GAKU: TOWARDS DYNAMIC COGNITION ON KNOWLEDGE GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge graphs (KGs) provide structured and reliable factual knowledge, yet large language models (LLMs) often struggle to align their parametric knowledge with the semantics, conventions, and coverage of a specific KG. This mismatch is particularly problematic in domain-specific knowledge reasoning, where conventional adaptation often requires substantial annotation and fine-tuning. In this paper, we propose GAKU, a multi-agent framework that builds dynamic cognition over KGs. GAKU follows a human-like cognitive loop: inspecting local graph evidence, reflecting on mistakes, inducing ontology-level cognition, validating them, and reusing the validated knowledge in subsequent decisions. The resulting cognition state bridges episodic experience and reusable reasoning skills, enabling an LLM to adapt to unfamiliar KG-specific semantics without parameter fine-tuning. Extensive experiments on KG fact verification and completion show that GAKU consistently outperforms strong fine-tuning and training-free baselines, using only 100 labeled examples to construct dynamic cognition. Further analyses demonstrate that the gains generalize across diverse backbone LLMs and support cross-model cognition transfer, highlighting explicit cognitive adaptation as a data-efficient and parameter-free approach to KG-grounded reasoning.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities across diverse tasks, drawing on vast knowledge acquired during training (Achiam et al., 2023; Touvron et al., 2023). However, such knowledge is implicit and may be incomplete, outdated, or misaligned with a specific context. Knowledge graphs (KGs), by contrast, represent facts explicitly in structured form, offering a reliable source of contextual knowledge. This complementarity has inspired extensive efforts to integrate LLMs with KGs (Pan et al., 2023; Zhu et al., 2023; Zhang et al., 2024; Guo et al., 2024).

Yet, integrating KGs with LLMs is not simply a matter of providing missing knowledge. LLMs enter downstream reasoning with strong priors acquired during pretraining, which may diverge from the semantics, conventions, or coverage of a particular KG. When such a mismatch arises, the model may remain overconfident in its own knowledge and underuse the graph, even when the task explicitly calls for KG-grounded reasoning (Liu et al., 2025).

Consider the example on the left of Figure 1, where the model is asked to verify the triplet $(\text{Budapest}, \textit{country}, \text{Austria-Hungary})$. An LLM may reject this triplet because its prior knowledge strongly associates Budapest with present-day Hungary. Yet, under the semantics of the target KG, historical affiliation can also constitute a valid instance of the relation *country*. The difficulty therefore does not arise from missing factual knowledge, but from a mismatch between the model's prior interpretation of the relation and the semantic boundary adopted by the KG.

Humans face similar mismatches when reasoning in unfamiliar domains: prior knowledge may initially mislead us, yet a few mistakes often suffice to reshape how we understand the underlying concepts and relations. Rather than memorizing each correction in isolation, we abstract recurring patterns, retain exceptional experiences, and reuse them in later decisions (Posner & Keele, 1968; Metcalfe, 2017). This suggests a different path for adapting LLMs to KGs: building an explicit and revisable cognition state that evolves through interaction with the target graph.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

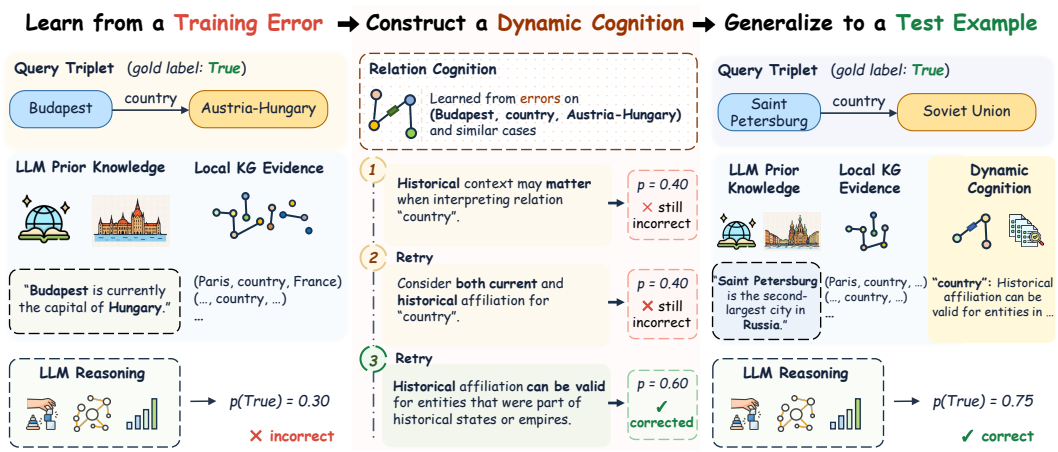


Figure 1: An illustrative example of how GAKU learns dynamic cognition from a reasoning error and generalizes it to a new triplet. **Left:** For the training triplet (Budapest, *country*, Austria-Hungary), the LLM initially relies on its prior knowledge and incorrectly rejects the historical affiliation. **Middle:** GAKU progressively revises the cognition of the relation *country*, eventually abstracting the reusable insight that historical affiliation may also constitute a valid relation instance. **Right:** The learned cognition is retrieved for the unseen triplet (Saint Petersburg, *country*, Soviet Union), helping the LLM reconcile its prior knowledge with the KG-specific semantics and reach the correct prediction.

Motivated by this observation, we propose GAKU (Graph-Adaptive Knowledge Understanding), a multi-agent framework that equips LLMs with dynamic cognition over KGs. Instead of adapting the model parameters, GAKU maintains an explicit cognition state that evolves from its reasoning experience, as illustrated in the middle of Figure 1. Through a cognitive loop of evidence inspection, reflection, abstraction, and validation, the model gradually reconciles its prior knowledge with KG-specific semantics and reuses the resulting cognition in subsequent reasoning.

We summarize our contributions as follows:

- We introduce **dynamic cognition** for KG-grounded reasoning, an explicit and revisable state that combines generalizable relation-level rules with provenance-aware entity experiences, enabling LLMs to adapt to KG-specific semantics without parameter fine-tuning.
- We propose GAKU, a multi-agent framework that builds dynamic cognition through an error-driven cognitive loop. By inspecting local graph evidence, generating and revising relation- and entity-level cognition, and validating each update, GAKU turns reasoning errors into reusable knowledge for subsequent decisions.
- We conduct extensive experiments on KG fact verification and completion across multiple datasets. With only a few training examples for cognition construction, GAKU consistently outperforms state-of-the-art methods, demonstrating the effectiveness of explicit cognitive adaptation to a target KG.

2 Related Work

We categorize related works into two groups:

LLMs for KG Reasoning Recent advances in LLMs have spurred growing interest in applying them to KG tasks (Pan et al., 2023; Yang et al., 2023; Yao et al., 2023; Guan et al., 2024; Guo et al., 2024; Pan et al., 2024; Zhang et al., 2024; Guo et al., 2025; Lin et al., 2025). A straightforward paradigm is to verbalize structural triplets and neighboring graph context as textual input (Yao et al., 2023). However, such graph-to-text conversion may obscure relational and structural information that is important for KG reasoning (Zhang et al., 2024). Recent studies further reveal that correct predictions do not necessarily indicate a robust grasp of KG knowledge. ProbeKGC (Liu et al., 2025), for example, constructs KG completion questions with controlled distractor difficulty and shows that LLM predictions can vary substantially as candidate difficulty changes.

To better preserve graph structures, another line of work explicitly injects structural representations into LLMs (Guo et al., 2024; Zhang et al., 2024; Lin et al., 2025). KoPA (Zhang et al., 2024) obtains KG embeddings with RotatE (Sun et al., 2019), while MKGL (Guo et al., 2024) employs PNA (Corso et al., 2020) as a structural encoder. SSQR (Lin et al., 2025) maps KG information into special code-token embeddings through quantized representations, and SIT-KGED (Li et al., 2026) further incorporates topological information for KG error detection. These approaches mainly provide static KG context or align structural representations with LLMs through parameter adaptation. In contrast, GAKU maintains an explicit and revisable cognition state that progressively adapts to KG-specific semantics through reasoning experience, without updating model parameters.

