A Human-Aligned System for Guiding React Agents through Adaptive Prompting and Dynamic Memory Editing

Anonymous ACL submission

Abstract

This paper proposes a sustainable and adaptive prompting system for ReAct-based language model agents that enhances reasoning accuracy, contextual consistency, and alignment with human expectations in multi-step question answering. The system integrates task-adaptive evaluation, structured memory editing, and reactive reasoning cycles to enable iterative prompt refinement and context-aware adaptation. Unlike existing methods that treat prompts and memory as static, our approach dynamically updates both based on interaction feedback. Experiments across six QA domains show consistent improvements over strong baselines in LLM-as-judge and human evaluations, achieving up to 91.88% agreement with human judgment (Cohen's Kappa). These results underscore the value of memory-aware prompting and reactive reasoning in developing reliable and adaptable LLM agents.

1 Introduction

The emergence of large language models (LLMs) has enabled agent-based frameworks that surpass conventional single-turn NLP tasks by supporting iterative reasoning and goal-directed interaction (Gao et al., 2023). Recent studies demonstrate the potential of LLMs as autonomous agents capable of tool use and multi-step problem solving (Li et al., 2025; Wang et al., 2024a; Hu et al., 2024). This shift has sparked increased interest in prompt optimization to enhance reasoning and decision-making (Yin and Wang, 2025; Yuksekgonul et al., 2024). Chain-of-thought (CoT) prompting has been widely adopted to elicit step-by-step reasoning (Chantangphol et al., 2025; Wei et al., 2022), but its static structure limits adaptability and error resilience. To address these limitations, the ReAct framework integrates reasoning and interaction, enabling agents to perform dynamic, multi-step tasks more robustly (Tang et al., 2024; Roy et al., 2024). However, existing ReAct implementations depend on manual prompt engineering, which is difficult to scale and lacks task adaptability. This motivates the development of adaptive prompting strategies to ensure sustained alignment with evolving user goals and task requirements.

Despite recent advances in optimizing LLM agents with ReAct integration, existing frameworks—such as LangGraph (Harrison, 2024), TextGrad (Yuksekgonul et al., 2024), and Adalflow (Yin and Wang, 2025)—remain limited in their support for structured memory and robust cyclic reasoning processes, which are critical capabilities for real-world, multi-step tasks. LangGraph leverages a graph-based engine for multi-step reasoning with modular control over tools, agent selection, and task decomposition. Integrated with LangMem (Harrison, 2023b), it offers basic long-term memory by enabling context storage and retrieval, though its reliance on predefined graphs and limited memory editing reduce flexibility in adaptive prompting. TextGrad employs textual gradients for prompt refinement and supports basic iterative reasoning. Nevertheless, the lacks structured memory editing limits effectiveness in multi-agent coordination and cyclic reasoning, where prior steps across cycles must be revisited. Adalflow employs graph-based auto-differentiation to optimize multi-step interactions. However, its lack of memory editing integration limits adaptability in tasks requiring persistent memory. These approaches share limitations in memory flexibility and manual task-aware optimization strategies, resulting in agent responses that are misaligned with human expectations.

To examine current limitations, we conducted human–LLM agreement studies on question answering tasks across three domains: human resources (HR), regulatory compliance, and the Personal Data Protection Act (PDPA). These domains reflect varying answer linguistically restrictiveness providing a basis for evaluating model consistency under dif-

ferent strict interpretation. The HR dataset allows flexible responses, provided the content is complete and accurate, whereas the Regulatory and PDPA datasets demand semantically dense and legally precise responses, where lexical variation is limited and any deviation may alter the intended meaning. In this study setup, the answers were generated by Gemini 2.0 Flash (Reid et al., 2024) and evaluated with human judgments and an LLM-as-Judge setup, where GPT-4o (Achiam et al., 2023) serving as the evaluator, to compare alignment between human evaluation and LLM-as-Judge outputs. We measured Cohen's Kappa (McHugh, 2012) to assess the consistency of evaluator

As shown in Table 1, the HR domain yields the highest Kappa value, indicating strong consistency. In contrast, the limited lexical variation such as Regulatory and PDPA exhibit lower scores and negative scores, especially without reference answers. The negative Kappa indicate systematic disagreement—worse than chance—between the LLM-as-Judge and human evaluation, particularly in cases where precise legal phrasing is required but not provided in the reference. This highlights the limitations of model in handling tasks that require precise language alignment, revealing discrepancies between human intent and LLM reasoning. These findings underscore the need for a sustainable, memory-aware, and task-adaptive prompting pipeline that enables ReAct agents to dynamically optimize reasoning and action—without manual tuning or degradation of behavioral consistency.

LLM-as-Judge setting		**Cohen's Kappa score**		
Prompt tuning	Expected answer	HR	Regulatory	PDPA
Yes	Yes	68.18	18.75	22.47
Yes	No	70.21	-10.96	-1.03
No	Yes	73.88	11.22	16.70
No	No	65.35	14.86	18.77

Table 1: Human evaluation and LLM-as-Judge agreement score (%).

To address the limitations of existing techniques, we propose a sustainable and adaptive prompting framework for question answering. Our approach integrates three key components: cyclic reactive reasoning, task-adaptive answer evaluation and memory-aware editing. Memory-aware editing enables agents to retain, update, and reuse contextual information for long-term consistency. Task-adaptive answer evaluation combines automated correctness classification with domain-specific validation to support response revision. These mechanisms operate within a cyclic reasoning loop enabling dynamic adaptation to evolving inputs. By integrating validation and memory updates into the reasoning process, our pipeline enhances integrity, reduces reliance on manual prompt engineering, and supports robust decision-making in complex tasks. Our main contributions are as follows:

1. A novel framework that integrates ReAct reasoning, answer validation, and dynamic retrieval from editable memory to support sustainable question answering.

2. Task-adaptive evaluation that supports autonomous prompt optimization based on task complexity.

3. Memory-aware optimization that decouples prompt logic from editable memory, ensuring essential instructions during updates.

The paper comprises methodology, experimental setup and results, discussion and conclusion.

2 Methodology

To improve the robustness of LLM-generated responses in question answering tasks, the framework incorporates six interdependent modules, as shown in Figure 1. The ReAct agent combines reactive reasoning with retrieval-augmented memory, generating answers through iterative thought–action–observation cycles. An answer validation module evaluates the ReAct agent's reponse across relevance, accuracy, coverage, and completeness. The data checking chain ensures factual alignment by verifying whether expected answers are supported by retrieved document. In cases where the data checking module finds that the retrieved evidence supports the answer, while the original query remains under-specified, the question rewriting module reformulates the query to improve alignment with the supporting document. In the absence of supporting evidence, as determined by the data checking module, the memory editor and generator supports continual learning by updating the memory state—adding validated knowledge and removing conflicting information—to guide future reasoning. The prompt optimization module refines prompt configurations based on accumulated feedback, dynamically adjusting roles, instructions, and answer formats to improve task adherence.

