Beyond Prefixes: Graph-as-Memory Cross-Attention for Knowledge Graph Completion with Large Language Models

Anonymous ACL submission

Abstract

Fusing Knowledge Graphs with Large Language Models (LLMs) is crucial for knowledge-intensive tasks like knowledge graph completion. Existing LLM-based approaches typically inject graph information via prefix concatenation, resulting in shallow interactions that fail to support fine-grained evidence retrieval during generation. Beyond prefixes, we propose Graph-as-Memory Tuning (GMT), a new paradigm that represents local graph structure as explicit graph memory and injects it into LLMs via deep, token-wise cross-attention. Specifically, GMT first employs a Semantic Graph Module to encode context-aware semantics from local neighborhoods guided by knowledge-enhanced relations, and compresses them into a fixed number of graph memory tokens. A Graph-as-Memory Cross-Attention Fusion Module then integrates these tokens into multiple Transformer layers, allowing LLM hidden state to dynamically retrieve relevant graph evidence. To enable efficient adaptation, GMT applies LoRA only to the memory cross-attention while keeping the base LLM frozen. Extensive experiments show that GMT significantly outperforms prefix-tuning and other strong baselines, providing more potent signals for robust reasoning.

1 Introduction

Knowledge Graphs (KGs) organize complex facts into structured triples (h, r, t), empowering applications ranging from recommendation systems (Cui et al., 2025; Chen et al., 2025) to question answering (Omar et al., 2023; Lu et al., 2025). However, their inherent incompleteness necessitates Knowledge Graph Completion (KGC) to infer missing links from existing facts (Pan et al., 2024; Li et al., 2025a).

Traditionally, KGC has relied on *embedding-based models*, including geometric approaches (Bordes et al., 2013; Sun et al., 2019) and graph neural networks (Dettmers et al., 2018; Zhu et al., 2021). While proficient at encoding static structural patterns, these methods often overlook the rich textual semantics inherent in entities and relations. To address this, recent research has pivoted towards *LLM-based paradigms* (Yao et al., 2019; Li et al., 2024; Zhang et al., 2024a), which leverage the generative and semantic capabilities of pre-trained models. However, existing integration strategies, predominantly based on prefix-tuning, typically employ a shallow fusion approach that simply concatenates structural embeddings with textual inputs (Zhang et al., 2024a). This shallow interaction fails to deeply align structural signals with textual representations, imposing a heavy implicit reasoning burden on the LLM and often leading to hallucinations or context-insensitive predictions.

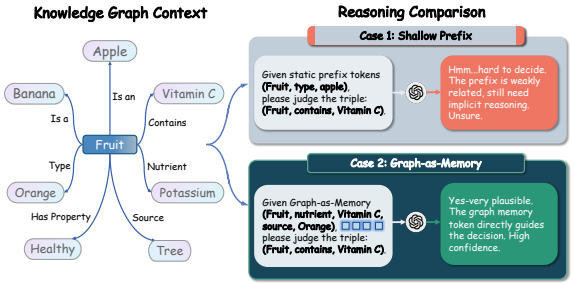

Figure 1: The guiding effect of structured information on LLMs. The semantics of the relation "Treats" change dynamically based on the graph context.

Consequently, a critical challenge remains: *how to effectively fuse explicit KG structure with implicit LLM semantics at a deep, feature-interactive level*? As illustrated in Figure 1, relational semantics are dynamic and context-dependent. The relation "Treats" implies distinct mechanisms, such as "Symptomatic Relief" or "Pathogen Targeted Treatment", depending entirely on the local graph neighborhood (e.g., *Aspirin* vs. *Oseltamivir*). Capturing these nuanced shifts requires more than static concatenation; it demands a mechanism that can

dynamically modulate the LLM's perception based on structural context.

To bridge this gap, we propose **Graph-as-Memory Tuning (GMT)**, a memory-centric framework that reframes local graph structure as an explicit graph memory and integrates it into LLMs through deep, token-wise cross-attention. GMT comprises two core components: 1) A **Semantic Graph Module** that uses a relation centric message passing mechanism to extract a dense and context-aware semantics from the local neighborhood, guided by knowledge enhanced relation descriptions, and compresses them into a fixed number of graph memory tokens. 2) A **Graph-as-Memory Cross-Attention Fusion Module** that injects these graph memory tokens into multiple Transformer layers, enabling each prompt token hidden representation to dynamically retrieve relevant evidence from the graph memory during generation. To maintain parameter efficiency, GMT keeps the base LLM frozen and applies LoRA only to the memory cross-attention pathway. Our main contributions are summarized as follows:

- We propose **GMT**, a deep fusion paradigm that replaces shallow concatenation with memory-based, token-wise retrieval via cross-attention, bridging graph structure and LLM semantics. (Section 3)
- We introduce a **Semantic Graph Module** that leverages knowledge-enhanced relation semantics to guide neighborhood aggregation and construct compact graph memory tokens. (Section 3.2)
- We design a **Graph-as-Memory Cross-Attention Fusion Module** that performs multi-layer memory injection and token-wise retrieval, and enable parameter-efficient training by applying LoRA-based adaptation to align graph memory with a frozen LLM. (Section 3.3)
- Empirical results confirm that GMT achieves state-of-the-art performance on multiple KGC benchmarks, validating the efficacy of deep injection. (Section 4)

2 Related Work

2.1 Knowledge Graph Completion

Traditional KGC methods predominantly rely on embedding-based paradigms to capture structural patterns. Geometric models, such as TransE (Bordes et al., 2013) and RotatE (Sun et al., 2019), map entities to continuous spaces using translational or rotational scoring functions. Extensions like HAKE (Zhang et al., 2020) and BoxE (Abboud et al., 2020) further incorporate hierarchical and logical constraints. Parallelly, GNN-based approaches like ConvE (Dettmers et al., 2018) and NBFNet (Zhu et al., 2021) capture local topology through message passing. Despite their structural efficacy, these methods assign static representations to entities, limiting their ability to generalize to unseen data or leverage the rich semantics inherent in KGs (Chang et al., 2024).