Multi-Agent Systems for KGs Multi-agent LLM systems have recently attracted increasing attention (Li et al., 2023; Wu et al., 2023; Chen et al., 2024; Hong et al., 2024; Qian et al., 2024), whereas their use in KG-centric tasks remains comparatively underexplored. Existing efforts mainly organize specialized agents around KG construction, interaction, and knowledge-intensive question answering. KARMA (Lu et al., 2025) coordinates agents for entity discovery, relation extraction, schema alignment, and conflict resolution to automate KG enrichment. AGENTiGraph (Zhao et al., 2025) develops a multi-agent KG framework for interactive, domain-specific LLM chatbots. For KG question answering, Debate over Mixed-knowledge (Liu et al., 2026) coordinates KG- and retrieval-based agents to integrate structured and external knowledge under incomplete KGs, while AgentRouter (Zhang et al., 2026) constructs a heterogeneous graph over queries, contextual entities, and agents to guide collaborative routing.

Overall, these methods primarily use agents to construct, access, or coordinate reasoning over KGs, while relatively little attention has been paid to how agents can continuously adapt an LLM to KG-specific semantics. In contrast, GAKU organizes reasoning, cognition generation, and validation agents into an error-driven loop that transforms reasoning failures into explicit, validated, and reusable cognition without parameter updates.

3 METHODOLOGY

In this section, we present GAKU and describe how it builds and exploits dynamic cognition for KG reasoning. We first formulate the KG reasoning task and introduce the representation of dynamic cognition. We then describe the error-driven cognitive loop through which GAKU acquires and revises cognition from training experience. Finally, we explain how the learned cognition is retrieved and incorporated into reasoning over unseen triplets.

3.1 PRELIMINARIES

Knowledge Graph A knowledge graph is denoted as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ and $\mathcal{R}$ denote the sets of entities and relations, respectively, and $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of factual triplets. Each triplet is represented as (h, r, t), where $h, t \in \mathcal{E}$ are the head and tail entities and $r \in \mathcal{R}$ is their relation.

KG Fact Verification KG fact verification, also known as triplet classification, aims to determine whether a query triplet $q = (h, r, t)$ is valid under a target KG $\mathcal{G}$. Formally, the model estimates

$$p(y = 1 \mid q, \mathcal{G}), \tag{1}$$

where $y \in \{0, 1\}$ denotes the validity. We consider an open-world setting in which the accessible graph context is partial and deliberately excludes the query triplet itself. Therefore, failure to retrieve q, or absence of relevant evidence, should not be interpreted as evidence that q is false.

KG Completion Following (Liu et al., 2025), we model KG completion as multiple-choice question answering for LLMs, where the model selects the correct tail or head entity from four candidates given $(h, r, ?)$ or $(?, r, t)$.

3.2 DYNAMIC COGNITION REPRESENTATION

The key idea of GAKU is to maintain an explicit cognition state that captures how a target KG interprets its entities and relations. Rather than representing cognition as a collection of isolated

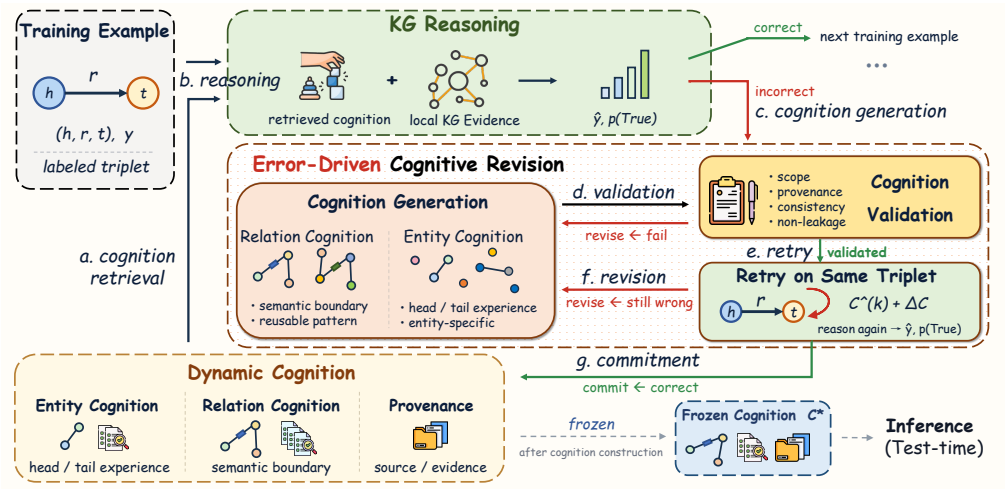


Figure 2: Architecture of GAKU. **(a)** Relevant cognition is retrieved from the dynamic cognition state. **(b)** The reasoning agent combines the retrieved cognition with local KG evidence to estimate $p(\mathrm{True})$. **(c)** The cognition generator analyzes the failed reasoning and proposes cognition updates. **(d)** The validation agent examines the proposed cognition for scope, provenance, consistency, and non-leakage. **(e)** Validated cognition is supplied back to the reasoning agent to retry the same triplet. **(f)** If the retry remains incorrect, the failed attempt is fed back for further cognition revision. **(g)** Once the error is corrected, the validated cognition is committed to the persistent cognition state; after cognition construction, the resulting state is frozen as $\mathcal{C}^*$ and reused at test time.

rules, we distinguish between *relation cognition*, which captures reusable semantic boundaries and reasoning patterns, and *entity cognition*, which preserves entity-specific experiences that may not generalize into universal rules.

Formally, the cognition state is defined as

$$\mathcal{C} = \mathcal{C}_R \cup \mathcal{C}_E, \tag{2}$$

where $\mathcal{C}_R$ and $\mathcal{C}_E$ denote relation- and entity-level cognition, respectively. Their cognition items are represented as

$$c_r = (r, m, \mathcal{P}_r), \tag{3}$$

$$c_{e,r} = (e, r, m, \mathcal{P}_{e,r}), \tag{4}$$

where m is a natural-language description of the learned semantic boundary or experience, and $\mathcal{P} = i_1, \ldots, i_k$ denotes the provenance set containing the training examples from which the cognition item is induced.

This separation allows GAKU to abstract recurring errors into relation-level knowledge while retaining exceptional or ambiguous cases as entity-level experience. Each cognition item is admitted into $\mathcal{C}$ only after validation, making the resulting state explicit, revisable, and provenance-aware.

Relation and Entity Cognition Relation cognition captures patterns that can be generalized across triplets sharing the same relation, such as semantic boundaries, cardinality assumptions, and recurring reasoning cues. Entity cognition, in contrast, preserves experiences that are specific to individual entities. This distinction is important because a reasoning error does not necessarily reveal a universal regularity: some errors expose a systematic misunderstanding of a relation, while others arise from exceptional or ambiguous entities. By retaining both forms of cognition, GAKU can generalize recurring experience without overgeneralizing isolated observations.

3.3 Error-Driven Cognitive Loop

As illustrated in Figure 2, GAKU learns dynamic cognition through an error-driven process. Given a labeled training triplet, the reasoning agent first makes an initial prediction using the current cognition

state and accessible KG evidence. If the prediction is correct, no update is made. Otherwise, the error triggers cognitive revision: a cognition generation agent analyzes the failed reasoning, inspects relevant graph evidence when necessary, and proposes updates to the cognition state. These updates are examined by an independent validation agent before being reused by the reasoning agent to reconsider the same triplet. The process continues until the error is corrected or a predefined revision budget is reached.