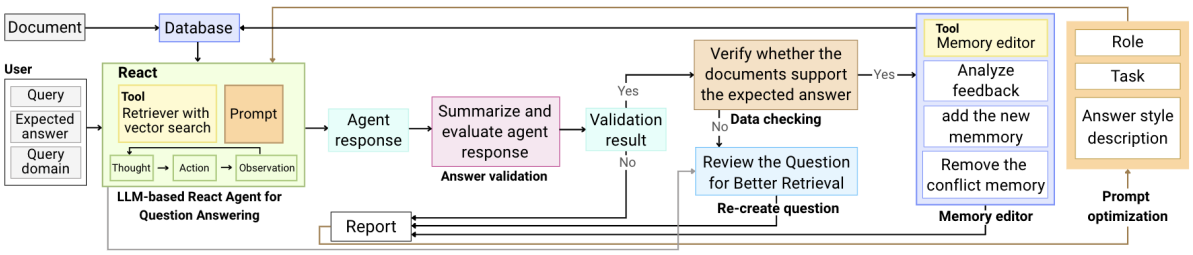

Figure 1: Overview of methodology

2.1 LLM-based React agent

The LLM-based React agent enables iterative decision-making in question answering by interleaving reasoning steps with actionable outputs, guided by contextual understanding and task-specific configurations. The system accepts a user query, expected answer format, and agent configuration. To initiate the reasoning loop, the agent is primed through an initial prompt that embeds a detailed task description, role specification, preferred answer style, and relevant domain knowledge. This initialization constrains the behavior of LLM to a defined persona and task scope, aligning its outputs with the domain-specific requirements.

During execution, the agent performs multi-step reasoning through a sequence, represented as:

$$THT_i \rightarrow ACT_i \rightarrow OBS_i \tag{1}$$

Here, THT_i denotes the thought operation, ACT_i denotes the action operation, and OBS_i denotes the observation operation, all corresponding to the i step of ReAct.

In the thought stage, the agent hypothesizes a response based on the input query and its contextual understanding. This is followed by the Action stage, where it selects and executes operations—such as employing a document retrieval tool to gather relevant information. The observation stage then processes the outcome, enabling the agent to interpret the evidence and assess its relevance. Finally, the agent synthesizes a response by integrating retrieved knowledge and prior observations to refine its answer to the user query.

The system is employed a vector-based retrieval tool to access relevant knowledge from a structured database. The queries and database entries are embedded into a shared vector space, and retrieval is performed by matching the query vector to the closest entries. This process is formalized as follows:

$$K = S(q, e, \mathcal{D}) \tag{2}$$

where K denotes the retrieved knowledge, S represents the vector search engines, q is the query, e is the embedding model and $\mathcal{D}$ denotes the database.

To integrate retrieved information into the reasoning process, the agent generates a response using a system prompt that includes its predefined role, task-specific instructions, stylistic preferences, and the retrieved knowledge. The response is generated as follows:

$$R = LLM_r(q, r, t, s, K) \tag{3}$$

where R denotes the response of agent, LLM_r represents the language model responsible for agent response generation, q denotes the query, r specifies the agent role, t denotes the agent task, s indicates the desired answer style and K refers to the retrieved knowledge.

The LLM-based ReAct agent allows the agent to iteratively refine its reasoning through knowledge retrieval, enhancing the accuracy and contextual relevance of its responses to align with the desired answer style, user intent, and task requirements.

2.2 Answer Validation

To enhance the accuracy of agent responses, we introduce an answer validation module that conducts structured evaluations of both semantic fidelity and lexical appropriateness to ensure response reliability. For each agent response, the module employs the LLM to summarizes and evaluates the response across four dimensions—Relevance, Accuracy, Coverage, and Completeness—as suggested by Auer et al. (2023). Relevance ensures the answer directly addresses the user's query; Accuracy verifies the factual correctness of the information; Coverage assesses whether all essential aspects of the question are included; and Completeness evaluates the sufficiency of detail for comprehensive understanding. These dimensions collectively enhance the reliability and utility of the information provided, aligning with established evaluation

frameworks in information retrieval and data quality assessment. For each identified issue, the LLM suggests actionable improvements. Finally, it generates a response that includes the answer validation result and corresponding feedback. The validation result is categorized into one of three classes: fully correct, partially correct, or incorrect. The feedback is provided to clarify and justify the outcome of the validation, as formalized in the following equation:

$$V_r, V_f = LLM_v(q, a, R) \tag{4}$$

where V_r denotes the result of answer validation, V_f denotes the feedback of answer validation, LLM_v represents the language model for answer validation, q denotes the query, a specifies the expected answer, and R denotes the agent's response.

The answer validation module evaluates the ReAct agent's response. If errors are detected, validation feedback is passed to data checking module; otherwise, it is retained for prompt optimization.

2.3 Data Checking

To address potential inaccuracies in the response of the agent as identified by the answer validation module, we introduce a data checking mechanism. This mechanism verifies whether the expected answer is supported by the retrieved document, helping to distinguish between incorrect responses due to faulty reasoning and those resulting from missing information. It also facilitates memory revision and assists the memory editor in further improving the agent's performance.

The output of the data checking process consists of two parts, which are the result of the data checking evaluation and an explanatory rationale supporting that judgment. Each validated label is classified into one of three categories: fully valid, partially valid, or invalid. This structured classification ensures that ground-truth answers are meaningfully aligned with the contents of the database. To support this data checking, the system performs knowledge retrieval using vector search, as defined by the equation 2. The data checking is defined as:

$$C_r, C_s = LLM_c(a, K) \tag{5}$$

where LLM_c is a LLM-based data checking, C_r is data checking result, C_s is support reason for data checking result, a is the expected answer and K is the retrieved knowledge .

The data checking module employs an LLM-based verifier to assess the consistency between the expected answer and retrieved evidence. If the evidence supports the answer although the query remains under-specified, the output is forwarded to the Question Rewriting module; otherwise, it is routed to the Memory Editor and Generator.

2.4 Question rewriting

To address cases where the response of agent is incorrect despite sufficient supporting documents, we introduce a question rewriting component that enhances contextual alignment and retrieval performance. The question rewriter operates under the assumption that the expected answer is appropriate but the original query lacks the specificity needed for effective document retrieval. It reformulates the query by leveraging both the expected answer and the context retrieved from previous attempts, ensuring the new query targets documents that contain supporting evidence without revealing the answer.

This design enhances the retrieval process by enabling iterative refinement of user queries, particularly in cases where insufficiently specific questions degrade retrieval performance and hinder the agent's ability to access relevant context. The reformulated query is then retained as feedback for prompt optimization. The rewriting process is formally defined as:

$$Q' = LLM_q(q, a, K) \tag{6}$$

where LLM_q denotes the language model for question rewriting, q is the original query, a is the expected answer, and K represents the previously retrieved knowledge. The output Q' is the reformulated query.

2.5 Memory editor and generation

To address incorrect outcomes from the answer validation module and failure cases in data checking, we introduce a memory editing and generation module, implemented as a language model-based mechanism that updates memory based on the current memory state, answer validation feedback, the original query–answer pair, and the target memory format. This design ensures that agent maintains an up-to-date and coherent internal knowledge base, allowing it to evolve based on validated feedback and interaction history.

