RISE: ENHANCING KNOWLEDGE GRAPH COMPLETION WITH LARGE LANGUAGE MODELS VIA ROLLBACK-GUIDED INFLUENTIAL SUBGRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge graph completion (KGC) with large language models (LLMs) can benefit from query-specific structural context, yet two challenges remain: identifying structures that are truly relevant to a particular prediction and integrating them into LLMs without verbose graph descriptions. To address these challenges, we propose RISE [1], a Rollback-guided Influential Subgraph-Enhanced framework for KGC with LLMs. RISE traces triple-level knowledge graph embedding (KGE) optimization updates and uses rollback-induced prediction changes to prioritize query-candidate paths, which are aggregated into compact influential subgraphs. These subgraphs are then encoded and fused with candidate representations to form latent structural prefixes for LLM-based reranking, complementing global KGE representations and textual semantics with query-specific local structure. Experiments on four datasets show that RISE achieves the best performance on most evaluation metrics among the compared methods, while further analyses demonstrate the benefits of prediction-relevant subgraph selection and compact latent conditioning.

1 INTRODUCTION

Knowledge graphs (KGs) represent factual knowledge as triples (h, r, t), where h, r, and t denote the head entity, relation, and tail entity, respectively. As structured knowledge repositories, KGs support a wide range of downstream applications (Cao et al., 2024; Wang et al., 2026). However, real-world KGs are often incomplete, motivating knowledge graph completion (KGC) to infer missing facts, typically by predicting the missing entity in queries such as $(h, r, ?)$ or $(?, r, t)$. Some approaches rely on entity representations or fact-level information (Sun et al., 2019; Wei et al., 2023), while others additionally retrieve or construct query-specific paths and subgraphs to provide structural context (Das et al., 2018; Xiao et al., 2025).

Traditional embedding models primarily capture global relational regularities through entity and relation representations (Sun et al., 2019; Cheng et al., 2024), while recent large language model (LLM)-based approaches additionally exploit textual semantics (Zhang et al., 2024; Luo et al., 2026). As illustrated in Figure 1(a-b), these two paradigms may still prioritize entities that are strongly associated with the query but are not correct targets: embedding-based methods can favor geometrically related entities, while LLM-based methods may rely on misleading semantic associations, as exemplified by the association between *Iron Man* and its actor. Instead, Figure 1(c) shows that query-specific substructures expose local relational connections more relevant to the target prediction, providing complementary structural context for distinguishing plausible targets. Nevertheless, not all connecting structures are equally useful; selection based on connectivity, local plausibility, or heuristic rules may still retain weakly relevant substructures.

This observation highlights two key challenges. *First, selecting informative query-specific substructures is difficult.* The number of candidate paths and subgraphs grows rapidly as graph connectivity increases, while structurally plausible substructures may still be redundant or weakly informative for target predictions. Effective retrieval should therefore prioritize prediction-relevant substructures. *Second, integrating such local structures into LLMs remains challenging.* Verbalizing graph

[1] Anonymous code is available at `https://anonymous.4open.science/r/RISE-1DAC`

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

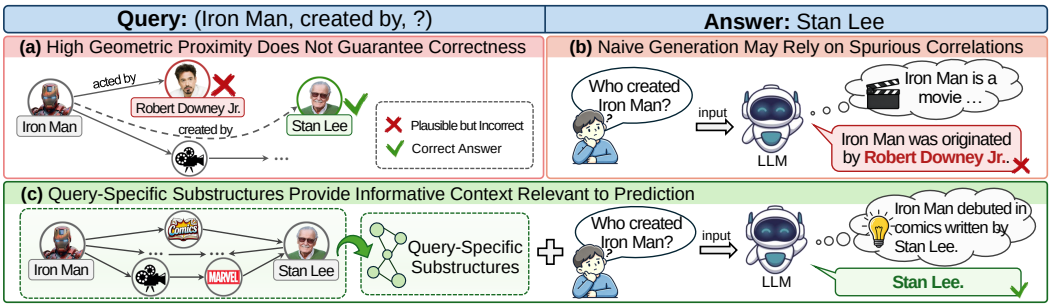


Figure 1: Motivation for query-specific substructures for KGC. (a) Geometric proximity and (b) semantic associations may favor incorrect entities, whereas (c) query-specific substructures provide complementary relational context for target prediction.

structures (Wei et al., 2023; Ye et al., 2024) can produce lengthy contexts and require LLMs to infer relational dependencies from serialized facts, whereas soft prompts based primarily on entity-level representations (Geng et al., 2025; Luo et al., 2026) may not explicitly capture query-specific structural dependencies. A compact encoding mechanism is thus needed to preserve useful structural information while avoiding excessive input length.

To realize such prediction-aware substructure selection, we exploit KGE optimization trajectories (Lawrence et al., 2021), i.e., the triple-associated embedding updates accumulated during KGE training. By rolling back the updates associated with a substructure, we measure the resulting change in a candidate prediction, providing a prediction-sensitive signal beyond connectivity or structural plausibility. Building on this intuition, we propose RISE, a **R**ollback-guided **I**nfluential **S**ubgraph-**E**nhanced framework for KGC with LLMs. RISE consists of three stages. First, *Trajectory-Aware KGE Modeling* learns KGE representations, retrieves candidates, and records triple-associated trajectories. Second, *Influential Subgraph Discovery* combines structural and trajectory-based preselection with rollback scoring to construct candidate-specific influential subgraphs. Finally, to address the integration challenge, *Latent Structural Conditioning* encodes and fuses these subgraphs into compact soft prefixes for LLM-based reranking. In summary, our main contributions are as follows:

- We propose RISE, a rollback-guided framework that augments LLM-based KGC with query-specific influential subgraphs derived from KGE optimization trajectories, complementing global structural representations with local context prioritized according to KGE prediction sensitivity.
- We develop a rollback-guided subgraph discovery mechanism that prioritizes multi-hop paths according to rollback-induced changes in KGE predictions, together with a compact latent conditioning architecture that incorporates the resulting structural context into LLM-based candidate reranking.
- Experiments on four datasets demonstrate that RISE achieves the best performance on most evaluation metrics among the compared methods. Further analyses show that query-specific subgraph augmentation provides complementary gains beyond KGE-informed candidate representations.

2 Related Work

Knowledge Graph Completion. We broadly categorize recent KGC methods by whether they retrieve or construct query-specific KG substructures for prediction. **(1) Query-Specific Substructure-Agnostic Methods:** Classical KGE models, such as TransE (Bordes et al., 2013) and RotatE (Sun et al., 2019), capture relational regularities through latent entity and relation representations. To incorporate textual semantics, C-LMKE (Wang et al., 2022) and ST-KGLP (Luo et al., 2026) align structural and textual representations, while another line of work conditions language models through structural prompts or tokens (Chen et al., 2023; Zhang et al., 2024; Geng et al., 2025; Lin et al., 2025). Beyond representation-based integration, KICGPT (Wei et al., 2023) retrieves relevant facts for in-context reasoning. Despite these different forms of structural integration, these methods primarily rely on entity-level representations or fact-level context without explicitly modeling query-specific substructures. Consequently, query-specific topological dependencies useful for distinguishing plausible targets may not be fully captured. **(2) Query-Specific Substructure-Aware Methods:** These meth-

ods retrieve or construct query-specific substructures as structural context for prediction. Path-based approaches capture relational dependencies: MINERVA (Das et al., 2018) learns query-conditioned traversal policies, while SRP-KGC (Hou et al., 2025) models soft reasoning paths when explicit paths are unavailable. Neighborhood-based approaches incorporate local structures, with InstructGLM (Ye et al., 2024) verbalizing multi-scale neighborhoods and RAA-KGC (Yuan et al., 2025) selecting relation-aware one-hop anchors. At the subgraph level, FtG (Liu et al., 2025a) and DrKGC (Xiao et al., 2025) combine subgraph selection with graph-to-language adaptation, whereas SAT (Liu et al., 2025b) and FuST-KGC (Han et al., 2026) emphasize structure-text alignment and joint encoding, respectively. However, connectivity, structural plausibility, or heuristic compatibility alone does not ensure that selected substructures provide informative context for distinguishing plausible targets. Retaining irrelevant connections or omitting useful dependencies may therefore limit the utility of the resulting substructures for candidate ranking. In contrast, our RISE combines structural and trajectory cues with rollback sensitivity to select substructures more relevant to the candidate prediction.

Explainability in KGC. Explainability in KGC seeks to identify the factors underlying model predictions (Rajabi & Etminni, 2024). GR (Lawrence et al., 2021) traces training updates and uses gradient rollback to estimate the influence of individual training facts, while Kelpie (Rossi et al., 2022) extracts necessary and sufficient fact-based explanations. Power-Link (Chang et al., 2024) and KGExplainer (Ma et al., 2025) organize explanations into paths and connected subgraphs, respectively. GradPath (Di Mauro et al., 2024) further uses model gradients to construct path-based explanations for KGC. Unlike these explainability-oriented methods, our RISE does not interpret rollback as fine-grained attribution over individual training facts or path components. Instead, we first preselect paths according to their structural plausibility and optimization-trace magnitude, and then use rollback-based prediction sensitivity for candidate-specific prioritization. The selected paths are subsequently encoded as structural context for LLM-based candidate reranking.