Reasoning and Error Detection For each training triplet $q = (h, r, t)$, the reasoning agent makes an initial prediction conditioned on the current cognition state $\mathcal{C}$ and locally retrieved KG evidence. Rather than exposing the entire graph, GAKU allows the agent to selectively inspect a bounded set of entity neighborhoods, relation-specific examples, and short paths relevant to the query. Importantly, these retrieval operations never reveal the query triplet itself; retrieved evidence is therefore treated as contextual support rather than a membership test.

The reasoning agent estimates a correctness probability p_q, from which the prediction is obtained as

$$\hat{y}_q = \mathbb{I}\left[p_q > \tau\right], \tag{5}$$

where τ denotes the decision threshold. During training, $\hat{y}_q$ is compared with the gold label y_q. A correct prediction leaves the cognition state unchanged, whereas an incorrect prediction triggers the subsequent cognitive revision process.

Cognition Generation When an initial prediction is incorrect, GAKU invokes a cognition generation agent to identify the source of the mismatch and propose the smallest useful update to the current cognition state. Conditioned on the failed reasoning trace, the gold label, and optionally retrieved KG evidence, the agent determines whether the error reflects a general relation-level pattern or an entity-specific exception. Relation cognition is preferred when the error reveals reusable semantics, such as relation boundaries, exclusivity, or value multiplicity, whereas entity cognition is introduced only when the observation is specific to the head or tail entity. Each generated item is associated with its corresponding relation or entity and retains the provenance of the training triplet from which it is induced. Rather than encoding the gold answer directly, the generated cognition captures what should be remembered to avoid the same reasoning mistake in future triplets.

Let $\mathcal{H}_q^{(k)}$ denote the interaction history at the k-th revision, including the failed reasoning trace and retrieved KG evidence. The cognition generation process can be abstracted as

$$\Delta\mathcal{C}_q^{(k)} = \mathsf{Gen}\left(q, y_q, \mathcal{H}_q^{(k)}, \mathcal{C}\right), \tag{6}$$

where $\Delta\mathcal{C}_q^{(k)}$ denotes the set of candidate cognition items proposed for the current reasoning error.

Cognition Validation and Revision Candidate cognition is subsequently examined by an independent validation agent before being incorporated into the dynamic cognition state. The validator checks whether each update has a valid scope and provenance, avoids direct label leakage, and does not overgeneralize a local observation or unsupported KG evidence. Approved cognition is then supplied back to the reasoning agent to reconsider the same training triplet. If the revised reasoning reaches the correct prediction, the cognition is promoted into the persistent cognition state and becomes available for future reasoning. Otherwise, the failed retry is fed back to the cognition generation agent, which revises the proposed cognition and repeats the generate–validate–retry cycle until the error is corrected or the revision budget is exhausted.

Let $\mathsf{Val}(\cdot)$ denote the validation process. At revision k, the validated subset is

$$\mathcal{A}_q^{(k)} = \left\{ c \in \Delta\mathcal{C}_q^{(k)} \mid \mathsf{Val}(c; q, y_q) = 1 \right\}, \tag{7}$$

$$\mathcal{C} \leftarrow \begin{cases} \mathcal{C} \cup \mathcal{A}_q^{(k)}, & \hat{y}_q^{(k)} = y_q, \\ \mathcal{C}, & \text{otherwise}, \end{cases} \tag{8}$$

where $\hat{y}_q^{(k)}$ is the prediction obtained by retrying the reasoning agent with the validated cognition. Thus, a cognition update is retained only when it passes validation and successfully corrects the original reasoning error.

3.4 Cognition Retrieval and KG Reasoning

Cognition Retrieval Given an unseen triplet $q = (h, r, t)$, GAKU retrieves a small set of relevant cognition from the learned state $\mathcal{C}$. Relation cognition is retrieved by exact matching on r, since its semantics are explicitly tied to a particular relation. For entity cognition, exact matches to h or t receive the highest priority, while the remaining candidates are retrieved by semantic similarity under the same relation. This relation constraint prevents superficially similar entity experiences from being transferred across unrelated relations.

Formally, the relation-level cognition for q is retrieved as

$$\mathcal{C}_q^R = \{c \in \mathcal{C}_R \mid r_c = r\}, \tag{9}$$

where r_c denotes the relation associated with cognition item c. For entity cognition, we retrieve items that either exactly match the query entities or are semantically relevant to the query, while restricting candidates to the query relation:

$$\mathcal{C}_q^E = \{c \in \mathcal{C}_E \mid r_c = r,\ s(e_c, q) \geq \delta\}, \tag{10}$$

where e_c denotes the entity associated with c, $s(\cdot, \cdot)$ is the retrieval similarity score, and δ is a similarity threshold. Exact entity matches are assigned a score of 1.0, while non-exact candidates are scored by semantic similarity to the query context.

The final cognition context is obtained by ranking the union of both sources:

$$\mathcal{C}_q = \mathrm{TopK}\left(\mathcal{C}_q^R \cup \mathcal{C}_q^E\right). \tag{11}$$

KG-Grounded Reasoning The retrieved cognition is combined with local KG evidence to guide reasoning over the query triplet. Rather than exposing the entire graph, GAKU allows the reasoning agent to actively query a bounded read-only toolbox, including entity neighborhoods, relation-specific examples, and short paths between the head and tail entities. The agent integrates these observations with the retrieved cognition and its own prior knowledge to estimate the correctness probability p_q. Since both graph retrieval and cognition retrieval provide partial evidence, neither is treated as decisive in isolation. In particular, the absence of a matching fact is never interpreted as evidence of falsehood. The final prediction is then obtained using the decision rule in Equation 5.

3.5 Data-Efficient Learning and Inference

Unlike parameter fine-tuning, GAKU does not require the entire training set to adapt an LLM to a target KG. Instead, we construct dynamic cognition from a small subset $\mathcal{D}_{\mathrm{cog}} \subset \mathcal{D}$. In our main experiments, only $|\mathcal{D}_{\mathrm{cog}}| = 100$ labeled triplets are used for cognition construction, which are sufficient to achieve state-of-the-art performance across multiple benchmarks. These examples are processed sequentially, yielding an evolving cognition state

$$\mathcal{C}^{(0)} \to \mathcal{C}^{(1)} \to \cdots \to \mathcal{C}^{(|\mathcal{D}_{\mathrm{cog}}|)}, \tag{12}$$

where correctly reasoned triplets leave the state unchanged, while reasoning errors may introduce new cognition through the generate–validate–retry process. The resulting state is denoted by $\mathcal{C}^*$.

At inference time, $\mathcal{C}^*$ is frozen. For each unseen triplet, GAKU retrieves relevant cognition together with local KG evidence and uses them only as contextual guidance for reasoning. No test triplet or prediction is written back into the cognition state.

4 Experiments

We conduct extensive experiments to evaluate the effectiveness and behavior of GAKU. We first assess its performance on KG fact verification and challenging KG completion, and then analyze the contribution, transferability, and evolution of dynamic cognition through ablation and further studies.

4.1 Settings

Datasets and Metrics For KG fact verification, we evaluate GAKU on CoDeX-S and FB15K-237N (Zhang et al., 2024), using only 100 labeled training triplets to construct dynamic cognition

Table 1: The main results on KG verification. "-" denotes the unavailable entries. For Acc/F1, boldface highlights the results of GAKU, while underlining denotes the strongest competing baseline. † denotes results reproduced using the same Qwen3.8-27B as the default GAKU configuration.