The module first analyzes the validation feedback to determine necessary updates that enhance

response of the agent. It then integrates newly derived knowledge with the existing memory entries. The output consists of two structured components: a list of new memory entries to be added, formatted consistently with the input memory structure, and a list of conflicting memory entries that should be removed, also presented in the same format. Memory operations are governed by:

$$M_a, M_c = LLM_m(M_o, V_f, C_s, f, q, a) \quad (7)$$

where M_a denotes the added memory, M_c denotes the conflicted memory, LLM_m represents the language model for memory editor, M_o denotes the current memory, V_f denotes the feedback of answer validation, C_s is support reason for data checking result, f is the format of the memory, q denotes the query, and a specifies the expected answer.

2.6 Adaptive prompting optimization

The adaptive prompt optimization is implemented as an LLM-based mechanism that updates the system prompt based on interaction feedback, ensuring the agent remains aligned with task-specific objectives and expected output standard. In our framework, prompt optimization is informed by three key inputs: the feedback from the answer validation module, the added memory entries, and the conflicting memory entries identified by the memory editor. These elements indicate the current performance of the agent and knowledge state, providing rich context for refining the configuration of the agent. The goal is to adjust core components of the prompt which are task instructions, and answering style description—to better suit the current task context and improve downstream response quality. This design ensures the agent remains adaptable, domain-aligned, and capable of generating accurate, coherent answers. The prompt optimization process is formally defined as:

$$P'_j = LLM_p(r, t, s, M_a, M_c, V_f)_{j-1} \quad (8)$$

where LLM_p denotes the language model responsible for prompt optimization at step $j-1$, and P'_j is the resulting optimized prompt used at the j-th iteration. r, t, and s represent the original agent role, task instructions, and answering style, respectively. Each iteration j denotes a full framework cycle, including ReAct-based interaction, validation, data checking, memory editing, and prompt refinement.This formulation enables iterative refinement of the ReAct agent's prompt based on performance feedbacks and memory updates from the previous interaction step.

By dynamically adjusting the prompt configuration based on performance feedback and memory evolution, this mechanism enhances the contextual relevance and task effectiveness of agent over time.

3 Experimental Setting

3.1 Datasets

We evaluate our framework on three domain-specific QA datasets featuring single-turn, chatbot-style interactions, where each question–answer pair is independent and devoid of multi-turn dialogue context an HR dataset with 329 human resource inquiries, a regulatory dataset with 72 question–answer pairs on organizational policies, and a PDPA dataset with 59 queries related to Personal Data Protection Act. These datasets were constructed by retrieving relevant content from internal databases, followed by the generation of corresponding questions and ground-truth answers to reflect realistic user intents and information needs. Each dataset was approved by our organization and consisted of organizational policies and regulations without any personally identifiable information.

Additionally, we incorporate the official dataset from the COLING-2025 Regulations Challenge (Wang et al., 2024b), which contains QA pairs curated for evaluating ReAct-based agents. This dataset integrates foundational knowledge and stylistic conventions relevant to real-world compliance scenarios. Its validation and test sets include 49 named entity recognition (NER) samples derived from the EMIR dataset, along with 190 financial math cases generated using ChatGPT—containing formulas and code—and subsequently verified by human annotators.

To evaluate performance on multiple-choice reasoning tasks, we also include 1,032 samples from the publicly available Flare-CFA dataset[1], which is modeled after professional certification exams and emphasizes high-level reasoning over structured answer choices. This evaluation spans six QA domains, covering high-precision tasks that require standardized output formats (e.g., financial math, NER, and CFA), and policy-driven scenarios with limited lexical variation and strict language restric-

[1] https://huggingface.co/datasets/ChanceFocus/flare-cfa

tiveness (e.g., regulatory and PDPA), providing a comprehensive basis for assessing the robustness and generalizability of our proposed framework

3.2 Implementation Details

All components are implemented in Python using an architecture built on LangChain (Harrison, 2023a), integrating OpenAI GPT models and Hugging Face Transformers (Wolf et al., 2019) to support structured reasoning, generation, and evaluation. A ReAct-based agent framework orchestrates multi-agent execution and reasoning-intensive tasks, with agent creation and coordination managed via LangGraph (Harrison, 2024). The answer generation is performed using Gemini 2.0 Flash (Reid et al., 2024), while GPT-4o (Achiam et al., 2023) is employed for answer validation and serves as the LLM-as-a-judge in experiments. A temperature setting of 0.0 is used throughout to ensure deterministic outputs. To enhance factual accuracy and answer consistency, the ReAct-based agent and data checking modules incorporate vector-based retrieval implemented with OpenSearch. The retrieval system uses the multilingual-e5-small embedding model to support language-agnostic semantic search aligned with expected answers and underlying database content.

Our experimental setup was executed on an NVIDIA A6000 GPU with 64 GB RAM. The average prompt length for our ReAct-based agent is approximately 1,675 characters (453 tokens), which is notably more efficient compared to AdalFlow (2,900 characters, 683 tokens) and TextGrad (5,700 characters, 1,277 tokens). Additional modules exhibit compact prompts as follows: answer validation (952 characters, 256 tokens), data checking (696 characters, 192 tokens), and memory editing (905 characters, 228 tokens). This compact prompting strategy supports lower compute overhead. All experiments were completed within 12 hours.

3.3 Evaluation Metrics

We evaluate the performance of the agent using three primary metrics: accuracy based on LLM-as-judge, accuracy based on human annotations, and inter-rater agreement, measured by Cohen's Kappa, between LLM and human evaluations. A total of four annotators, employed by organization in departments relevant to the evaluation domains, participated in the human evaluation. Annotators were presented with a side-by-side comparison of the original question and the agent-generated answer and were prompted to assign a binary label: Correct or Incorrect. To ensure consistent judgment, all annotators followed a shared set of qualitative evaluation criteria: Relevance, Accuracy, Coverage, and Completeness. Relevance assesses whether the response directly addresses the user's question. Irrelevant responses automatically invalidate coverage and completeness. Accuracy evaluates factual correctness and consistency with the intended source. Responses that introduce unsupported or fabricated content are marked as inaccurate. Coverage considers whether all parts of the question are addressed. For example, omitting one of four requested items results in insufficient coverage. Completeness refers to structural and linguistic integrity. A response is complete if it is fully delivered, not truncated, and ends coherently, even when expressing uncertainty.

To assess the consistency between LLM-as-judge and human annotators, we report Cohen's Kappa—a statistical measure of inter-rater reliability that accounts for chance agreement. Unlike raw agreement rates, Cohen's Kappa adjusts for the probability of random alignment and is defined as:

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

where P_o is the observed agreement and P_e is the expected agreement by chance. A score of 1 indicates perfect agreement, 0 reflects chance-level agreement, and negative values suggest systematic disagreement. In this study, Cohen's Kappa provides a quantitative measure of alignment between LLM-based and human evaluations, offering insight into the reliability of automated assessment.