3 METHODOLOGY

We study KGC over a KG $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, with entity and relation vocabularies shared across training and evaluation. Here, $\mathcal{E}$ is the entity set, $\mathcal{R}$ the relation set, and $\mathcal{T} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$ the set of factual triples. In this work, we focus on tail-entity prediction: given an incomplete query $(h, r, ?)$, the goal is to identify the missing tail entity t. The formulation can be symmetrically extended to head-entity prediction. As shown in Figure 2, our RISE contains three stages: Trajectory-Aware KGE Modeling, Influential Subgraph Discovery, and Latent Structural Conditioning.

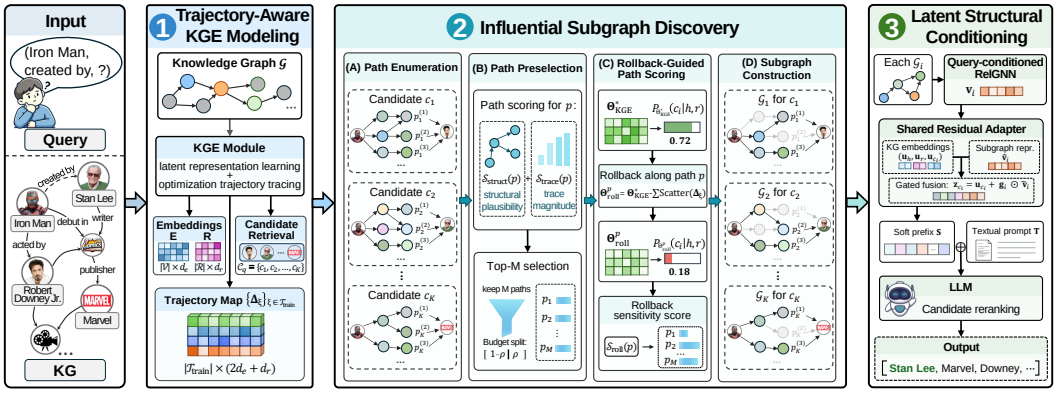


Figure 2: Overview of RISE: (1) Trajectory-Aware KGE Modeling for structural representation learning, candidate retrieval, and trajectory tracing; (2) Influential Subgraph Discovery via rollback-guided path scoring; and (3) Latent Structural Conditioning for LLM-based candidate reranking.

3.1 TRAJECTORY-AWARE KGE MODELING

This stage centers on two components: **candidate retrieval**, which narrows the search space using optimized KGE embeddings, and **trajectory tracing**, which records triple-level updates for subsequent rollback-based path evaluation.

Latent Representation Learning. We employ a lightweight KGE to capture structural patterns, parameterized by the entity and relation embedding matrices $\Theta_{\text{KGE}} = \{\mathbf{E} \in \mathbb{R}^{|\mathcal{E}|\times d_e}, \mathbf{R} \in \mathbb{R}^{|\mathcal{R}|\times d_r}\}$. These embeddings are optimized over the training triples $\mathcal{T}_{\text{train}} \subseteq \mathcal{T}$ via a self-adversarial negative-sampling loss $\mathcal{L}_{\text{KGE}}$:

$$\Theta^*_{\text{KGE}} = \arg\min_{\Theta_{\text{KGE}}} \mathcal{L}_{\text{KGE}}(\mathcal{T}_{\text{train}}; \Theta_{\text{KGE}}). \tag{1}$$

We adopt shallow KGE models with independently parameterized entity and relation embeddings, which facilitate localized trajectory tracking and subsequent KGE-side rollback sensitivity analysis.

Candidate Retrieval. We use the KGE to retrieve the top K candidates for each query $q = (h, r, ?)$, avoiding exhaustive LLM reranking over $|\mathcal{E}|$ entities:

$$\mathcal{C}_q = \text{argTopK}_{t\in\mathcal{E}}\ \mathcal{F}_{\text{score}}(\mathbf{E}[h],\ \mathbf{R}[r],\ \mathbf{E}[t];\ \Theta^*_{\text{KGE}}), \tag{2}$$

where $\mathcal{F}_{\text{score}}$ is the scoring function of KGEs (e.g., $\mathcal{F}_{\text{score}} = -\|\mathbf{E}[h] \circ \mathbf{R}[r] - \mathbf{E}[t]\|$ for RotatE). In practice, $K = 30$ (cf. Figure 5), substantially reducing the downstream candidate space ($K \ll |\mathcal{E}|$).

Optimization Trajectory Tracing. Final KGE embeddings combine information from training facts without explicitly retaining their associated updates. Inspired by (Lawrence et al., 2021), we record triple-associated embedding changes, linking facts to the update directions used for subsequent prediction-sensitive path selection. For a training triple $\xi = (h, r, t) \in \mathcal{T}_{\text{train}}$ and each component $x \in \{h, r, t\}$, we accumulate its parameter updates over the optimization steps involving ξ:

$$\Delta_{\xi,x} = \sum\nolimits_{\tau\in\mathcal{U}(\xi)} \left(\mathbf{w}_x^{(\tau,+)} - \mathbf{w}_x^{(\tau,-)}\right), \qquad x \in \{h, r, t\}, \tag{3}$$

where $\mathbf{w}_x^{(\tau,-)}$ and $\mathbf{w}_x^{(\tau,+)}$ denote the embedding of x immediately before and after the corresponding optimization step, respectively, and $\mathcal{U}(\xi)$ is the set of optimization steps regarding ξ. We concatenate these component-wise updates into a triple-level record $\Delta_\xi = [\Delta_{\xi,h}; \Delta_{\xi,r}; \Delta_{\xi,t}]$, which stores only the embedding updates associated with the head, relation, and tail of ξ. Table 7 provides a concrete example of such a trajectory record, while Table 8 reports the corresponding storage overhead. The trajectory map is constructed only over training triples, so its size scales with $|\mathcal{T}_{\text{train}}|$ rather than the full KG triple set $|\mathcal{T}|$.

3.2 Influential Subgraph Discovery

To identify candidate-relevant structures, we use **dual-criterion preselection** based on structural plausibility and optimization-trace magnitude, followed by **rollback-guided prioritization** according to the induced drop in candidate KGE log-probability. The top-ranked paths are then merged into candidate-specific subgraphs for structural encoding.

Multi-hop Path Preselection. For each candidate $c_i \in \mathcal{C}_q$, we enumerate multi-hop paths $p = (\xi_1, \ldots, \xi_\ell)$ connecting the query head h to c_i, where $\ell \le L_{\max}$. Paths may traverse KG edges in either direction, while each triple $\xi_j = (h_j, r_j, t_j)$ preserves its original head-relation-tail orientation. We characterize each path from two complementary perspectives: structural plausibility and optimization-trace magnitude,

$$\mathcal{S}_{\text{struct}}(p) = \frac{1}{\ell}\sum\nolimits_{i=1}^{\ell} P_{\Theta^*_{\text{KGE}}}(t_i \mid h_i, r_i), \qquad \mathcal{S}_{\text{trace}}(p) = \frac{1}{\ell}\sum\nolimits_{i=1}^{\ell} \|\Delta_{\xi_i}\|_2, \tag{4}$$

where $P_{\Theta^*_{\text{KGE}}}(t_i \mid h_i, r_i)$ denotes the KGE score normalized over all entities, and Δ_{ξ_i} denotes the accumulated parameter updates associated with training triple ξ_i. For each query-candidate pair, M denotes the path preselection budget, limiting the candidate pool to at most M paths for subsequent evaluation. Approximately $(1-\rho)M$ slots are filled by the highest-$\mathcal{S}_{\text{struct}}$ paths, while the remaining ρM slots are reserved for high-$\mathcal{S}_{\text{trace}}$ paths. The sensitivity to M is analyzed in Figure 9.

Rollback-Based Subgraph Construction. For each path $p = (\xi_1, \ldots, \xi_\ell)$ with update records, we roll back the KGE parameters Θ^*_{KGE} by subtracting the recorded updates of all triples along the path:

$$\Theta^p_{\text{roll}} = \text{Rollback}(\Theta^*_{\text{KGE}}, p). \tag{5}$$

Specifically, for each triple $\xi = (h, r, t) \in p$, we subtract $\Delta_{\xi,h}$, $\Delta_{\xi,r}$, and $\Delta_{\xi,t}$ from the corresponding embedding rows of h, r, and t, respectively (e.g., $\mathbf{E}[h] \leftarrow \mathbf{E}[h] - \Delta_{\xi,h}$). If multiple triples along the path involve the same entity or relation, their updates are accumulated before subtraction.