	Method	CoDeX-S				FB15K-237N			
		Acc	**P**	**R**	**F1**	**Acc**	**P**	**R**	**F1**
Conventional	TransE (Bordes et al., 2013)	72.07	71.91	72.42	72.17	69.71	70.80	67.11	68.91
	DistMult (Yang et al., 2015)	66.79	69.67	59.46	64.16	58.66	58.98	56.84	57.90
	ComplEx (Trouillon et al., 2016)	67.64	67.84	67.06	67.45	65.70	66.46	63.38	64.88
	RotatE (Sun et al., 2019)	75.68	75.66	75.71	75.69	68.46	69.24	66.41	67.80
Fine-tuning	KG-LLaMA (Yao et al., 2023)	79.43	78.67	80.74	79.69	74.81	67.37	96.23	79.25
	Vanilla IT (Zhang et al., 2024)	81.18	77.01	88.89	82.52	73.50	65.87	97.53	78.63
	Structure-aware IT (Zhang et al., 2024)	81.27	77.14	88.40	82.58	76.42	69.56	93.95	79.94
	KoPA (Zhang et al., 2024)	82.74	77.91	91.41	<u>84.11</u>	77.65	70.81	94.09	80.81
	SSQR (Lin et al., 2025)	-	-	-	-	<u>79.80</u>	75.90	87.20	<u>81.10</u>
	SIT-KGED (Li et al., 2026)	<u>84.57</u>	86.87	81.46	84.08	77.84	80.39	73.64	76.87
Training-free	Zero-shot (Zhang et al., 2024)	54.68	69.13	16.94	27.21	60.15	86.62	24.01	37.59
	ICL (Zhang et al., 2024)	49.86	49.86	50.59	50.17	54.54	53.67	66.35	59.34
	RAG† (Liu et al., 2025)	84.38	93.20	74.46	82.78	74.41	85.07	57.21	68.42
	CoT+RAG† (Liu et al., 2025)	81.92	81.38	83.15	82.26	78.66	79.27	75.78	77.49
	GAKU (Qwen3.8-27B)	**85.45**	82.50	89.99	**86.08**	**83.97**	95.24	71.52	**81.69**
	GAKU (DeepSeek4.1-Flash)	**91.03**	90.98	91.08	**91.03**	**86.63**	86.02	86.45	**86.23**

and the original test splits for evaluation. We report Accuracy (Acc), Precision (P), Recall (R), and F1, with Acc and F1 as the primary metrics. For KG completion, we use ProbeKGC (Liu et al., 2025) and report Accuracy on its *Hard* split, whose highly competitive distractors provide a more discriminative evaluation than the near-saturated Easy and Medium settings. Dataset statistics and preprocessing details are provided in Appendix B.

Baselines For KG fact verification, we compare against conventional KG methods, including TransE, DistMult, ComplEx, and RotatE (Bordes et al., 2013; Yang et al., 2015; Trouillon et al., 2016; Sun et al., 2019); fine-tuning methods, including KG-LLaMA (Yao et al., 2023), Vanilla IT, Structure-aware IT, and KoPA (Zhang et al., 2024), SSQR (Lin et al., 2025), and SIT-KGED (Li et al., 2026); and training-free methods, including zero-shot prompting, ICL (Dong et al., 2022), RAG, and CoT+RAG (Liu et al., 2025).

Implementation Details Unless otherwise specified, we use Qwen3.8-27B (Yang et al., 2025) as the backbone in GAKU. During cognition construction and inference, the agents access a bounded read-only KG toolbox containing entity neighborhoods, relation-specific examples, and short paths, while the query triplet itself is excluded from retrieval. We use $\tau = 0.5$ for fact verification. Further implementation details, including retrieval and revision budgets, are provided in Appendix C.

4.2 KG Verification Results

Table 1 reports the main results on KG fact verification. Overall, GAKU achieves state-of-the-art performance on both CoDeX-S and FB15K-237N, consistently outperforming conventional KG embedding methods, fine-tuning approaches, and training-free LLM baselines. Replacing the backbone with DeepSeek4.1-Flash further boosts GAKU to 91.03 Accuracy/F1 on CoDeX-S and 86.63 Accuracy with 86.23 F1 on FB15K-237N, surpassing the strongest previous results by 6.46/6.92 and 6.83/5.13 points in Accuracy/F1, respectively. Notably, these gains require only 100 labeled triplets to construct dynamic cognition and involve no parameter updates. The improvements also remain consistent across different backbone LLMs and over our reproduced RAG-based baselines under the same backbone, indicating that the benefit is not simply due to increased model capacity.

Most existing LLM-based KG reasoning methods rely on fine-tuning to encode task-specific KG information into model parameters. In contrast, GAKU adapts through an explicit cognition state that accumulates validated relation- and entity-level experience from reasoning errors. These results demonstrate that such parameter-free cognitive adaptation provides an effective and data-efficient alternative for grounding LLM reasoning in a target KG.

Table 2: Ablation studies on CoDeX-S and FB15K-237N. The best Acc/F1 results are **boldfaced**.

Method	CoDeX-S				FB15K-237N			
	Acc	P	R	F1	Acc	P	R	F1
GAKU	**85.45**	82.50	89.99	**86.08**	**83.97**	95.24	71.52	**81.69**
w/o Validator	83.01	84.66	80.98	82.78	74.47	73.02	75.03	74.01
w/o Retry	82.47	84.48	79.89	82.12	74.53	73.05	75.16	74.09
w/o Dynamic Cognition	79.45	83.85	73.37	78.26	73.50	79.83	60.60	68.90

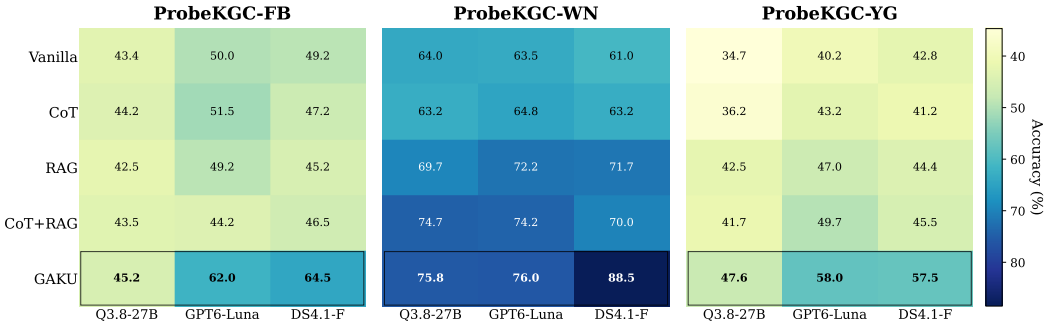


Figure 3: Accuracy on the **Hard** split of ProbeKGC. Q3.8-27B and DS4.1-F denote Qwen3.8-27B and DeepSeek4.1-Flash, respectively.

4.3 KG Completion Results

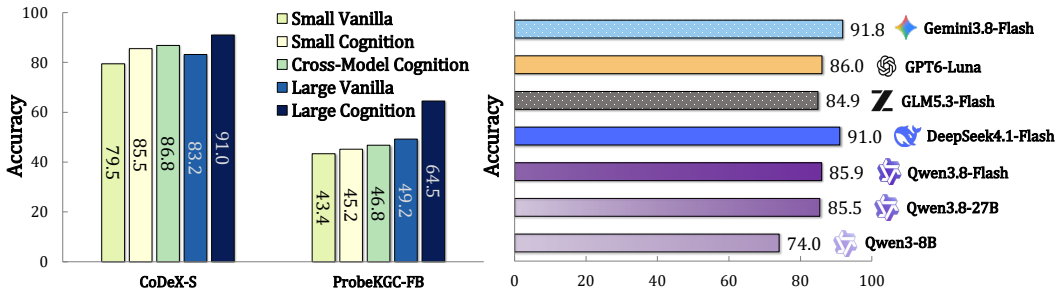
To examine whether dynamic cognition generalizes beyond fact verification, we further evaluate GAKU on the Hard split of ProbeKGC. As shown in Figure 3, GAKU achieves the best accuracy across all nine dataset–backbone combinations, consistently outperforming Vanilla, CoT, RAG, and CoT+RAG. The gains are particularly pronounced with DeepSeek4.1-Flash, where GAKU surpasses the strongest baseline by 15.30, 16.80, and 12.07 points on ProbeKGC-FB, ProbeKGC-WN, and ProbeKGC-YG, respectively. Similar improvements are observed with the other backbones, demonstrating that the benefit of dynamic cognition is not tied to a particular LLM.