2.2 LLMs for KGC

The adoption of Large Language Models has significantly advanced semantic reasoning in KGC. Early LLM-based discriminative approaches, such as KG-BERT (Yao et al., 2019) and SimKGC (Wang et al., 2022), formulate KGC as a sequence classification task but do not explicitly model graph topology. In the generative setting, instruction-tuned frameworks including KICGPT (Wei et al., 2023) and KG-LLaMA (Zhu and De Meo, 2025) directly predict tail entities, while later methods such as MKGL (Guo et al., 2024b) and KG-FIT (Jiang et al., 2024) incorporate multi-view learning and few-shot inductive reasoning capabilities. More recent frameworks seek to integrate explicit structural information with the semantic reasoning capabilities of LLMs. Representative approaches include KoPA (Zhang et al., 2024a), which project structural embeddings into the textual space as prefix tokens; SSQR (Lin et al., 2025), which optimizes this via quantized representations; and GLTW (Luo et al., 2025), which employs a joint graph transformer architecture. Nevertheless, these methods largely rely on shallow integration strategies such as prefix concatenation or textualization, failing to establish a deep, feature-level interaction in which graph structure dynamically modulates the internal representations of the LLM.

3 Methodology

3.1 Preliminaries

A knowledge graph $\mathcal{G}$ is defined as a set of triples $\mathcal{T} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the entity and relation sets, respectively. Knowledge Graph Completion (KGC) aims to infer a missing element in an incomplete triple, e.g., predicting t for a query $(h, r, ?)$. In LLM-based KGC, a model $\mathcal{M}$ is prompted with a textualized query and trained to generate (or select) the missing

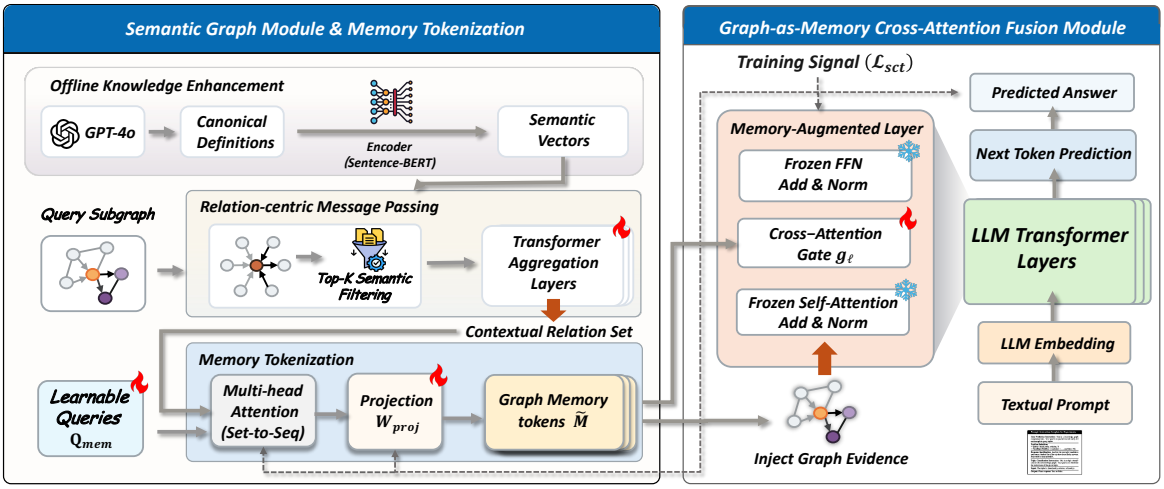

Figure 2: The GMT framework.

entity.

As shown in Figure 2, our GMT conditions the LLM on a graph memory built from the local semantic subgraph around the query entities, and injects such memory into multiple Transformer layers through cross-attention, enabling deep and token-wise retrieval of graph evidence during generation.

3.2 Semantic Graph Module

The first stage of GMT is to transform the local neighborhood structure into a dense set of contextual semantic representations that can serve as external memory for the LLM. Unlike methods that directly rely on pre-trained entity embeddings (Zhang et al., 2024a), our Semantic Graph Module (SGM) performs *relation-centric* message passing to extract semantically filtered relational evidence around the entities.

Relation-centric Message Passing. Inspired by (Wang et al., 2021; Li et al., 2025b), we treat relations as the primary carriers of KG semantics[1]. For a given query triple (h, r, t), we extract the local neighborhoods around h and t (masking the predicted entity for link prediction). For each central edge e_c incident to h or t, we aggregate information from its neighboring edges $e_n \in \mathcal{N}(e_c)$. To mitigate noise from indiscriminate aggregation, we perform Top-K neighbor filtering using explicit semantic relevance. Specifically, we first conduct an offline **Knowledge Enhancement** step for each relation type using a strong LLM (GPT-4o[2]), producing canonical definitions (Prompt in Appendix A), and encode them with a embedding model (e.g., Sentence-BERT[3] (Reimers and Gurevych, 2019)) to obtain a semantic vector for each relation. During message passing, for each central edge e_c and a neighbor edge e_n, we compute relevance by cosine similarity between their pre-computed semantic vectors $\mathbf{s}_c$ and $\mathbf{s}_n$:

$$\text{Score}(e_c, e_n) = \frac{\mathbf{s}_c \cdot \mathbf{s}_n}{\|\mathbf{s}_c\|_2 \, \|\mathbf{s}_n\|_2}. \quad (1)$$

We then select $\mathcal{N}_K(e_c)$ as the Top-K neighbors with highest scores and aggregate their states (e.g., mean pooling) to obtain $\bar{\mathbf{s}}^{\mathcal{N}_K(e_c)}$.

We refine each central edge representation via a Transformer-style update, where $\mathbf{s}_c^l$ attends to the aggregated neighborhood context:

$$\mathbf{s}_c^{l+1} = \text{TransformerLayer}(\mathbf{s}_c^l, \bar{\mathbf{s}}^{\mathcal{N}_K(e_c)}), \quad (2)$$

and iterate for L layers to obtain contextual edge representations around h and t.