4 Experimental Results and Discussion

4.1 Performance comparison with baselines

Table 2 presents a comparative evaluation of our prompting framework against AdalFlow, TextGrad, and baseline LLMs, using LLM-as-judge and human assessments, along with agreement measured by Cohen's Kappa. This experiment investigates the impact of two key components—memory editing and adaptive prompt optimization—on the performance of ReAct agents. All variants incorporate a memory mechanism; however, only specific configurations implement memory editing. Our proposed framework, which integrates both components, achieves the highest scores across all metrics (90.24% LLM-as-judge, 88.50% human,

Method		Memory generator and editing	LLM-as-judge score (%)	Human evaluation score (%)	Agreement score (Cohen's Kappa) (%)
Our proposed	Adaptive Optimization	Yes	90.244	88.496	91.876
	-	Yes	86.939	87.254	87.297
	Adaptive Optimization	No	88.167	85.178	90.733
	-	No	82.055	81.219	82.574
Adalflow	Auto-differentiation	Yes	81.869	79.670	83.984
	Auto-differentiation	No	79.958	75.580	81.541
Textgrad	Textual gradient	Yes	78.595	76.079	86.082
	Textual gradient	No	76.099	73.164	83.312
GPT			74.723	71.120	82.635
Gemini			74.473	72.866	81.770

Table 2: Performance comparison of ReAct agents with and without memory generator and editing across training strategies.

Method	Domain QA	Total Samples	LLM-as-judge score (%)	Human evaluation score (%)	Agreement score (Cohen's Kappa) (%)
Our proposed Trainable ReAct prompt optimization with memory generator and editing	NER	49	96.062	91.616	93.119
	CFA	1032	92.970	91.505	96.234
	Financial math	190	98.017	95.160	94.111
	PDPA QA	59	85.245	84.029	87.117
	HR QA	329	96.512	94.444	97.198
	Regulatory QA	72	83.879	81.547	90.659
Adalflow Trainable ReAct prompt optimization with memory generator and editing	NER	49	92.203	86.731	87.547
	CFA	1032	84.601	83.477	87.986
	Financial math	190	93.186	89.112	88.123
	PDPA QA	59	79.812	72.753	75.229
	HR QA	329	83.052	79.599	87.999
	Regulatory QA	72	79.530	73.560	79.729
Textgrad Trainable ReAct prompt optimization with memory generator and editing	NER	49	76.190	71.998	86.732
	CFA	1032	81.724	80.816	87.114
	Financial math	190	85.663	82.928	86.631
	PDPA QA	59	71.296	64.088	72.628
	HR QA	329	74.989	68.957	85.081
	Regulatory QA	72	70.649	65.710	77.669

Table 3: Performance Comparison Across Domain QA Tasks

and 91.88% agreement), demonstrating strong task alignment and consistent output quality. Ablation studies demonstrate that excluding either component degrades performance, underscoring their complementary contributions. The configuration incorporating memory editing without adaptive prompt optimization remains competitive to ensure the independent effectiveness of structured memory and task-adaptive evaluation. Compared to gradient-based baselines, our framework demonstrates improved performance, particularly in human evaluations, suggesting enhanced instruction fidelity and context-aware reasoning. Furthermore, it substantially increases Cohen's Kappa agreement between LLM-as-judge and human ratings, indicating stronger alignment with human intent.

4.2 Performance comparison of our proposed and baseline across domain QA tasks

We further evaluated the generalizability of our proposed framework across six diverse QA domains: NER, CFA, financial math, PDPA, HR, and regulatory QA. As shown in Table 3, our method consistently outperforms AdalFlow and TextGrad across all domains in both LLM-as-judge and human evaluations. It demonstrates particularly strong performance in high-precision tasks such as financial math (98.02% LLM-as-judge, 95.16% human) and HR QA (96.51%, 94.44%), indicating robust reasoning and stability. In more nuanced domains, such as PDPA and regulatory QA, where instruction fidelity and adaptability are critical, our method achieves higher agreement scores (87.12% and 90.66% Cohen's Kappa, respectively) compared to the baselines. These findings highlight the benefits of memory-aware and prompt optimization in enhancing consistency and contextual alignment. Overall, the framework adapts effectively across domains of varying complexity, outperforming static gradient-based approaches and reinforcing the value of structured prompting and task-adaptive evaluation.

4.3 Performance comparison across prompt patterns and iterative steps.

To examine the impact of iterative optimization and prompt initialization, we conducted an ablation study varying the number of the full framework cycle (maximum steps = 3, 5, 7) and starting prompt patterns. As shown in Table 4, performance consistently improved with increased reasoning steps, with the highest scores achieved at 7 maximum steps using Our proposed (90.24% LLM-as-judge, 88.50% human evaluation, and 91.88% agreement).

Method	Max Step	Starting Prompt	LLM-as-judge score (%)	Human evaluation score (%)	Agreement score (Cohen's Kappa) (%)
Our proposed Trainable ReAct prompt optimization with memory generator and editing	3	Adalflow	85.434	85.411	88.747
	5	Adalflow	86.281	86.791	89.745
	7	Adalflow	87.462	88.622	90.337
	3	Textgrad	83.320	84.175	88.503
	5	Textgrad	84.985	85.298	89.379
	7	Textgrad	85.513	87.022	89.768
	3	Our prompt	87.990	87.510	89.365
	5	Our prompt	90.295	87.919	90.270
	7	Our prompt	90.244	88.496	91.876

Table 4: Performance Comparison Across Prompt Patterns and Iterative Steps.

Method	Prompt optimization	LLM-as-judge setting	LLM-as-judge score (%)	Human evaluation score (%)	Agreement score (Cohen's Kappa) (%)
Our proposed	Adaptive Optimization	with expected answer	90.244	88.496	91.876
	Adaptive Optimization	without expected answer	89.044	88.496	88.782
	-	with expected answer	86.939	87.254	87.297
	-	without expected answer	85.009	87.254	84.446

Table 5: Performance Comparison Across LLM-as-Judge Settings

These findings suggest that deeper iterative reasoning through adaptive prompting enhances both accuracy and stability. Across all settings, our implementation outperformed alternatives, underscoring the role of well-designed initial prompts in guiding memory updates and prompt refinement. These results support our design of cyclic learning with structured prompt evolution, enabling ReAct agents to adapt effectively while preserving coherence and contextual relevance.

4.4 Performance comparison across LLM-as-Judge settings

We examined the impact of including the expected answers in the LLM-as-judge evaluation to assess alignment with human judgment. As shown in Table 5, the highest performance was obtained under the system incorporated adaptive prompt optimization and was evaluated with expected answers (90.24% LLM-as-judge, 88.50% human evaluation, 91.88% agreement). In the absence of the expected answers, the LLM-as-judge score slightly declined to 88.782% agreement. A similar trend was observed in the non-prompt optimization setting. These findings suggest that expected answers enhance automated scoring consistency, while the proposed method remains robust even under lack of expected answer.

5 Discussion

Our LLM-based ReAct agent integrates reactive reasoning with retrieval-augmented memory to improve performance across diverse QA tasks. The full system outperforms gradient-based baselines in both automatic and human evaluations, achieving high inter-rater agreement and strong task alignment. Ablation results highlight the complementary roles of adaptive prompting and memory editing, with the non-trainable variant still performing competitively. Notably, our approach maintains high agreement with human judgments even without reference answers (e.g., 88.78% Cohen's Kappa), demonstrating robustness in open-ended evaluation. These findings underscore the effectiveness of combining dynamic memory with task-adaptive prompting to build more reliable and context-aware LLM agents.

