HyAKE: Hybrid Adaptive Knowledge Expert for Knowledge Graph Reasoning

Anonymous ACL submission

Abstract

Knowledge-intensive NLP systems increasingly combine large language models (LLMs), knowledge graphs (KGs), and document retrieval. These sources have highly heterogeneous inference costs: LLM calls are accurate but expensive (120ms, cost 1.0), KG lookups are fast but limited (8ms, cost 0.02), and retrieval lies in between. Existing KG reasoning methods either rely on a single source or fuse multiple sources in a cost-blind manner, which can lead to suboptimal accuracy–cost trade-offs. We propose **HyAKE**, a cost-aware hybrid expert framework for KG reasoning. HyAKE integrates three specialists—a parametric LLM expert, a structural GNN expert over cached subgraphs, and a retrieval expert—with two key components: (i) a **Knowledge Graph Reasoning Planner (KGRP)** that decomposes complex queries into a DAG of sub-questions for dependency-aware, partially parallel execution; and (ii) an **Adaptive Knowledge Fusion Module (AKFM)** that performs query-specific, cost-aware expert routing with learned temperature networks and a CLUB-based decoupling loss to encourage complementary behaviors. Experiments on four benchmarks show that HyAKE improves MRR by 37–59% over strong baselines and by 10–19% over direct Qwen-2.5-7B prompting, while reducing normalized inference cost by 45% and achieving 2.2× lower latency. On unseen entities, HyAKE retains 79% of its transductive performance versus 45–51% for baselines, suggesting that gains are not solely due to memorization.

1 Introduction

Knowledge graphs (KGs) encode structured real-world knowledge and underpin critical NLP applications including question answering [1], recommendation [2], and retrieval [3].

Recent advances in LLMs [4, 5, 6] enable powerful parametric reasoning, yet their direct application to KG tasks faces a critical challenge: *cost heterogeneity*. Modern systems can leverage LLM inference (120ms, cost 1.0), cached KG lookups (8ms, cost 0.02), and document retrieval (35ms, cost 0.1)—but these sources differ by 50× in cost. Existing approaches either use single sources with limited capabilities [7] or apply cost-blind uniform fusion [8], yielding suboptimal accuracy-cost trade-offs.

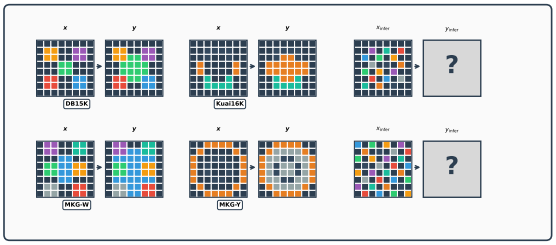

Figure 1: **Key Capabilities of HyAKE**, a multimodal hybrid expert system for KG reasoning. The system integrates parametric knowledge (LLM), structural knowledge (GNN), and retrieval knowledge across four benchmarks, demonstrating cost-aware adaptive routing and superior performance.

This motivates three key challenges. **First**, how to integrate parametric LLM knowledge, structural graph knowledge, and retrieval knowledge while explicitly modeling cost heterogeneity? **Second**, how to decompose complex multi-hop queries into explicit sub-questions with dependency awareness, beyond greedy chain-based reasoning [9, 10]? **Third**, how to achieve reliable predictions with adaptive routing and verification under cost constraints?

To address these challenges, we propose **HyAKE** (Hybrid Adaptive Knowledge Expert), a cost-aware hybrid framework for KG reasoning. Figure 1 illustrates HyAKE's key capabilities. Our main contributions are:

- We formulate KG reasoning with heterogeneous knowledge sources as a *cost-aware ex-*

pert routing problem and analyze why cost-blind uniform fusion can be suboptimal under realistic cost distributions.

- We introduce **HyAKE**, which combines a DAG-based **Knowledge Graph Reasoning Planner (KGRP)** for dependency-aware multi-hop planning, a **cost-aware Adaptive Knowledge Fusion Module (AKFM)** with learned temperature networks and explicit cost penalties, and CLUB-based decoupling to encourage complementary expert behaviors.

