

From Imitation to Reasoning: Sample-Efficient LLM Agents for Knowledge Graph Completion

Chenjun Zheng¹, Zhigang Wang¹, Wenjia Zhang¹, Zhe Wang^{1*}, Xiaolin Yang^{1*}

¹Institute of Basic Medical Sciences & School of Basic Medicine,
Chinese Academy of Medical Sciences & Peking Union Medical College
s2024005050@student.pumc.edu.cn, wangzhe@ibms.pumc.edu.cn,
yangxl@pumc.edu.cn

Abstract

While large language models (LLMs) have been introduced to knowledge graph completion (KGC) to improve multi-hop reasoning, most existing approaches still rely on embedding models for candidate pruning, which limits their structured reasoning capability. In contrast, fully path-based reasoning better reflects the nature of graph inference but suffers from extremely low sample efficiency. We propose AdaG-Reason, a path-based LLM-KGC framework. We formulate reasoning data synthesis as a non-stationary multi-armed bandit problem and introduce Active Reasoning Distillation to select informative trajectories by jointly considering distribution alignment, relation difficulty, and uncertainty. We further extend this bandit-based sampling principle to the reinforcement learning phase, adaptively allocating rollout queries by per-relation difficulty. For training, we propose Implicit Reasoning Distillation optimized via Step-wise Group Relative Policy Optimization (GRPO), enabling direct and stable learning of pure graph navigation policies without value function estimation. Experiments on YAGO3-10 and NELL-995 show that AdaG-Reason achieves competitive MRR and state-of-the-art Hits@1 on YAGO3-10, demonstrating the effectiveness of active data selection and pure policy learning for path-based LLM-KGC.

1 Introduction

Knowledge Graph Completion (KGC) aims to infer missing facts in a knowledge graph and is a fundamental problem in representation learning and symbolic reasoning. While embedding-based methods dominate current benchmarks, they perform inference through static similarity matching in latent spaces, lacking explicit reasoning processes and struggling to model multi-hop dependencies or provide interpretable decisions (Bordes et al., 2013; Sun et al., 2019; Dettmers et al., 2018).

To address these limitations, path-based reasoning methods reformulate KGC as a sequential

decision-making problem on graphs (Xiong et al., 2018), where an agent navigates from a source entity by iteratively selecting relations and entities. Representative work such as MINERVA formulates multi-hop reasoning as a POMDP and enables reinforcement learning-based graph navigation (Das et al., 2018). However, traditional path-based methods remain constrained by exponentially growing search spaces and limited generalization: policies learned over symbolic or embedding-level actions often overfit to local structural patterns and fail to generalize across semantically equivalent but topologically distinct paths (Ma et al., 2025; Lin et al., 2018).

Recent advances in large language models (LLMs) offer a natural extension to path-based reasoning (Wei et al., 2023). By operating in a semantic space, LLMs can abstract relational meanings and functional roles of paths, enabling stronger semantic generalization than conventional policy networks (Ahn et al., 2022). In parallel, direct fine-tuning approaches such as KG-FIT (Jiang et al., 2024) and SAT (Liu et al., 2025) encode KG triples as textual sequences, leveraging the generative capacity of LLMs without explicit graph navigation. LLM-based graph reasoning frameworks leverage step-wise interaction (Yao et al., 2023) to perform multi-hop inference, demonstrating improved flexibility in complex decision making. However, this semantic expressiveness comes at a substantial cost. In graph reasoning, decision spaces grow combinatorially, and multiple reasoning paths may lead to the same correct answer (Das et al., 2018; Lei et al., 2020; Lin et al., 2018). As a result, naively training LLM-based agents to imitate full trajectories is inefficient, redundant, and often misaligned with the final reasoning objective.

These observations motivate a key question: how can LLMs acquire path-based reasoning without imitating full trajectories? We argue that effective training should focus on high-information decision

signals rather than exhaustive path imitation (Russo and Roy, 2014; Auer et al., 2002). Our contributions are:

- **Active Reasoning Distillation (ARD):** a non-stationary bandit formulation for trajectory synthesis that jointly models distribution alignment, relation difficulty, and uncertainty, adaptively allocating a fixed budget toward learnable relations.
- **Implicit Reasoning Distillation (IRD):** we identify a semantic-topological mismatch in teacher-generated reasoning. Among 5,000 analyzed trajectories, only 2.2% contain explicit topological violations, whereas 89.6% are classified as implicit semantic misalignment: their graph actions remain valid, yet the resulting reasoning trajectory leads to an incorrect prediction. IRD addresses this mismatch through action-centric masking, training the policy on grounded actions without imitating the teacher’s full CoT.
- **Step-wise GRPO with MAB-guided roll-out:** a critic-free RL stage that unifies bandit-based sampling across both training phases, achieving state-of-the-art Hits@1 on YAGO3-10 (0.572) while training on trajectories covering only 6.6% of entities.

2 Related Work

Knowledge Graph Completion. Early research was dominated by **embedding-based methods**, which map entities and relations into vector spaces. Representative models like TransE (Bordes et al., 2013), ComplEx (Trouillon et al., 2016), and RotatE (Sun et al., 2019), along with structure-aware extensions (e.g., Schlichtkrull et al. (2017) and Vashishth et al. (2020)), achieve strong performance on benchmarks like YAGO3-10 (Suchanek et al.). To address the lack of explicit reasoning, **path-based approaches** formulate KGC as a multi-step decision process (e.g., MultiHop (Lin et al., 2018), M-Walk (Shen et al., 2018), and RuleGuider (Lei et al., 2020)). Notably, MINERVA (Das et al., 2018) models reasoning as a POMDP via reinforcement learning, establishing an end-to-end paradigm. Despite improving interpretability, these methods grapple with exponentially large search spaces, unstable training dynamics (Ma et al., 2025), and limited semantic

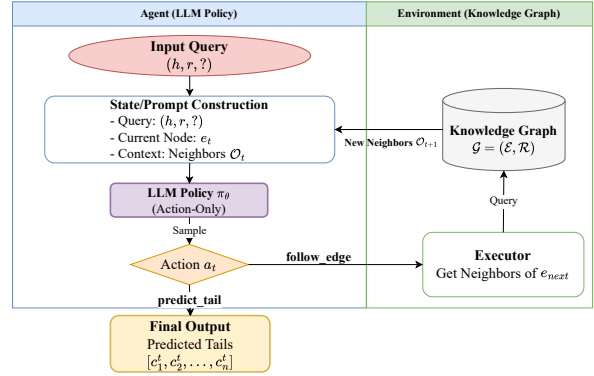


Figure 1: Illustration of the ReAct-style sequential decision process on the knowledge graph.

generalization due to their reliance on symbolic representations (Lin et al., 2018).