In contrast, the effectiveness of CoT and RAG varies substantially across datasets and backbones: additional reasoning or retrieved evidence can improve performance in some settings while degrading it in others, and their combination does not consistently yield additive gains. GAKU instead adapts the reasoning process through reusable cognition distilled from previous errors, providing more consistent improvements under highly competitive distractors. These results suggest that explicit KG-specific cognition complements the parametric knowledge of different LLMs more reliably than simply adding longer reasoning traces or additional retrieved context.

4.4 Ablation Study

We further ablate the key components of GAKU to examine their individual contributions. Specifically, *w/o Validator* removes the independent validation stage and directly reuses generated cognition; *w/o Retry* disables the retry-and-revision mechanism after cognition generation; and *w/o Dynamic Cognition* removes the persistent cognition state entirely, leaving the reasoning agent to rely on its prior knowledge and local KG evidence.

As shown in Table 2, removing any component consistently degrades performance on both datasets. The largest drop occurs without dynamic cognition, reducing Accuracy/F1 from $85.45/86.08$ to $79.45/78.26$ on CoDeX-S and from $83.97/81.69$ to $73.50/68.90$ on FB15K-237N, confirming that persistent cognition is the primary source of adaptation. Removing either validation or retry also causes substantial degradation, especially on FB15K-237N, where Accuracy drops by more than 9 points. These results indicate that effective cognition requires not only generating reusable experience, but also validating its reliability and iteratively refining it through feedback.



a. cross-model cognition transfer b. different backbone LLMs (CoDeX-S)

Figure 4: Backbone analysis of GAKU. **a.** Comparison of five reasoning–cognition configurations using Qwen3.8-27B as the smaller model and DeepSeek4.1-Flash as the stronger model, including self-cognition and cross-model cognition transfer. **b.** Performance of GAKU across seven backbone LLMs on CoDeX-S, where each backbone is used for both reasoning and cognition construction.

4.5 Backbone Analysis

Cross-Model Cognition Transfer Since dynamic cognition is maintained explicitly outside the LLM parameters, the models used for reasoning and cognition construction need not be identical. We therefore investigate both cross-model cognition transfer and backbone sensitivity in Figure 4. Figure 4a compares five settings using Qwen3.8-27B as the smaller model and DeepSeek4.1-Flash as the stronger model: *Small Vanilla* uses the smaller model for reasoning without cognition; *Small Cognition* uses it for both reasoning and cognition construction; *Cross-Model Cognition* retains the smaller reasoning model but employs the stronger model for cognition generation and validation; *Large Vanilla* directly reasons with the stronger model; and *Large Cognition* uses the stronger model for both components.

Figure 4a reveals a consistent trend across the five configurations: dynamic cognition improves both smaller and stronger reasoners, while stronger cognition construction provides further gains even when the reasoning backbone remains unchanged. In particular, Cross-Model Cognition improves Small Cognition from 85.5 to 86.8 on CoDeX-S and from 45.2 to 46.8 on ProbeKGC-FB. This improvement requires the stronger model only during cognition construction over a small number of training examples. Test-time reasoning is still performed entirely by the smaller model. Cross-model cognition therefore offers a practical way to exploit stronger external models without increasing test-time model capacity, while retaining the efficiency of a smaller model during deployment.

Different Backbone LLMs Figure 4b further evaluates GAKU with seven backbone LLMs, using the same backbone for reasoning and cognition construction. The accuracy remains strong across diverse model families, with most backbones achieving above 80.0 accuracy on CoDeX-S and Gemini3.8-Flash reaching 91.8. Qwen3-8B is a notable exception at 74.0, suggesting that smaller models may have more difficulty supporting the multi-agent reasoning, cognition generation, and validation processes required by GAKU.

Overall, these results demonstrate that GAKU generalizes well across backbone LLMs, while stronger agent capabilities can further improve dynamic cognition and downstream reasoning.

5 Conclusion

In this work, we propose GAKU, a multi-agent framework that adapts LLMs to KG-specific semantics through explicit and revisable dynamic cognition. Instead of updating model parameters, GAKU transforms reasoning errors into reusable relation- and entity-level cognition through generation, validation, and revision. Experiments on KG fact verification and completion show that GAKU achieves strong performance with only 100 labeled examples and no fine-tuning, while remaining effective across different backbone LLMs and cross-model cognition transfer settings. These results demonstrate that explicit cognitive adaptation provides a practical and data-efficient alternative to parameter-centric adaptation for KG-grounded reasoning.

AI Use Statement

In this work, we used generative AI tools to assist with language editing and polishing of the manuscript and appendix, including improvements to grammar, clarity, and presentation. All AI-assisted content was carefully reviewed, revised, and verified by the authors. The authors take full responsibility for the final content of the paper, including all claims, results, and conclusions.

Ethics Statement

This work is conducted entirely on publicly available KG datasets and does not involve human subjects, private personal data, or newly collected sensitive information. We do not identify any specific ethical concerns arising from the datasets, methodology, or experimental setting used in this study. All experiments are conducted for research purposes, and the authors take responsibility for the proper use and reporting of the results.

Reproducibility Statement

To support reproducibility, we provide the source code and processed datasets through an anonymous repository at `https://anonymous.4open.science/r/gaku-release-87E3/`. Detailed experimental settings, hyper-parameters, and implementation choices are documented in the Appendix and are also included in the released code. These materials provide the necessary information and resources for reproducing the main experimental results reported in this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pp. 2787–2795, 2013.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *ICLR*, 2024.

Peizhuang Cong, Qizhi Chen, Haochen Zhao, and Tong Yang. BATON: Enhancing batch-wise inference efficiency for large language models via dynamic re-batching. In *WWW*, pp. 2309–2318, 2025. doi: 10.1145/3696410.3714950.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *NeurIPS*, 33:13260–13271, 2020.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Jiayi Geng and Graham Neubig. Effective strategies for asynchronous software engineering agents. *CoRR*, abs/2603.21489, 2026. doi: 10.48550/ARXIV.2603.21489.

Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *AAAI*, pp. 18126–18134, 2024.

Lingbing Guo, Zhongpu Bo, Zhuo Chen, Yichi Zhang, Jiaoyan Chen, Yarong Lan, Mengshu Sun, Zhiqiang Zhang, Yangyifei Luo, Qian Li, Qiang Zhang, Wen Zhang, and Huajun Chen. MKGL: mastery of a three-word language. In *NeurIPS*, 2024.

Lingbing Guo, Yichi Zhang, Zhongpu Bo, Zhuo Chen, Mengshu Sun, Zhiqiang Zhang, Wen Zhang, and Huajun Chen. K-ON: stacking knowledge on the head layer of large language model. In *AAAI*, pp. 11745–11753. AAAI Press, 2025.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *ICLR*, 2024.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *NeurIPS*, 2023.

Ting Li, Xingyi Mao, Yipeng Yu, and Liang Yao. SIT-KGED: simply inject topology into LLM for knowledge graph error detection. In *WWW*, pp. 8497–8500. ACM, 2026.

Qika Lin, Tianzhe Zhao, Kai He, Zhen Peng, Fangzhi Xu, Ling Huang, Jingying Ma, and Mengling Feng. Self-supervised quantized representation for seamlessly integrating knowledge graphs with large language models. In *ACL (1)*, pp. 13587–13602. Association for Computational Linguistics, 2025.