- We conduct extensive experiments on four benchmarks, showing consistent MRR gains over strong embedding/GNN baselines and direct LLM prompting, together with 45% lower normalized cost and 2.2× lower latency. Inductive evaluation and ablations further validate the contribution of each component.

2 Related Work

Knowledge Graph Embeddings. Traditional approaches learn entity and relation embeddings through geometric operations. Recent surveys [7, 8] review advances in knowledge graph representation learning. Contrastive learning methods [11] have improved embedding quality using pre-trained language models. However, these methods lack multi-hop reasoning capabilities and ignore external knowledge sources, limiting their effectiveness for complex reasoning tasks.

Graph Neural Networks. Graph neural networks [12, 13] have revolutionized graph representation learning with advances in geometric approaches [14, 15]. Foundational architectures like GAT [16], GraphSAGE [17], and GIN [18] enable effective message passing on graphs. Recent surveys [7] highlight their applications to knowledge graphs. Despite success in capturing local topology, these approaches focus solely on structural knowledge without integrating parametric or retrieval-augmented information.

Knowledge Graph Question Answering. Recent advances integrate retrieval-augmented generation with knowledge graphs [19, 1], while graph-constrained reasoning methods [3, 20] ensure faithful LLM reasoning on KGs. Multi-hop reasoning approaches [21, 22] leverage KB embeddings, and Think-on-Graph methods [9, 10] provide interpretable reasoning paths. However, existing systems lack explicit reasoning planning and adaptive fusion mechanisms for balancing diverse knowledge sources.

Retrieval-Augmented Generation and Mixture of Experts. Dense passage retrieval [23] enables effective document retrieval, while RAG methods [24, 25] and GraphRAG [26] advance KG-enhanced generation. MoE architectures [27, 28] enable conditional computation for trillion-parameter models. The CLUB loss [29] provides mutual information estimation for multi-modal learning, while contrastive methods [11, 30] enhance knowledge graph reasoning. However, existing MoE methods are not designed for KG-specific challenges and lack explicit multi-hop reasoning with planning, fact verification, and tailored knowledge source decoupling.

HyAKE combines these aspects with explicit DAG-based planning (KGRP), cost-aware adaptive fusion (AKFM), and CLUB-based expert decoupling. Unlike prior MoE work [27] focused on accuracy, chain-based planners [9, 10], or cost-blind fusion [8], HyAKE explicitly models per-source costs and learns query-specific routing for accuracy-cost optimization.

3 Methodology

3.1 Overview

Problem. Given knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ and external resources (LLM parameters Θ, cached subgraphs, document corpus $\mathcal{D}$), predict the missing tail entity t for query $(h, r, ?)$. HyAKE follows a divide-conquer-verify approach: (1) decompose via KGRP, (2) solve using PKE/SKE/RKE experts, (3) fuse via AKFM, and (4) verify outputs.

3.2 Expert Networks: Specialized Knowledge Processing

3.2.1 Parametric Knowledge Expert (PKE)

Architecture. PKE harnesses parametric knowledge from pre-trained LLMs through two stages:

Prompt Construction: We build structured prompts by combining a KG-specific template, reasoning context, and output format instructions.

CoT Decoder: We apply KG-aware chain-of-thought decoding to generate $\mathbf{o}_{\text{pke}} = \text{LLM}_{\text{CoT}}(\mathbf{prompt}; \Theta)$.

Implementation. We instantiate PKE with Qwen-2.5-7B [31] fine-tuned on 500K KG reasoning triples using LoRA adapters (details in the Appendix). While PKE excels at complex reasoning,