PLMs and LLM Agents for KGC. Pretrained Language Models (PLMs) such as KG-BERT (Yao et al., 2019), SaTR (Wang et al., 2021), and SimKGC (Wang et al., 2022) enhance semantic modeling by encoding triples as text. However, they remain fundamentally similarity-based, serving as semantic encoders rather than performing explicit sequential reasoning. Recently, Large Language Models (LLMs) have evolved into agents capable of interleaving reasoning and action (Zhao et al., 2025; Tan et al., 2025). Frameworks like ReAct (Yao et al., 2023), combined with policy optimization algorithms such as PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024), have demonstrated effectiveness in complex reasoning tasks (Shinn et al., 2023). In parallel, direct LLM fine-tuning approaches such as KG-FIT (Jiang et al., 2024) and SAT (Liu et al., 2025) encode KG triples as textual sequences for generative training, achieving competitive results by leveraging the semantic capacity of pretrained LMs without explicit graph traversal. Nevertheless, adapting these agent frameworks to large-scale KGC remains challenging due to rigid structural constraints, vast discrete action spaces, and low sample efficiency.

3 Methodology

3.1 Preliminaries

A knowledge graph $G = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ consists of entities $\mathcal{E}$, relations $\mathcal{R}$, and triples $\mathcal{T} = \{(h, r, t)\}$. KGC answers queries $(h, r, ?)$ by ranking candidate tail entities.

We formulate multi-hop reasoning as a POMDP (Figure 1). Given a query $q = (h, r, ?)$, at step t

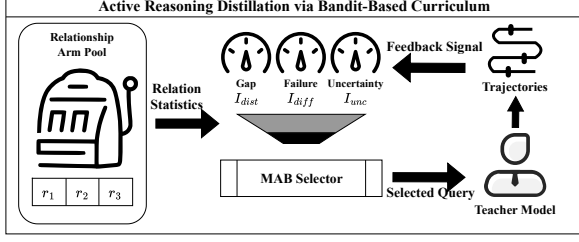


Figure 2: Overview of the **Active Reasoning Distillation** framework. The MAB Selector dynamically prioritizes relations through a hierarchical acquisition function. Teacher-generated trajectories are validated and fed back to update the bandit statistics in a closed loop.

the agent positioned at entity e_t observes a local subgraph O_t and conditions on $x_t = (q, O_t)$. The LLM parameterizes a stochastic policy over the vocabulary $\mathcal{V}$ (full prompt template in Appendix E):

$$P_\theta(w | x_t) = \text{Softmax}(\text{Logits}_\theta(x_t)). \quad (1)$$

The action space has two types. A *path-expansion* action $a_t^f = (r', e')$ follows an outgoing edge $(e_t, r', e') \in \mathcal{G}$. A *termination* action $a_t^p = L_t$ outputs a ranked candidate list $L_t = [c_1, \dots, c_k]$ ($k \leq K_{\max}$) in descending confidence, with probability:

$$P(a_t^p | x_t) \propto \prod_{i=1}^k P_\theta(c_i | x_t, c_{<i}). \quad (2)$$

A trajectory $\tau = (o_0, a_0, \dots, a_T)$ ends with a terminal `predict_tail` action producing L_T . The objective maximizes the probability of ranking the ground-truth entity e^* in L_T :

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [\mathbb{I}(e^* \in L_T)]. \quad (3)$$

Directly optimizing this objective is prohibitive: the graph branches exponentially, and rewards are sparse (binary success at trajectory termination). We address this through two stages—first **bootstrapping** the policy from selectively distilled teacher trajectories (Section 3.2), then **reinforcement fine-tuning** (Section 3.3).

3.2 Distillation-Based Policy Bootstrapping

3.2.1 Active Reasoning Distillation (ARD)

We cast trajectory synthesis under a fixed budget as a non-stationary multi-armed bandit (Figure 2), where the reward distribution shifts as per-relation success rates evolve with accumulated trajectories. Each relation $r \in \mathcal{R}$ is an arm; pulling arm r

randomly selects a head entity h from the training triples of r to form query $(h, r, ?)$, then queries a teacher LLM for a trajectory. The goal is to allocate budget toward relations yielding *learnable* trajectories rather than uniform exploration.

Composite Acquisition Function. The priority score for relation r at iteration t combines three terms:

$$S_t(r) = \mathcal{I}_{\text{dist}}(r) + \lambda_1 \mathcal{I}_{\text{diff}}(r) + \lambda_2 \mathcal{I}_{\text{unc}}(r), \quad (4)$$

where $\lambda_1 > \lambda_2$ prioritizes learnability-driven sampling over pure exploration.

Distribution Alignment. Let $P^*(r)$ denote the empirical relation frequency in the training set, N_{total} the total budget, and $N_t(r)$ the number of valid trajectories collected for r up to iteration t .

$$\mathcal{I}_{\text{dist}}(r) = \max\left(0, \frac{N_{\text{total}}P^*(r) - N_t(r)}{N_{\text{total}}P^*(r) + \epsilon}\right) \quad (5)$$

This term prioritizes relations whose trajectory count trails their target quota.

Difficulty Awareness. Beyond coverage, not all relations are equally learnable. Let $\mu_t(r) = V_t(r)/A_t(r)$ be the empirical success rate for r , where $A_t(r)$ and $V_t(r)$ count total attempts and valid trajectories. A relation is *abandoned* when $A_t(r) > A_{\text{abandon}}$ yet $\mu_t(r) < \mu_{\min}$. We model learnability via a Gaussian window centered at μ_{target} :

$$\mathcal{I}_{\text{diff}}(r) = \begin{cases} 0, & \text{abandoned,} \\ \exp\left(-\frac{(\mu_t(r) - \mu_{\text{target}})^2}{2\sigma^2}\right), & \text{otherwise.} \end{cases} \quad (6)$$

Gated Uncertainty Exploration. We encourage exploration of under-sampled relations while suppressing unlearnable ones by gating the UCB term with $\mu_t(r)^\gamma$:

$$\mathcal{I}_{\text{unc}}(r) = \sqrt{\frac{2 \ln(\sum_{r'} N_t(r') + 1)}{N_t(r) + 1}} \cdot \mu_t(r)^\gamma. \quad (7)$$

Closed-Loop Distillation. ARD proceeds iteratively: S_t is Softmax-transformed into a sampling distribution; a batch of relations is drawn, queries are instantiated and submitted to the teacher; $N_t(r)$ and $\mu_t(r)$ are updated from trajectory validity. The three acquisition terms interact dynamically: early iterations are dominated by distribution alignment (broad coverage), mid-stage by difficulty awareness (concentrating on the learnability window),

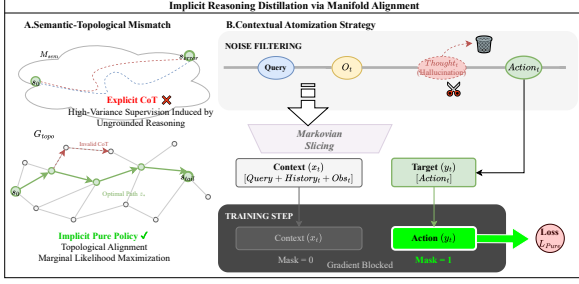


Figure 3: Implicit Reasoning Distillation (IRD). (A) Semantic-Topological Mismatch: semantically plausible teacher reasoning does not always translate into correct graph-constrained decisions. (B) Action-centric masking: gradients are restricted to grounded action tokens, reducing supervision from potentially misaligned CoT.

and late-stage by gated uncertainty (suppressing exhausted or hopeless relations). This phase transition, visualized in Appendix C, yields an adaptive curriculum under a fixed budget. Each synthesized trajectory pairs grounded actions with the teacher’s CoT reasoning. As we show next, the free-form reasoning can be semantically plausible yet misaligned with the graph-constrained decision process.