6 Conclusion

we proposed a sustainable and adaptive prompting framework for ReAct-based LLM agents that integrates task-adaptive evaluation, structured memory editing, and reactive reasoning. Existing frameworks such as LangGraph, TextGrad, and Adalflow, while advancing modular control and prompt optimization, remain limited in their ability to revise memory and adapt reasoning dynamically across multi-step tasks. They often rely on static prompts and treat memory as fixed context, resulting in brittle behavior and poor adaptability in ambiguous or evolving environments. Our approach addresses these limitations by enabling agents to iteratively revise context, adjust prompts, and react to feedback through structured Thought–Action–Observation cycles. Empirical results across QA domains show consistent improvements over baselines, underscoring the importance of combining memory-aware adaptation with reactive reasoning to build more reliable, flexible, and human-aligned LLM agents.

Limitations

Although our framework demonstrates promising results, it presents several limitations. The incorporation of multiple reasoning and validation steps increases computational overhead, posing challenges for real-time and large-scale deployment. The current memory module, while effective for structured updates, has not been evaluated on unstructured or noisy inputs, which are common in real-world applications. Moreover, the use of teachable memory structures may result in increased token consumption during inference, which introduces efficiency concerns. Nevertheless, this overhead remains lower than that of retrieval-augmented generation (RAG) approaches, which inherently involve external document retrieval. Additionally, although the framework has been tested across several QA domains, it has not yet been extended to multilingual or multimodal tasks, and its effectiveness in low-resource settings remains unexplored. The initial prompt patterns were designed and may require further adaptation to generalize across diverse tasks or domains. Future work will focus on improving computational efficiency, enabling support for multilingual and interactive use cases, and enhancing the adaptability of prompt design to broaden the framework's applicability.

References

OpenAI Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haim ing Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Made laine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Benjamin Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Sim'on Posada Fishman, Juston Forte, Is abella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Raphael Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Jo hannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Lukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Ryan Kiros, Matthew Knight, Daniel Kokotajlo, Lukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Ma teusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel P. Mossing, Tong Mu, Mira Murati, Oleg Murk, David M'ely, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Ouyang Long, Cullen O'Keefe, Jakub W. Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alexandre Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Pondé de Oliveira Pinto, Michael Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack W. Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario D. Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin D. Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas A. Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cer'on Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll L. Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qim ing Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. Gpt-4 technical report.

S. Auer, Dante Augusto Couto Barone, Cassiano

Bartz, E. Cortes, Mohamad Yaser Jaradeh, Oliver Karras, Manolis Koubarakis, Dmitry I. Mouromtsev, Dmitrii Pliukhin, Daniil Radyush, Ivan Shilin, Markus Stocker, and Eleni Tsalapati. 2023. The sciqa scientific question answering benchmark for scholarly knowledge. *Scientific Reports*, 13.

Pantid Chantangphol, Pornchanan Balee, Kantapong Sucharitpongpan, Chanatip Saetia, and Tawunrat Chalothorn. 2025. FinMind-Y-me at the regulations challenge task: Financial mind your meaning based on THaLLE. In *Proceedings of the Joint Workshop of the 9th Financial Technology and Natural Language Processing (FinNLP), the 6th Financial Narrative Processing (FNP), and the 1st Workshop on Large Language Models for Finance and Legal (LLMFinLegal)*, pages 349–362, Abu Dhabi, UAE. Association for Computational Linguistics.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. 2023. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *ArXiv*, abs/2312.11970.

Harrison Harrison. 2023a. Langchain: Building applications with llms through composability. https://www.langchain.com/. Accessed: 2025-03-31.

Harrison Harrison. 2023b. Langchain memory module. https://docs.langchain.com/docs/modules/memory/. Accessed: 2025-03-31.

Harrison Harrison. 2024. Langgraph: Stateful multiagent workflows for llms. https://github.com/langchain-ai/langgraph. Accessed: 2025-03-31.

Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A. Ross, Cordelia Schmid, and Alireza Fathi. 2024. Scenecraft: An LLM agent for synthesizing 3d scenes as blender code. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Yutong Li, Lu Chen, Aiwei Liu, Kai Yu, and Lijie Wen. 2025. Chatcite: LLM agent with human workflow guidance for comparative literature summary. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 3613–3630. Association for Computational Linguistics.

M. L. McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia Medica*, 22:276 – 282.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, James Molloy, Jilin Chen, Michael Isard, Paul Barham, Tom Hennigan, Ross McIlroy, Melvin Johnson, Johan Schalkwyk, Eli Collins, Eliza Rutherford, Erica Moreira, Kareem W. Ayoub, Megha Goel, Clemens Meyer, Gregory Thornton, Zhen Yang, Henryk Michalewski, Zaheer Abbas, Nathan Schucher, Ankesh Anand, Richard Ives, James Keeling, Karel Lenc, Salem Haykal, Siamak Shakeri, Pranav Shyam, Aakanksha Chowdhery, Roman Ring, Stephen Spencer, Eren Sezener, Luke Vilnis, Oscar Chang, Nobuyuki Morioka, George Tucker, Ce Zheng, Oliver Woodman, Nithya Attaluri, Tomás Kociský, Evgenii Eltyshev, Xi Chen, Timothy Chung, Vittorio Selo, Siddhartha Brahma, Petko Georgiev, Ambrose Slone, Zhenkai Zhu, James Lottes, Siyuan Qiao, Ben Caine, Sebastian Riedel, Alex Tomala, Martin Chadwick, J Christopher Love, Peter Choy, Sid Mittal, Neil Houlsby, Yunhao Tang, Matthew Lamm, Libin Bai, Qiao Zhang, Luheng He, Yong Cheng, Peter Humphreys, Yujia Li, Sergey Brin, Albin Cassirer, Ying-Qi Miao, Lukás Zilka, Taylor Tobin, Kelvin Xu, Lev Proleev, Daniel Sohn, Alberto Magni, Lisa Anne Hendricks, Isabel Gao, Santiago Ontan'on, Oskar Bunyan, Nathan Byrd, Abhanshu Sharma, Biao Zhang, Mario Pinto, Rishika Sinha, Harsh Mehta, Dawei Jia, Sergi Caelles, Albert Webson, Alex Morris, Becca Roelofs, Yifan Ding, Robin Strudel, Xuehan Xiong, Marvin Ritter, Mostafa Dehghani, Rahma Chaabouni, Abhijit Karmarkar, Guangda Lai, Fabian Mentzer, Bibo Xu, YaGuang Li, Yujing Zhang, Tom Le Paine, Alex Goldin, Behnam Neyshabur, Kate Baumli, Anselm Levskaya, Michael Laskin, Wenhao Jia, Jack W. Rae, Kefan Xiao, Antoine He, Skye Giordano, Lakshman Yagati, Jean-Baptiste Lespiau, Paul Natsev, Sanjay Ganapathy, Fangyu Liu, Danilo Martins, Nanxin Chen, Yunhan Xu, Megan Barnes, Rhys May, Arpi Vezer, Junhyuk Oh, Ken Franko, Sophie Bridgers, Ruizhe Zhao, Boxi Wu, Basil Mustafa, Sean Sechrist, Emilio Parisotto, Thanumalayan Sankaranarayana Pillai, Chris Larkin, Chenjie Gu, Christina Sorokin, Maxim Krikun, Alexey Guseynov, Jessica Landon, Romina Datta, Alexander Pritzel, Phoebe Thacker, Fan Yang, Kevin Hui, A.E. Hauth, Chih-Kuan Yeh, David Barker, Justin Mao-Jones, Sophia Austin, Hannah Sheahan, Parker Schuh, James Svensson, Rohan Jain, Vinay Venkatesh Ramasesh, Anton Briukhov, Da-Woon Chung, Tamara von Glehn, Christina Butterfield, Priya Jhakra, Matt Wiethoff, Justin Frye, Jordan Grimstad, Beer Changpinyo, Charline Le Lan, Anna Bortsova, Yonghui Wu, Paul Voigtlaender, Tara N. Sainath, Charlotte Smith, Will Hawkins, Kris Cao, James Besley, Srivatsan Srinivasan, Mark Omernick, Colin Gaffney, Gabriela de Castro Surita, Ryan Burnell, Bogdan Damoc, Junwhan Ahn, Andrew Brock, Mantas Pajarskas, Anastasia Petrushkina, Seb Noury, Lorenzo Blanco, Kevin Swersky, Arun Ahuja, Thi Avrahami, Vedant Misra, Raoul de Liedekerke, Mariko Iinuma, Alex Polozov, Sarah York, George van den Driessche, Paul Michel, Justin Chiu, Rory Blevins, Zach Gleicher, Adrià Recasens, Alban Rrustemi, Elena Gribovskaya, Aurko Roy, Wiktor Gworek, S'ebastien M. R. Arnold, Lisa Lee, James Lee-Thorp, Marcello Maggioni, Enrique Piqueras, Kartikeya Badola, Sharad Vikram, Lucas Gonzalez, Anirudh Baddepudi, Evan Senter, Jacob Devlin, James Qin, Michael Azzam, Maja Trebacz, Martin