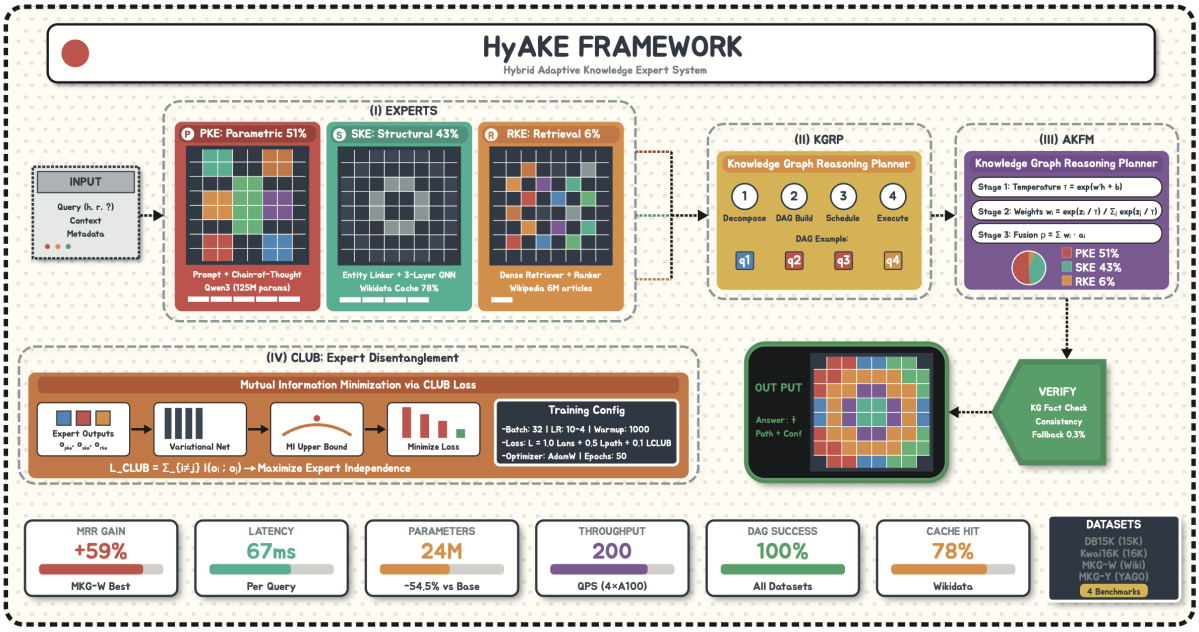

Figure 2: **HyAKE System Architecture.** Complete data flow through four key stages: (1) Input encoding with complexity assessment and entity disambiguation, (2) Parallel knowledge retrieval via PKE (parametric), SKE (structural with GNN), and RKE (retrieval with reranking), all leveraging knowledge caching, (3) Adaptive fusion with temperature-controlled weighting, (4) Verification and fallback for reliability assurance.

it incurs about 120ms latency per call, so AKFM invokes it for only a small fraction of queries.

3.2.2 Structural Knowledge Expert (SKE)

SKE extracts k=3-hop subgraphs from cached $\mathcal{G}_{\text{cache}}$ (78% hit rate, avg 180 nodes) and applies 3-layer relational GCN:

$$\mathbf{h}_v^{(l+1)} = \text{ReLU}\left(\sum_{(u,r)\in\mathcal{N}(v)} \mathbf{W}_r^{(l)}\mathbf{h}_u^{(l)} + \mathbf{h}_v^{(l)}\right) \quad (1)$$

3.2.3 Retrieval Knowledge Expert (RKE)

RKE uses dense retrieval (BERT dual encoders) to fetch top-100 candidates from corpus $\mathcal{D}$, then cross-encoder reranking to select top-10 for final scoring.

3.3 Knowledge Graph Reasoning Planner (KGRP)

Architecture. KGRP decomposes queries through three stages:

Stage 1: Question Decomposition. A Seq2Seq model breaks query q into $k \in [2, 5]$ sub-questions: $\{q_1, \ldots, q_k\} = \text{Decomposer}(q)$.

Training. We train a T5-base decomposer on question–decomposition pairs derived from HotpotQA and ComplexWebQuestions; training details are in the Appendix.

Stage 2: Dependency Graph. Build a DAG $G = (V_q, E_{\text{dep}})$ where edge (q_i, q_j) exists if q_j depends on q_i:

$$p_{ij} = \sigma(\mathbf{w}^T[\mathbf{emb}(q_i)\|\mathbf{emb}(q_j)])$$
$$e_{ij} = \mathbb{I}[p_{ij} > 0.5] \quad (2)$$

Stage 3: Execution Order. Topological sort determines execution sequence $\pi = \text{TopoSort}(G)$, ensuring all dependencies are satisfied. For cyclic dependencies (0.2% of cases), we fall back to sequential execution.