3.2.2 Implicit Reasoning Distillation (IRD)

To mitigate the semantic-topological mismatch introduced by explicit CoT supervision (quantified in Appendix B; Figure 3), we propose *Implicit Reasoning Distillation (IRD)*. Rather than directly supervising the student with the teacher’s full free-form reasoning trace, IRD treats reasoning as a latent variable and optimizes the policy through grounded action supervision, reducing its dependence on potentially misaligned CoT.

Theoretical Motivation. LLM reasoning can be understood as an interaction between two manifolds: the continuous semantic manifold $\mathcal{M}_{\text{sem}}$ from pretraining, and the discrete topological manifold $\mathcal{G}_{\text{topo}}$ of the knowledge graph. Teacher-generated traces z_{syn} , sampled from $\mathcal{M}_{\text{sem}}$, may diverge from the structured decisions required by $\mathcal{G}_{\text{topo}}$. We distinguish explicit topological violations, where a navigation action does not correspond to a valid KG edge, from grounded but outcome-misaligned trajectories, where all executed edges are valid but the final prediction is incorrect. Across 5,000 teacher trajectories on YAGO3-10, 89.6% fall into the latter category, while only 2.2% contain explicit topological vi-

olations (Appendix B).

This mismatch creates two problems for explicit CoT supervision. First, *spurious correlation*: when the teacher’s free-form reasoning is misaligned with the graph-constrained decision while the associated action remains useful for supervision, directly optimizing the full CoT can encourage the student to model irrelevant or misleading reasoning signals. Second, *high-variance gradients*: the CoT trace spans many tokens whose gradient $\nabla \log P_{\theta}(z_{\text{syn}} | x)$ competes with the action gradient, diluting supervision on the decisions that matter. By masking CoT tokens and backpropagating only through actions, IRD reduces exposure to both sources of noise and concentrates the training signal on the state-to-action mapping.

Action-Centric Optimization. We implement IRD via step-wise behavioral cloning with an action-centric masking strategy. Given a flattened trajectory sequence $X = [x_t; y_t]$, we construct a binary mask M where $M_i = 1$ if $X_i \in \text{Tokens}(y_t)$ and 0 otherwise. The objective is:

$$\mathcal{L}_{\text{BC}}(\theta) = -\mathbb{E}_{(x,y) \sim \mathcal{D}_{\text{step}}} \sum_i M_i \log P_{\theta}(X_i | X_{<i}). \quad (8)$$

Importantly, IRD primarily serves as an action-grounded policy initialization rather than the final optimization stage. On YAGO3-10, the IRD/SFT-only policy achieves 0.324 Hits@1, which is subsequently improved to 0.568 by GRPO. By concentrating supervision on executable graph actions, IRD provides a lower-noise initialization for subsequent policy optimization.

3.3 Reinforcement Fine-Tuning with Bandit-Guided Rollout

3.3.1 Step-wise GRPO

Starting from the IRD-initialized policy π_{BC} , we refine it via Step-wise Group Relative Policy Optimization (GRPO), a critic-free algorithm using group-based relative feedback (Figure 4).

Rank-Aware Reward & Advantage. To mitigate the “hedging” phenomenon where models generate excessive candidates, we define a Rank-Aware Reward that strictly enforces Top-1 precision while penalizing path length:

$$R(\tau) = \frac{\mathbb{I}(e_{\text{gold}} \in L_{\text{pred}})}{\text{Rank}(e_{\text{gold}})} - \lambda_{\text{len}}(|\tau| - 1). \quad (9)$$

For each query, we sample a group of G trajectories. The advantage A_i for trajectory τ_i is computed by

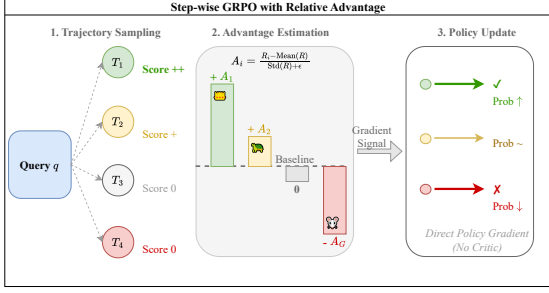


Figure 4: Step-wise GRPO with Relative Advantage. The process iterates through: (1) sampling a group of trajectories; (2) computing relative advantages based on rank-aware rewards; and (3) updating the policy directly to reinforce high-quality reasoning paths without a critic network.

normalizing rewards against the group statistics: $A_i = (R_i - \mu_R)/(\sigma_R + \epsilon)$. To address temporal credit assignment, we modulate this advantage with an exponential decay weight $\omega_{i,t} = \gamma^{T_i-t}$, assigning higher credit to actions closer to the final prediction—critical in graph navigation, where early exploration decisions determine which sub-graph is reachable. The GRPO objective maximizes the weighted likelihood of step-wise actions while penalizing deviation from the reference policy π_{ref} :

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\frac{1}{G} \sum_{i=1}^G \sum_{t=1}^{|\tau_i|} \omega_{i,t} \left[A_i \log \pi_{\theta}(a_{i,t} | x_{i,t}) - \beta D_{\text{KL}}(\pi_{\theta} || \pi_{\text{ref}})_t \right], \quad (10)$$

where $D_{\text{KL}}(\cdot)_t$ is the per-step KL penalty for training stability.

3.3.2 MAB-Guided Rollout Sampling

Each GRPO iteration requires sampling rollout queries. Uniform sampling wastes budget on relations the policy already masters. We extend the bandit principle to the RL phase: per-relation success rates from the preceding evaluation determine sampling weights, assigning higher weight to relations where the policy struggles. However, naively concentrating all rollout budget on the hardest relations introduces a training stability risk: in early RL iterations, when the policy is still fragile, batches dominated by near-impossible queries produce sparse, noisy rewards that can destabilize the policy before it consolidates basic graph navigation patterns.

To mitigate this, we introduce a *Relation Buffer* term that guarantees every relation a minimum sam-

Dataset	Entity	Relation	Train	Valid	Test
YAGO3-10	123,182	37	1,079,040	5,000	5,000
NELL-995	75,492	200	154,213	543	2,818

Table 1: Summary statistics of benchmark datasets.

pling probability, preventing the distribution from collapsing to extreme difficulty. Formally, given the policy’s per-relation success rate $\mu_t^{\text{eval}}(r)$ at RL iteration t , the sampling weight for relation r is

$$w(r) = \underbrace{(1 - \mu_t^{\text{eval}}(r))}_{\text{difficulty}} + \underbrace{\beta}_{\text{buffer}}, \quad (11)$$

where $\beta > 0$ controls the minimum allocation share. The rollout query distribution is then

$$P(r) = \frac{w(r)}{\sum_{r'} w(r')}. \quad (12)$$

When β is small, allocation is dominated by failure rates, concentrating updates on weaknesses; when β is large, the distribution approaches uniformity, ensuring even mastered relations ($\mu_t^{\text{eval}} \rightarrow 1$) retain a non-negligible sampling probability.

Together with ARD, this forms a unified bandit-based data selection strategy spanning both training stages.