We then quantify the rollback sensitivity of path p by the length-normalized reduction in the target log-probability:

$$\mathcal{S}_{\mathrm{roll}}(p) = \frac{1}{\ell}\left[\log P_{\Theta^*_{\mathrm{KGE}}}(c_i \mid h, r) - \log P_{\Theta^p_{\mathrm{roll}}}(c_i \mid h, r)\right]. \tag{6}$$

A larger $\mathcal{S}_{\mathrm{roll}}(p)$ indicates greater rollback sensitivity for candidate c_i. From at most M preselected paths, we retain the top N_{pth} positively scored ones to form an influential subgraph $\mathcal{G}_i$. Here, M controls preselection breadth, while N_{pth} controls the final subgraph size.

We emphasize that $\mathcal{S}_{\mathrm{roll}}$ measures path-level rollback sensitivity rather than fine-grained attribution to individual components. In shallow KGE models, updates involving the query head or candidate can influence its prediction score more directly than updates on entities or relations in the middle of the path. We therefore use $\mathcal{S}_{\mathrm{roll}}$ only for path prioritization, with further analysis in Appendix F.

3.3 Latent Structural Conditioning

To integrate local structures without verbose graph descriptions, we use **latent structural encoding** to preserve query-specific multi-hop information and **compact conditioning** to compress it into only $K + 2$ soft tokens. Specifically, the selected subgraphs are encoded, projected to the LLM hidden space, and fused with candidate representations for reranking.

Structural Prefix Encoding. We encode each query and its candidate-specific subgraphs into compact structural representations. Let d_e and d_{lm} denote the KGE and the LLM hidden dimension, respectively. We first obtain a normalized query representation $\mathbf{q} = Q([\mathbf{e}_h; \mathbf{e}_r]) \in \mathbb{R}^{d_e}$, where Q is an MLP over the KGE embeddings of the query head h and relation r.

For each non-empty candidate-specific subgraph $\mathcal{G}_i$, a query-conditioned relational graph encoder captures its local structure. Node representations incorporate structural-role and directed-degree features, while message passing uses relation embeddings, edge directions, and path positions. Conditioned on $\mathbf{q}$, the encoder produces $\mathbf{X}_i = \mathrm{RelGNN}(\mathcal{G}_i, \mathbf{q})$, which is aggregated as:

$$\mathbf{v}_i = \mathrm{LayerNorm}\left(\mathrm{MLP}_{\mathrm{out}}\left([\mathbf{x}_{c_i}; \mathrm{AttnPool}(\mathbf{X}_i, \mathbf{q}); \mathbf{q}]\right)\right) \in \mathbb{R}^{d_e}, \tag{7}$$

where $\mathbf{x}_{c_i}$ is the encoded candidate node and $\mathrm{AttnPool}$ performs query-conditioned attention over the subgraph nodes. When $\mathcal{G}_i$ is empty, the candidate is retained and $\mathbf{v}_i$ is replaced with a learnable null-subgraph representation $\mathbf{v}_\emptyset \in \mathbb{R}^{d_e}$, which is processed by the same shared adapter and gated fusion as non-empty subgraphs.

We then employ a shared residual adapter $A : \mathbb{R}^{d_e} \to \mathbb{R}^{d_{lm}}$ to project both global KGE embeddings and query-specific subgraph representations into the LLM hidden space:

$$\begin{gathered}\mathbf{u}_x = A(\mathbf{e}_x), \quad x \in \{h, r, c_1, \ldots, c_K\}; \qquad \tilde{\mathbf{v}}_i = A(\mathbf{v}_i), \quad i \in \{1, \ldots, K\}, \\ A(\mathbf{a}) = \mathrm{LayerNorm}(\boldsymbol{\alpha}(\mathbf{a}) \odot \mathrm{MLP}_A(\mathbf{a}) + [1 - \boldsymbol{\alpha}(\mathbf{a})] \odot \mathrm{Linear}_A(\mathbf{a})), \quad \mathbf{a} \in \{\mathbf{e}_x, \mathbf{v}_i\}.\end{gathered} \tag{8}$$

Here, $\boldsymbol{\alpha}(\mathbf{a}) = \sigma(\mathrm{Linear}_\alpha \mathbf{a}))$ is an element-wise gating vector that balances the nonlinear and linear projection branches. To augment each candidate c_i with its query-specific structural context $\tilde{\mathbf{v}}_i$, we apply gated residual fusion:

$$\mathbf{z}_{c_i} = \mathbf{u}_{c_i} + \mathbf{g}_i \odot \tilde{\mathbf{v}}_i, \qquad \text{where } \mathbf{g}_i = \sigma(\mathrm{Linear}_g([\mathbf{u}_{c_i}; \tilde{\mathbf{v}}_i])). \tag{9}$$

We then assemble structural tokens for the query head, relation, and each candidate into the soft prefix

$$\mathbf{S} = [\mathbf{u}_h; \mathbf{u}_r; \mathbf{z}_{c_1}; \ldots; \mathbf{z}_{c_K}] \in \mathbb{R}^{(K+2)\times d_{lm}}. \tag{10}$$

The prefix $\mathbf{S}$ is prepended to the token embeddings of the textual query and candidate-list prompt, allowing the LLM to jointly exploit textual semantics and the learned query-specific structural representations for candidate reranking.

Textual Context Construction. We construct a concise textual prompt $\mathcal{P}$ containing only the task instruction, query semantics, and candidate entities. For a query $q = (h, r, ?)$ and candidate set $\mathcal{C}_q$, $\mathcal{P}$ concatenates a task instruction $\mathcal{D}_{\text{task}}$, a query text $\mathcal{D}_q$, and a candidate list $\mathcal{D}_{\mathcal{C}_q}$ (see Table 12 for the exact template). The prompt is mapped to token embeddings as

$$\mathbf{T} = \mathrm{Embed}_{\mathrm{LLM}}(\mathcal{P}) \in \mathbb{R}^{N_{\text{token}} \times d_{lm}}. \tag{11}$$

We deliberately exclude textualized paths or subgraphs from $\mathcal{P}$, leaving query-specific structural information to be supplied exclusively through the latent structural prefix $\mathbf{S}$.

Autoregressive Candidate Reranking. We concatenate the latent structural prefix $\mathbf{S}$ with the textual token embeddings $\mathbf{T}$ and feed the joint sequence to the LLM. Conditioned on this input, the model autoregressively generates a reordered sequence over the retrieved candidate set:

$$P_{\text{LLM}}(Y \mid [\mathbf{S}; \mathbf{T}]) = \prod_{\tau=1}^{N_{\text{out}}} P_{\text{LLM}}\big(y_\tau \mid y_{<\tau}, [\mathbf{S}; \mathbf{T}]\big), \tag{12}$$

where $Y = (y_1, \ldots, y_{N_{\text{out}}})$ denotes the generated candidate-ranking token sequence and $y_{<\tau}$ the preceding tokens. The output is then parsed into an ordering of the original candidate entities, from which the final ranking is obtained for KGC evaluation.

3.4 Optimization and Complexity Analysis

Model Optimization. Algorithm 1 summarizes the optimization procedure of RISE. We jointly optimize the model with an autoregressive candidate-reranking objective and an auxiliary query-subgraph contrastive objective:

$$\mathcal{L}_{\text{total}} = \lambda \widetilde{\mathcal{L}}_{\text{gen}} + (1 - \lambda) \widetilde{\mathcal{L}}_{\text{sub}}, \tag{13}$$

where $\lambda \in [0, 1]$ controls the contribution of the generation objective, while $1 - \lambda$ weights the auxiliary subgraph objective. The corresponding raw losses are defined as

$$\mathcal{L}_{\text{gen}} = -\sum_{\tau=1}^{N_{\text{out}}} \log P_{\text{LLM}}\big(y_\tau \mid y_{<\tau}, [\mathbf{S}; \mathbf{T}]\big), \;\; \mathcal{L}_{\text{sub}} = -\log \frac{\exp(s^+)}{\exp(s^+) + \sum_{m=1}^{N_{\text{neg}}} \exp(s_m^-)}. \tag{14}$$

To mitigate scale imbalance, we use the EMA-normalized losses $\widetilde{\mathcal{L}}_{\text{gen}}$ and $\widetilde{\mathcal{L}}_{\text{sub}}$ in Eq.(13). For $\mathcal{L}_{\text{sub}}$, $s^+ = \text{sim}(\mathbf{q}, \mathbf{v}^+)$ measures query compatibility with the ground-truth candidate subgraph, while $s_m^- = \text{sim}(\mathbf{q}, \mathbf{v}_m^-)$ is computed from structurally corrupted variants. Thus, $\mathcal{L}_{\text{gen}}$ supervises candidate reranking, whereas $\mathcal{L}_{\text{sub}}$ encourages query-compatible structural representations. Notably, $\mathcal{L}_{\text{sub}}$ is applied before candidate fusion and does not directly supervise the final structural prefix.

