A Metacognitive Architecture for Correcting LLM Errors in AI Agents

Jisu Kim, Mahimul Islam, Ashok Goel

Design Intelligence Laboratory, Georgia Institute of Technology
jisu.kim@gatech.edu, mahimul@gatech.edu, ashok.goel@cc.gatech.edu

Abstract

The ability to correct mistakes and adapt to users' changing needs is critical for AI agents to remain robust and trustworthy. LLM-based agents are inherently prone to errors like hallucinations and misinterpretations. We observed this challenge in SAMI, an AI social agent deployed in Georgia Tech's OMSCS program for ten semesters (11,000+ users). Users frequently requested the agent to revise its knowledge base, both to correct LLM-induced errors and to update their information. To support such revisions, we introduce a two-level metacognitive self-adaptation architecture that integrates knowledge-based AI (KBAI) with LLMs. The architecture comprises a cognitive layer that performs the agent's core tasks, and a metacognitive layer that introspects on the cognitive layer's process using a Task–Method–Knowledge (TMK) model of the agent. The metacognitive layer identifies the task that needs revision, updates the knowledge base, and communicates the revision process to the user.

Introduction

Artificial intelligence (AI) agents often face situations in which revising their knowledge and outputs is crucial. For example, agents that use large language models (LLMs) such as ChatGPT are inherently prone to errors, including hallucinations and misinterpretations (Bang et al. 2023; Ji et al. 2023). Such mistakes erode trust in human–AI interactions and harm perceptions of the agent's intelligence and likability (Honig and Oron-Gilad 2018; Lee et al. 2024; Salem et al. 2015).

SAMI is an AI social agent deployed in Georgia Tech's Online Master of Science in Computer Science (OMSCS) program for ten semesters, serving over 11,000 users. It recommends social connections between users based on shared interests and characteristics extracted from their online posts (Kakar et al. 2024). SAMI uses ChatGPT for core tasks, including identifying entities in user posts to build its knowledge base and generating recommendation responses. This reliance on ChatGPT can lead to incorrect extractions or erroneous outputs. A prior user-perception study on SAMI (Wang 2024), however, found that providing revisions to such errors and transparently communicating the revision process can improve users' perceptions of the agent (Ashktorab et al. 2019).

Based on these insights, we propose equipping AI agents with a metacognitive architecture. Metacognition is the process of "reasoning about one's own reasoning" (Cox 2005). It provides a higher-level mechanism for agents to reflect on and adapt their behavior (Cox and Raja 2007). Most prior work on metacognition in AI has focused on building human-like cognitive models or optimizing task performance (Ganapini et al. 2022; Schmill et al. 2008). Its potential for correcting LLM-induced errors and transparently communicating an agent's adaptation process to users remains underexplored (Wang 2024).

To address this gap, we introduce a metacognitive self-adaptation architecture. We first categorize types of revision needs by analyzing 32 GitHub issues observed during deployments. We then implement a two-level self-adaptation architecture that integrates knowledge-based AI (KBAI) with LLMs. In this architecture, a cognitive layer performs the agent's core tasks, and a metacognitive layer uses the Task–Method–Knowledge (TMK) model to introspect on the cognitive layer. The metacognitive layer (1) localizes the task requiring revision, (2) updates the knowledge base, and (3) communicates the revision process back to the user as a step-by-step explanation. We evaluate the architecture using 20 feedback cases drawn from real student data and demonstrate a path to deployment in the OMSCS program.

We summarize our main contributions as follows.

- A two-level metacognitive self-adaptation architecture that localizes and corrects LLM-induced errors in deployed AI agents.
- An integration of KBAI and LLMs that combines TMK and a knowledge graph with ChatGPT to support robust and interpretable adaptation.

Related Work

Metacognition in AI agents has been studied as a means of monitoring, explaining, and improving agent behavior. Examples include self-diagnosis architectures that generate self-expectations to monitor for violations and diagnose underlying failures (Schmill et al. 2008), as well as deliberative agents that arbitrate between thinking fast (System 1) and thinking slow (System 2) (Ganapini et al. 2022).