Graph Memory Tokenization. A single pooled vector inevitably bottlenecks diverse evidence. Instead, we expose a **set of memory tokens** to the LLM. Let $\mathcal{S} = \{\mathbf{s}_{h,i}^L\}_i \cup \{\mathbf{s}_{t,j}^L\}_j \in \mathbb{R}^{N \times d_c}$ denote the set of contextual relation states collected from both sides (with N variable per query). We compress $\mathcal{S}$ into a fixed-length graph memory with m tokens using a learnable set-to-sequence tokenizer. Concretely, we introduce m learnable memory queries $\mathbf{Q}_{mem} \in \mathbb{R}^{m \times d_c}$ and compute:

$$\mathbf{M} = \text{Attn}(\mathbf{Q}_{mem}, \mathcal{S}, \mathcal{S}) \in \mathbb{R}^{m \times d_c}, \quad (3)$$

[1]For instance, the rule (A, is_father_of, B) ∧ (C, is_wife_of, A) → (C, is_mother_of, B) is resolved by relational interplay. Moreover, an entity's identity is often implicitly encoded by its relational context, making relation-centric aggregation effective.

[2]https://github.com/openai/openai-python.

[3]https://github.com/UKPLab/sentence-transformers

where $\mathrm{Attn}(\cdot)$ is standard multi-head attention. $\mathbf{M}$ serves as a compact yet expressive **Graph-as-Memory** representation for the subsequent fusion module. Finally, we project $\mathbf{M}$ to the LLM hidden size:

$$\tilde{\mathbf{M}} = \mathbf{M}\mathbf{W}_{proj}, \quad \mathbf{W}_{proj} \in \mathbb{R}^{d_c \times d_{llm}}. \quad (4)$$

3.3 Graph-as-Memory Cross-Attention Fusion

The core challenge is to inject the structured graph evidence into the LLM in a *deep* and *token-wise* manner. We propose a Graph-as-Memory Cross-Attention Fusion Module that allows each token to retrieve relevant KG evidence from $\tilde{\mathbf{M}}$ at multiple layers.

Memory-augmented Transformer Layers. Given the input prompt embeddings $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, the LLM produces hidden states $\mathbf{H}^0 = \mathbf{X}$ and updates them through L_{llm} Transformer layers. For a subset of layers $\mathcal{L}_{mem} \subseteq \{1, \ldots, L_{llm}\}$, we insert a cross-attention sub-layer after self-attention:

$$\hat{\mathbf{H}}^{\ell} = \mathbf{H}^{\ell} + \mathrm{SelfAttn}(\mathrm{LN}(\mathbf{H}^{\ell})), \quad (5)$$

$$\mathbf{H}^{\ell+1} = \hat{\mathbf{H}}^{\ell} + g_{\ell} \cdot \mathrm{CrossAttn}\big(\hat{\mathbf{H}}^{\ell}, \tilde{\mathbf{M}}\big), \quad (6)$$

followed by the standard FFN block. Here $\mathrm{CrossAttn}(\cdot)$ uses queries from token states and keys/values from the graph memory, enabling each token to selectively retrieve graph evidence. We use a learnable gate g_{ℓ} for training stability, allowing the model to gradually incorporate memory.

Low-Rank Adaptation of Cross-Attention. To adapt the memory-reading pathway efficiently, we apply LoRA to the projection matrices of cross-attention (and only these parameters, unless otherwise stated). For each projection matrix $\mathbf{W} \in \{\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v, \mathbf{W}_o\}$ in the cross-attention module, LoRA parameterizes:

$$\mathbf{W} = \mathbf{W}_0 + \Delta\mathbf{W}, \quad \Delta\mathbf{W} = \mathbf{B}\mathbf{A}, \quad (7)$$

where $\mathbf{W}_0$ is frozen, $\mathbf{A} \in \mathbb{R}^{r \times d}$, $\mathbf{B} \in \mathbb{R}^{d \times r}$, and $r \ll d$. Only $\mathbf{A}, \mathbf{B}$ are trained. This design keeps the base LLM frozen while opening a dedicated, parameter-efficient channel to align graph memory with the LLM latent space.

At inference, we construct graph memory $\tilde{\mathbf{M}}$ from the query neighborhood and generate the missing entity autoregressively:

$$\mathcal{A} = \arg\max_{\mathcal{A}} P_{\mathcal{M}}(\mathcal{A} \mid \mathcal{I}_{\mathrm{GMT}}, \mathbf{X}, \tilde{\mathbf{M}}), \quad (8)$$

where $\mathcal{I}_{GMT}$ is the instruction template and $\mathcal{A}$ is the predicted answer.

3.4 Training Strategy

We adopt a two-stage training paradigm. Stage 1 pre-trains the graph module to capture structural/semantic regularities, providing a strong initialization. Stage 2 aligns the graph memory and the LLM through memory-augmented fine-tuning with LoRA on cross-attention.

3.4.1 Stage 1: Self-Supervised Pre-training

We first pre-train SGM on a self-supervised link prediction objective. Given (h, r, t), the graph module produces contextualized representations for the head and tail, denoted as $\mathbf{e}_h$ and $\mathbf{e}_t$ (obtained by mean pooling the final relation states associated with each entity). We define a plausibility score:

$$F_{\mathcal{G}}(h, r, t) = \langle \mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t \rangle, \quad (9)$$

where $\mathbf{e}_r$ is the knowledge-enhanced relation semantic vector (from GPT-4o and Sentence-BERT). With negative sampling, for each positive triple $(h, r, t) \in \mathcal{T}$, we construct corrupted triples $\mathcal{T}' = \{(h'_i, r, t'_i)\}_{i=1}^{k}$. The loss is:

$$\mathcal{L}_{\text{Graph}} = -\sum_{(h,r,t)\in\mathcal{T}} \Big[\log \sigma(F_{\mathcal{G}}(h, r, t)) + \sum_{i=1}^{k} \log \sigma\big(-F_{\mathcal{G}}(h'_i, r, t'_i)\big) \Big]. \quad (10)$$

This stage equips SGM with robust relational semantics before interacting with the LLM.