Complexity Analysis. We analyze online inference after offline subgraph preparation, with full complexity analysis in Appendix E. RISE compresses structural context into $K + 2$ soft tokens, giving $N_{\text{seq}} = N_{\text{token}} + K + 2$, independent of N_{pth} and $L_{\max}$ for fixed K. The LLM prefill costs $\mathcal{O}(N_{\text{seq}}^2 d_{lm})$ per layer, while the graph encoder costs $\mathcal{O}(L_{\text{gnn}} K N_{\text{pth}} L_{\max} d_e^2)$. *Thus, richer subgraphs increase graph-encoding cost approximately linearly, while keeping the LLM sequence length fixed*, avoiding the quadratic attention growth caused by textualizing additional paths. This yields a *favorable trade-off between structural richness and efficiency*. Figure 7 reports the runtime.

4 Experiments

4.1 Experimental Setups

Datasets and Metrics. Our experiments span four benchmarks with diverse scales and domains, from small, domain-specific KGs (**Kinship-MH** (Hinton, 1986) and **UMLS** (McCray, 2003)) to large, open-domain KGs (**FB15K-237** (Toutanova & Chen, 2015) and **CoDEx-L** (Safavi & Koutra, 2020)). Dataset statistics are reported in Table 5, with details in Appendix B. We report filtered Mean Reciprocal Rank (**MRR**) and **Hits@**N (H@N, $N \in \{1, 3, 10\}$) over tail-entity prediction queries as percentages (%). For reranking models, targets outside the initial top-K retain their retriever ranks, while omitted in-set targets are conservatively calibrated; the complete protocol is given in Appendix D. We additionally report dataset-wise RotatE **Recall@**K, which measures ground-truth coverage in the retrieved top-K candidates, with further analysis in Appendix G.

Comparative Baselines. We divide the comparative baselines into query-specific substructure-agnostic and substructure-aware methods, based on whether they retrieve or construct query-specific substructures as inference context. **(1) Query-specific substructure-agnostic baselines**: RotatE (Sun et al., 2019), C-LMKE (Wang et al., 2022), CSProm-KG (Chen et al., 2023) (hereafter CSPr-KG), KICGPT (Wei et al., 2023), KoPA (Zhang et al., 2024), PDKGC (Geng et al., 2025), SSQR (Lin et al., 2025), and ST-KGLP (Luo et al., 2026). **(2) Query-specific substructure-aware baselines**: MINERVA (Das et al., 2018), InstructGLM (Ye et al., 2024) (hereafter InstrGLM), FtG (Liu et al.,

2025a), RAA-KGC (Yuan et al., 2025), SRP-KGC (Hou et al., 2025), DrKGC (Xiao et al., 2025), SAT (Liu et al., 2025b), and FuST-KGC (Han et al., 2026). Furthermore, in Section 4.4, we also evaluate several standard **KGE methods** (i.e., TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), and ComplEx (Trouillon et al., 2016)) to examine how the choice of KGE backbone affects our model. Further details on these baselines are provided in Appendix C.

Implementation Details. Exhaustive experimental settings and architectural specifications are detailed in Appendix D. Overall, our RISE integrates RotatE (Sun et al., 2019) as the KGE backbone with Qwen3-8B (Yang et al., 2025) as the LLM backbone, fine-tuned via LoRA (Hu et al., 2022). To control for backbone capacity, all applicable LLM-based baselines are re-trained and evaluated using the same Qwen3-8B. We strictly confine background graph construction to the training set to prevent data leakage, and adopt dataset-specific hyperparameters for path construction and loss weighting.

4.2 Main Results

Table 1 compares performance across four datasets. Overall, our RISE achieves the best results on most metrics against both baseline groups. Several patterns emerge beyond the overall gains. The substructure-agnostic ST-KGLP achieves higher MRR than all substructure-aware baselines on UMLS, indicating that access to substructures alone does not ensure better predictions. The substructure-selection analysis in Figure 3 further shows that randomly selected substructures underperform no subgraph context on FB15K-237, highlighting the importance of selecting useful structural context. Our framework follows this principle by augmenting KGE-informed candidate representations with subgraphs selected for each prediction. On Kinship-MH, it improves MRR over DrKGC by 6.3 percentage points while both achieve 100.0% H@10, reflecting better candidate ordering rather than greater top-10 coverage. Its MRR gains over both groups on FB15K-237 and CoDEx-L further support its effectiveness beyond small benchmarks. Although FtG retains a 0.6-point H@1 advantage on CoDEx-L, our RISE places more correct entities within the top-3 and 10.

Table 1: The MRR and H@{1, 3, 10} scores. Our model is reported in the last row. Items with the optimal values are marked in **bold**, and the runners-up are underlined.

Methods	Kinship-MH				UMLS				FB15K-237				CoDEx-L			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
RotatE	61.9	45.7	73.9	87.0	66.3	48.9	81.1	91.8	35.4	26.0	39.0	54.4	37.5	24.5	43.4	64.5
C-LMKE	52.9	41.3	54.4	82.6	77.8	67.3	85.3	93.2	39.8	31.0	43.7	59.9	38.0	30.4	40.3	53.2
CSPr-KG	45.5	32.6	52.2	71.7	40.8	28.9	45.5	64.2	37.0	27.5	40.4	56.3	42.6	33.7	47.5	59.3
KICGPT	18.7	4.4	15.2	52.2	22.0	10.1	20.4	49.3	33.9	23.9	38.3	55.2	45.1	33.5	51.2	66.1
KoPA	20.2	10.9	19.6	54.4	26.5	12.9	27.5	69.7	41.6	34.2	42.7	56.8	40.2	30.3	42.7	61.0
PDKGC	42.2	23.9	41.3	87.0	75.5	66.3	80.7	92.3	36.9	28.7	40.4	52.9	32.9	26.1	35.8	45.6
SSQR	19.7	8.7	15.2	43.5	76.9	67.9	82.3	92.4	45.2	38.8	47.3	58.9	38.2	28.4	41.4	57.3
ST-KGLP	69.9	60.9	78.3	91.3	88.1	86.7	88.8	92.9	43.5	35.9	47.6	57.3	45.4	34.5	52.5	65.1
MINERVA	55.9	43.5	58.7	87.0	79.7	69.6	88.2	93.8	29.7	21.1	32.9	45.7	36.4	25.7	42.5	56.6
InstrGLM	60.8	43.5	73.9	87.0	76.8	70.1	80.2	91.7	35.5	26.1	38.7	55.0	42.7	34.1	48.3	60.2
FtG	62.5	45.7	78.3	80.4	57.3	49.8	65.5	66.6	40.7	33.4	42.1	56.3	44.4	**38.0**	49.4	57.1
RAA-KGC	44.6	30.4	47.8	87.0	82.9	74.4	88.2	94.3	38.4	29.4	41.3	56.5	44.5	35.6	47.9	62.1
SRP-KGC	39.1	17.4	52.2	89.1	44.8	37.7	47.5	60.8	44.2	36.5	47.0	60.3	39.7	29.7	43.6	59.1
DrKGC	80.9	67.4	93.5	**100.0**	80.7	74.4	85.2	94.1	43.3	34.4	46.3	56.7	39.7	28.6	49.8	67.6
SAT	28.9	15.2	28.3	60.9	80.4	74.0	86.4	93.2	37.9	31.6	41.8	55.6	43.2	33.9	49.6	60.6
FuST-KGC	44.8	28.3	52.2	91.3	71.7	60.5	78.5	94.1	42.3	32.7	46.0	61.7	40.9	30.9	45.3	60.6
RISE	**87.2**	**76.1**	**97.8**	**100.0**	**90.9**	**89.1**	**91.2**	**95.0**	**47.8**	**41.5**	**50.9**	**62.2**	**48.5**	37.4	**57.3**	**71.3**

4.3 Ablation Study

Table 2 shows that RISE outperforms all variants on both datasets. Removing soft prompts (- *soft prompts*) causes the largest MRR drops, from 90.9 to 75.1 on UMLS and from 47.8 to 33.7 on FB15K-237, highlighting the importance of explicitly injecting structural information into the LLM. Removing subgraph embeddings (- *subgraph embeds*) further reduces MRR to 83.4 and 43.4, respectively, showing that global KGE representations alone are insufficient and that query-specific local structures provide complementary context. Similarly, removing query conditioning (- *query conditioning*) or the auxiliary loss (- *auxiliary loss* $\mathcal{L}_{\text{sub}}$) consistently degrades performance, supporting the

need to encode subgraphs in a query-aware and discriminative manner. In contrast, directly adding textualized subgraphs (+ *subgraph text*) lowers MRR to 80.7 and 39.1, indicating that simply verbalizing graph structures is less effective than compact latent conditioning. Overall, these results echo the challenges: effectively exploiting query-specific structural context and integrating it into LLMs without relying on verbose graph descriptions.

Table 2: Ablation study of RISE variants.

Methods	UMLS		FB15K-237	
	MRR	H@1	MRR	H@1
RISE	**90.9**	**89.1**	**47.8**	**41.5**
- soft prompts	75.1	71.1	33.7	27.8
- subgraph embeds	83.4	80.3	43.4	36.0
- query conditioning	89.7	87.8	46.4	39.5
- auxiliary loss $\mathcal{L}_{\text{sub}}$	88.5	86.2	45.7	38.3
+ subgraph text	80.7	76.9	39.1	31.2

4.4 Model Analysis

This section analyzes the robustness of RISE to path sources in subgraphs, variations in KGE backbones, candidate counts, and LLM backbones. Additional analyses are provided in Appendix H.