4 Experiment

4.1 Experimental Setup

Datasets We evaluate our method on two distinct benchmarks: YAGO3-10 (Suchanek et al.) and NELL-995 (Xiong et al., 2018). YAGO3-10 is a large-scale subset of YAGO3, characterized by a high number of entities and a long-tail relation distribution, serving as a stress test for scalability. NELL-995, derived from the Never-Ending Language Learning system, serves as a standard testbed for multi-hop reasoning. It features complex path dependencies and specific reasoning tasks that rigorously test the agent’s ability to perform logical deduction over multiple steps. Table 1 summarizes the statistics of these datasets. We exclude FB15k-237 and WN18RR; justification is provided in Appendix A.

Evaluation Metrics We assess performance using standard KGC metrics: Mean Reciprocal Rank (MRR) and Hits@ k ($k = 1, 10$). Hits@ k measures the proportion of test triples where the ground-truth entity is ranked within the top- k candidates. We employ the *filtered ranking setting* proposed by

Bordes et al. (2013), excluding other known true triples $(h, r, t') \in \mathcal{G}_{\text{train}} \cup \mathcal{G}_{\text{valid}} \cup \mathcal{G}_{\text{test}}$ ($t' \neq t$) during evaluation.

For LLM-based prediction, entity names returned by `predict_tail` are mapped to KG entity IDs using case-insensitive exact matching, and the ranked valid candidates are used for metric computation. For SC@10, we independently sample 10 reasoning trajectories with temperature 0.7 and aggregate their final predictions via majority voting.

Implementation Details. We employ DeepSeek-V3.2 (DeepSeek-AI, 2025) as the teacher to synthesize ARD-curated trajectories, yielding 4,498 valid trajectories for YAGO3-10 and 12,654 for NELL-995 (detailed synthesis statistics in Appendix D). The student model, Qwen3-8B (Yang et al., 2025), undergoes full-parameter SFT followed by Step-wise GRPO, both on $8 \times A100$ GPUs with DeepSpeed ZeRO-3. Full hyperparameters are provided in Appendix D.

Table 2 details the coverage profile of the ARD-synthesized trajectories relative to the full training graphs.

Table 2: Trajectory coverage statistics. *Unique Query Triples* counts distinct $(h, r, ?)$ queries; *Unique Entities* counts distinct entities appearing in any state or observation; *Unique Edges* counts distinct (h, r, t) traversed. Percentages denote coverage of the full-graph entity set (123,182 for YAGO3-10, 75,492 for NELL-995).

	YAGO3-10	NELL-995
Synthesized trajectories	4,498	12,654
Unique query triples	4,439	5,996
Unique entities	8,113	7,055
(% of full graph)	(6.6%)	(9.3%)
Unique edges traversed	5,483	6,262
Relations covered	34/37	182/200
Full-graph training triples	1,079,040	154,213

On YAGO3-10, the ARD process yields 4,498 valid trajectories covering 4,439 distinct queries across 34 of 37 relations; the three uncovered relations receive negligible allocation due to persistently low teacher success rates—a consequence of $\mathcal{I}_{\text{dist}}$ and $\mathcal{I}_{\text{diff}}$ in Eq. 4 that jointly suppress unproductive exploration. On NELL-995, 12,654 trajectories span 182 of 200 relations; the remaining 18 are extreme long-tail cases where the teacher failed to produce valid reasoning chains despite extended exploration. This coverage profile highlights a key distinction from embedding-based methods: AdaG-Reason trains exclusively on relation-diverse, ARD-

filtered trajectories rather than the full triple set. The resulting trajectories cover only 6.6% and 9.3% of graph entities, respectively, indicating that the student is trained from a sparse subset of the available graph supervision. Here, *sample efficiency* refers to the amount and coverage of retained supervision used for student training, rather than end-to-end computational efficiency; detailed synthesis and training costs are reported in Appendix D.5.

4.2 Main Results

Table 3 presents the performance comparison against a broad range of state-of-the-art baselines on both datasets. We include classical and contemporary embedding-based models, recent neural graph reasoning approaches, and text-based methods.

Precision and the Role of Design Choices.

AdaG-Reason achieves the highest Hits@1 on both datasets (0.572 on YAGO3-10; 0.518 on NELL-995), surpassing KNOWFORMER, and reaches MRR 0.597 on YAGO3-10. This precision follows from the design choices in Section 3. ARD’s bandit curriculum (Section 3.2) ensures the policy is bootstrapped from learnable trajectories rather than uniformly sampled noise. IRD’s action-centric masking (Section 3.2.2) avoids directly supervising the student with potentially misaligned CoT tokens, reducing the impact of semantic-topological mismatch during distillation. GRPO with MAB-guided rollout (Section 3.3) further sharpens the policy by concentrating reinforcement signals on relations where the model is most uncertain. On NELL-995, this precision focus comes at a trade-off in MRR (0.558 vs. KNOWFORMER 0.566), reflecting the rank-aware reward’s structural bias toward Top-1 accuracy. Each component addresses a specific source of optimization noise, and the consistent Top-1 accuracy across two structurally distinct benchmarks suggests that their combined effect favors decisive, high-confidence predictions.

Relational Complexity vs. Entity Scale.

YAGO3-10 (123k entities, 37 relations) achieves higher absolute performance than NELL-995 (75k entities, 200 relations) under the same training paradigm. This counter-intuitive result—the larger entity set yielding better performance—aligns with the bandit formulation of ARD: the difficulty of path-based KGC is not determined by the number of entities but by the diversity of the relation space. YAGO3-10’s 37 hierarchically clear re-

Table 3: Main Results on YAGO3-10 and NELL-995. The best results are highlighted in **bold**, and the second-best are underlined. $\pm$ denotes standard deviation over 3 training seeds. Direct: greedy single-path decoding; SC@10: self-consistency decoding with 10 paths.

Method	YAGO3-10			NELL-995		
	MRR	H@1	H@10	MRR	H@1	H@10
TransE (Bordes et al., 2013)	0.510	0.413	0.681	0.507	0.424	0.648
RotatE (Sun et al., 2019)	0.495	0.402	0.670	0.508	0.448	0.608
DistMult (Yang et al., 2015)	0.566	0.491	0.704	0.510	0.438	0.636
House (Li et al., 2022)	0.571	0.491	0.714	0.519	0.458	0.618
AdaProp (Zhang et al., 2023)	0.573	0.510	0.685	0.554	0.493	0.655
NBFNet (Zhu et al., 2022)	0.563	0.480	0.708	0.525	0.451	0.639
DRUM (Sadeghian et al., 2019)	0.531	0.453	0.676	0.532	0.460	<u>0.662</u>
CompGCN (Vashishth et al., 2020)	0.421	0.392	0.577	0.463	0.383	0.596
SimKGC (Wang et al., 2022)	0.211	0.141	0.351	0.501	0.426	0.653
KNOWFORMER (Liu et al., 2024)	0.615	0.547	0.734	0.566	0.502	0.675
KG-FIT (Jiang et al., 2024)	0.568	0.474	0.718	-	-	-
SAT (Liu et al., 2025)	<u>0.616</u>	0.525	<u>0.726</u>	-	-	-
AdaG-Reason (Direct)	0.597 $\pm$ 0.007	0.568 $\pm$ 0.008	0.640 $\pm$ 0.006	0.539 $\pm$ 0.008	0.518 $\pm$ 0.009	0.565 $\pm$ 0.007
AdaG-Reason (SC@10)	0.617$\pm$0.004	0.572$\pm$0.005	0.678 $\pm$ 0.004	<u>0.558$\pm$0.005</u>	0.518$\pm$0.005	0.613 $\pm$ 0.004

lations present a concentrated learning problem, while NELL-995’s 200 diverse relations demand complex, long-range reasoning chains that stretch the capacity of both teacher and student. This observation suggests that relation diversity and reasoning depth may be more informative indicators of difficulty than entity count alone in agent-based KGC.