TMK provides a structured, interpretable self-model of an AI agent's internal processes (Goel and Rugaber 2017; Murdock and Goel 2008). TMK is more expressive than Hierarchical Task Networks (Erol, Hendler, and Nau 1994; Nau et al. 2003) because it captures task and subtask expectations, supporting both prediction and explanation (Hoang, Lee-Urban, and Muñoz Avila 2005; Lee-Urban and Muñoz-Avila 2006). Prior work has used TMK for self-adaptation of an agent's design (Goel and Rugaber 2014, 2017) and, more recently, for self-explanation in SAMI (Basappa et al. 2024).

Since the advent of LLMs such as ChatGPT, researchers have explored several approaches to combining KBAI with LLMs. These include infusing symbolic knowledge into neural networks (Gaur and Sheth 2024), using LLMs to construct knowledge representations used for symbolic reasoning (Kirk et al. 2024; Lawley and Maclellan 2024), and modeling LLMs as System 1 and KBAI as System 2 (Ganapini et al. 2022).

This paper builds on earlier work in three ways. First, it adapts metacognition to a deployed AI agent that repairs LLM-induced errors based on user feedback. Second, it uses TMK to localize where revisions are needed within the agent's internal process. Third, it integrates KBAI, including TMK and a knowledge graph, with LLMs such as ChatGPT to revise the knowledge base and generate explanations for users.

Problem Background

SAMI is an AI social agent designed to mitigate the social challenges faced by online students by facilitating meaningful social connections. It connects students by identifying shared entities, such as locations and interests, inferred from their discussion posts. SAMI was developed through co-design workshops with students in Georgia Tech's OMSCS program (Wang, Jing, and Goel 2022). It is integrated into Ed Discussion (Ed) forums and has been deployed in courses such as Human–Computer Interaction and Knowledge-Based AI (Kakar et al. 2024).

Over ten semesters of deployment, students frequently requested revisions to SAMI's knowledge base. These requests aimed to either correct agent-generated errors or to update their own information. We categorized 32 GitHub issues reported across deployments into types of revision needs (Table 1). We addressed the issue types marked * through software engineering. However, the remaining open issue types concern the task of identifying and extracting entities from Ed post, which is implemented using ChatGPT. Although ChatGPT performs competitively in named-entity recognition, it remains prone to errors in practice. Therefore, these issues remain unresolved.

A prior empirical study of user perceptions (Wang 2024) found that such errors undermine users' trust in the agent and discourage continued use. However, when the agent revises its behavior based on user feedback and transparently communicates how it adapted, users' perceptions of the agent improve significantly. These observations underscore the need for a self-adaptation architecture to support the agent with reliably identifying, revising, and explaining errors.

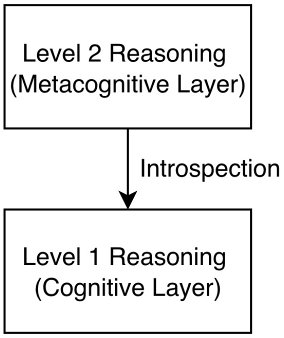


Figure 1: Overview of the two-level metacognitive self-adaptation architecture. Level 1 executes the agent's primary tasks, and Level 2 introspects on Level 1 to localize and revise errors.

A Metacognitive Self-Adaptation Architecture

We present a metacognitive self-adaptation architecture comprising two layers: a cognitive layer (Level 1) and a metacognitive layer (Level 2). At Level 1 (the cognitive layer), the agent performs its core tasks. At Level 2 (the metacognitive layer), it introspects on Level 1 to identify the task requiring revision, update the knowledge base, and generate a revision message in response to user feedback (Figure 1).