Analysis of Path Sources in Subgraphs. Figure 3 directly examines the challenge: not all connecting paths are equally informative for a specific query-candidate pair. The *Dual* strategy, which combines $\mathcal{S}_{\text{struct}}$ and $\mathcal{S}_{\text{trace}}$ without rollback prioritization, already improves MRR/H@1 by 3.5/4.6 points on UMLS and 1.4/2.7 points on FB15K-237, showing that structural plausibility and optimization trajectories provide complementary signals for path preselection. Applying rollback-based prioritization in *Ours* further increases the gains to 7.5/8.8 and 4.4/5.5 points, respectively, indicating that prediction-sensitive rollback helps identify paths more relevant to the target candidate. In contrast, shortest-path selection yields only marginal gains, while random paths consistently hurt performance. These results show that connectivity alone is insufficient, supporting prediction-relevant substructure selection.

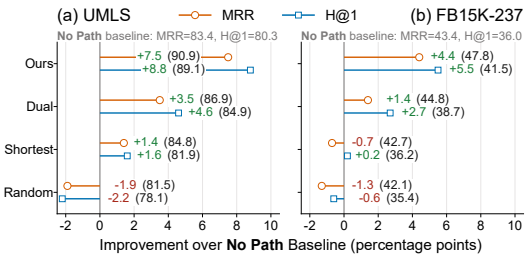


Figure 3: Impact of path source.

Effects of KGE models. Figure 4 shows that RISE consistently improves MRR and H@1 across all four KGE backbones on both datasets. Variants built on RotatE and ComplEx outperform those based on TransE and DistMult, with RotatE achieving the highest scores throughout. Substantial gains are also observed with simpler KGE models: RISE raises TransE's H@1 on UMLS from 38.7 to 82.5, a relative improvement of 113.2%. On FB15K-237, relative improvements range from 25.1% to 35.0% in MRR and from 44.6% to 59.6% in H@1 across the four KGE models. These results highlight the influence of KGE model selection on final performance while demonstrating that the benefits of RISE extend across different KGE representations.

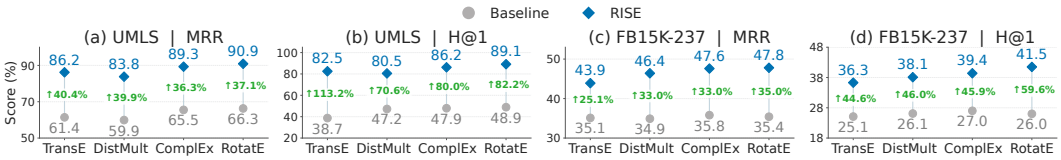


Figure 4: Performance of RISE variants built upon various KGE models.

Sensitivity to Candidate Count K. Table 9 reports RotatE Recall@K, which measures the coverage of ground-truth entities in the retrieved candidate set and thus bounds subsequent reranking performance. As shown in Figure 5, increasing K from 10 to 30 consistently improves both MRR and H@1 on UMLS and FB15K-237. Beyond $K = 30$, the trend becomes dataset-dependent. On UMLS, Recall@K is already 97.0% at $K = 30$ and increases only to 98.0% and 98.8% at $K = 40$ and 50, while reranking performance peaks at $K = 40$ and then declines. In contrast, FB15K-237 continues to improve up to $K = 50$, although the marginal gains beyond $K = 30$ are modest. Therefore, we choose $K = 30$ as a practical trade-off between candidate coverage, reranking accuracy, and computational cost rather than as a universal saturation point.

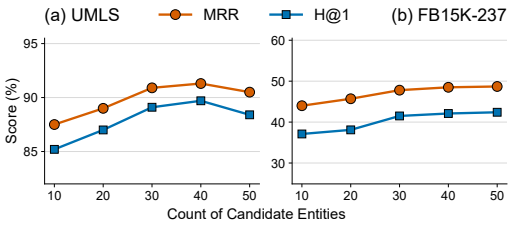


Figure 5: Impact of candidate count.

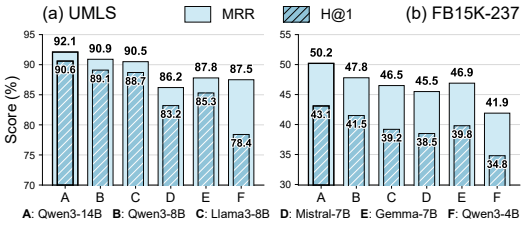


Figure 6: Effect of different LLM backbones.

Effects of LLM Backbones. Figure 6 compares RISE across different LLM backbones. Qwen3-14B achieves the best performance on both datasets, reaching 92.1/90.6 MRR/H@1 on UMLS and 50.2/43.1 on FB15K-237. Among the 7B-8B models, Qwen3-8B and Llama3-8B remain competitive, while Mistral-7B and Gemma-7B show larger variations, indicating that performance depends on both model scale and backbone architecture. Qwen3-4B exhibits the clearest degradation, particularly in H@1, which drops to 78.4 on UMLS and 34.8 on FB15K-237. Overall, larger backbones generally improve ranking quality, but the gains from Qwen3-8B to Qwen3-14B are moderate compared with the gap to the 4B model, suggesting diminishing returns from further scaling.

4.5 Case Study

Table 3 presents a case study for the query *(Marco, husband of, ?)*, whose ground-truth entity is *Lucia*. Unlike InstrGLM and SRP-KGC, which rely on isolated paths or flattened neighborhoods, both DrKGC and RISE capture connecting structures and successfully rank *Lucia* first. Compared with DrKGC, however, RISE retrieves more targeted multi-hop paths, including explicit familial connections through *in-laws*, while introducing fewer perturbations to the remaining candidates. This example illustrates how rollback-guided subgraph extraction provides fine-grained structural information for accurate and stable candidate reranking.

Query: (Marco, husband of, ?) **Answer:** Lucia

Methods	Initial Candidates	Query-specific Substructures (Top-2)	Predicted List
InstrGLM	[Marco, Sophia, Lucia, Alfonso, . . .]	Marco $\xleftarrow{\text{sister of}}$ Angela $\xleftarrow{\text{husband of}}$ Tomaso Marco $\xleftarrow{\text{son-in-law of}}$ Roberto $\xleftarrow{\text{granddaughter of}}$ Sophia	[Marco, Sophia, **Lucia**, Alfonso, . . .]
DrKGC	[Marco, Sophia, Lucia, Alfonso, . . .]	Lucia $\xrightarrow{\text{daughter-in-law of}}$ Francesca $\xrightarrow{\text{mother of}}$ Marco Lucia $\xleftarrow{\text{sister-in-law of}}$ Angela $\xrightarrow{\text{sister of}}$ Marco	[**Lucia**, ~~Angela~~, ~~Gina~~, Marco, . . .]
SRP-KGC	No candidates	Marco's Neighbors: [Pierro, Roberto, Sophia, Tomaso . . .] Lucia's Neighbors: [Pierro, Roberto, Sophia, Emilio . . .]	[Gina, Angela, Francesca, Emilio, . . .]
RISE	[Marco, Sophia, Lucia, Alfonso, . . .]	Lucia $\xrightarrow{\text{daughter of}}$ Maria $\xleftarrow{\text{husband of}}$ Roberto $\xrightarrow{\text{father-in-law of}}$ Marco Marco $\xrightarrow{\text{son-in-law of}}$ Roberto $\xrightarrow{\text{husband of}}$ Maria $\xleftarrow{\text{daughter of}}$ Lucia	[**Lucia**, Sophia, ~~Penelope~~, ~~Maria~~, . . .]

Table 3: Case study of query-specific substructures and prediction lists in different models.

5 Conclusion

In this paper, we present RISE, a rollback-guided framework that enhances LLM-based KGC with query-specific structural context. RISE traces KGE optimization updates and uses rollback-induced prediction changes to prioritize query-candidate paths, which are aggregated into influential subgraphs and encoded as compact structural prefixes for candidate reranking. This design complements global KGE representations and textual semantics with prediction-relevant local structures. Experiments on four datasets show that RISE achieves the best performance on most evaluation metrics among the compared methods, while further analyses demonstrate the benefits of both rollback-guided subgraph selection and latent structural conditioning.

RISE has several limitations. Trajectory tracing and rollback-based subgraph discovery introduce additional offline computation and storage, and the current framework is limited to static, transductive KGs. Its performance also depends on the initial KGE retriever providing sufficient candidate recall. Future work will explore more efficient subgraph discovery, dynamic or inductive settings, and tighter integration between candidate retrieval and reranking.

AI USE STATEMENT

Large language models were used as components of the proposed method and its experimental evaluation, as described in the paper. Beyond this research use, generative AI tools were used for grammar checking, proofreading, language polishing, and automated pre-submission feedback through the ICLR-provided Paper Assistant Tool (PAT). The feedback was used to identify potential issues in presentation and technical clarity, and all suggestions were independently evaluated by the authors before incorporation. The authors take full responsibility for the final content of the paper.