Jilong Liu, Pengyang Shao, Wei Qin, Fei Liu, Yonghui Yang, and Richang Hong. Debate over mixed-knowledge: A robust multi-agent reasoning framework for incomplete knowledge graph question answering. In *AAAI*, pp. 15333–15341, 2026. doi: 10.1609/AAAI.V40I18.38559.

Yang Liu, Zequn Sun, Zhoutian Shao, Yuanning Cui, and Wei Hu. Are llms really knowledgeable for knowledge graph completion? In *ISWC (2)*, volume 16141 of *Lecture Notes in Computer Science*, pp. 77–95. Springer, 2025.

Yuxing Lu, Wei Wu, Xukai Zhao, Rui Peng, and Jinzhuo Wang. KARMA: Leveraging multi-agent llms for automated knowledge graph enrichment. In *NeurIPS*, 2025.

Janet Metcalfe. Learning from errors. *Annual Review of Psychology*, 68:465–489, 2017. doi: 10.1146/annurev-psych-010416-044022.

Jeff Z Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, et al. Large language models and knowledge graphs: Opportunities and challenges. *arXiv preprint arXiv:2308.06374*, 2023.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

Michael I. Posner and Steven W. Keele. On the genesis of abstract ideas. *Journal of Experimental Psychology*, 77(3):353–363, 1968. doi: 10.1037/h0025953.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. In *ACL*, pp. 15174–15186, 2024. doi: 10.18653/V1/2024.ACL-LONG.810.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*, 2019. URL `https://openreview.net/forum?id=HkgEQnRqYQ`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, pp. 2071–2080, 2016.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. AutoGen: Enabling next-gen llm applications via multi-agent conversation framework. *CoRR*, abs/2308.08155, 2023. doi: 10.48550/ARXIV.2308.08155.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*, 2015. URL http://arxiv.org/abs/1412.6575.

Linyao Yang, Hongyang Chen, Zhao Li, Xiao Ding, and Xindong Wu. Chatgpt is not enough: Enhancing large language models with knowledge graphs for fact-aware language modeling. *arXiv preprint arXiv:2306.11489*, 2023.

Liang Yao, Jiazhen Peng, Chengsheng Mao, and Yuan Luo. Exploring large language models for knowledge graph completion. *arXiv preprint arXiv:2308.13916*, 2023.

Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. Making large language models perform better in knowledge graph completion. In *ACM Multimedia*, pp. 233–242. ACM, 2024.

Zheyuan Zhang, Kaiwen Shi, Zhengqing Yuan, Zehong Wang, Tianyi Ma, Keerthiram Murugesan, Vincent Galassi, Chuxu Zhang, and Yanfang Ye. AgentRouter: A knowledge-graph-guided llm router for collaborative multi-agent question answering. In *ACL*, pp. 788–809, 2026. doi: 10.18653/V1/2026.ACL-LONG.33.

Xinjie Zhao, Moritz Blum, Fan Gao, Yingjian Chen, Boming Yang, Luis Marquez-Carpintero, Monica Pina-Navarro, Yanran Fu, So Morikawa, Yusuke Iwasawa, Yutaka Matsuo, Chanjun Park, and Irene Li. AGENTiGraph: A multi-agent knowledge graph framework for interactive, domain-specific llm chatbots. In *CIKM*, pp. 6757–6761, 2025.

Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. *arXiv preprint arXiv:2305.13168*, 2023.

Table 3: Statistics of the KG fact verification and completion datasets. Train KG denotes the positive-only graph accessible to the read-only retrieval tools. Relations and unique entities are computed over the corresponding Train KG. Test denotes the number of evaluation instances, and ProbeKGC statistics correspond to the *Hard* setting.

Dataset	Train Cognition Examples	Train KG	Relations	Entities	Test
CoDeX-S	100	32,888	42	2,034	3,656
FB15K-237N	100	87,282	93	12,701	16,452
ProbeKGC-FB	100	272,115	237	14,541	4,000
ProbeKGC-WN	100	86,835	11	40,943	4,000
ProbeKGC-YG	100	1,079,040	37	123,182	4,000

A Limitations

A current limitation of GAKU is that cognition construction is inherently sequential: each training example is processed against the cognition state accumulated from previous examples, which limits native parallelism. In our main experiments, this overhead remains modest because only 100 labeled examples are used for cognition construction. Nevertheless, scaling GAKU to substantially larger or continuously arriving data streams may require more efficient update mechanisms. A promising direction is to develop batch-wise or asynchronous cognition construction, inspired by recent advances in dynamic LLM batching and asynchronous multi-agent execution (Cong et al., 2025; Geng & Neubig, 2026), where multiple reasoning errors can be processed in parallel and subsequently consolidated into a consistent cognition state for scalable online adaptation.

B Datasets

Table 3 summarizes the datasets used in our experiments. Across all benchmarks, we use only 100 labeled cognition examples to construct dynamic cognition, while evaluation is performed on the complete corresponding test set.

KG Fact Verification We evaluate KG fact verification on CoDeX-S and FB15K-237N. For both datasets, the read-only retrieval graph consists of all positive triplets from the original training split and remains fixed across all experimental conditions. The resulting Train KG contains 32,888 triplets, 42 relations, and 2,034 unique entities for CoDeX-S, and 87,282 triplets, 93 relations, and 12,701 unique entities for FB15K-237N. The complete CoDeX-S test split contains 1,828 positive and 1,828 negative triplets, while FB15K-237N contains 8,226 triplets of each label. Test examples are used exclusively for evaluation and never update the cognition state.

ProbeKGC ProbeKGC formulates KG completion as four-choice entity prediction with *Easy*, *Medium*, and *Hard* distractors. We focus on the *Hard* setting, which introduces the strongest semantic competition among candidate entities. Each benchmark contains 4,000 Hard questions, evenly split between head and tail prediction. ProbeKGC-FB is constructed from a Train KG with 272,115 triplets, 237 relations, and 14,541 unique entities; ProbeKGC-WN contains 86,835 triplets, 11 relations, and 40,943 unique entities; and ProbeKGC-YG contains 1,079,040 triplets, 37 relations, and 123,182 unique entities.

Sampling Strategy For cognition construction, we sample 100 labeled examples from each dataset using a fixed random seed of 42. For CoDeX-S, we sample across relation–label strata while prioritizing examples that introduce previously unseen entities. After ensuring broad relation coverage, the remaining budget is allocated according to the square root of relation frequency. The resulting cognition examples cover all 42 relations and 196 unique entities. For FB15K-237N, we select one positive and one negative example for each of the 50 most frequent training relations, again prioritizing entity diversity, resulting in 197 unique entities. In both datasets, no entity appears more than twice among the selected cognition examples.

For KG completion, we follow the same relation-aware sampling principle. When the number of relations is small, the sampling budget first ensures coverage of every relation and then allocates the

remaining examples according to the square root of relation frequency. When the relation space is larger, we prioritize the most frequent relations and distribute the budget evenly among them. Within each relation, examples involving less frequently selected entities are preferred to encourage entity diversity. The resulting 100 cognition examples are balanced between head and tail prediction, with 50 questions of each type.

For each sampled cognition-construction question in ProbeKGC, the correct entity is paired with three negative candidates drawn from entities observed in the same relation role. Candidates that form known valid facts are removed. The remaining candidates are ranked by the cosine similarity between their relation-participation patterns and that of the correct entity, and the three most similar candidates are retained. This construction produces structurally plausible alternatives that are difficult to distinguish from the correct answer.