Self-Consistency and the Single-Path Trade-off.

Direct decoding trails global embedding methods on Hits@10, reflecting the recall limitation inherent in greedy single-path inference. Self-consistency decoding (SC@10) bridges this gap without retraining: sampling 10 independent paths and aggregating via majority voting raises YAGO3-10 MRR to 0.617, surpassing KNOWFORMER, and lifts Hits@10 to 0.678. That an opt-in, training-free decoding strategy closes much of the gap suggests that a substantial portion of the Hits@10 deficit arises from the high-confidence search behavior encouraged by the rank-aware reward (Section 3.3.1), which prioritizes Top-1 precision over recall. Notably, SC@10 yields marginal H@1 improvement on both datasets (+0.004 on YAGO3-10, +0.000 on NELL-995), indicating that the policy’s top-ranked prediction is already accurate and stable; the gain from self-consistency lies in recovering correct candidates that fall outside the top-1, not in correcting the top-1 itself.

4.3 Ablation Studies

We conduct comprehensive ablation studies on the YAGO3-10 dataset to validate our architectural

choices and training mechanisms.

4.3.1 Impact of Reasoning Modality

We contrast our implicit reasoning paradigm against three baselines: (1) Zero-shot ReAct (using Qwen3-32B-Instruct as a comparable-scale reference; the 8B-Instruct variant was unavailable at the time of experiments); (2) Direct SFT, which treats KGC as internal memorization; and (3) Explicit Chain-of-Thought (CoT) ReAct. All fine-tuned variants use Qwen3-8B and are evaluated after the IRD/SFT stage. Results on YAGO3-10 and NELL-995 are shown in Table 4.

Table 4: Ablation on reasoning modality on YAGO3-10 and NELL-995. Models are evaluated after the IRD/SFT stage under full-parameter training.

Variant	YAGO3-10		NELL-995	
	MRR	H@1	MRR	H@1
Zero-shot ReAct	0.067	0.066	0.026	0.026
Memorization SFT	0.063	0.063	0.120	0.116
CoT ReAct SFT	0.152	0.147	0.216	0.209
AdaG-Reason SFT only	0.324	0.324	0.339	0.339

ReAct vs. Memorization. Zero-shot ReAct and Direct SFT both perform poorly, supporting the premise of our agent design: large-scale KGC is fundamentally a reasoning-and-acting task rather than one that relies solely on internal knowledge recall. Parametric memory alone is insufficient to cover the rich factual relations in large-scale knowledge graphs, supporting the POMDP formulation in Section 3.1 that requires active graph ex-

ploration.

Implicit vs. Explicit Reasoning. Across both datasets, CoT ReAct SFT substantially underperforms IRD, providing further support for the semantic-topological mismatch analysis in Section 3.2.2. Explicit reasoning traces generated by the teacher in semantic space may deviate from effective decision paths in the graph environment; forcing the student to imitate these intermediate reasoning traces can therefore introduce additional optimization noise. By restricting supervision to grounded actions, IRD enables the student to learn a more direct mapping from graph states to structurally valid actions, yielding a stronger initialization for subsequent policy optimization.

4.3.2 Impact of Learning Paradigm

We isolate the contribution of the RL stage by comparing the supervised baseline (IRD/SFT-Only) versus the full pipeline (SFT + GRPO). Results are shown in Table 5. Full-parameter SFT alone yields

Table 5: Ablation on Impact of Learning Paradigm. Full-parameter SFT vs. SFT + GRPO.

Dataset	Stage	MRR	H@1	H@10
YAGO3-10	Full SFT only	0.324	0.324	0.325
YAGO3-10	Full SFT + GRPO	0.597	0.568	0.640
YAGO3-10	+ SC@10	0.617	0.572	0.678
NELL-995	Full SFT only	0.339	0.339	0.340
NELL-995	Full SFT + GRPO	0.539	0.518	0.565
NELL-995	+ SC@10	0.558	0.518	0.613

near-identical H@1 and H@10 on both datasets, indicating that the policy gains little benefit from ranking additional candidates at this stage. GRPO substantially improves MRR and Hits@ k on both datasets, demonstrating the importance of subsequent policy optimization beyond supervised initialization. The rank-aware reward (Section 3.3.1) is designed to reward precise Top-1 predictions while penalizing hedging, encouraging the policy to place greater confidence on high-ranked candidates. SC@10 further lifts YAGO3-10 MRR to 0.617, showing that additional inference-time sampling can further improve candidate ranking without retraining.

4.3.3 Impact of Bandit-Based Adaptive Sampling

We evaluate bandit-based adaptive sampling at both the data synthesis and RL rollout stages on YAGO3-10 (Table 6). At the synthesis stage, we compare

topology-only BFS sampling, uniform teacher distillation, and ARD. At the RL stage, we independently vary the rollout strategy under both uniform and ARD-based SFT initializations.

Table 6: Data synthesis and RL rollout strategies on YAGO3-10.

SFT	Rollout	MRR	H@1
<i>SFT only</i>			
BFS Random	–	0.047	0.043
Uniform	–	0.206	0.204
ARD	–	0.324	0.324
<i>SFT + GRPO</i>			
Uniform	Uniform	0.456	0.423
Uniform	MAB	0.525	0.491
ARD	Uniform	0.526	0.505
ARD	MAB	0.597	0.568
ARD	MAB + SC@10	0.617	0.572

Effect of Adaptive Sampling. At the synthesis stage, BFS Random produces topologically valid but semantically unguided trajectories and performs poorly, highlighting the importance of teacher-guided trajectory generation. Uniform teacher sampling substantially improves over BFS Random, while ARD further improves the SFT initialization by adaptively allocating the synthesis budget toward more learnable relations. Importantly, this advantage persists after subsequent GRPO training.

At the RL stage, MAB-guided sampling consistently outperforms uniform sampling under both SFT initialization strategies, suggesting that relation-difficulty-aware rollout allocation provides additional gains beyond those obtained from adaptive distillation. Overall, adaptive allocation is beneficial at both data synthesis and RL exploration stages, with ARD and MAB-guided rollout achieving the strongest training-time performance. SC@10 further improves the final result at inference time without additional training.

ARD Allocation Behavior. The ARD bandit policy dynamically allocates the synthesis budget across three regimes: easy relations are exploited early and throttled upon convergence, medium-difficulty relations receive the majority of investment, and persistently hard relations are suppressed to prevent low-quality trajectories from diluting the training signal. The complete analysis and visualization are provided in Appendix C.

4.4 Reasoning Depth Analysis

To analyze the actual reasoning depth of the learned policy, we group test trajectories by the number of graph-navigation steps executed before the final prediction and report the trajectory ratio and Hits@1 at each depth.