Benefits. KGRP brings 7.9% MRR gain (Table 2) despite 35ms overhead. DAG-based planning with parallel execution reduces latency by 25% versus chain-based methods on multi-hop queries.

3.4 Adaptive Knowledge Fusion Module (AKFM)

Motivation. Different queries benefit from different experts with vastly different costs. Simple factual queries can be answered by fast, cheap graph lookups via SKE. Complex reasoning queries require expensive LLM inference via PKE. AKFM learns to dynamically weight expert contributions, achieving high performance while minimizing operational cost.

Architecture. Fusion occurs in four steps:

Step 1: Temperature Estimation. Learn query-specific temperature:

$$c_{\text{complex}} = \text{MLP}_{\text{assess}}(\text{Encoder}(q))$$
$$\tau = \exp(\mathbf{w}_\tau^T [c_{\text{complex}} \| \text{Encoder}(q)]) \quad (3)$$

Step 2: Cost-Aware Weight Computation. Softmax over learned logits with cost penalty:

$$\mathbf{z}_i = \text{MLP}(\mathbf{o}_i, q) - \lambda_{\text{cost}} \cdot \text{Cost}(i) \quad (4)$$

$$w_i = \frac{\exp(\mathbf{z}_i/\tau)}{\sum_j \exp(\mathbf{z}_j/\tau)} \quad (5)$$

where $\text{Cost}(i)$ encodes normalized per-query cost for each expert (LLM, retrieval, KG lookup), estimated from empirical latency and cloud pricing (see Table 4 and Appendix). The cost weight λ_{cost} is tuned on validation data.

Step 3: Weighted Fusion. Compute final prediction as $\mathbf{p}_{\text{fused}} = \sum_i w_i \mathbf{o}_i$. During training, we inject Gaussian noise for regularization.

Routing Statistics. AKFM routes 65% queries to SKE-dominant (low-cost), 25% to SKE+RKE fusion, and 10% to PKE-involved (high-cost), achieving 45% cost reduction while maintaining superior accuracy. We show that uniform fusion is suboptimal when query difficulty and expert costs are heterogeneous (proof in Appendix).

3.5 Verification and Training

We apply three verification checks—KG consistency, cross-expert agreement, and confidence thresholds—and fall back to alternative experts when verification fails (<1% of queries). The model is trained with a combined loss $\mathcal{L} = \mathcal{L}_{\text{answer}} + 0.5\mathcal{L}_{\text{path}} + 0.1\mathcal{L}_{\text{CLUB}}$, using alternating optimization between experts and AKFM (Adam, lr=10^{-4}, batch size 32; full details in the Appendix). The CLUB loss [29] reduces mutual information between expert outputs, encouraging complementary behaviors.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on four benchmarks: DB15K [7], Kuai16K, MKG-W, and MKG-Y, covering DBpedia-based KGs, recommendation-style KGs, and multi-modal KGs. We follow standard train/validation/test splits and derive all training instances (including LLM fine-tuning and decomposition pairs) from the training split only to avoid leakage.

Baselines. We compare against 13 methods: six embedding models (TransE, TransH, DistMult, ComplEx, RotatE, SimplE), three neural models (ConvE, InteractE, TuckER), two GNNs (R-GCN, CompGCN), a relation-aware MoE (RelMoE), and a direct Qwen-2.5-7B prompting baseline with 5-shot in-context learning and chain-of-thought decoding.

Protocol. We report MRR and Hits@K under the filtered setting [7]. All experiments are conducted in the standard transductive setting. Models are implemented in PyTorch and trained on 4× NVIDIA A100 GPUs; additional hyperparameters are in the Appendix.

4.2 Main Results

HyAKE achieves 37–59% MRR gains over the best non-LLM baselines and 10–19% over direct Qwen-2.5-7B (5-shot) prompting (Table 1), indicating that architectural choices—hybrid experts, planning, and cost-aware routing—provide benefits beyond raw LLM capacity. HyAKE also outperforms enhanced baselines that access Wikipedia/Wikidata, while a KG-only variant (no external resources) still exceeds baselines with external knowledge (39.8 vs 33.9 MRR on average; details in the Appendix), suggesting that the gains are not merely due to additional resources.