C Implementation Details

Backbone and Inference Unless otherwise specified, we use `Qwen3.8-27B` as the shared backbone for the reasoning, cognition generation, and validation agents. The model is served with vLLM through an OpenAI-compatible endpoint in BF16 precision with an 8,192-token context window. Decoding is greedy with temperature zero and a maximum of 384 output tokens. For evaluation, we use 10 concurrent workers across two NVIDIA A800 GPUs. In the backbone analysis, the backbone models are replaced according to the corresponding experimental setting, while the remaining implementation settings are kept unchanged unless otherwise specified.

Agent and Tool Budgets Each training error may trigger at most three generate–validate–retry revisions. The reasoning and cognition generation agents are each allowed at most three tool calls per revision. Each KG retrieval returns at most six results, and retrieved graph paths contain no more than three hops. If a reasoning output is malformed, we allow up to two additional finalization attempts. For fact verification, the reasoning agent predicts $p = \texttt{probability_correct} \in [0, 1]$, and the prediction is True if and only if $p > 0.5$.

The read-only KG toolbox provides entity neighborhoods, relation-specific examples, and short directed paths over the positive training graph. The query triplet itself is removed from all KG-tool retrieval results. Consistent with the open-world setting, an empty retrieval result is treated only as missing evidence and never as evidence of falsehood.

Cognition Retrieval Relation cognition is retrieved by exact matching on the query relation. For entity cognition, exact matches to the query head or tail receive a score of 1.0 and the highest retrieval priority. The remaining entity cognition candidates are restricted to the same relation and scored by semantic similarity to the query context. Semantic similarity is computed on CPU using a local `all-MiniLM-L6-v2` SentenceTransformer with a threshold of 0.35. Candidates below the threshold are discarded, after which relation- and entity-level cognition are ranked jointly and at most six items are returned for each query. After cognition construction, the resulting cognition state is frozen throughout evaluation and is never updated with test examples.

For controlled comparisons with our reproduced training-free baselines, we keep the backbone, data, Train KG, decoding parameters, decision threshold, and retrieval budgets fixed whenever applicable. Method-specific prompting follows the corresponding baseline design, while GAKU additionally retrieves validated cognition as contextual guidance for reasoning.

D Analysis on Hyperparameter Sensitivity

Table 4 examines the sensitivity of GAKU to three key hyperparameters on CoDeX-S. *Tool Calls* controls the maximum number of KG-tool invocations available within each reasoning or revision attempt, determining how extensively an agent can inspect entity neighborhoods, relation examples, and graph paths. *Top-k* controls the maximum number of retrieved cognition items supplied to the reasoning agent, and therefore directly affects the amount of previously learned KG-specific experience exposed for each query. *Max Revisions* specifies the maximum number of generate–validate–retry cycles allowed after an incorrect prediction, controlling how many times the proposed cognition can

Table 4: Hyperparameter sensitivity of GAKU on CoDeX-S. Higher values indicate better performance. Color intensity is normalized within each row, with darker shades indicating better values.

Metric	Tool Calls			Top-k			Max Revisions		
	1	3	5	3	6	9	1	3	5
Accuracy	89.38	91.03	90.21	88.01	91.03	88.84	91.03	91.03	89.38
Precision	89.77	90.98	91.73	87.78	90.98	90.09	91.88	90.98	90.21
Recall	88.91	91.08	88.37	88.37	91.08	87.28	90.00	91.08	88.37
F1 Score	89.33	91.03	90.00	88.07	91.03	88.65	90.92	91.03	89.27

Table 5: Detailed backbone performance of GAKU on CoDeX-S. The same backbone is used for both reasoning and cognition construction. Shaded cells denote the top four results within each metric, with darker shades indicating higher rankings.

Backbone	Accuracy	Precision	Recall	F1 Score
Qwen3-8B	73.99	80.60	63.18	70.84
Qwen3.8-27B	85.45	82.50	89.99	86.08
Qwen3.8-Flash	85.93	82.16	92.30	86.93
DeepSeek4.1-Flash	91.03	90.98	91.08	91.03
GLM5.3-Flash	84.93	97.08	72.28	82.87
GPT6-Luna	86.03	95.24	76.09	84.59
Gemini3.8-Flash	91.78	94.25	89.13	91.62

be refined before the current training example is abandoned. We vary one hyperparameter at a time while keeping the others fixed at their default values of 3, 6, and 3, respectively.

Overall, the default configuration provides the most balanced performance. Increasing the tool-call budget from 1 to 3 improves Accuracy/F1 from $89.38/89.33$ to $91.03/91.03$, whereas increasing it further to 5 yields no additional gain and slightly reduces Recall and F1, suggesting that excessive evidence exploration may introduce distracting or redundant information. A similar pattern appears for cognition retrieval: $k = 6$ performs best, while both $k = 3$ and $k = 9$ degrade performance, reflecting a trade-off between insufficient cognition and overly broad, potentially noisy context. Increasing the revision budget from 1 to 3 produces a slightly more balanced Precision–Recall profile, whereas 5 revisions noticeably reduce overall performance. Notably, the tool-call budget is reset after each retry in our implementation. Consequently, the effective number of tool invocations can grow approximately with the number of retries times the per-attempt tool-call budget. The revision budget therefore affects not only how often cognition can be refined, but also how much additional KG evidence may be explored, which may partly explain why allowing more revisions does not monotonically improve performance.

E Detailed Backbone Results

Table 5 provides the complete results underlying the backbone comparison in Figure 4b, including Precision, Recall, and F1 in addition to Accuracy. Consistent with the main-text analysis, GAKU remains effective across diverse LLM families when the same backbone is used for both reasoning and cognition construction. Except for Qwen3-8B, all evaluated backbones achieve approximately 85% or higher Accuracy, with Gemini3.8-Flash and DeepSeek4.1-Flash reaching 91.78% and 91.03%, respectively. In contrast, Qwen3-8B achieves only 73.99% Accuracy, suggesting that smaller backbones may have greater difficulty supporting the reasoning, cognition generation, and validation processes required by GAKU.

The detailed metrics further reveal distinct Precision–Recall characteristics across backbones. Qwen3.8-Flash favors Recall, reaching 92.30%, whereas GLM5.3-Flash and GPT6-Luna achieve substantially higher Precision but lower Recall. DeepSeek4.1-Flash and Gemini3.8-Flash provide

Table 6: Detailed accuracy results on the **Hard** split of ProbeKGC. Q3.8-27B and DS4.1-F denote Qwen3.8-27B and DeepSeek4.1-Flash, respectively. The best and second-best results in each setting are **boldfaced** and underlined.

Method	ProbeKGC-FB			ProbeKGC-WN			ProbeKGC-YG		
	Q3.8-27B	GPT6-Luna	DS4.1-F	Q3.8-27B	GPT6-Luna	DS4.1-F	Q3.8-27B	GPT6-Luna	DS4.1-F
Vanilla	43.4	50.0	49.2	64.0	63.5	61.0	34.7	40.2	42.8
CoT	44.2	51.5	47.2	63.2	64.8	63.2	36.2	43.2	41.2
RAG	42.5	49.2	45.2	69.7	72.2	71.7	42.5	47.0	44.4
CoT+RAG	43.5	44.2	46.5	74.7	74.2	70.0	41.7	49.7	45.5
GAKU	**45.2**	**62.0**	**64.5**	**75.8**	**76.0**	**88.5**	**47.6**	**58.0**	**57.5**