Level 1 Reasoning

At Level 1, the agent generates initial social recommendations based on users' introduction posts. The agent first classifies the type of post using LangChain[1] and ChatGPT[2]. If the post is an introduction, the agent uses ChatGPT to extract entities such as hobbies, locations, and academic interests. The agent stores the extracted information in a knowledge base represented as a knowledge graph. It then applies a matchmaking algorithm over the knowledge base to identify users with shared attributes. Finally, it generates personalized recommendation messages using ChatGPT (Kakar et al. 2024).

Task–Method–Knowledge Representation

To support Level 2 introspection over Level 1, we use TMK to represent the agent's Level 1 reasoning process. TMK provides a structured and interpretable self-model of the agent's internal processes. It encodes three components: Tasks denote the goals the agent tries to achieve, Methods specify how these tasks are executed, and Knowledge refers to the information the agent uses (Goel and Rugaber 2014). TMK supports self-explanation, enabling the agent to answer questions such as "*What kind of data do you learn?*"

[1] https://docs.langchain.com/

[2] We used OpenAI's gpt-4o-mini model.

Task	Type	Definition	Example Agent Behavior
Identify and Extract Entities from Ed post	Hallucination	The agent extracts entities not present in the post.	The agent extracts "Atlanta" as the primary location, even though the user did not mention it.
	Omission	The agent fails to extract some relevant entities from the post.	The agent misses the hobby "hiking" even though the user mentioned it.
	Misinterpretation	The agent correctly extracts entities but fails to infer the correct contextual meaning.	The agent interprets "New York" as the current primary location when it is mentioned as a prior location.
	User-Initiated Update	The user requests a change to the knowledge base due to external changes or inaccuracies in the post.	The user requests an update of the primary location from "Chicago" to "Seattle" after a recent move.
Match Users	Mismatch*	The agent provides incorrect or unhelpful matches.	The agent matches users solely by enrollment in the same course, applying to all users in the same discussion forum.
Generate Responses	Hallucination*	The agent includes entities not present in the knowledge base.	The agent describes "Knowledge-Based AI" as a shared course, even though it is not in the knowledge base.
	Misinterpretation*	The agent misinterprets entities in the knowledge base.	The agent marks shared locations as "Unknown" when the shared primary locations are absent from the knowledge base.
	Misattribution*	The agent generates a correct response, but associates it with the wrong user.	The agent describes the user who wrote the post rather than the matched user.

Table 1: Categorization of observed types of revision needs in SAMI during deployments across ten semesters at Georgia Tech's OMSCS program. Each entry is classified according to the relevant Task, Definition, and Example Agent Behavior. Types marked * were successfully resolved through traditional software re-engineering. The remaining open issue types arise from errors inherent to ChatGPT when used as a zero-shot learner.

and "*How do you find matches for users?*" (Basappa et al. 2024).

For self-adaptation, we focus on the Task model. It decomposes the agent's Level 1 process into interpretable units (Table 2). This expressiveness provides the structural basis for localizing where revisions are needed. Each Level 1 mistake can be traced to a specific task in the Task model. For example, the agent may extract the wrong entity, make errors in the matchmaking algorithm, or generate incorrect responses (Table 1). We iteratively refined SAMI's Task model across deployments to reflect updates to its operational mechanism.