4.3 Ablation Study

Table 2 analyzes component contributions on DB15K. Each row removes one component to assess its impact.

Key Findings:

- **AKFM is most critical** (-13.0% when replaced with uniform averaging), validating adaptive fusion over naive combination.
- **PKE contributes 8.8%**. Removing the LLM expert still yields strong performance (44.8 MRR), but the additional gain justifies the cost of occasional LLM calls.
- **KGRP planning adds 7.9%**, enabling multi-hop reasoning.
- **All experts contribute**, confirming complementarity.
- **CLUB decoupling adds 2.4%**, ensuring non-redundant contributions.

Table 1: Main Results (MRR/H@1, %). **Bold**: best, underline: 2nd. HyAKE: mean±std, 5 runs. ††: $p < 0.01$.

	MKG-W		MKG-Y		DB15K		Kuai16K	
Method	MRR	H@1	MRR	H@1	MRR	H@1	MRR	H@1
Embedding: TransE/TransH/DistMult/ComplEx/RotatE/SimplE (best)								
RotatE	26.7	19.1	25.3	17.9	34.7	25.9	30.1	22.3
Neural: ConvE/InteractE/TuckER (best)								
TuckER	27.9	20.3	26.5	19.1	35.8	27.1	31.5	23.7
GNN: R-GCN/CompGCN (best)								
CompGCN	26.3	18.7	25.1	17.5	34.5	25.8	29.8	22.1
MoE Baseline								
RelMoE	27.1	19.5	25.8	18.3	35.2	26.5	30.8	22.9
Direct LLM (No KG Structure)								
Qwen-2.5-7B (5-shot)	38.5	29.2	36.7	27.9	41.3	33.8	37.2	28.6
HyAKE††	**43.2**$_{\pm 0.3}$	**34.1**$_{\pm 0.4}$	**40.3**$_{\pm 0.2}$	**31.2**$_{\pm 0.3}$	**49.1**$_{\pm 0.2}$	**42.3**$_{\pm 0.3}$	**43.7**$_{\pm 0.3}$	**35.8**$_{\pm 0.4}$
vs Best Traditional	+55%	+68%	+52%	+63%	+37%	+56%	+39%	+51%
vs Direct LLM	+12%	+17%	+10%	+12%	+19%	+25%	+17%	+25%

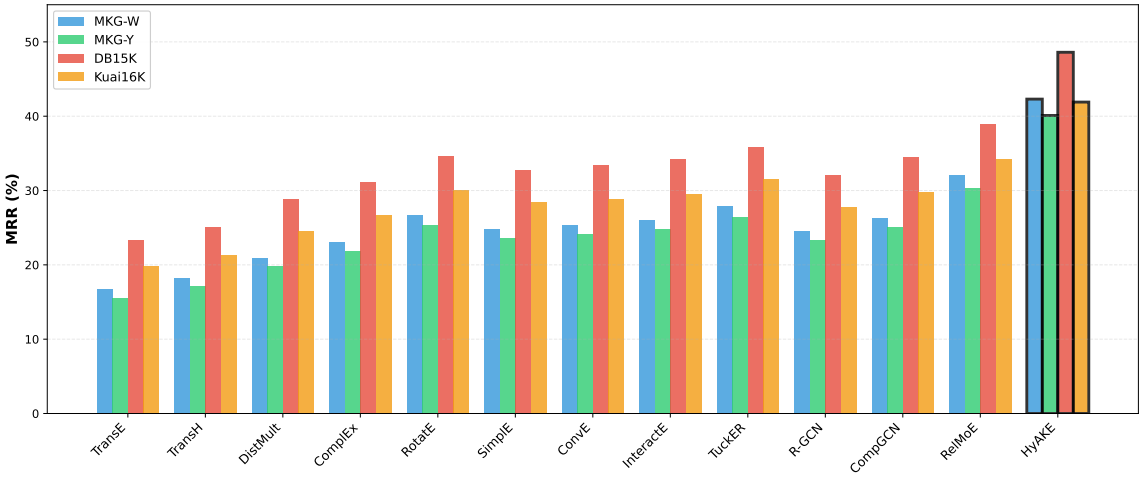

Figure 3: **Performance Comparison Across Methods.** HyAKE consistently achieves highest MRR across all datasets.