3.4.2 Stage 2: Memory-Augmented Alignment with the LLM

Starting from the pre-trained SGM, we fine-tune the full GMT pipeline on the KGC objective while keeping the base LLM frozen. For each training query, we build the graph memory $\tilde{\mathbf{M}}$ using Eq. (3) and feed the textual prompt into the LLM equipped with memory cross-attention layers (Eq. (6)). We optimize the auto-regressive next-token prediction loss:

$$\mathcal{L}_{\mathrm{GMT}} = -\sum_{i=1}^{|\mathcal{S}_{GMT}|} \log P(s_i \mid s_{<i}), \quad (11)$$

where $\mathcal{S}_{GMT} = \mathcal{I}_{GMT} \oplus \mathbf{X} \oplus \mathcal{A}$. Gradients from $\mathcal{L}_{\mathrm{GMT}}$ update: (i) the graph memory tokenizer parameters and projection $\mathbf{W}_{proj}$, and (ii) the LoRA weights of cross-attention in the selected layers. This training aligns the graph memory space with

Algorithm 1 Memory-Augmented Fine-tuning Stage of GMT

Input: Knowledge graph $\mathcal{G}$; training triple (h, r, t); pre-trained SGM_θ; memory tokenizer parameters ω (including $\mathbf{Q}_{mem}$); memory projection $\mathbf{W}_{proj}$; frozen base LLM; cross-attn LoRA parameters $\psi_{\text{mem-lora}}$.
Output: Updated $(\omega, \mathbf{W}_{proj}, \psi_{\text{mem-lora}})$.
Prepare textual input:
  $S_{in} \leftarrow Textualize(h, r)$;
  $\mathbf{X} \leftarrow Embedding_{LLM}(S_{in})$;
Build graph memory (mask the predicted entity for link prediction):
  $\mathcal{S} \leftarrow SGM_\theta(h, t, \mathcal{G})$; {$\mathcal{S}$: contextual relation set}
  $\mathbf{M} \leftarrow \text{Attn}(\mathbf{Q}_{mem}, \mathcal{S}, \mathcal{S})$;
  $\tilde{\mathbf{M}} \leftarrow \mathbf{M}\mathbf{W}_{proj}$;
Memory-augmented forward with LoRA cross-attention:
  $P(\cdot \mid \mathbf{X}, \tilde{\mathbf{M}}) \leftarrow LLM(\mathbf{X}; \tilde{\mathbf{M}}, \psi_{\text{mem-lora}})$;
Compute loss and update parameters:
  $\mathcal{L}_{\text{GMT}} \leftarrow -\sum_{j=1}^{|t|} \log P(t_j \mid \mathbf{X}, \tilde{\mathbf{M}}, t_{<j})$;
  Update $(\omega, \mathbf{W}_{proj}, \psi_{\text{mem-lora}})$ by backprop;
Return updated parameters.

Table 1: Statistics of the benchmark datasets. For triple classification datasets (UMLS, CoDeX-S, FB15k-237N), the validation/test splits are denoted as positive/negative samples.

Dataset	Task	#Entities	#Relations	#Train	#Valid	#Test
WN18RR	Link	40,943	11	86,835	3,034	3,134
FB15k-237	Link	14,541	237	272,115	17,535	20,466
UMLS	Triple	135	46	5216	652/652	661/661
CoDeX-S	Triple	2034	42	32888	1827/1827	1828/1828
FB15k-237N	Triple	13,104	93	87,282	7041/7041	8226/8226

the LLM's internal representations through a dedicated memory-reading pathway, enabling deep fusion without modifying the base model weights. The detailed workflow is presented in Algorithm 1.

4 Experiments

To evaluate the effectiveness of GMT, we designed a series of experiments to address the following research questions:

- **RQ1:** How significantly can GMT enhance LLMs' performance in KGC tasks?
- **RQ2:** What are the distinct contributions of GMT's key components to its overall performance?
- **RQ3:** How Does Context from Semantic Graphs Enhance the Reasoning of LLMs?

4.1 Experiment Settings

Datasets. We evaluate our GMT framework on various widely recognized KGC benchmark datasets:

WN18RR (Dettmers et al., 2018) and **FB15k-237** (Toutanova and Chen, 2015): These datasets are used for the **link prediction** task. Both have inverse relations removed, making them standards for evaluating complex reasoning abilities.

UMLS, **CoDeX-S** and **FB15k-237N** (Lv et al., 2022): These are employed for the **triple classification** task. It provides high-quality negative samples for each triple, making them ideal for assessing a model's ability to distinguish factual correctness. Detailed statistics are provided in Table 1.

Baselines. Our selection of baselines covers a wide spectrum of representative methods, from traditional KGE models to the latest LLM-based approaches. Many results are directly cited from the SSQR (Lin et al., 2025), Kopa (Zhang et al., 2024a) and GLTW (Luo et al., 2025) to maintain consistency.

For Link Prediction. We compare against (1) *Emb-based Methods*: TransE (Bordes et al., 2013), CompGCN (Vashishth et al., 2020), AdaProp (Zhang et al., 2024b), MA-GNN (Xu et al., 2023), TCRA (Guo et al., 2024a), ARR (Chen et al., 2024), and DiffusionE (Cao et al., 2024); (2) *LLM-based Methods*: KICGPT (Wei et al., 2023), CSProm-KG-CD (Li et al., 2023), KG-FIT (Jiang et al., 2024), MKGL (Guo et al., 2024b), SSQR-LLaMA2 (Lin et al., 2025).

For Triple Classification. We compare against (1) *Emb-based Methods*: TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), ComplexE (Trouillon et al., 2016), and RotatE (Sun et al., 2019); (2) *LLM-based Methods*: KG-LLaMA (Zhu and De Meo, 2025), KG-Alpaca (Zhu and De Meo, 2025), KOPA (Zhang et al., 2024a), SSQR-LLaMA2 (Lin et al., 2025) and SAT (Liu et al., 2025).