Table 7: Performance across reasoning depths. Ratio denotes the proportion of test trajectories terminating at each depth.

Steps	YAGO3-10		NELL-995	
	Ratio	H@1	Ratio	H@1
1	5.7%	0.341	4.1%	0.291
2	23.2%	0.769	17.6%	0.688
3	33.9%	0.738	30.5%	0.651
4	23.9%	0.436	27.1%	0.491
5	13.4%	0.188	20.7%	0.228

Across both datasets, most trajectories involve multi-step graph navigation, while only a small fraction terminate after a single step. Performance is strongest at intermediate depths of 2–3 steps and degrades noticeably as trajectories become longer. This pattern indicates that the policy can effectively exploit moderate-depth multi-hop paths, while longer reasoning chains remain more susceptible to path uncertainty and error accumulation.

5 Conclusion

We present AdaG-Reason, a framework where LLM agents acquire graph reasoning through actively curated demonstrations (ARD), action-grounded supervision (IRD), and trial-and-error refinement (Step-wise GRPO). Our analysis identifies a semantic–topological mismatch in teacher supervision: explicit topological violations are rare (2.2%), whereas 89.6% of analyzed trajectories exhibit implicit semantic misalignment. This finding motivates IRD’s action-centric supervision, which avoids direct imitation of the teacher’s full CoT. Across YAGO3-10 and NELL-995, AdaG-Reason consistently improves Top-1 prediction through adaptive data selection and policy optimization. On YAGO3-10, it achieves state-of-the-art Hits@1 (0.572) while training on trajectories covering only 6.6% of graph entities.

Acknowledgments

This work was supported by the Beijing Natural Science Foundation (7254504), the Noncommunicable Chronic Diseases–National Science and

Technology Major Project (2024ZD0532900), the National Natural Science Foundation of China (No. 32670888), and the CAMS Innovation Fund for Medical Sciences (CIFMS, No. 2021-I2M-1-057).

Limitations

ARD relies on the teacher producing at least a minimal number of valid trajectories per relation; relations where the teacher consistently fails receive little or no allocation (3/37 on YAGO3-10 and 18/200 on NELL-995), limiting the framework on queries beyond the teacher’s reasoning competence. Although the retained supervision used for student training is sparse, ARD still incurs substantial teacher-side synthesis cost because trajectory generation requires repeated API calls and filtering.

The current framework is primarily designed and optimized for graph reasoning settings represented by YAGO3-10 and NELL-995, where explicit multi-step navigation and relational composition play an important role. Its effectiveness may not directly transfer to datasets with substantially different graph structures or reasoning characteristics, such as regimes dominated by short-range associations, highly hierarchical relations, or different connectivity patterns. Adapting the sampling strategy, navigation policy, or training objective to such graph regimes remains an important direction for future work.

The single-path formulation favors Top-1 accuracy at the expense of candidate recall; although SC@10 partially mitigates this trade-off at inference time, it introduces additional latency. More generally, AdaG-Reason incurs higher inference cost than embedding-based methods because each query requires sequential model interaction and multiple forward passes.

References

- Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, and 26 others. 2022. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*.
- Peter Auer, Nicol O Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*.

- Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’13, page 27872795, Red Hook, NY, USA. Curran Associates Inc.
- Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. [Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning](#). *arXiv preprint*. ArXiv:1711.05851 [cs].
- DeepSeek-AI. 2025. Deepseek-v3.2: Pushing the frontier of open large language models.
- Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. [Convolutional 2D Knowledge Graph Embeddings](#). *arXiv preprint*. ArXiv:1707.01476 [cs].
- Pengcheng Jiang, Lang Cao, Cao Xiao, Parminder Bhatia, Jimeng Sun, and Jiawei Han. 2024. [Kg-fit: Knowledge graph fine-tuning upon open-world knowledge](#). *Preprint*, arXiv:2405.16412.
- Deren Lei, Gangrong Jiang, Xiaotao Gu, Kexuan Sun, Yuning Mao, and Xiang Ren. 2020. [Learning Collaborative Agents with Rule Guidance for Knowledge Graph Reasoning](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8541–8547, Online. Association for Computational Linguistics.
- Rui Li, Jianan Zhao, Chaozhuo Li, Di He, Yiqi Wang, Yuming Liu, Hao Sun, Senzhang Wang, Weiwei Deng, Yanming Shen, Xing Xie, and Qi Zhang. 2022. [House: Knowledge graph embedding with householder parameterization](#). *Preprint*, arXiv:2202.07919.
- Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. [Multi-Hop Knowledge Graph Reasoning with Reward Shaping](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3243–3253, Brussels, Belgium. Association for Computational Linguistics.
- Junnan Liu, Qianren Mao, Weifeng Jiang, and Jianxin Li. 2024. [Knowformer: Revisiting transformers for knowledge graph reasoning](#). *Preprint*, arXiv:2409.12865.
- Yu Liu, Yanan Cao, Xixun Lin, Yanmin Shang, Shi Wang, and Shirui Pan. 2025. [Enhancing large language model for knowledge graph completion via structure-aware alignment-tuning](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20970–20984, Suzhou, China. Association for Computational Linguistics.
- Ying Ma, Owen Burns, Mingqiu Wang, Gang Li, Nan Du, Laurent El Shafey, Liqiang Wang, Izhak Shafran, and Hagen Soltau. 2025. [Knowledge Graph Reasoning with Self-supervised Reinforcement Learning](#). *IEEE Transactions on Audio, Speech and Language Processing*, 33:1508–1519. ArXiv:2405.13640 [cs].
- Daniel Russo and Benjamin Van Roy. 2014. [Learning to optimize via information directed sampling](#). *CoRR*, abs/1403.5556.
- Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. 2019. [Drum: End-to-end differentiable rule mining on knowledge graphs](#). *Preprint*, arXiv:1911.00055.
- Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. [Modeling Relational Data with Graph Convolutional Networks](#). *arXiv preprint*. ArXiv:1703.06103 [stat].
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal Policy Optimization Algorithms](#). *arXiv preprint*. ArXiv:1707.06347 [cs].
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models](#). *arXiv preprint*. ArXiv:2402.03300 [cs].
- Yelong Shen, Jianshu Chen, Po-Sen Huang, Yuqing Guo, and Jianfeng Gao. 2018. [M-Walk: Learning to Walk over Graphs using Monte Carlo Tree Search](#). *arXiv preprint*. ArXiv:1802.04394 [cs].
- Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. [Reflexion: Language Agents with Verbal Reinforcement Learning](#). *arXiv preprint*. ArXiv:2303.11366 [cs].
- Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia.
- Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. [RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space](#). *arXiv preprint*. ArXiv:1902.10197 [cs].
- Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, and Wenjie Zhang. 2025. [Paths-over-Graph: Knowledge Graph Empowered Large Language Model Reasoning](#). *arXiv preprint*. ArXiv:2410.14211 [cs].
- Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. [Complex embeddings for simple link prediction](#). *CoRR*, abs/1606.06357.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2020. [Composition-based Multi-Relational Graph Convolutional Networks](#). *arXiv preprint*. ArXiv:1911.03082 [cs].

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, and Yi Chang. 2021. [Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion](#). In *Proceedings of the Web Conference 2021*, pages 1737–1748. ArXiv:2004.14781 [cs].