ETHICS STATEMENT

The data used in this paper are sourced exclusively from publicly available KGC benchmark datasets (see Appendix B). We do not collect private data or conduct experiments involving human participants. Accordingly, this work raises no apparent ethical concerns related to data privacy or human-subject research.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed pseudocode for the proposed algorithm in Appendix I. Dataset descriptions and preprocessing procedures, implementation details, and hyperparameter settings are documented in Appendices B and D. The source code and accompanying documentation for setup and reproduction are available in the anonymous repository linked in the abstract.

REFERENCES

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*, pp. 2787–2795, Lake Tahoe, Nevada, 2013.

Jiahang Cao, Jinyuan Fang, Zaiqiao Meng, and Shangsong Liang. Knowledge graph embedding: A survey from the perspective of representation spaces. *ACM Computing Surveys*, 56(6):159:1–159:42, 2024.

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

Chen Chen, Yufei Wang, Aixin Sun, Bing Li, and Kwok-Yan Lam. Dipping PLMs sauce: Bridging structure and text for effective knowledge graph completion via conditional soft prompting. In *Findings of the Association for Computational Linguistics, ACL 2023*, pp. 11489–11503, Toronto, Canada, 2023. Association for Computational Linguistics.

Kewei Cheng, Nesreen K Ahmed, Ryan A Rossi, Theodore L Willke, and Yizhou Sun. Neural-symbolic methods for knowledge graph reasoning: A survey. *ACM Transactions on Knowledge Discovery from Data*, 18(9):225:1–225:44, 2024.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *Proceedings of the 6th International Conference on Learning Representations, ICLR*, Vancouver, BC, Canada, 2018. OpenReview.net.

Antonio Di Mauro, Zhao Xu, Wiem Ben Rim, Timo Sztyler, and Carolin Lawrence. Generating and evaluating plausible explanations for knowledge graph completion. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pp. 12106–12118, Bangkok, Thailand, 2024. Association for Computational Linguistics.

Yuxia Geng, Jiaoyan Chen, Yuhang Zeng, Zhuo Chen, Wen Zhang, Jeff Z Pan, Yuxiang Wang, and Xiaoliang Xu. Prompting disentangled embeddings for knowledge graph completion with pre-trained language model. *Expert Systems with Applications*, 268:126175, 2025.

Daojun Han, Mengxin Jin, Juntao Zhang, Linkun Fan, Qinglin Su, and Bendong Qiao. FuST-KGC: Fusing sub-graph structures and textual semantics for knowledge graph completion. *Neurocomputing*, 671:132690, 2026.

Geoffrey E Hinton. Learning distributed representations of concepts. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 8, 1986. URL https://escholarship.org/uc/item/79w838g1.

Yanning Hou, Sihang Zhou, Ke Liang, Lingyuan Meng, Xiaoshu Chen, Ke Xu, Siwei Wang, Xinwang Liu, and Jian Huang. Soft reasoning paths for knowledge graph completion. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence, IJCAI*, pp. 2937–2945, Montreal, Canada, 2025. ijcai.org.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Proceedings of the 10th International Conference on Learning Representations, ICLR*, Online, 2022. OpenReview.net.

Carolin Lawrence, Timo Sztyler, and Mathias Niepert. Explaining neural matrix factorization with gradient rollback. In *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI*, pp. 4987–4995, Online, 2021. AAAI Press.

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

Ben Liu, Jihai Zhang, Fangquan Lin, Cheng Yang, and Min Peng. Filter-then-Generate: Large language models with structure-text adapter for knowledge graph completion. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING*, pp. 11181–11195, Abu Dhabi, UAE, 2025a. Association for Computational Linguistics.

Yu Liu, Yanan Cao, Xixun Lin, Yanmin Shang, Shi Wang, and Shirui Pan. 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, EMNLP*, pp. 20970–20984, Suzhou, China, 2025b. Association for Computational Linguistics.

Shijie Luo, Xinyuan Lu, Qinpei Zhao, and Weixiong Rao. Combining structural and textual knowledge for knowledge graph link prediction via large language models. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining, WSDM*, pp. 479–488, Boise, ID, USA, 2026. Association for Computing Machinery.

Tengfei Ma, Xiang Song, Wen Tao, Mufei Li, Jiani Zhang, Xiaoqin Pan, Yijun Wang, Bosheng Song, and Xiangxiang Zeng. Towards synergistic path-based explanations for knowledge graph completion: Exploration and evaluation. In *The Thirteenth International Conference on Learning Representations, ICLR*, Singapore, 2025. OpenReview.net.

Alexa T McCray. An upper-level ontology for the biomedical domain. *Comparative and Functional Genomics*, 4(1):80–84, 2003. URL https://doi.org/10.1002/cfg.255.

Enayat Rajabi and Kobra Etminni. Knowledge-graph-based explainable AI: A systematic review. *Journal of Information Science*, 50(4):1019–1029, 2024.

Andrea Rossi, Donatella Firmani, Paolo Merialdo, and Tommaso Teofili. Explaining link prediction systems based on knowledge graph embeddings. In *Proceedings of the 2022 International Conference on Management of Data, SIGMOD*, pp. 2062–2075, Philadelphia, PA, USA, 2022. Association for Computing Machinery.

Tara Safavi and Danai Koutra. CoDEx: A comprehensive knowledge graph completion benchmark. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pp. 8328–8350, Online, 2020. Association for Computational Linguistics.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. RotatE: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the 7th International Conference on Learning Representations, ICLR*, New Orleans, LA, USA, 2019. OpenReview.net.

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

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning, ICML*, volume 48, pp. 2071–2080, New York, New York, USA, 2016. PMLR.

Xintao Wang, Qianyu He, Jiaqing Liang, and Yanghua Xiao. Language models as knowledge embeddings. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence, IJCAI*, pp. 2291–2297, Vienna, Austria, 2022. ijcai.org.

Yingxu Wang, Shiqi Fan, Mengzhu Wang, Siyang Gao, Chao Wang, and Nan Yin. DAMR: Efficient and adaptive context-aware knowledge graph question answering with LLM-guided MCTS. In *Proceedings of the 14th International Conference on Learning Representations, ICLR*. OpenReview.net, 2026.

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

Yongkang Xiao, Sinian Zhang, Yi Dai, Huixue Zhou, Jue Hou, Jie Ding, and Rui Zhang. DrKGC: Dynamic subgraph retrieval-augmented LLMs for knowledge graph completion across general and biomedical domains. In *Findings of the Association for Computational Linguistics, EMNLP 2025*, pp. 16432–16445, Suzhou, China, 2025. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025. URL `https://arxiv.org/abs/2505.09388`.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR*, San Diego, CA, USA, 2015.

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. In *Findings of the Association for Computational Linguistics, EACL 2024*, pp. 1955–1973, St. Julian's, Malta, 2024. Association for Computational Linguistics.

Duanyang Yuan, Sihang Zhou, Xiaoshu Chen, Dong Wang, Ke Liang, Xinwang Liu, and Jian Huang. Knowledge graph completion with relation-aware anchor enhancement. In *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI*, volume 39, pp. 15239–15247, Philadelphia, PA, USA, 2025. AAAI Press.

Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. Making large language models perform better in knowledge graph completion. In *Proceedings of the 32nd ACM International Conference on Multimedia, MM*, pp. 233–242, Melbourne, VIC, Australia, 2024. Association for Computing Machinery.

APPENDIX CONTENTS

A Notations . . . 14
B Datasets . . . 15
C Comparative Baselines . . . 15
D Implementation Details . . . 17
D Model Configuration . . . 17
Training and Subgraph Settings . . . 17
Baseline and Candidate Settings . . . 17
Evaluation Protocol . . . 17
E Complexity Analysis . . . 17
Influential Subgraph Construction . . . 17
Model Training . . . 18
Model Inference . . . 18
Trajectory Map Storage . . . 18
F Analysis of Rollback Sensitivity . . . 18
G Candidate Recall and Size Selection . . . 19
H Analysis Experiments . . . 19
Influence of Loss Weight λ . . . 19
Effects of Candidate Path Budget M . . . 19
Impact of Max Path Length $L_{\max}$ in Subgraphs . . . 20
Sensitivity to Path Count N_{pth} in Subgraphs . . . 20
Effect of Trajectory Reservation Ratio ρ . . . 20
Sensitivity to Training Set Size. . . . 20
Impact of Soft Prompt Position . . . 20
Structural Encoding vs. Subgraph Text . . . 21
I Overall Algorithm . . . 21
J LLM Prompt Template . . . 21

A NOTATIONS

Table 4 summarizes the main symbols used in this paper.

Table 4: Summary of the main notations used in this paper.