Implementation Details. We build GMT on Alpaca-7B with the base LLM frozen. We pre-train the SGM for 20 epochs with self-supervised link prediction using negative sampling (k=64), and then fine-tune GMT with the base LLM frozen. We use K=8 for Top-K neighbor filtering and m=32 memory tokens, injecting memory via cross-attention in the top layers $\mathcal{L}_{mem}$={25, . . . , 32} with a learnable gate initialized to zero. In Stage 2, we train only the memory tokenizer, the memory projection, and LoRA on cross-attention projections $\{W_q, W_k, W_v, W_o\}$ (rank r=64, α=128, dropout 0.01). We tune epochs in {3,4,5} and learning rate in $\{1e{-}4, 3e{-}4, 5e{-}4\}$, using AdamW with batch size 12, bf16, and gradient clipping 100.0. Experiments are run on Nvidia A800-80GB

Table 2: Link Prediction on FB15k-237 and WN18RR datasets. Best results are in **bold**, second best are underlined.

Type	Model	WN18RR				FB15k-237			
		MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
Emb-based	TransE	0.223	0.014	0.401	0.529	0.330	0.231	0.369	0.528
	CompGCN	0.479	0.443	0.494	0.546	0.355	0.264	0.390	0.535
	AdaProp	0.562	0.499	–	0.671	0.417	0.331	–	0.585
	MA-GNN	0.565	0.507	0.592	0.679	0.379	0.282	0.415	0.569
	TCRA	0.496	0.457	0.511	0.574	0.367	0.275	0.403	0.554
	ARR	0.521	–	0.607	–	0.398	–	0.436	–
	DiffusionE	0.557	0.504	–	0.658	0.376	0.294	–	0.539
LLM-based	KICGPT	0.549	0.474	0.585	0.641	0.412	0.327	0.448	0.554
	CSProm-KG-CD	0.559	0.508	0.578	0.660	–	–	–	–
	KG-FIT	0.553	0.488	0.595	0.695	0.362	0.275	0.485	0.572
	MKGL	0.552	0.500	0.577	0.656	0.415	0.325	0.454	0.591
	SSQR-LLaMA2	0.591	0.548	0.618	0.673	0.449	0.374	0.491	0.597
	GLTW$_{7b}$	0.593	0.556	0.649	0.690	0.469	0.351	0.481	0.614
Ours	**GMT**	**0.621**	**0.569**	**0.667**	**0.703**	**0.488**	**0.394**	**0.505**	**0.629**

Table 3: Triple classification on UMLS, CoDeX-S, and FB15K-237N datasets. Best results are in **bold**, second best are underlined.

Type	Model	UMLS				CoDeX-S				FB15K-237N			
		Acc	P	R	F1	Acc	P	R	F1	Acc	P	R	F1
Emb-based	TransE	84.49	86.53	81.69	84.04	72.07	71.91	72.42	72.17	69.71	70.80	67.11	68.91
	DistMult	86.38	87.06	86.53	86.79	66.79	69.67	59.46	64.16	58.66	58.98	56.84	57.90
	ComplEx	90.77	89.92	91.83	90.87	67.64	67.84	67.06	67.45	65.70	66.46	63.38	64.88
	RotatE	92.05	90.17	94.41	92.23	75.68	75.66	75.71	75.69	68.46	69.24	66.41	67.80
LLM-based	Alpaca$_{0-shot}$	52.64	51.55	87.69	64.91	50.62	50.31	**99.83**	66.91	56.06	53.32	97.37	68.91
	GPT-3.5$_{0-shot}$	67.58	88.04	40.71	55.67	54.68	69.13	16.94	27.21	60.15	86.62	24.01	37.59
	ICL$_{8-shot}$	55.52	55.85	52.65	54.21	50.62	50.31	**99.83**	66.91	59.23	57.23	73.02	64.17
	KG-LLaMA	85.77	87.84	83.05	85.38	79.43	78.67	80.74	79.69	74.81	67.37	96.23	79.25
	KG-Alpaca	86.01	**94.91**	76.10	84.46	80.25	79.38	81.73	80.54	69.91	62.71	**98.28**	76.56
	KoPA	92.58	90.85	94.70	92.70	82.74	77.91	91.41	84.11	77.65	70.81	94.09	80.81
	SAT	92.24	91.05	93.99	93.17	85.55	**83.38**	89.31	86.54	82.71	82.30	85.24	83.28
Ours	**GMT**	**94.55**	91.86	**95.74**	**93.76**	**89.01**	83.27	92.43	**87.61**	**84.10**	**83.14**	91.27	**87.02**

GPUs. All closed-source models used for knowledge enhancement are accessed via their official APIs, using default inference hyperparameters (e.g., temperature) with no additional tuning.

4.2 Overall Performance Comparison: RQ1

Link Prediction Classic link prediction is a ranking task that evaluates the position of the gold entity among candidates. We fine-tune GMT with an instruction template $\mathcal{I}_{GMT}$ (Appendix A) and follow SSQR-LLaMA2's candidate setting by using a pre-trained AdaProp to retrieve 20 candidates per query. As shown in Table 2, GMT achieves state-of-the-art results on both WN18RR and FB15k-237. On **WN18RR**, GMT achieves state-of-the-art performance with **0.621** MRR and **0.703** Hits@10, outperforming the strongest LLM-based baseline (MRR 0.593). On **FB15k-237**, GMT also ranks first with **0.488** MRR and **0.629** Hits@10, surpassing the best competitor (MRR 0.469). These consistent gains across both datasets validate the effectiveness of deep, memory-based graph evidence retrieval.

Table 4: The ablation results for the link prediction task.