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022. [SimKGC: Simple Contrastive Knowledge Graph Completion with Pre-trained Language Models](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4281–4294, Dublin, Ireland. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. [Chain-of-Thought Prompting Elicits Reasoning in Large Language Models](#). *arXiv preprint*. ArXiv:2201.11903 [cs].

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2018. [DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning](#). *arXiv preprint*. ArXiv:1707.06690 [cs].

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, and 41 others. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. [Embedding entities and relations for learning and inference in knowledge bases](#). *Preprint*, arXiv:1412.6575.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. [KG-BERT: BERT for Knowledge Graph Completion](#). *arXiv preprint*. ArXiv:1909.03193 [cs].

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. [ReAct: Synergizing Reasoning and Acting in Language Models](#). *arXiv preprint*. ArXiv:2210.03629 [cs].

Yongqi Zhang, Zhanke Zhou, Quanming Yao, Xiaowen Chu, and Bo Han. 2023. [Adaprop: Learning adaptive propagation for graph neural network based knowledge graph reasoning](#). *Preprint*, arXiv:2205.15319.

Qi Zhao, Hongyu Yang, Qi Song, Xinwei Yao, and Xiangyang Li. 2025. [KnowPath: Knowledge-enhanced Reasoning via LLM-generated Inference Paths over Knowledge Graphs](#). *arXiv preprint*. ArXiv:2502.12029 [cs].

Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. 2022. [Neural bellman-ford networks: A general graph neural network framework for link prediction](#). *Preprint*, arXiv:2106.06935.

A Applicability Scope and Dataset Selection

In this work, we primarily study explicit logical reasoning over graph structures, where inference requires multi-hop traversal and structured decision-making, rather than implicit semantic similarity matching. Accordingly, we conduct experiments on YAGO3-10 and NELL-995, which are commonly used benchmarks for path-based and rule-like reasoning. In this appendix, we clarify the scope of our evaluation and discuss the considerations behind excluding FB15k-237 and WN18RR.

Reasoning Type: Logical Inference vs. Semantic Correlation. FB15k-237 contains a large number of relations whose predictions can often be resolved through direct semantic associations between entities (e.g., /award/award_nominee $\leftrightarrow$ /award/award_winner). Prior work has shown that embedding-based models (e.g., RotatE, TuckER) are particularly effective in such settings by capturing symmetric or anti-symmetric patterns in a continuous latent space. In contrast, our method focuses on explicit path-based reasoning, where correct predictions depend on composing multiple relational steps (e.g., $A \xrightarrow{\text{bornIn}} B \xrightarrow{\text{locatedIn}} C \Rightarrow A \xrightarrow{\text{nationality}} C$). YAGO3-10 and NELL-995 contain a higher proportion of such multi-hop, rule-like relational patterns, making them more suitable for evaluating sequential decision-making agents. As a result, we do not expect the advantages of explicit reasoning policies to be fully reflected on FB15k-237.

Graph Topology and Relational Diversity. WN18RR exhibits a predominantly hierarchical structure with a low average node degree, where reasoning is largely constrained to hypernym and hyponym relations. While this dataset is well-suited for studying hierarchical representation learning, it provides limited relational diversity for evaluating general-purpose graph navigation agents. By contrast, YAGO3-10 and NELL-995 exhibit heterogeneous and long-tailed relation distributions, requiring models to generalize from frequent relations to sparsely observed ones. This property aligns with our focus on sample-efficient reasoning under sparse supervision, which constitutes a central motivation of our work.

Scope of LLM-based Agents. Our framework is designed for scenarios in which reasoning re-

quires explicit multi-step decision-making under structural constraints. In datasets where answers are frequently reachable within one hop or can be inferred via surface-level semantic similarity, the computational overhead of LLM-based agents may not translate into clear empirical gains. Therefore, we restrict our evaluation to benchmarks where structured reasoning is expected to play a more prominent role. We leave a systematic study of extending AdaG-Reason to other dataset regimes, including dense or hierarchy-dominated graphs, to future work.

B Semantic-Topological Mismatch Quantification

To quantify semantic-topological mismatch in teacher-generated trajectories, we analyze 5,000 reasoning traces produced by DeepSeek-V3.2 on YAGO3-10. The classification is deterministic and requires no manual labeling. It is based on whether each executed `follow_edge` action corresponds to an edge available in the current KG observation and whether the terminal prediction matches the ground-truth tail. Each trajectory is assigned to one of the following categories:

- **Successful** (207, 4.1%): The trajectory produces the correct target entity through graph-based reasoning.
- **Explicit Topological Violation** (109, 2.2%): At least one executed `follow_edge` action attempts to traverse an edge that is not present in the current KG observation.
- **Implicit Semantic Misalignment** (4,480, 89.6%): All executed `follow_edge` actions correspond to valid KG edges, but the trajectory ultimately produces an incorrect tail prediction.
- **Single-Step Guess** (204, 4.1%): The agent invokes `predict_tail` without graph traversal.

The breakdown reveals a marked asymmetry: explicit violations of graph structure are rare (2.2%), whereas the dominant failure mode consists of trajectories that remain grounded in valid graph edges but terminate at an incorrect entity (89.6% of all analyzed trajectories). This asymmetry motivates IRD (Section 3.2.2): rather than requiring the student to imitate the teacher’s full intermediate reasoning text, IRD restricts supervision to grounded,

executable actions, reducing the influence of reasoning signals that are not aligned with the final graph decision.

C ARD Resource Allocation Analysis

Figure 5 illustrates how the MAB policy optimizes the synthesis budget on NELL-995. The joint analysis of temporal evolution (Panel A) and final efficiency (Panel B) reveals a strategic transition from correctness to utility via three mechanisms:

Saturation-Aware Throttling. Easy relations are exploited early but throttled upon convergence (Panel A, Red), producing a counter-intuitive inversion where the highest-performing tasks receive minimal final investment (Panel B, Top-Left) to prevent overfitting on trivial patterns.

Dynamic Stratification. Resources shift to the medium-difficulty learnability window (Panel A, Green); the majority of the budget concentrates on these high-gradient tasks (Panel B, Warm Colors), maximizing their contribution to the student model.

Cost-Effective Pruning. Hard relations undergo active suppression (Panel A, Gray) into sustained low-resource exploration (Panel B, Bottom), preventing high-uncertainty failures from diluting data quality while maintaining coverage of long-tail knowledge.

D Implementation Details

D.1 Hardware and Environment

All training experiments were conducted on $8 \times$ NVIDIA A100 80G GPUs using DeepSpeed ZeRO-3. The software environment includes PyTorch 2.6, CUDA 12.6, and HuggingFace Transformers. For GRPO results, we report the mean and standard deviation over 3 independent runs with different random seeds (42, 123, 456). Due to the non-deterministic nature of the DeepSeek API, teacher-generated trajectories may vary across API calls; we will therefore release all synthesized trajectories alongside our code to facilitate reproducibility.

D.2 Active Reasoning Distillation (ARD)

Hyperparameters. In the composite acquisition function (Eq. 4), we set the coefficients for difficulty and uncertainty as $\lambda_1 = 1$ and $\lambda_2 = 0.3$, giving distribution alignment and difficulty equal emphasis while moderating the uncertainty-driven exploration to prevent unproductive sampling from dominating the acquisition score.