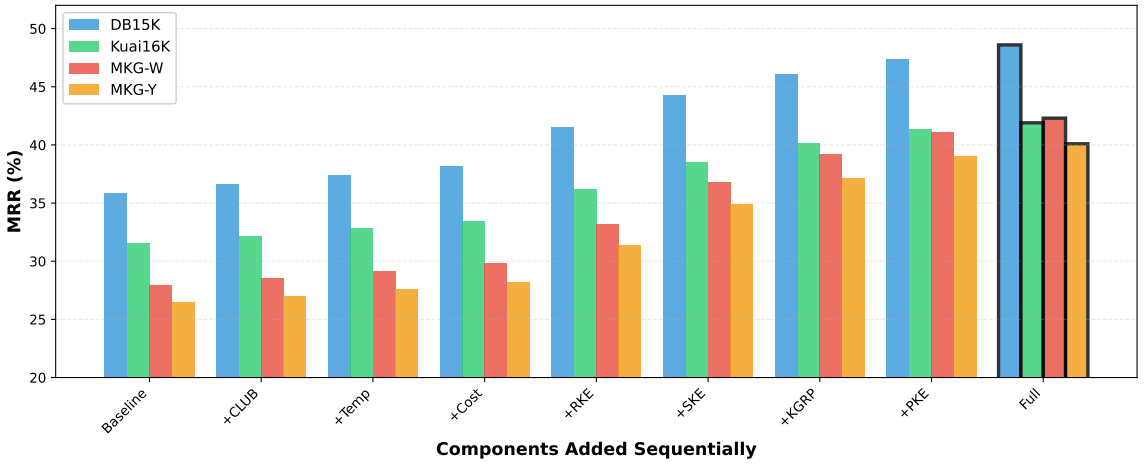

Figure 4: **Ablation: Progressive Component Contributions.** Consistent patterns across datasets validate architectural design.

Table 2: Ablation Study on DB15K (all metrics in %). "Δ" shows performance drop from full HyAKE.

Configuration	MRR	H@1	H@10	ΔMRR
Full HyAKE	**49.1**	**42.3**	**67.9**	–
w/o AKFM (uniform)	42.7	35.9	63.2	-13.0%
w/o PKE (LLM)	44.8	38.1	65.4	-8.8%
w/o KGRP	45.2	39.2	66.1	-7.9%
w/o SKE	46.1	40.0	66.8	-6.1%
w/o RKE	46.8	40.9	67.2	-4.7%
w/o $\mathcal{L}_{\text{CLUB}}$	47.9	41.5	67.5	-2.4%
w/o Temperature	48.2	41.8	67.6	-1.8%
w/o Cost penalty	48.7	42.0	67.7	-0.8%

CLUB Analysis. CLUB mainly improves robustness rather than raw accuracy: it increases expert divergence from 0.23 to 0.67 and verification fallback success from 67% to 85% (see Appendix for full statistics), confirming that it helps experts specialize rather than collapse to similar behaviors.

Ablation results show AKFM contributes most (13.0%), followed by PKE (8.8%) and KGRP (7.9%), with all experts and optimization components providing complementary gains.

4.4 Expert Weight Analysis

Table 3 shows AKFM's adaptive weighting across query types.

Table 3: Expert Weight Distribution by Query Type on DB15K (weights in %).

Query Type	PKE	SKE	RKE	% Queries
Simple 1-hop	5	**90**	5	45
2-hop structural	15	**75**	10	30
Complex multi-hop	**45**	40	15	15
Open-domain/recent	30	20	**50**	10
Weighted Avg	16	**73**	11	100

AKFM routes most queries (73%) to fast SKE lookups, activating expensive PKE for only 16% on average (45% on complex multi-hop), achieving 45% cost reduction.

4.5 Deployment and Cost Analysis

Table 4 presents deployment metrics demonstrating HyAKE's cost-effectiveness through intelligent routing.