Model	MRR	Hits@1	Hits@3	Hits@10
WN18RR				
GMT	**0.621**	**0.569**	**0.667**	**0.703**
w/o Semantics	0.603	0.541	0.639	0.695
w/o Fusion	0.558	0.516	0.607	0.639
FB15k-237				
GMT	**0.488**	**0.394**	**0.505**	**0.629**
w/o Semantics	0.464	0.368	0.479	0.602
w/o Fusion	0.425	0.331	0.450	0.584

Triple Classification. We further evaluate GMT on triple classification over UMLS, CoDeX-S, and FB15K-237N, using a task-specific instruction template $\mathcal{I}_{GMT}$ that differs from link prediction (Appendix A). As shown in Table 3, GMT delivers consistent state-of-the-art performance across the three benchmarks. In particular, GMT achieves the best

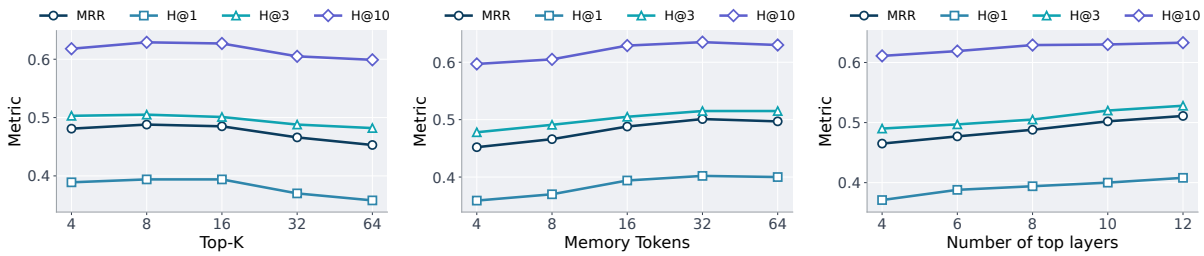

Figure 3: Impact of Key Hyperparameters on GMT Performance

overall results on **UMLS** (Acc **94.55**, F1 **93.76**), ranks first on **CoDeX-S** (Acc **89.01**, F1 **87.61**), and also leads on the more challenging **FB15K-237N** (Acc **84.10**, F1 **87.02**). These results demonstrate that GMT generalizes beyond link prediction and remains robust across datasets with diverse relational patterns and label distributions.

4.3 Ablation study: RQ2

To assess the contribution of each component, we compare GMT with two variants: **w/o Semantics**, which replaces SGM with conventional pre-trained KGE embeddings, and **w/o Fusion**, which removes the Graph-as-Memory Cross-Attention Fusion Module and injects memory by simple prefix concatenation.

Table 4 shows that both components are necessary. Removing SGM (**w/o Semantics**) consistently degrades performance on WN18RR and FB15k-237, indicating that relation-centric, context-aware semantics provide a stronger query-specific signal than static KG embeddings. More importantly, replacing cross-attention fusion with prefix injection (**w/o Fusion**) causes the largest drop (e.g., FB15k-237 MRR: 0.488 to 0.425), confirming that deep, token-wise memory retrieval is crucial and cannot be substituted by shallow concatenation.

Scoring Function in Stage-1 Pre-training. We further study the scoring function $F_{\mathcal{G}}(h, r, t)$ used to pre-train SGM. As reported in Table 5, a RotatE-style scoring function performs best on FB15k-237 (MRR **0.471**), while alternatives such as TransE/DistMult/ComplEx or an MLP lead to lower accuracy. We therefore adopt RotatE by default in our implementation. The superiority of RotatE can likely be attributed to its inherent ability to model complex relational patterns.

Hyperparameter Sensitivity. We analyze the sensitivity of GMT to three key hyperparameters

Table 5: The scoring function for the link prediction task.

Model	MRR	Hits@1	Hits@3	Hits@10
FB15k-237				
GMT(w/ RotaE)	**0.471**	**0.380**	**0.488**	**0.623**
w TransE	0.459	0.375	0.483	0.610
w DistMult	0.454	0.366	0.481	0.615
w ComplexE	0.455	0.368	0.481	0.613
w MLP	0.447	0.358	0.475	0.606

Table 6: Impact of LLM Relational Knowledge Enhancement.

Model	MRR	Hits@1	Hits@3	Hits@10
FB15k-237				
GMT	**0.488**	**0.394**	**0.505**	**0.629**
w/o Enhancement	0.454	0.370	0.475	0.601

in Figure 3: (i) Top-K neighbor selection in SGM, (ii) the number of graph memory tokens m, and (iii) the number of top Transformer layers equipped with memory cross-attention. **Top-K** shows a clear sweet spot at a moderate neighborhood size (around K=8–16); too small loses context, while too large introduces noise and degrades MRR/Hits@1. Increasing the number of **memory tokens** improves performance and saturates around m=32, after which gains become marginal. Finally, injecting memory into more **top Transformer layers** consistently boosts all metrics, with the best results at the largest setting, supporting the benefit of deep, multi-layer memory retrieval.

4.4 Analysis of Semantics: RQ3

A key design in SGM is relation knowledge enhancement: we use an LLM to generate relation definitions and embed them as semantic vectors for Top-K neighbor selection. We evaluate **GMT w/o Enhancement**, which encodes only raw relation names with the same Sentence-BERT.

GMT w/o Enhancement. As shown in Table 6, removing enhancement consistently degrades performance on FB15k-237 (MRR drops from 0.488

Table 7: Case study: comparison of the top-5 neighbor relations for a query, before and after applying Knowledge Enhancement. Abbreviations stand for full relation names, e.g., **Gov Position (Title)** for `.../government_position_held/basic_title`. The Change column highlights the significant re-ranking based on semantic understanding.

Without Knowledge Enhancement (Lexical Matching)		**With Knowledge Enhancement (Semantic Relevance)**		
Neighbor Relation	**Score**	**Neighbor Relation**	**Score**	**Change**
Gov Position (Title)	0.859	Gov Position (Title)	0.611	—
Gov Position (Sessions)	0.828	Gov Position (Sessions)	0.460	—
Person (Profession)	0.250	**Person (Nationality)**	0.385	↑ **Up (New)**
Person (Places Lived)	0.206	Person (Employment)	0.377	↑ **Up**
Person (Employment)	0.184	**Person (Profession)**	0.338	↓ **Down**

Table 8: Robustness analysis across different LLM generators. Best results are in **bold**, second best are underlined.