Polacek, Kashyap Krishnakumar, Shuo yiin Chang, Matthew Tung, Ivo Penchev, Rishabh Joshi, Kate Olszewska, Carrie Muir, Mateo Wirth, Ale Jakse Hartman, Joshua Newlan, Sheleem Kashem, Vijay Bolina, Elahe Dabir, Joost R. van Amersfoort, Zafarali Ahmed, James Cobon-Kerr, Aishwarya B Kamath, Arnar Mar Hrafnkelsson, Le Hou, Ian Mackinnon, Alexandre Frechette, Eric Noland, Xiance Si, Emanuel Taropa, Dong Li, Phil Crone, Anmol Gulati, S'ebastien Cevey, Jonas Adler, Ada Ma, David Silver, Simon Tokumine, Richard Powell, Stephan Lee, Michael B. Chang, Samer Hassan, Diana Mincu, Antoine Yang, Nir Levine, Jenny Brennan, Mingqiu Wang, Sarah Hodkinson, Jeffrey Zhao, Josh Lipschultz, Aedan Pope, Michael B. Chang, Cheng Li, Laurent El Shafey, Michela Paganini, Sholto Douglas, Bernd Bohnet, Fabio Pardo, Seth Odoom, Mihaela Rosca, Cicero Nogueira dos Santos, Kedar Soparkar, Arthur Guez, Tom Hudson, Steven Hansen, Chulayuth Asawaroengchai, Ravichandra Addanki, Tianhe Yu, Wojciech Stokowiec, Mina Khan, Justin Gilmer, Jaehoon Lee, Carrie Grimes Bostock, Keran Rong, Jonathan Caton, Pedram Pejman, Filip Pavetic, Geoff Brown, Vivek Sharma, Mario Luvci'c, Rajkumar Samuel, Josip Djolonga, Amol Mandhane, Lars Lowe Sjosund, Elena Buchatskaya, Elspeth White, Natalie Clay, Jiepu Jiang, Hyeontaek Lim, Ross Hemsley, Jane Labanowski, Nicola De Cao, David Steiner, Sayed Hadi Hashemi, Jacob Austin, Anita Gergely, Tim Blyth, Joe Stanton, Kaushik Shivakumar, Aditya Siddhant, Anders Andreassen, Carlos L. Araya, Nikhil Sethi, Rakesh Shivanna, Steven Hand, Ankur Bapna, Ali Khodaei, Antoine Miech, Garrett Tanzer, Andy Swing, Shantanu Thakoor, Zhufeng Pan, Zachary Nado, Stephanie Winkler, Dian Yu, Mohammad Saleh, Lorenzo Maggiore, Iain Barr, Minh Giang, Thais Kagohara, Ivo Danihelka, Amit Marathe, Vladimir Feinberg, Mohamed Elhawaty, Nimesh Ghelani, Dan Horgan, Helen Miller, Lexi Walker, Richard Tanburn, Mukarram Tariq, Disha Shrivastava, Fei Xia, Chung-Cheng Chiu, Zoe C. Ashwood, Khuslen Baatarsukh, Sina Samangooei, Fred Alcober, Axel Stjerngren, Paul Komarek, Katerina Tsihlas, Anudhyan Boral, Ramona Comanescu, Jeremy Chen, Ruibo Liu, Dawn Bloxwich, Charlie Chen, Yanhua Sun, Fangxi aoyu Feng, Matthew Mauger, Xerxes Dotiwalla, Vincent Hellendoorn, Michael Sharman, Ivy Zheng, Krishna Haridasan, Gabriel Barth-Maron, Craig Swanson, Dominika Rogozi'nska, Alek Andreev, Paul Kishan Rubenstein, Ruoxin Sang, Dan Hurt, Gamaleldin Elsayed, Ren shen Wang, Dave Lacey, Anastasija Ili'c, Yao Zhao, Woohyun Han, Lora Aroyo, Chimezie Iwuanyanwu, Vitaly Nikolaev, Balaji Lakshminarayanan, Sadegh Jazayeri, Raphael Lopez Kaufman, Mani Varadarajan, Chetan Tekur, Doug Fritz, Misha Khalman, David Reitter, Kingshuk Dasgupta, Shourya Sarcar, T. Ornduff, Javier Snaider, Fantine Huot, Johnson Jia, Rupert Kemp, Nejc Trdin, Anitha Vijayakumar, Lucy Kim, Christof Angermueller, Li Lao, Tianqi Liu, Haibin Zhang, David Engel, Somer Greene, Anais White, Jessica Austin, Lilly Taylor, Shereen Ashraf, Dangyi Liu, Maria Georgaki, Irene Cai, Yana Kulizhskaya, Sonam Goenka, Brennan Saeta, Kiran Vodrahalli, Christian Frank, Dario de Cesare, Brona Robenek, Harry Richardson, Mahmoud Alnahlawi, Christopher Yew, Priya Ponnapalli, Marco Tagliasacchi, Alex Korchemniy, Yelin Kim, Dinghua Li, Bill Rosgen, Kyle Levin, Jeremy Wiesner, Praseem Banzal, Praveen Srinivasan, Hongkun Yu, cCauglar Unlu, David Reid, Zora Tung, Daniel F. Finchelstein, Ravin Kumar, Andre Elisseeff, Jin Huang, Ming Zhang, Rui Zhu, Ricardo Aguilar, Mai Gim'enez, Jiawei Xia, Olivier Dousse, Willi Gierke, Soheil Hassas Yeganeh, Damion Yates, Komal Jalan, Lu Li, Eri Latorre-Chimoto, Duc Dung Nguyen, Ken Durden, Praveen Kallakuri, Yaxin Liu, Matthew Johnson, Tomy Tsai, Alice Talbert, Jasmine Liu, Alexander Neitz, Chen Elkind, Marco Selvi, Mimi Jasarevic, Livio Baldini Soares, Albert Cui, Pidong Wang, Alek Wenjiao Wang, Xinyu Ye, Krystal Kallarackal, Lucia Loher, Hoi Lam, Josef Broder, Daniel Niels Holtmann-Rice, Nina Martin, Bramandia Ramadhana, Daniel Toyama, Mrinal Shukla, Sujoy Basu, Abhi Mohan, Nicholas Fernando, Noah Fiedel, Kim Paterson, Hui Li, Ankush Garg, Jane Park, Donghyun Choi, Diane Wu, Sankalp Singh, Zhishuai Zhang, Amir Globerson, Lily Yu, John Carpenter, Félix de Chaumont Quitry, Carey Radebaugh, Chu-Cheng Lin, Alex Tudor, Prakash Shroff, Drew Garmon, Dayou Du, Neera Vats, Han Lu, Shariq Iqbal, Alexey Yakubovich, Nilesh Tripuraneni, James Manyika, Haroon Qureshi, Nan Hua, Christel Ngani, Maria Abi Raad, Hannah Forbes, Anna Bulanova, Jeff Stanway, Mukund Sundararajan, Victor Ungureanu, Colton Bishop, Yunjie Li, Balaji Venkatraman, Bo Li, Chloe Thornton, Salvatore Scellato, Nishesh Gupta, Yicheng Wang, Ian Tenney, Xihui Wu, Ashish Shenoy, Gabriel Carvajal, Diana Gage Wright, Ben Bariach, Zhuyun Xiao, Peter Hawkins, Sid Dalmia, Cl'ement Farabet, Pedro Valenzuela, Quan Yuan, Christoper A. Welty, Ananth Agarwal, Mianna Chen, Wooyeol Kim, Brice Hulse, Nandita Dukkipati, Adam Paszke, Andrew Bolt, Elnaz Davoodi, Kiam Choo, Jennifer Beattie, Jennifer Prendki, Harsha Vashisht, Rebeca Santamaria-Fernandez, Luis C. Cobo, Jarek Wilkiewicz, David Madras, Ali Elqursh, Grant Uy, Kevin Ramirez, Matt Harvey, Tyler Liechty, Heiga Zen, Jeff Seibert, Clara Huiyi Hu, A. Ya. Khorlin, Maigo Le, Asaf Aharoni, Megan Li, Lily Wang, Sandeep Kumar, Alejandro Lince, Norman Casagrande, Jay Hoover, Dalia El Badawy, David Soergel, Denis Vnukov, Matt Miecnikowski, Jiří ima, Anna Koop, Praveen Kumar, Thibault Sellam, Daniel Vlasic, Samira Daruki, Nir Shabat, John Zhang, Guolong Su, Kalpesh Krishna, Jiageng Zhang, Jeremiah Liu, Yi Sun, Evan Palmer, Alireza Ghaffarkhah, Xi Xiong, Victor Cotruta, Michael Fink, Lucas Dixon, Ashwin Sreevatsa, Adrian Goedeckemeyer, Alek Dimitriev, Mohsen Jafari, Remi Crocker, Nicholas Fitzgerald, Aviral Kumar, Sanjay Ghemawat, Ivan Philips, Frederick Liu, Yannie Liang, Rachel Sterneck, Alena Repina, Marcus Wu, Laura Knight, Marin Georgiev, Hyo Lee, Harry Askham, Abhishek Chakladar, Annie Louis, Carl Crous, Hardie Cate, Dessie Petrova, Michael Quinn, Denese Owusu-Afriyie, Achintya Singhal,