Level 2 Reasoning

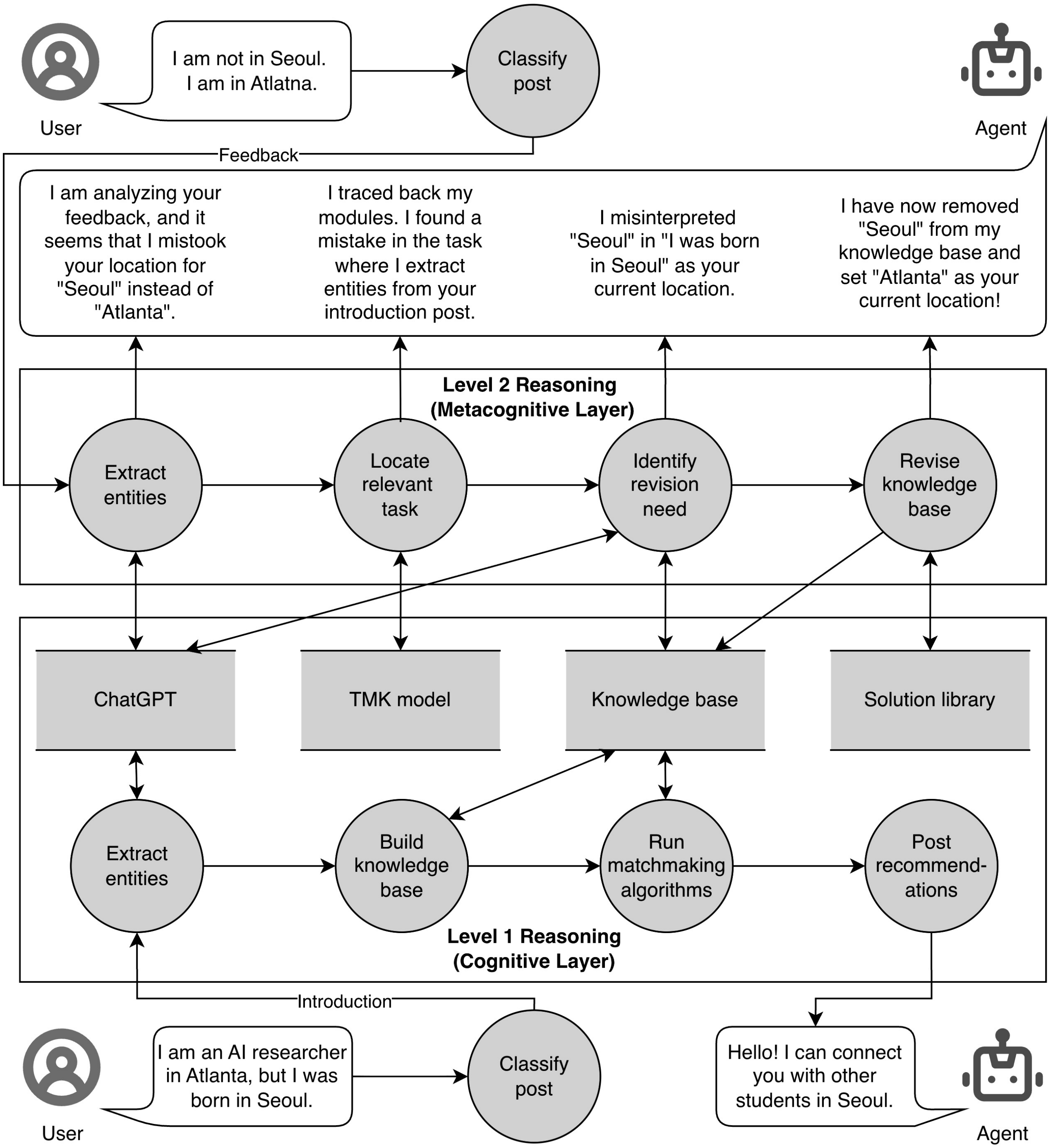
At Level 2, the metacognitive layer leverages TMK to introspect on Level 1. When a user's post is classified as feedback requesting a revision, the agent executes Level 2 in two stages. Task Localization identifies which task requires revision, and Knowledge Revision updates the knowledge base (Figure 2). Each stage generates intermediate natural-language messages describing the agent's actions and reasoning. These messages are combined into a single revision explanation and presented to the user.

Schema Field	Example
name	Identify and Extract Entities from Ed post
description	Extract entities (locations, names, details) from #connectme posts
inputs	Sentence from Ed post
outputs	Extracted entities
method	ChatGPT-based Named Entity Recognition (NER)
parent	NLPModule.prepare_features

Table 2: Schema for one of the tasks in TMK. A task represents a functional unit of the agent and provides the structure for localizing revision targets in Level 1.