Source	**LLM Generator**	**WN18RR**				**FB15k-237**				**FB15k-237N**			
		MRR	**H@1**	**H@3**	**H@10**	**MRR**	**H@1**	**H@3**	**H@10**	**Acc**	**P**	**R**	**F1**
Closed	GPT-4o (Ours)	**0.621**	0.569	**0.667**	**0.703**	0.488	**0.394**	0.505	0.629	**84.10**	**83.14**	91.27	**87.02**
	Claude-3.5-Sonnet	0.617	**0.572**	0.660	0.702	**0.490**	0.390	**0.511**	**0.635**	84.05	83.11	**91.35**	86.98
	Gemini-1.5-Pro	0.612	0.565	0.656	0.697	0.485	0.390	0.502	0.626	83.95	82.98	91.10	86.85
Open	Qwen3-32B-Instruct	0.619	0.567	0.664	0.700	0.486	0.391	0.503	0.627	83.92	82.95	91.08	86.82
	Qwen3-8B-Instruct	0.613	0.561	0.658	0.694	0.481	0.386	0.497	0.620	83.52	82.55	90.68	86.42
	Llama-3-8B-Instruct	0.610	0.558	0.655	0.690	0.478	0.383	0.494	0.616	83.25	82.26	90.35	86.11

to 0.454; Hits@1 drops from 0.394 to 0.370), verifying that relation names alone provide insufficient semantics for reliable neighbor filtering.

Robustness Across LLM Generators. We further investigate whether GMT relies on specific proprietary models for definition generation. Table 8 compares performance across various closed-source and open-source LLMs. Results show that GMT maintains consistent performance, with even lightweight open-source models yielding negligible drops. This confirms that GMT benefits from the explicit semantic guidance rather than the capabilities of a specific model.

Case Study. To highlight the role of relational semantics, we examine the query (Barack Obama, /government/politician/government_positions_held..., ?), which describes the geographical or administrative scope of a political position. As shown in Table 7, knowledge enhancement substantially re-orders the top-5 neighbors, shifting retrieval from surface lexical matching to semantic relevance. Without enhancement, the ranking is dominated by relations with shared string prefixes (e.g., `Gov Position (Title)` and `Gov Position (Sessions)`), which captures naming overlap rather than the intended semantics. With knowledge enhancement, neighbor retrieval becomes semantics-driven. In particular, `Person (Nationality)` appears and ranks highly, aligning with the query's notion of jurisdiction rather than string overlap. Meanwhile, semantically related relations (e.g., `Person (Employment)`) are promoted and less relevant ones (e.g., `Person (Places Lived)`) are suppressed, yielding a cleaner evidence set. Overall, the case study shows that SGM goes beyond lexical matching and builds a more semantically coherent graph memory, providing more reliable signals for LLM reasoning.

5 Conclusion

In this paper, we propose Graph-as-Memory Tuning (GMT), a memory-centric framework that integrates knowledge graphs with LLMs beyond shallow prefix fusion. GMT builds query-specific graph memory tokens using a Semantic Graph Module (SGM) with knowledge-enhanced relation semantics, and injects them into multiple Transformer layers via a Graph-as-Memory Cross-Attention Fusion Module, enabling token-wise retrieval of graph evidence during generation. We further adopt LoRA on memory cross-attention for parameter-efficient adaptation with a frozen base LLM.

Experiments on link prediction and triple classification benchmarks show that GMT achieves state-of-the-art or highly competitive performance, and ablations confirm the effectiveness of both SGM and cross-attention fusion, while additional analyses show GMT is robust to different LLMs used for relation knowledge enhancement.

Limitations

Despite its effectiveness, GMT still has several method-specific limitations. First, its semantic graph construction relies on an offline knowledge-enhancement step that uses external LLM-generated relation definitions and precomputed sentence embeddings, which introduces external dependency and may make the relation semantics less adaptive to highly context-dependent queries.

Second, GMT builds graph memory from query-local neighborhoods with Top-K semantic filtering, so it may overlook useful long-range evidence, and its performance remains sensitive to the neighborhood size and memory configuration.

Third, GMT compresses variable-size subgraph evidence into a fixed number of memory tokens and aligns it with a frozen base LLM through LoRA only on cross-attention, which may limit its ability to preserve fine-grained graph information.

In future work, we plan to develop a more adaptive graph memory mechanism beyond the current query-local neighborhood construction and fixed-size memory tokenization, so that GMT can capture longer-range structural dependencies and support more complex multi-hop reasoning.

References

Ralph Abboud, Ismail Ilkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. BoxE: A box embedding model for knowledge base completion. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020)*. Curran Associates Inc.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Zongsheng Cao, Jing Li, Zigan Wang, and Jinliang Li. 2024. Diffusione: Reasoning on knowledge graphs via diffusion-based graph neural networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 222–230, New York, NY, USA. Association for Computing Machinery.

Heng Chang, Jiangnan Ye, Alejo Lopez-Avila, Jinhua Du, and Jia Li. 2024. Path-based explanation for knowledge graph completion. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 231–242.

Zefeng Chen, Wensheng Gan, Jiayang Wu, Kaixia Hu, and Hong Lin. 2025. Data scarcity in recommendation systems: A survey. *ACM Transactions on Recommender Systems*, 3(3):1–31.

Zexin Chen, Wenjie Peng, Zixuan Zhang, and Laks VS Lakshmanan. 2024. ARR: A continuous-time dynamic KG embedding model for asynchronous and recurring relations. In *Proceedings of the 2024 International Conference on Management of Data (SIGMOD/PODS)*, pages 1–26.

Ziqiang Cui, Yunpeng Weng, Xing Tang, Fuyuan Lyu, Dugang Liu, Xiuqiang He, and Chen Ma. 2025. Comprehending knowledge graphs with large language models for recommender systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, page 1229–1239, New York, NY, USA. Association for Computing Machinery.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 3814–3820.