Symbol	Description
$\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$	Knowledge graph with entity set $\mathcal{E}$, relation set $\mathcal{R}$, and factual triple set $\mathcal{T}$
$\xi = (h, r, t)$	A factual triple with head entity h, relation r, and tail entity t
$q = (h, r, ?)$	A KGC query with an unknown tail entity
$\Theta_{\text{KGE}} = \{\mathbf{E}, \mathbf{R}\},\ \Theta^*_{\text{KGE}}$	KGE parameters and their optimized state; $\mathbf{E}$ and $\mathbf{R}$ denote entity and relation embedding matrices
$d_e,\ d_r,\ d_{lm}$	Entity embedding, relation embedding, and LLM hidden dimensions
$\mathcal{C}_q,\ K$	Retrieved candidate set for query q and its candidate budget
$\mathcal{F}_{\text{score}},\ P_{\Theta^*_{\text{KGE}}}$	KGE scoring function and its normalized candidate probability under parameters Θ^*_{KGE}
Δ_ξ	Triple-level update vector formed by concatenating the accumulated embedding updates of the head, relation, and tail in triple ξ
$\Delta_{\xi,x}$	Accumulated optimization update associated with triple ξ on component $x \in \{h, r, t\}$
$p = (\xi_1, \ldots, \xi_\ell)$	A path of length ℓ connecting the query head to a candidate entity
$L_{\max}$	Maximum path length considered during path enumeration
$\mathcal{S}_{\text{struct}}(p)$	Structural plausibility score of path p based on normalized KGE predictions
$\mathcal{S}_{\text{trace}}(p)$	Optimization-trace magnitude of path p based on accumulated triple-level updates
$M,\ \rho$	Candidate path budget and the fraction reserved for trajectory-based preselection
Θ^p_{roll}	Rollback-perturbed KGE parameter state obtained by retracting the recorded updates associated with path p
$\mathcal{S}_{\text{roll}}(p)$	Rollback sensitivity score measuring the prediction change induced by retracting path-related updates
$N_{\text{pth}},\ \mathcal{G}_i$	Count of retained paths and the influential subgraph constructed for candidate c_i
$\mathbf{q}$	Query representation derived from the KGE embeddings of the query head and relation
$\mathbf{X}_i,\ \mathbf{v}_i$	Relational graph encoder node representations and the aggregated representation of candidate-specific subgraph $\mathcal{G}_i$
$A(\cdot),\ \mathbf{u}_x$	Shared residual adapter and the adapted KGE representation of component x in the LLM hidden space
$\mathbf{z}_{c_i}$	Candidate representation augmented with its query-specific subgraph representation
$\mathbf{S},\ \mathbf{T}$	Latent structural soft prefix and textual token embeddings supplied to the LLM
$Y = (y_1, \ldots, y_{N_{\text{out}}})$	Autoregressively generated candidate-ranking token sequence
$\mathcal{L}_{\text{gen}},\ \mathcal{L}_{\text{sub}},\ \lambda$	Candidate-reranking loss, query-subgraph contrastive loss, and generation-objective weight

B DATASETS

Table 5 summarizes the statistics of four experimental datasets. **Kinship-MH** is constructed from the original Kinship dataset (Hinton, 1986), which contains 24 individuals from two families and 12 fundamental relations. To evaluate multi-hop reasoning, we augment the relation schema with 10 composite relations deterministically defined by the logical rules in Table 6. To prevent deductive leakage, the original triples are partitioned before relation augmentation, and composite triples in the training graph are generated exclusively from training-set premises. Validation and test facts are never used to construct training edges. We denote this augmented benchmark as Kinship-MH to distinguish it from the vanilla Kinship dataset and all methods are evaluated on the same augmented benchmark for fair comparison. **UMLS** (McCray, 2003), maintained by the U.S. National Library of Medicine, serves as a unifying *semantic* network that merges various biomedical vocabularies into a cohesive graph providing standardized representations of medical concepts. **FB15K-237** (Toutanova & Chen, 2015) is a subset of Freebase designed to eliminate data leakage found in earlier benchmarks. By removing inverse and symmetric relations, it forces models to rely on *compositional* reasoning rather than trivial pattern matching. **CoDEx-L** (Safavi & Koutra, 2020) is a Wikidata-based benchmark that strictly removes inverse relations to eliminate train/test leakage. By minimizing easy-to-predict frequency patterns, it ensures that performance reflects genuine *relational* learning rather than trivial statistical matching.

Table 5: Statistics of experimental datasets.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}_{\text{train}}\rvert$	$\lvert\mathcal{T}_{\text{valid}}\rvert$	$\lvert\mathcal{T}_{\text{test}}\rvert$	$\lvert\mathcal{T}\rvert$
Kinship-MH	24	22	120	18	46	184
UMLS	135	46	5,216	652	661	6,529
FB15K-237	14,541	237	272,115	17,535	20,466	310,116
CoDEx-L	77,951	69	551,193	30,622	30,622	612,437

Table 6: Construction rules for the additional composite relations in Kinship-MH. All composite relations are deterministically derived from the original Kinship relations.

Composite Relation	Logical Construction Rule
brother-in-law	$\text{brother}(x,y) \wedge [\text{husband}(y,z) \vee \text{wife}(y,z)] \Rightarrow \text{brother-in-law}(x,z)$
sister-in-law	$\text{sister}(x,y) \wedge [\text{husband}(y,z) \vee \text{wife}(y,z)] \Rightarrow \text{sister-in-law}(x,z)$
father-in-law	$\text{father}(x,y) \wedge [\text{husband}(y,z) \vee \text{wife}(y,z)] \Rightarrow \text{father-in-law}(x,z)$
mother-in-law	$\text{mother}(x,y) \wedge [\text{husband}(y,z) \vee \text{wife}(y,z)] \Rightarrow \text{mother-in-law}(x,z)$
son-in-law	$\text{husband}(x,y) \wedge \text{daughter}(y,z) \Rightarrow \text{son-in-law}(x,z)$
daughter-in-law	$\text{wife}(x,y) \wedge \text{son}(y,z) \Rightarrow \text{daughter-in-law}(x,z)$
grandfather	$\text{father}(x,y) \wedge [\text{father}(y,z) \vee \text{mother}(y,z)] \Rightarrow \text{grandfather}(x,z)$
grandmother	$\text{mother}(x,y) \wedge [\text{father}(y,z) \vee \text{mother}(y,z)] \Rightarrow \text{grandmother}(x,z)$
grandson	$\text{son}(x,y) \wedge [\text{son}(y,z) \vee \text{daughter}(y,z)] \Rightarrow \text{grandson}(x,z)$
granddaughter	$\text{daughter}(x,y) \wedge [\text{son}(y,z) \vee \text{daughter}(y,z)] \Rightarrow \text{granddaughter}(x,z)$

C COMPARATIVE BASELINES

We divide the competitors into two categories: query-specific substructure-agnostic and substructure-aware methods, based on whether query-relevant local structures (e.g., neighbors, paths or subgraphs) are leveraged during inference.

Query-Specific Substructure-Agnostic Baselines. These methods infer entity relationships through implicit vector alignment, textual sequence semantic matching, or global knowledge encoding, without relying on local topological structures. These models avoid the traversal of multi-hop neighborhoods or the extraction of subgraph structures, encompassing a diverse range of paradigms from algebraic mathematical transformations to cross-modal alignment based on LLMs.

- **TransE** (Bordes et al., 2013) interprets relationships as translation operations in a low-dimensional vector space, modeling the tail entity as the result of the head entity vector plus the relation vector.
- **DistMult** (Yang et al., 2015) represents entities as vectors and relations as diagonal matrices to model head-tail interactions through bilinear scoring, achieving high efficiency and strong generalization.

- **ComplEx** (Trouillon et al., 2016) utilizes complex-valued embeddings and the Hermitian dot product to efficiently model both symmetric and anti-symmetric relations, providing a scalable latent factorization approach for link prediction.
- **RotatE** (Sun et al., 2019) represents entities as complex vectors and defines relations as element-wise rotations in the complex vector space, enabling the implicit mathematical modeling of crucial relation patterns such as symmetry, inversion, and composition.
- **C-LMKE** (Wang et al., 2022) optimizes description-based knowledge graph completion through a contrastive learning framework, utilizing in-batch negative sampling to efficiently align the textual representations of queries and target entities using pre-trained language models.
- **CSProm-KG** (Chen et al., 2023) (hereafter CSPr-KG) bridges structural and textual knowledge by generating conditional soft prompts from the structural embeddings of entities and relations, which are then appended to text inputs and fed into a frozen pre-trained language model for link prediction.
- **KICGPT** (Wei et al., 2023) integrates an LLM with a triple-based knowledge retriever, utilizing an in-context learning strategy called Knowledge Prompt to encode structural facts and textual descriptions into demonstrations, thereby guiding the LLM for zero-shot or few-shot link prediction without costly fine-tuning.
- **KoPA** (Zhang et al., 2024) improves KGC by pre-training structural embeddings and mapping them through a knowledge prefix adapter into virtual tokens prepended to the LLM input, enabling structure-aware reasoning during instruction tuning and inference.
- **PDKGC** (Geng et al., 2025) performs knowledge graph completion on a frozen pre-trained language model by combining a text-based hard task prompt with trainable continuous soft prompts, which are generated by aggregating disentangled neighborhoods.
- **SSQR** (Lin et al., 2025) learns discrete entity codes that jointly encode KG structural and semantic information through self-supervised quantization, and feeds these codes into LLMs as compact tokens for instruction tuning and candidate reranking.
- **ST-KGLP** (Luo et al., 2026) improves link prediction by employing a knowledge aligner to bridge the information gap between structural embeddings and textual representations, followed by a query-aware adaptive weighting strategy to dynamically re-rank candidate entities based on query context.