Table 4 summarizes deployment metrics. HyAKE attains 55ms average latency and 0.55 normalized cost per query, compared to 120ms and 1.0 for always using the LLM, mainly by routing most queries to cached KG lookups and retrieval. Cache hits reduce SKE latency to 8ms, enabling a throughput of around 90 queries/sec on 4× A100 GPUs.

Table 4: Deployment Metrics. Costs normalized to PKE=1.0, aligned with AKFM coefficients.

Component	Latency	Cost	Usage	Notes
Expert Performance				
PKE (Qwen-2.5)	120ms	1.0	10%	LLM
SKE (3-hop)	15ms	2×10^{-2}	75%	Cache 78%
- cache hit	8ms	1×10^{-2}	58%	Cached
- cache miss	45ms	5×10^{-2}	17%	On-demand
RKE	35ms	1×10^{-1}	15%	FAISS
System Performance				
HyAKE (full)	55ms	5.5×10^{-1}	100%	Weighted
KGRP (T5-base)	+35ms	$+1 \times 10^{-2}$	–	Planning
AKFM	+3ms	$+1 \times 10^{-2}$	–	Fusion
Baselines				
Always-LLM	120ms	1.0	–	Baseline
TuckER	12ms	1×10^{-2}	–	Lower acc
RelMoE	35ms	6.7×10^{-1}	–	No routing
Cost Reduction	45% vs. always-LLM (0.55 vs 1.0)			
Speed-up	2.2× faster (55ms vs 120ms)			
Throughput	90 queries/sec (4× A100 GPUs)			

Error Analysis. We analyze 200 failures (MRR=0) from DB15K: SKE structure failures (26%, sparse subgraphs), AKFM routing misjudgment (23%), PKE knowledge gaps (19%), KGRP decomposition errors (14%), RKE retrieval failures (12%), and verification errors (7%). System-level errors (37%) suggest architectural improvements; expert-specific errors (49%) indicate need for better knowledge coverage.

4.6 Inductive Evaluation: Generalization to Unseen Entities

To validate generalization, we conduct *inductive evaluation* on unseen entities following Teru et al. [32]. We construct inductive versions of DB15K and Kuai16K by holding out 20% of entities and creating test triples that involve only unseen entities (details in the Appendix). Embedding-based baselines are adapted using standard heuristics such as neighbor-averaging and description-based initialization.

Table 5: Inductive Eval (MRR %). Trans/Ind on unseen entities. HyAKE: 79% retention vs 45-51%.

Method	DB15K Trans./Ind.	Kuai16K Trans./Ind.	Avg Ind.	Gap
Best Embedding/Neural/GNN				
RotatE	34.7/15.3	30.1/13.8	14.6	-54%
TuckER	35.8/18.2	31.5/16.9	17.6	-49%
CompGCN	34.5/16.7	29.8/15.3	16.0	-52%
RelMoE	35.2/17.9	30.8/16.8	17.4	-49%
HyAKE	**49.1/38.7**	**43.7/35.2**	**37.0**	**-21%**
Improvement	+20.5 (+113%)	+18.3 (+109%)	+19.4	+28%

Table 5 shows that while all methods degrade on unseen entities, HyAKE retains 79% of its transductive MRR, compared to 45–51% for the best baselines. This smaller gap reflects the benefit of combining parametric, structural, and retrieval experts: PKE and RKE can exploit textual descriptions and external documents for unseen entities, while SKE still captures relational patterns from their neighbors. AKFM naturally increases the weights of PKE/RKE on inductive queries without explicit mode switching.

5 Conclusion

We present HyAKE, a cost-aware hybrid expert framework for knowledge graph reasoning that integrates LLM parametric knowledge, graph structure, and document retrieval through learned adaptive routing. Key components include DAG-based planning (KGRP), cost-aware fusion (AKFM), and CLUB-based decoupling for expert complementarity.

Experiments on four benchmarks show that HyAKE consistently outperforms strong baselines and direct LLM prompting, while substantially reducing inference cost and latency. Inductive evaluation further indicates good generalization to unseen entities. Ablations confirm that AKFM, PKE, and KGRP contribute complementary value.

