systems, which constantly change through guided reflections that are structured. Once this paradigm is fully developed, it will allow rendering high-performance AI more available,

flexible, and human values friendly, assuming that the issues of verification accuracy, reliability, efficiency, and safety are approached in a properly structured manner.

REFERENCES

Bai, Y., Duan, S., Huang, M., Yao, J., Liu, Z., Zhang, P., Lu, T., Yi, X., Sun, M., & Xie, X. (2025). IROTE: Human-like traits elicitation of large language model via in-context self-reflective optimization. *arXiv preprint*, arXiv:2508.08719. https://doi.org/10.48550/arXiv.2508.08719

Chuang, Y.-N., et al. (2025). Learning to route LLMs with confidence tokens. *arXiv preprint*, arXiv:2410.13284. https://doi.org/10.48550/arXiv.2410.13284

Dong, Z., et al. (2025). GreenTEA: Gradient descent with topic-modeling and evolutionary auto-prompting. *arXiv preprint*, arXiv:2508.16603. https://doi.org/10.48550/arXiv.2508.16603

Maity, A., Potamitis, N., & Arora, A. (2025). Reconciling divergent views through a critical analysis of iterative self-improvement in LLMs. Conference abstract presented at Aarhus University.

Shinn, N., et al. (2023). Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Yi, S., Khang, M., & Park, S. (2025). ZERA: Zero-init instruction evolving refinement agent – From zero instructions to structured prompts via principle-based optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 23323–23337). Association for Computational Linguistics. https://doi.org/10.18653/v1/2025.emnlp-main.1190

Zhang, Q., Hu, C., et al. (2025). Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint*, arXiv:2510.04618. https://doi.org/10.48550/arXiv.2510.04618

Zhang, Z., Lu, S., Qian, H., He, D., & Liu, Z. (2026). AgentFactory: A self-evolving framework through executable subagent accumulation and reuse. *arXiv preprint*, arXiv:2603.18000.