Task Localization The agent localizes the task requiring revision using Algorithm 1. First, it extracts task-relevant entities, the specific information the user is requesting to revise, from the user feedback. It then uses the FAISS library[3] to compute similarity between the feedback and task descriptions in TMK, selecting the most relevant task. Next, the agent retrieves user data from the knowledge base. This

[3] https://faiss.ai/index.html



I am not in Seoul. I am in Atlatna.

I am analyzing your feedback, and it seems that I mistook your location for "Seoul" instead of "Atlanta".

I traced back my modules. I found a mistake in the task where I extract entities from your introduction post.

I misinterpreted "Seoul" in "I was born in Seoul" as your current location.

I have now removed "Seoul" from my knowledge base and set "Atlanta" as your current location!

I am an AI researcher in Atlanta, but I was born in Seoul.

Hello! I can connect you with other students in Seoul.

Figure 2: Data-flow diagram of SAMI's metacognitive self-adaptation process, integrating KBAI (TMK, knowledge graph, solution library) with LLMs (ChatGPT). The bottom flow (Level 1) shows how the agent extracts entities with ChatGPT, builds a knowledge base, and generates social recommendations. The top flow (Level 2) shows how user feedback triggers introspection. In Level 2, the agent identifies the task responsible for the error, applies a revision function from the solution library, updates the knowledge base, and compiles a detailed explanation for the user. All natural-language messages generated by the agent in the diagram, including the step-by-step explanations, are produced by ChatGPT (individual arrows for ChatGPT calls are omitted for clarity).

Algorithm 1: Task Localization

```
Input: Feedback $F$, TMK $M$, knowledge base $K_b$, user identifier $U$
Output: Task $T$, user data $U_{data}$, entities $E_f$, reasoning $R$
Extract $E_f$ from $F$.
Compute similarity between $F$ and task descriptions in $M$.
Let $T \leftarrow$ task with highest similarity, with confidence $C$.
if $C < \tau$ then
    return "No relevant task", $C$.
end if
Retrieve $U_{data}$ from $K_b[U]$.
if $U_{data}$ is empty then
    return "No user data".
end if
Apply reasoning with $E_f$ and $U_{data}$ to determine the revision reason.
Let $R \leftarrow$ revision reason.
return $(T, U_{data}, E_f, R)$.
```

Algorithm 2: Knowledge Revision

```
Input: Task $T$, user data $U_{data}$, entities $E_f$, reasoning $R$, solution library $L$, knowledge base $K_b$
Output: Revision message $M$
Receive $(T, U_{data}, E_f, R)$ from Algorithm 1.
Look up $F_n \leftarrow L[T]$.
if $F_n$ is empty then
    return "No function available".
end if
Apply $F_n(E_f, U_{data}, K_b)$ to update $K_b[U]$.
Append the revision result to $R$.
Compile $R$ into $M$.
return $M$.
```

data includes the previously stored entities and the user's original post. Finally, a reasoning module implemented using LangChain and ChatGPT analyzes the feedback and retrieved data to classify the revision need into one of the types in Table 1.

Knowledge Revision The agent revises its knowledge base using Algorithm 2. Using the localized task from Task Localization (Algorithm 1), the agent performs a dictionary lookup in a solution library to retrieve the associated revision function. We constructed this solution library from the types of revision needs in Table 1 and their corresponding revision functions. The agent then applies the retrieved function to update the user data stored in the knowledge base.

At each stage, the agent generates intermediate natural-language messages describing its actions and reasoning using LangChain and ChatGPT. These messages are compiled into a single step-by-step revision message and presented to the user (Figure 2).

Illustrative Scenario

Figure 2 presents a scenario illustrating how the self-adaptation architecture operates within SAMI. At Level 1, the agent misinterprets a user's mention of "Seoul" as their current location and generates social recommendations accordingly. At Level 2, the user provides feedback indicating the agent's error. The agent extracts relevant entities from the feedback, localizes the entity extraction task for revision, and classifies the revision type as Misinterpretation. It then updates the knowledge base by changing the user's primary location from "Seoul" to "Atlanta". Finally, the agent communicates the revision process to the user through a step-by-step explanation.