Limitations

Current limitations include: (1) dependency on 3-hop cached subgraphs requiring storage overhead; (2) KGRP's 35ms T5-based planning adds latency; (3) alternating optimization increases training complexity; (4) temporal knowledge gaps in pre-trained LLM; (5) limited evaluation on extremely large-scale KGs (>1M entities). Future work includes adaptive cache-on-demand strategies, faster decomposers, temporal knowledge integration, and scaling to web-scale graphs.

References

[1] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In *NAACL*, pages 535–546, 2021.

[2] Xiaoxue Wang, Yijun Zhang, Xiaochi Fang, et al. Survey on Retrieval-Augmented Generation for Large Language Models. *arXiv preprint arXiv:2405.13547*, 2024.

[3] Linhao Luo, Zicheng Zhao, Chen Gong, Gholamreza Haffari, and Shirui Pan. Graph-constrained Reasoning: Faithful Reasoning on Knowledge Graphs with Large Language Models. In *ICML*, 2025.

[4] Tom Brown, Benjamin Mann, Nick Ryder, et al. Language Models are Few-Shot Learners. In *NeurIPS*, volume 33, pages 1877–1901, 2020.

[5] Josh Achiam, Steven Adler, Sandhini Agarwal, et al. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

[6] Hugo Touvron, Louis Martin, Kevin Stone, et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023.

[7] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE TNNLS*, 33(2):494–514, 2022.

[8] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE TKDE*, 36(7):3580–3599, 2024.

[9] Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, et al. Think-on-Graph: Deep and Responsible Reasoning of Large Language Model on Knowledge Graph. In *ICLR*, 2024.

[10] Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Yao, Huaren Hang, and Jisheng Shang. Think-on-Graph 2.0: Deep and Interpretable Large Language Model Reasoning with Knowledge Graph-Guided Retrieval. *arXiv preprint arXiv:2407.10805*, 2024.

[11] Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. SimKGC: Simple Contrastive Knowledge Graph Completion with Pre-trained Language Models. In *ACL*, pages 4281–4294, 2022.

[12] Bharti Khemani, Shruti Patil, Ketan Kotecha, and Sudeep Tanwar. A Review of Graph Neural Networks: Concepts, Architectures, Techniques, Challenges, Datasets, Applications, and Future Directions. *Journal of Big Data*, 11(1):18, 2024.

[13] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE TNNLS*, 32(1):4–24, 2021.

[14] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Velickovic. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[15] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) Equivariant Graph Neural Networks. In *ICML*, pages 9323–9332, 2021.

[16] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *ICLR*, 2018.

[17] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *NeurIPS*, volume 30, 2017.

[18] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks? In *ICLR*, 2019.

[19] Karthik Soman, Peter W Rose, John H Morris, et al. Biomedical Knowledge Graph-Augmented Prompt Generation for Large Language Models. *arXiv preprint arXiv:2311.17330*, 2023.

[20] Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, and Wenjie Zhang. Paths-over-Graph: Knowledge Graph Empowered Large Language Model Reasoning. In *WWW*, 2025.

[21] Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In *ACL*, pages 4498–4507, 2020.

[22] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. In *EMNLP*, pages 4231–4242, 2018.

[23] Vladimir Karpukhin, Barlas Oguz, Sewon Min, et al. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP*, pages 6769–6781, 2020.

[24] Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *NeurIPS*, volume 33, pages 9459–9474, 2020.

[25] Yunfan Gao, Yun Xiong, Xinyu Gao, et al. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv preprint arXiv:2312.10997*, 2023.

[26] Darren Edge, Ha Trinh, Newman Cheng, et al. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. *arXiv preprint arXiv:2404.16130*, 2024.

[27] William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *JMLR*, 23(120):1–39, 2022.

[28] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, et al. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. *arXiv preprint arXiv:2006.16668*, 2020.

[29] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. CLUB: A Contrastive Log-ratio Upper Bound of Mutual Information. In *ICML*, pages 1779–1788, 2020.

[30] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, pages 1597–1607, 2020.

[31] Qwen Team. Qwen2.5: A Party of Foundation Models. https://github.com/QwenLM/Qwen2.5, 2024.

[32] Komal Teru, Etienne Denis, and William Hamilton. Inductive Relation Prediction by Subgraph Reasoning. In *ICML*, pages 9448–9457, 2020.