Jingtao Guo, Chunxia Zhang, Lingxi Li, Xiaojun Xue, and Zhendong Niu. 2024a. A unified joint approach with topological context learning and rule augmentation for knowledge graph completion. In *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand. Association for Computational Linguistics.

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. 2024b. Mkgl: Mastery of a three-word language. In *Advances in Neural Information Processing Systems*, volume 37, pages 140509–140534. Curran Associates, Inc.

Yuxin Jiang, Zixuan Zhang, Zexin Chen, Yun Tong, Zonghan Wu, Nitesh V Chawla, and Laks VS Lakshmanan. 2024. KG-FIT: A framework of few-shot inductive reasoning on knowledge graphs. In *Proceedings of the ACM Web Conference 2024 (WWW)*, pages 111–122.

Binhang Li, Yizhou Liu, Zixuan Zhang, Rui Chen, and Laks VS Lakshmanan. 2023. CSProm-KG: A cross-modal self-supervised prompting for knowledge graph completion under cold-start setting. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11355–11368.

Qian Li, Zhuo Chen, Cheng Ji, Shiqi Jiang, and Jianxin Li. 2024. Llm-based multi-level knowledge generation for few-shot knowledge graph completion. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24.

Siyuan Li, Ruitong Liu, Te Sun, Yan Wen, Ruihao Zhou, Jingyi Kang, and Yunjia Wu. 2025a. Context-driven knowledge graph completion with semantic-aware

relational message passing. In *International Conference on Neural Information Processing*, pages 567–580. Springer.

Siyuan Li, Ruitong Liu, Yan Wen, Te Sun, Andi Zhang, Yanbiao Ma, and Xiaoshuai Hao. 2025b. Flow-modulated scoring for semantic-aware knowledge graph completion. *arXiv preprint arXiv:2506.23137*.

Qika Lin, Tianzhe Zhao, Kai He, Zhen Peng, Fangzhi Xu, Ling Huang, Jingying Ma, and Mengling Feng. 2025. Self-supervised quantized representation for seamlessly integrating knowledge graphs with large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13587–13602, Vienna, Austria. Association for Computational Linguistics.

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 20981–20995.

Yuyin Lu, Hegang Chen, Yanghui Rao, Jianxing Yu, Wen Hua, and Qing Li. 2025. An efficient fuzzy system for complex query answering on knowledge graphs. *IEEE Transactions on Knowledge and Data Engineering*, 37(9):4962–4976.

Kangyang Luo, Yuzhuo Bai, Cheng Gao, Shuzheng Si, Zhu Liu, Yingli Shen, Zhitong Wang, Cunliang Kong, Wenhao Li, Yufei Huang, Ye Tian, Xuantang Xiong, Lei Han, and Maosong Sun. 2025. GLTW: Joint improved graph transformer and LLM via three-word language for knowledge graph completion. In *Findings of the Association for Computational Linguistics*, Vienna, Austria. Association for Computational Linguistics.

Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. 2022. Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3570–3581. Association for Computational Linguistics.

Reham Omar, Ishika Dhall, Panos Kalnis, and Essam Mansour. 2023. A universal question-answering platform for knowledge graphs. *Proceedings of the ACM on Management of Data*, 1(1):1–25.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Cevaert. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 2071–2080.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*.

Hongwei Wang, Hongyu Ren, and Jure Leskovec. 2021. Relational message passing for knowledge graph completion. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 1697–1707, New York, NY, USA. Association for Computing Machinery.

Linyi Wang, Wen Zhao, Zhipeng Wei, and Jing 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.

Yanbin Wei, Qiushi Huang, Yu Zhang, and James Kwok. 2023. KICGPT: Large language model with knowledge in context for knowledge graph completion. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8667–8683, Singapore. Association for Computational Linguistics.

Hongcai Xu, Junpeng Bao, and Wenbo Liu. 2023. Double-branch multi-attention based graph neural network for knowledge graph completion. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 15257–15271.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations (ICLR)*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *ArXiv*, abs/1909.03193.

Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. 2024a. Making large language models perform better in knowledge graph completion. In *Proceedings of the 32nd ACM international conference on multimedia*, pages 233–242.

Zequn Zhang, Jian Cai, Yanzhi Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 3065–3072. AAAI Press.

Zixuan Zhang, Ying Shao, Dmitri Kalashnikov, Binhang Li, Yizhou Liu, and Laks VS Lakshmanan. 2024b. Adaprop: A plug-and-play approach for propagating large-scale updates in knowledge graphs. *Proceedings of the VLDB Endowment*, 17(5):1016–1029.

Jia Zhu and Pasquale De Meo. 2025. Exploring large language models for knowledge graph completion with auto-prompting. In *2025 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Zhaocheng Zhu, Zuo Bai Zhang, Louis-Pascal Xhonneux, and Jian Tang. 2021. Neural bellman-ford networks: A general graph neural network framework for link prediction. In *Advances in Neural Information Processing Systems*, volume 34, pages 29476–29490.

Appendix

A Prompt Template

Prompt: Knowledge Enhancement

Task: Generate a canonical and descriptive definition for a knowledge graph relation by following the provided examples. The definition must clearly explain the semantic relationship between a head entity and a tail entity in a neutral, factual sentence.

Relation Name: located_in
Definition: The head entity is a smaller geographical, physical, or conceptual area that is situated within the larger area of the tail entity.

Relation Name: relation_name
Definition: output

Prompt: Instruction Template for Link Prediction

Instruction: This is a knowledge graph completion task. Your goal is to predict the tail entity for an incomplete query triplet.

Problem Definition:

- **Query:** (head_entity, relation, ?)
- **Candidate Entities:** {candidate_1, ..., candidate_20}

Response Specification: Analyze the provided candidates and return a ranked list of the top three most likely answers, from most to least probable.

Prompt: Instruction Template for Triple Classification

Instruction: This is a triple classification task in knowledge graph. Your goal is to determine the correctness of the given triple.

Input:

- The triple is: (head_entity, relation, tail_entity)

Output: Please response True or False.