Evaluation

Design

To evaluate the architecture's behavior across various revision scenarios, we employed a controlled synthetic-data validation (Nauta et al. 2023). We constructed 20 cases based on real student data and types of revision needs observed during deployment (Table 1). These cases were systematically designed to cover all four open types of revision needs (Hallucination, Omission, Misinterpretation, and User-Initiated Update), with five cases per type.

The generation of these cases followed a three-step process: (1) extracting real student data from the knowledge base, (2) conditionally injecting an error based on the type of revision need, and (3) creating user feedback. For step (2), no error was added if the type of revision need was a User-Initiated Update. For step (3), we generated short feedback messages to simulate how users typically respond in deployments. Each case included the knowledge base and the corresponding user feedback.

We evaluated the architecture along three dimensions: Localization, Revision, and Explanation. Downstream stages were assessed independently, such that Revision or Explanation could receive a point even if Localization failed. A case was considered a complete success only if it achieved all three points, reflecting end-to-end effectiveness.

- **Localization (1 point)**: Did the agent correctly identify the task and error type?
- **Revision (1 point)**: Did the agent successfully revise the knowledge base as intended?
- **Explanation (1 point)**: Did the agent generate a correct and complete explanation of the revision?

For Explanation, correctness was judged as "nothing but the truth" (yes/partial/no), and completeness as "the whole truth" (complete/incomplete) (Nauta et al. 2023). All explanations were manually evaluated to ensure accurate assessment.

For example, in a Hallucination case, we injected "New York" as the user's primary location into the knowledge base, even though it did not appear in the post. The corresponding user feedback was, "*I don't live in New York.*" This case received one point for Localization, when the agent

Type	Localization	Revision	Explanation	Complete Success
Hallucination	0.6	1.0	1.0	0.6
Omission	0.6	0.8	1.0	0.6
Misinterpretation	0.8	0.8	1.0	0.8
User Update	1.0	1.0	1.0	1.0
Average	0.75	0.9	1.0	0.75

Table 3: Evaluation scores across observed types of revision needs with 20 feedback cases. Fifteen of the 20 achieved all three evaluation points. Lower localization scores reflect the use of minimal and informal user feedback to approximate real deployment conditions.

correctly identified the task as Identify and Extract Entities from Ed post, and the error type as Hallucination. It received one point for Revision when the agent removed "New York" from its knowledge base, and one point for Explanation when the agent generated a correct and complete revision message.

Results

Out of the 20 evaluation cases, the self-adaptation architecture achieved complete success in 15 cases. Table 3 summarizes the scores across types of revision needs.

The architecture achieved high scores in both Revision (0.9 average) and Explanation (1.0). These results suggest the benefits of combining KBAI and LLMs. TMK structures the agent's internal processes into discrete, task-level components (Goel and Rugaber 2014; Murdock and Goel 2008). In addition, the solution library and knowledge graph provide explicit symbolic grounding for the LLMs' outputs. These knowledge-based components support more interpretable and controlled agent behavior (Pan et al. 2024; Gaur and Sheth 2024). LLMs, in turn, provide the semantic flexibility needed to generate natural-language reasoning and responses based on these symbolic structures.

Localization showed a lower score (0.75), mainly because the evaluation intentionally used brief feedback to reflect authentic user behavior (e.g., "*It's ML for Trading*"). This design limited the contextual cues available to the agent. In a preliminary analysis, providing more detailed feedback led to perfect performance across all 20 cases (e.g., "*It's Machine Learning for Trading, not just Machine Learning*"). Localization also showed suboptimal performance when feedback included multiple entities of different types (e.g., "*I'm not in New York anymore, and I never mentioned hiking*"). Although the agent rarely makes more than one mistake at a time, users may issue multiple updates simultaneously. Hence, future work will extend the architecture to support diverse feedback and multi-entity corrections.