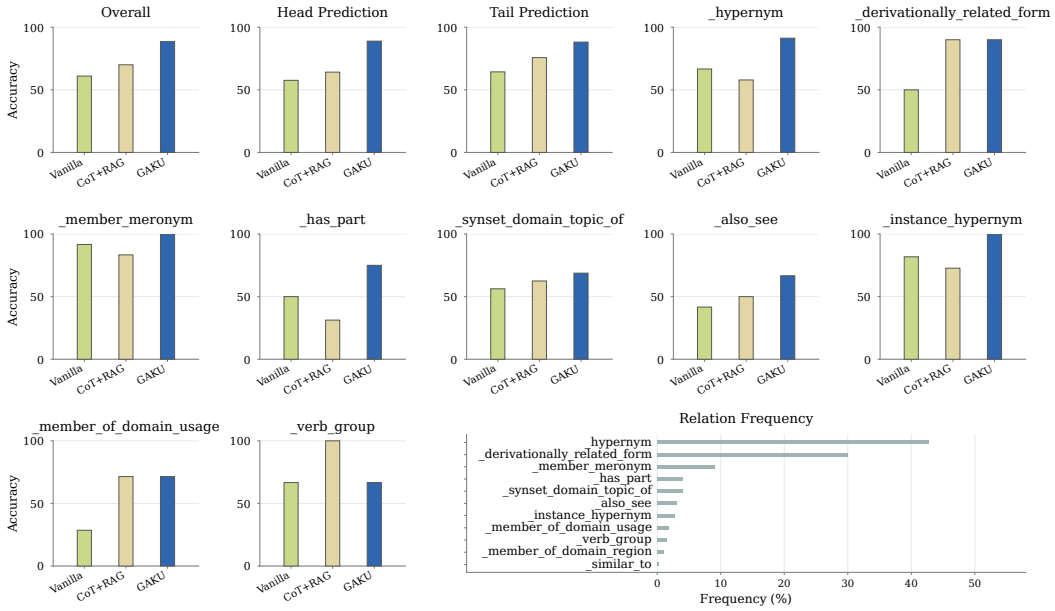


Figure 5: Fine-grained KG completion results on the **Hard** split of ProbeKGC-WN under the DeepSeek4.1-Flash backbone. The figure reports overall accuracy, head-prediction accuracy, tail-prediction accuracy, and relation-wise accuracy for Vanilla, CoT+RAG, and GAKU. The bottom-right panel shows the relation-frequency distribution of ProbeKGC-WN. The relations `_member_of_domain_region` and `_similar_to` are omitted from the relation-wise accuracy panels because of their very small sample sizes.

more balanced performance across the two metrics, resulting in the strongest overall F1 scores. These results complement the Accuracy-focused analysis in the main text by showing that GAKU generalizes across model families, while the underlying backbone can still influence the error profile and how effectively the learned cognition is exploited.

F Detailed KG Completion Results

We provide the complete KG completion results in Table 6, complementing the compact visualization in the main text. Evaluation is conducted on the *Hard* split of ProbeKGC, where semantically competitive distractors make entity prediction substantially more challenging. Across all three datasets and three backbone LLMs, GAKU achieves the best accuracy in every setting. The improvements are particularly pronounced with stronger backbones, while remaining consistent across ProbeKGC-FB, ProbeKGC-WN, and ProbeKGC-YG, demonstrating that dynamic cognition generalizes across both datasets and model families.

Figure 5 further provides a fine-grained analysis on ProbeKGC-WN with DeepSeek4.1-Flash. GAKU improves both head and tail prediction and shows strong performance across most major relations,

indicating that the gains are not concentrated on a single query direction or relation. The relation-frequency distribution is also included for reference.

G PROMPTS

We provide the prompts used by GAKU in the main KG fact verification experiments on CoDeX-S and FB15K-237N. Unless otherwise specified, the reasoning, cognition generation, and validation agents use the default backbone and inference settings described in Appendix C. Each agent prompt is issued as a single user message with temperature zero. Runtime content is represented by angle-bracketed placeholders. Provenance is attached deterministically by the framework from the current labeled training example and is therefore omitted from the generated JSON schema. For controlled comparisons, our reproduced training-free baselines follow their corresponding method-specific prompting protocols under matched inference and retrieval settings whenever applicable.

Reasoning Agent For each triplet, the reasoning agent combines the query with retrieved cognition and accumulated tool interactions to estimate its correctness probability. The same template is used for the initial prediction, cognition-guided retry, and test-time inference.

Reasoning Agent Prompt

```
Classify a knowledge-graph triplet under an OPEN-WORLD setting.
Retrieved facts are partial: missing or empty retrieval is not
evidence of false, and a relation need not be one-to-one. Tool
results are indirect evidence and never include the query triplet.
Dynamic cognition items are weak hints; judge their scope and relevance.

TRIPLET          = <TRIPLET>
COGNITION        = <COGNITION>
HISTORY          = <HISTORY>
TOOLS_REMAINING  = <TOOLS_REMAINING>
TOOLS            = <TOOLS>

Tool JSON:
{"action":"tool","tool":"listed tool name","arguments":{}}

Final JSON:
{"action":"final","probability_correct":0.0,"reason":"..."}

Return exactly one object. The prediction is True iff
probability_correct > 0.5.
```

The reasoning agent may inspect additional KG evidence through `get_neighborhood`, `get_relation_examples`, and `get_paths`. Tools are invoked sequentially, and the query triplet is removed from every KG-tool retrieval result. Retrieved cognition is supplied separately through the `COGNITION` field according to the cognition retrieval procedure described in the main text.

Cognition Generation Agent An incorrect prediction triggers the cognition generation agent. Given the labeled training example, the current failed reasoning trace, and the accumulated interaction history, the agent diagnoses the mismatch and proposes the smallest useful update to the cognition state. Relation cognition captures reusable semantic boundaries or reasoning patterns, whereas entity cognition is reserved for experience specific to the head or tail entity.

Cognition Generation Agent Prompt

```
The gold label is authoritative and must be assumed correct. Diagnose
why the current reasoning disagrees with the gold label, and generate
cognition that can correct this mismatch without encoding the label
itself. Never dispute, negate, or reinterpret the gold label.
```

```
Generate the smallest useful set of reusable cognition items under an
OPEN-WORLD setting. Empty retrieval is not evidence of false, and
relation values may be non-exclusive. Prefer relation cognition for
cardinality, exclusivity, or value-multiplicity errors; add entity
cognition only for genuinely entity-specific knowledge. A cognition
item is a weak hint, not a universal fact. Do not state the label in
its text.

SAMPLE           = <LABELED_TRAINING_SAMPLE>
REVISION         = <CURRENT_REVISION>/<MAX_REVISIONS>
HISTORY          = <HISTORY>
TOOLS_REMAINING  = <TOOLS_REMAINING>
TOOLS            = <TOOLS>

Tool JSON:
{"action":"tool","tool":"listed tool name","arguments":{}}

Final JSON:
{"action":"final","cognition":[
  {"scope":"relation or entity","key":"...","text":"..."}
]}

Return exactly one object. The cognition list may contain relation
cognition, entity cognition, or both.
```

Validation Agent Each proposed cognition item first undergoes deterministic checks for schema validity, scope, key consistency, training provenance, and direct label leakage. Candidates that pass these checks are then independently examined by the validation agent before being supplied to the reasoning agent for retry.

Validation Agent Prompt

```
Validate one cognition item assuming the gold label is authoritative.
The cognition item must address the reasoning error and support the
gold decision without stating or negating the label. Check its schema,
scope, key, provenance, label leakage, and consistency with the labeled
sample. An entity key must equal the sample head or tail; a relation
key must equal the sample relation. Entity names may appear in cognition
text, and relation cognition may be induced from a single example. Do
not reject an item solely for lacking confidence wording or broader
supporting evidence.

Missing retrieval is not evidence of false.

SAMPLE     = <LABELED_TRAINING_SAMPLE>
COGNITION  = <CANDIDATE_COGNITION>

Return exactly:
{"approved":true or false,"reason":"..."}
```

Finalization Once an agent exhausts its tool budget, it must finalize without further retrieval. An invalid reasoning response triggers at most two additional finalization attempts. If both attempts fail, we record a format error and assign a correctness probability of 0.5, which maps to False under the strict decision rule $p > 0.5$.