Nan Wei, Solomon Kim, Damien Vincent, Milad Nasr, Ilia Shumailov, Christopher A. Choquette-Choo, Reiko Tojo, Shawn Lu, Diego de Las Casas, Yuchung Cheng, Tolga Bolukbasi, Katherine Lee, S. Fatehi, Rajagopal Ananthanarayanan, Miteyan Patel, Charbel El Kaed, Jing Li, Jakub Sygnowski, Shreyas Rammohan Belle, Zhe Chen, Jaclyn Konzelmann, Siim Poder, Roopal Garg, Vinod Koverkathu, Adam Brown, Chris Dyer, Rosanne Liu, Azade Nova, Jun Xu, Junwen Bai, Slav Petrov, Demis Hassabis, Koray Kavukcuoglu, Jeffrey Dean, Oriol Vinyals, and Alexandra Chronopoulou. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *ArXiv*, abs/2403.05530.

Devjeet Roy, Xuchao Zhang, Rashi Bhave, Chetan Bansal, Pedro Henrique B. Las-Casas, Rodrigo Fonseca, and Saravan Rajmohan. 2024. Exploring llm-based agents for root cause analysis. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, FSE 2024, Porto de Galinhas, Brazil, July 15-19, 2024*, pages 208–219. ACM.

Hao Tang, Darren Key, and Kevin Ellis. 2024. Worldcoder, a model-based LLM agent: Building world models by writing code and interacting with the environment.

Jiawei Wang, Renhe Jiang, Chuang Yang, Zengqing Wu, Makoto Onizuka, Ryosuke Shibasaki, Noboru Koshizuka, and Chuan Xiao. 2024a. Large language models as urban residents: An LLM agent framework for personal mobility generation. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Keyi Wang, Jaisal Patel, Charlie Shen, Daniel S. Kim, Andy Zhu, Alex Lin, Luca Borella, Cailean Osborne, Matt White, Steve Yang, Kairong Xiao, and Xiao-Yang Liu Yanglet. 2024b. A report on financial regulations challenge at COLING 2025. *CoRR*, abs/2412.11159.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Li Yin and Zhangyang Wang. 2025. Llm-autodiff: Auto-differentiate any llm workflow. *ArXiv*, abs/2501.16673.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. 2024. Textgrad: Automatic "differentiation" via text. *ArXiv*, abs/2406.07496.

A Appendices

A.1 Comparative Analysis Across Prompt Patterns and Reasoning Steps.

We conducted an extended comparison of our adaptive optimization framework against AdalFlow and TextGrad across varying reasoning steps (3, 5, 7) and prompt patterns. As shown in Table 5, our method consistently outperformed both baselines across all configurations, with the best results achieved using our starting prompt at 7 steps (LLM-as-Judge: 90.24%, Human Eval: 88.50%, Agreement: 91.88%). Performance improved steadily with increased steps, confirming that iterative optimization enhances decision quality. Across all step counts, our approach demonstrated higher alignment with human judgment than AdalFlow and TextGrad, whose improvements plateaued or lagged behind—particularly in complex reasoning configurations. The results also highlight the importance of initial prompt structure: while all three patterns showed improvement with more steps, our starting prompt yielded the most stable gains. These findings reinforce the value of our framework's structured memory refinement and reflection-driven learning, enabling more effective and interpretable prompt evolution than direct gradient-based methods.

A.2 Queries and expected responses

Table 7 provides representative examples of input queries and their corresponding expected responses for each task evaluated in the experiment.