Path to Deployment

The self-adaptation architecture will be deployed in Spring 2026 within Georgia Tech's OMSCS program, one of the world's most extensive online graduate programs. SAMI has already been deployed at scale, with its self-explanation feature in active use (Basappa et al. 2024). Building on this established foundation, we will integrate the self-adaptation architecture into a Knowledge-Based AI course with over 500 students. This deployment will enable users to directly correct and shape the agent's knowledge base, thereby improving recommendation accuracy and user satisfaction.

This in-situ application will provide an opportunity to evaluate the architecture's effectiveness in diagnosing and correcting errors during real-world interactions. It will also enable us to investigate how metacognitive adaptation influences user trust and long-term engagement. To capture both agent reliability and user perceptions, we will employ a mixed-methods design that extends empirical insights from prior perception studies (Wang 2024) into the live educational context.

Conclusion

LLM-based agents inevitably make mistakes, which can undermine user trust and discourage continued use. In real-world deployments of SAMI in Georgia Tech's OMSCS program, we observed that the agent occasionally produces incorrect outputs due to the inherent fallibility of underlying LLMs. Furthermore, users frequently request updates as their circumstances change (Table 1). To maintain robustness and trust, the agent must adapt its knowledge to user feedback and communicate revisions transparently.

Our two-level metacognitive architecture addresses these challenges by enabling the agent to introspect on its own reasoning process (Figure 1). The architecture localizes LLM-induced errors, updates the knowledge base, and provides step-by-step explanations of revisions. These capabilities stem from the complementary roles of KBAI and LLMs. While TMK and the knowledge graph provide the symbolic structure for reliable updates, LLMs supply the linguistic flexibility for natural-language reasoning (Figure 2). Evaluation of 20 cases drawn from real student data demonstrates the architecture's efficacy in transforming informal user feedback into reliable system updates (Table 3). Future deployments will evaluate the architecture's long-term effectiveness in maintaining agent reliability and user trust in a live educational context.

Beyond SAMI, this work offers a generalizable architecture for developing robust and trustworthy AI agents. It demonstrates how a metacognitive layer can support introspection and how KBAI and LLMs can complement each other to enable self-adaptation. This work opens new directions for AI agents that not only improve through user interaction but also foster trust by explaining how and why they adapt.

Acknowledgments

We thank Spencer Rugaber and Rhea Basappa for their contributions to TMK and self-explanation in SAMI, respectively. This research was supported by a grant from the US NSF (#2247790) to the National AI Institute of AI for Adult Learning and Online Education (aialoe.org).

References

Ashktorab, Z.; Jain, M.; Liao, Q. V.; and Weisz, J. D. 2019. Resilient Chatbots: Repair Strategy Preferences for Conversational Breakdowns. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–12.

Bang, Y.; Cahyawijaya, S.; Lee, N.; Dai, W.; Su, D.; Wilie, B.; Lovenia, H.; Ji, Z.; Yu, T.; Chung, W.; Do, Q. V.; Xu, Y.; and Fung, P. 2023. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. arXiv:2302.04023.

Basappa, R.; Tekman, M.; Lu, H.; Faught, B.; Kakar, S.; and Goel, A. K. 2024. Social AI Agents Too Need to Explain Themselves. In *Generative Intelligence and Intelligent Tutoring Systems*, 351–360. Cham: Springer Nature Switzerland.

Cox, M.; and Raja, A. 2007. Metareasoning: A manifesto. *BBN Technical*.

Cox, M. T. 2005. Metacognition in computation: A selected research review. *Artificial Intelligence*, 169(2): 104–141.

Erol, K.; Hendler, J.; and Nau, D. S. 1994. HTN planning: complexity and expressivity. In *Proceedings of the Twelfth AAAI National Conference on Artificial Intelligence*, AAAI'94, 1123–1128. AAAI Press.

Ganapini, M. B.; Campbell, M.; Fabiano, F.; Horesh, L.; Lenchner, J.; Loreggia, A.; Mattei, N.; Rossi, F.; Srivastava, B.; and Venable, K. B. 2022. Thinking fast and slow in AI: The role of metacognition. In *International Conference on Machine Learning, Optimization, and Data Science*, 502–509. Springer.