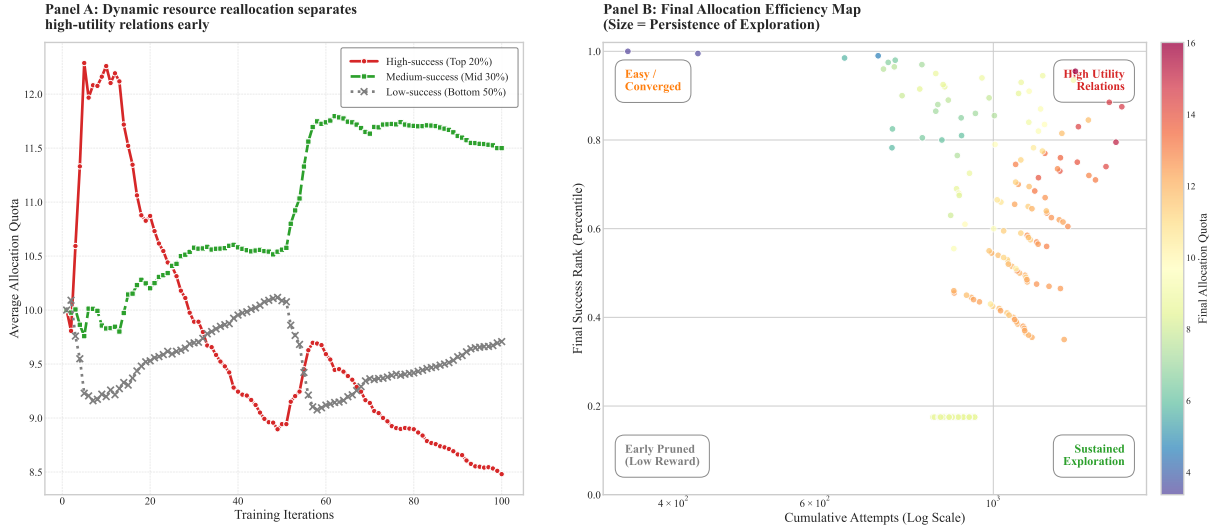


Figure 5: Visualization of Dynamic Resource Allocation in ARD. Panel A traces the temporal evolution of allocation quotas, illustrating the strategic shift from early exploration of simple relations to a focused exploitation of learnable, intermediate-difficulty ones. Panel B maps the final efficiency landscape.

Synthesis Statistics. We employed DeepSeek-V3.2 as the teacher model. The synthesis process details are as follows:

- **YAGO3-10:** We set the target retained-trajectory budget to 5,000. At each ARD iteration, a candidate pool of 1,000 queries was constructed and adaptively allocated across relations. The synthesis process consumed approximately 200,000 teacher API calls before early stopping, yielding 4,498 valid trajectories.
- **NELL-995:** We set the target retained-trajectory budget to 20,000. Each ARD iteration used a candidate pool of 1,000 queries. Due to the increasing concentration of sampling on difficult long-tail relations, the marginal yield of valid trajectories gradually diminished, and the process was terminated early after approximately 800,000 teacher API calls, yielding 12,654 valid trajectories.

D.3 Implicit Reasoning Distillation (IRD)

In the Supervised Fine-Tuning (SFT) phase, we apply **full-parameter fine-tuning** to the student model (Qwen3-8B) using DeepSpeed ZeRO-3. The detailed configuration is:

- **Optimization:** AdamW optimizer with a learning rate of 1×10^{-5} and a linear learning rate scheduler.
- **Regularization:** Weight decay of 0.02.

- **Training Config:** Effective batch size of 32, trained for 3 epochs (YAGO3-10 and NELL-995) or 5 epochs (FB15k-237).

D.4 Step-wise GRPO

For the reinforcement learning stage, we optimize all model parameters (no adapters) under DeepSpeed ZeRO-3. The specific hyperparameters are:

- **Group Size G :** 16 (sampling 16 trajectories per query).
- **KL Coefficient β :** 0.04 (to penalize deviation from the reference policy).
- **Temporal Decay γ :** 0.8 (for step-wise credit assignment).
- **Learning Rate:** Constant 1×10^{-6} .
- **Training Duration:** 10 iterations of the Stage 1–Stage 3 cycle, with 500 queries sampled per iteration.
- **Reward Design:** We use the rank-aware reward and step-wise credit assignment defined in Section 3.3.1.
- **Rollout Sampling:** We use the relation-difficulty-aware sampling distribution defined in Section 3.3.1.

D.5 Cost Analysis

Table 8 summarizes the resource consumption of AdaG-Reason across trajectory synthesis and student training. Teacher API calls denote the cumulative number of model calls made during ARD before early stopping, rather than the target number of retained trajectories or the candidate query pool size.

Table 8: Resource consumption for trajectory synthesis and student training. GPU hours include the full-parameter SFT and GRPO stages.

Resource	YAGO3-10	NELL-995
Teacher API calls	~200K	~800K
Retained trajectories	4,498	12,654
SFT GPU hours	5.0	11.5
GRPO GPU hours	160.0	80.0
Total training GPU hours	165.0	91.5

The synthesis cost reflects the iterative nature of ARD: the target retained-trajectory budgets are 5,000 and 20,000 for YAGO3-10 and NELL-995, respectively, while the actual number of teacher calls is determined by adaptive sampling and early stopping. Thus, the low coverage of the retained trajectories should be interpreted as efficiency in the amount of supervision consumed by the student, rather than low end-to-end synthesis cost.

At inference time, Direct decoding requires approximately 1.6 seconds per query, whereas SC@10 requires approximately 5.0 seconds per query, corresponding to about $3.1\times$ the latency. SC@10 incurs no additional training cost and is used only as an optional inference-time strategy to improve candidate recall.

E Prompt Design

The local graph state x_t is serialized into a natural language prompt to guide the LLM agent. Below is the template used for both the Teacher (during ARD) and the Student (during Inference). Note that for the Student model in the *Implicit Reasoning* setting, the output is strictly constrained to the action space tokens.

```
You are an agent that searches a Knowledge Graph
for a missing tail entity.
Your task: Through multi-step graph lookup,
decide whether to continue exploring (using
follow_edge) based on explored paths and
visible adjacent relationships.
If the current context is sufficient for a final
decision, execute predict_tail. You may
predict any entity from the set of already
explored entities.
```

```
You are given the following information:
Query (head, relation): ({elq_name}, {rq_name})
Current traversal node (et): {et_name}
```

```
You observe the following local subgraph (
context):
{context_text}
```

```
Allowed Actions (choose ONE in one answer):
- follow_edge('relation_name', 'tail_entity_name')
  -> Move one step along a visible edge from
      the current node et to its neighbor.
  -> The relation_name and tail_entity_name
      must be chosen from the current
      observation's neighbors.
- predict_tail(['tail1', 'tail2', ...])
  -> Finally return an ordered list of tail
      entity names, at most 20 items.
```

```
{output_format}
```

Note:

- Only response a specific Action, no other words.
- When using predict_tail, list results most-likely-first that appears in the context and past paths, max 20.
- When using follow_edge, pick a neighbor that appears in the context.