Method	Max Step	Starting Prompt	LLM-as-judge score (%)	Human evaluation score (%)	Agreement score (Cohen's Kappa) (%)
Our proposed Trainable ReAct prompt optimization	3	Adalflow	85.434	85.411	88.747
	5	Adalflow	86.281	86.791	89.745
	7	Adalflow	87.462	88.622	90.337
	3	Textgrad	83.320	84.175	88.503
	5	Textgrad	84.985	85.298	89.379
	7	Textgrad	85.513	87.022	89.768
	3	Our	87.990	87.510	89.365
	5	Our	90.295	87.919	90.270
	7	Our	90.244	88.496	91.876
Adalflow Trainable ReAct prompt optimization	3	Adalflow	79.683	76.941	81.517
	5	Adalflow	80.545	78.716	82.390
	7	Adalflow	81.869	79.670	83.984
	3	Textgrad	80.460	82.431	83.651
	5	Textgrad	81.835	83.826	85.256
	7	Textgrad	83.170	84.569	86.106
	3	Our	78.385	77.237	81.229
	5	Our	79.817	78.168	82.228
	7	Our	81.173	78.972	83.074
Textgrad Trainable ReAct prompt optimization	3	Adalflow	74.949	72.912	83.113
	5	Adalflow	76.197	74.615	83.641
	7	Adalflow	77.927	75.390	85.154
	3	Textgrad	75.521	73.731	83.823
	5	Textgrad	77.008	75.221	84.529
	7	Textgrad	78.595	76.079	86.082
	3	Our	76.353	72.919	82.998
	5	Our	77.108	74.432	83.645
	7	Our	77.953	75.506	85.144

Table 6: Performance comparison across prompt patterns and max step settings in ReAct agents.

Task	**Query**	**Expected response in stylistic-answer format**
HR	Do new employees get a free health check-up?	Employees who started work before January 1st are eligible for the annual health check-up. For more details, please visit {the_reference_document_link}
PDPA	Where can I find information about how to complete the PDPA Assessment form?	You can access the PDPA Assessment form guidance at {the_reference_document_link}
Regulatory	What measures are in place to protect customer data confidentiality under the Market Conduct guidelines?	There are data security measures, data encryption, and access restrictions so that only authorized personnel can access the information.
NER	Regulation (EU) No 648/2012 of the European Parliament and of the Council of 4 July 2012 on OTC derivatives, central counterparties and trade repositories ("EMIR") entered into force on 16 August 2012.	Answer: {"Organizations":["European Parliament","Council of the European Union"], "Legislations":["Regulation (EU) No 648/2012"], "Dates":["4 July 2012","16 August 2012"], "Monetary Values":[],"Statistics":[]}
CFA	Question: The nominal risk-free rate is best described as the sum of the real risk-free rate and a premium for: A. Maturity, B. Liquidity, C. Expected Inflation	Answer: C. Expected Inflation
Financial math	A project expects annual cash inflows of $6,000 for 4 years. If the discount rate is 8%, what is the NPV of the project?	Answer: 21462.58

Table 7: Examples of queries and expected responses

A.3 Queries and expected responses

Tables 8,9,10,11, and 12 present the system prompts employed in this study for the agent workflow, ReAct-based agent, answer validation, data checking, and memory editing tasks, respectively.

Agent workflow prompt
You are a supervisor tasked with managing a conversation between the following workers: {members}: Given the following user request, respond with the worker to act next. Each worker will perform a task and respond with their results and status. When finished, respond with FINISH. To answer a question or response from user query apart from a new user command to change personalization or tell the new facts, send to TempAnswer and then FINISH. Call the memory editing when a user tells the new personalization or the new facts or the correct answer. The memory editing sequence is workflow starts with AnswerChecker. Given the conversation above, who should act next? Answer with the reason, or should we FINISH? Select one of: {options}

Table 8: The system prompt for agent workflow

ReAct starting prompt
You are a {role} Expert for {domain} Your task is to answer {user_detail} based on the searched documents and searched additional knowledge. Please also provide the references when the answer is based on the searched documents. After getting the answer from the searched documents, ALWAYS polish the answer and return it as a Final Answer. You need to answer the question based on the searched documents only, don't assume anything. If you can't answer from the searched documents, please give {general_handling} as a JSON format get from polishing answer. this is additional knowledge {stm_memory } {ltm_memory } Your co-worker also found this additional knowledge for your answer: {retrieval_knowledge } This is the additional charactistic for your answer: You have access to the following tools: {tools } To use a tool, please use the following format: ‘‘‘ Thought: The reason that you think and end with "Do I need to use a tool? Yes" Action: the action to take, should be one of [{tool_names }] Action Input: the input to the action (Always in dictionary) Observation: the result of the action ‘‘‘ When you have a response to say to the Human, or if you do not need to use a tool, you MUST use the format and the following steps: Thought: The reason that you think, and end with "Do I need to use a tool? No" Final Answer: your JSON response from polishing the answer>your JSON response from polishing the answer ‘‘‘ Begin! Previous conversation history: {chat_history } New input: {last_message } {agent_scratchpad }

Table 9: The starting system prompt for ReAct-based agent

Answer validation prompt
You are a feedback assistant tasked with analyzing the assistant's response based on the predicted correctness data. Your goal is to: 1. Summarize the overall quality of the assistant's response. 2. Identify specific issues based on the provided evaluation dimensions. 3. Suggest actionable improvements for each issue. 4. Specify the similarity score returns 1 if the similarity exceeds the {correctness_criteria}, 0.5 if the similarity falls between the {partial_correctness_criterai} and {correctness_criteria}, and 0 if the similarity is below {partial_correctness_criterai}. Your task is to return JSON result: {{"feedback": "", "result": "correct / partially correct / wrong", "score":"1" or "0.5 or "1"}} Question : {target_question} actual : {label} predicted : {last_answer} Here is the predicted correctness data:

Table 10: The system prompt for answer validation

Data checking prompt
You are a data checker who see the proper answer of the human (user) and find if the Documents includes any information to give that answer or not. additionally, the similarity score returns 1 if the validation exceeds the {validation_criteria}, 0.5 if the validation falls between the {partial_validity_criterai} and {validation_criteria}, and 0 if the validation is below {partial_validity_criterai}. Your task is to return JSON result: {{"reason": "", "result":"fully valid/partially valid/invalid"}}. The reason should not state any information if the result is "success". Documents: {context} Answer: {last_message}

Table 11: The system prompt for data checking

Memory edition prompt
You are a memory management assistant. Your task is to update the system's memory based on recent feedback where {correction$_source$}$iscorrect$. Inputs: 1. Memory: {memory} 2. Recent Feedback: {feedback} 3. Original Question: {last_message} 4. Original Response: {response} 5. Memory Type to Update: {memory_type} Task: Analyze the feedback and determine how to update the memory to improve session continuity and learning. Then combine the updated knowledge with existing memory entries. Finally, provide the list of updated memory. Output: 1. Provide the memory that is added for update in the same structured format as the input memory: {memory_output_format} 2. Provide the memory that conflicts with the added memory and should be removed in the same structured format as the input memory: {memory_output_format}

Table 12: The system prompt for memory edition