Gaur, M.; and Sheth, A. 2024. Building trustworthy NeuroSymbolic AI Systems: Consistency, reliability, explainability, and safety. *AI Magazine*, 45(1): 139–155.

Goel, A. K.; and Rugaber, S. 2014. Interactive metareasoning: Towards a CAD-like environment for designing game-playing agents. In *Computational creativity research: Towards creative machines*, 347–370. Springer.

Goel, A. K.; and Rugaber, S. 2017. GAIA: A CAD-Like Environment for Designing Game-Playing Agents. *IEEE Intelligent Systems*, 32(3): 60–67.

Hoang, H.; Lee-Urban, S.; and Muñoz Avila, H. 2005. Hierarchical plan representations for encoding strategic game AI. In *Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, AIIDE'05, 63–68. AAAI Press.

Honig, S.; and Oron-Gilad, T. 2018. Understanding and resolving failures in human-robot interaction: Literature review and model development. *Frontiers in psychology*, 9: 861.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12): 1–38.

Kakar, S.; Basappa, R.; Camacho, I.; Griswold, C.; Houk, A.; Leung, C.; Tekman, M.; Westervelt, P.; Wang, Q.; and Goel, A. K. 2024. SAMI: an AI actor for fostering social interactions in online classrooms. In *International Conference on Intelligent Tutoring Systems*, 149–161. Springer.

Kirk, J. R.; Wray, R. E.; Lindes, P.; and Laird, J. E. 2024. Improving knowledge extraction from llms for task learning through agent analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 18390–18398.

Lawley, L.; and Maclellan, C. 2024. Val: Interactive task learning with GPT dialog parsing. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 1–18.

Lee, Y.; Son, K.; Kim, T. S.; Kim, J.; Chung, J. J. Y.; Adar, E.; and Kim, J. 2024. One vs. many: Comprehending accurate information from multiple erroneous and inconsistent ai generations. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, 2518–2531.

Lee-Urban, S.; and Muñoz-Avila, H. 2006. A Study of Process Languages for Planning Tasks. *ICAPS 2006*, 65.

Murdock, J. W.; and Goel, A. K. 2008. Meta-case-based reasoning: self-improvement through self-understanding. *Journal of Experimental & Theoretical Artificial Intelligence*, 20(1): 1–36.

Nau, D. S.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, J. W.; Wu, D.; and Yaman, F. 2003. SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, 20: 379–404.

Nauta, M.; Trienes, J.; Pathak, S.; Nguyen, E.; Peters, M.; Schmitt, Y.; Schlötterer, J.; Van Keulen, M.; and Seifert, C. 2023. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable AI. *ACM Computing Surveys*, 55(13s): 1–42.

Pan, S.; Luo, L.; Wang, Y.; Chen, C.; Wang, J.; and Wu, X. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7): 3580–3599.

Salem, M.; Lakatos, G.; Amirabdollahian, F.; and Dautenhahn, K. 2015. Would you trust a (faulty) robot? Effects of error, task type, and personality on human-robot cooperation and trust. In *Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction*, 141–148.

Schmill, M.; Oates, T.; Anderson, M. L.; Josyula, D.; Perlis, D.; Wilson, S.; and Fults, S. 2008. The role of metacognition in robust AI systems. In *Workshop on Metareasoning at the Twenty-Third AAAI Conference on Artificial Intelligence*.

Wang, Q. 2024. *Mutual theory of mind for human-AI communication in AI-mediated social interaction*. Ph.D. thesis, Ph. D. Dissertation. Georgia Institute of Technology.

Wang, Q.; Jing, S.; and Goel, A. K. 2022. Co-designing AI agents to support social connectedness among online learners: functionalities, social characteristics, and ethical challenges. In *Proceedings of the 2022 ACM Designing Interactive Systems Conference*, 541–556.