Query-Specific Substructure-Aware Baselines. These models explicitly leverage structural information such as connectivity between entities, multi-hop reasoning paths, or local subgraph topologies to construct a rich reasoning context. By explicitly capturing the neighborhood dependency and structural constraints, these methods move beyond simple latent factorization or global textual alignment, enabling more nuanced multi-hop inference and structural pattern recognition.

- **MINERVA** (Das et al., 2018) formulates knowledge graph completion as a sequential decision-making problem, employing a reinforcement learning agent to dynamically navigate graph structures and discover reasoning paths conditioned on the input query.
- **InstructGLM** (Ye et al., 2024) (hereafter InstrGLM) verbalizes multi-scale graph geometric structures (including node features, k-hop neighbors, and intermediate reasoning paths) into natural language prompts, and fine-tunes an LLM to perform graph tasks directly without any specialized graph neural network encoders.
- **FtG** (Liu et al., 2025a) follows a filter-then-generate paradigm that first retains top-ranked candidates with a conventional KGC model, then combines a pruned ego-graph serialization with an adapter-derived soft graph token to guide LLM-based entity prediction.
- **RAA-KGC** (Yuan et al., 2025) enhances text-based knowledge graph completion by explicitly selecting relation-aware 1-hop neighbors as contextual anchors and appending their textual descriptions to the input query to enrich the pre-trained language model with local structural semantics.
- **SRP-KGC** (Hou et al., 2025) enhances path-based reasoning by introducing soft reasoning paths that generalize relation-specific path semantics, ensuring robust link prediction even when explicit multi-hop reasoning paths are absent.
- **DrKGC** (Xiao et al., 2025) enhances large language models for knowledge graph completion by dynamically retrieving rule-guided subgraphs and encoding them via a graph convolutional network adapter into structural prompts, integrating explicit graph topology with text representation.

- **SAT** (Liu et al., 2025b) bridges graph and language spaces through node- and subgraph-level contrastive alignment, then couples unified graph instructions with a lightweight knowledge adapter to support structure-aware reasoning across multiple KGC tasks.
- **FuST-KGC** (Han et al., 2026) constructs informative local subgraphs through dynamic degree-guided sampling, linearizes their multi-hop paths with entity descriptions, and jointly encodes the resulting structure–text sequences using a contrastively trained dual-encoder PLM.

D Implementation Details

Model Configuration. We adopt RotatE (Sun et al., 2019) as the KGE backbone, whose independently parameterized entity and relation embeddings facilitate localized tracking of triple-associated optimization updates. The KGE backbone choice is analyzed in Figure 4. For the LLM component, we use Qwen3-8B[2] as the unified backbone for RISE and all applicable LLM-based baselines, thereby controlling for backbone capacity; its impact is further analyzed in Figure 6. KGE embeddings have a dimension of 400 and are mapped to the 4096-dimensional LLM hidden space. All MLP modules contain three layers with ReLU activation, while the gating networks use sigmoid activation.

Training and Subgraph Settings. To prevent data leakage, the background graph used for path discovery and structural encoding during training, validation, and testing is constructed exclusively from training triples. We set the candidate path budget $M = 50$, trajectory reservation ratio $\rho = 0.3$, maximum path length $L_{\max} = 3$, and retained path count $N_{\text{pth}} = 3$ for Kinship-MH and UMLS, and $M = 100$, $\rho = 0.3$, $L_{\max} = 3$, and $N_{\text{pth}} = 5$ for FB15K-237 and CoDEx-L. The LLM is fine-tuned with LoRA (Hu et al., 2022) using rank 32 and dropout 0.05, with a learning rate of 1×10^{-5} and a 100-step warm-up across all datasets. Based on Figure 8, the generation-loss weight λ is set to 0.8 for Kinship-MH and UMLS and 0.9 for FB15K-237 and CoDEx-L. RISE is implemented in PyTorch, and all experiments are conducted on four NVIDIA RTX 4090 (24GB) GPUs.

Baseline and Candidate Settings. We follow the method-specific configurations of each baseline, while re-training and evaluating all LLM-based baselines with the same Qwen3-8B backbone used by RISE. For methods requiring an initial ranking, we use identical retrieved candidate lists with a fixed size of $K = 30$ ($K = |\mathcal{E}| = 24$ in Kinship-MH). During training, if the ground-truth entity falls outside the retrieved top-K, it is inserted at the first position and the list is truncated to K entities. The supervision target places the ground truth first while preserving the retriever's relative ordering among the remaining candidates, whereas the input candidate list is randomly shuffled to prevent positional shortcuts. At test time, the retrieved top-K candidate set is kept unchanged; if the ground-truth entity is absent, it is not manually added.

Evaluation Protocol. Let c^* denote the ground-truth entity, $\mathcal{C}_q$ the initial top-K candidate set, and $\mathcal{L}_q$ the valid re-ranked output. Its final rank is calibrated as

$$r_{\text{final}}(c^*) = \begin{cases} r_{\text{rerank}}(c^*), & c^* \in \mathcal{L}_q, \\ \max(N_{\text{list}}, r_{\text{ret}}), & c^* \in \mathcal{C}_q \setminus \mathcal{L}_q, \\ r_{\text{ret}}, & c^* \notin \mathcal{C}_q, \end{cases} \tag{15}$$

where $r_{\text{rerank}}(c^*)$ is the rank of c^* in $\mathcal{L}_q$, $N_{\text{list}} = |\mathcal{L}_q|$, and r_{ret} its original retriever rank.

E Complexity Analysis

Influential Subgraph Construction. For each query–candidate pair, bounded iterative-deepening search enumerates N_{init} paths of length at most $L_{\max}$. Candidate paths are preselected using structural plausibility $\mathcal{S}_{\text{struct}}$ and optimization-trace magnitude $\mathcal{S}_{\text{trace}}$. By caching full-entity KGE predictions for repeated head–relation pairs, preselection costs $\mathcal{O}(U|\mathcal{E}|d_e + N_{\text{init}}L_{\max}d_e + N_{\text{init}} \log N_{\text{init}})$, where U denotes the number of distinct head-relation pairs requiring KGE evaluation. Retaining at most M paths per length gives $N_{\text{pre}} \leq \min(N_{\text{init}}, ML_{\max})$. Rollback scoring then evaluates $\mathcal{S}_{\text{roll}}$ for the retained paths by comparing the target probability under Θ^* and the rollback-perturbed

[2] https://modelscope.cn/models/Qwen/Qwen3-8B

state Θ^{p}_{roll}, incurring $\mathcal{O}(N_{\text{pre}}|\mathcal{E}|d_e)$ for KGE evaluation and $\mathcal{O}(N_{\text{pre}} \log N_{\text{pre}})$ for ranking. The top N_{pth} positively scored paths are finally merged into the influential subgraph. All path discovery and rollback scoring are performed offline.

Model Training. For each query, an L_{gnn}-layer relational graph encoder processes K candidate subgraphs, each containing at most $N_{\text{pth}}L_{\max}$ path-induced triple occurrences. The structural-prefix construction therefore costs $C_{\text{pref}} = \mathcal{O}\big(L_{\text{gnn}}KN_{\text{pth}}L_{\max}d_e^2 + Kd_t^2\big)$, where the second term accounts for adapter projection and candidate fusion. The query-subgraph contrastive objective additionally encodes N_{neg} corrupted subgraphs, requiring $\mathcal{O}(L_{\text{gnn}}N_{\text{neg}}N_{\text{pth}}L_{\max}d_e^2)$ per supervised query. Let N_{train} denote the total number of structural-prefix, prompt, and target tokens, and N_{layer} the number of Transformer layers in the LLM. The LLM forward and backward computation scales as $\mathcal{O}(N_{\text{layer}}N_{\text{train}}d_t(N_{\text{train}} + d_t + d_{\text{ffn}}))$. Thus, over H epochs, the total training cost is $\mathcal{O}\big(H|\mathcal{Q}_{\text{train}}|\left[C_{\text{pref}} + L_{\text{gnn}}N_{\text{neg}}N_{\text{pth}}L_{\max}d_e^2 + N_{\text{layer}}N_{\text{train}}d_t(N_{\text{train}} + d_t + d_{\text{ffn}})\right]\big)$. LoRA reduces the number of trainable backbone parameters while preserving the leading-order forward and backward computation. Hence, the structural modules introduce additional training cost beyond standard LoRA fine-tuning.

Model Inference. Online inference includes structural-prefix construction and LLM generation, with influential subgraphs prepared offline. Prefix construction costs $\mathcal{O}(C_{\text{pref}})$. Since RISE uses one soft token for the query head, relation, and each candidate, $N_{\text{seq}} = N_{\text{token}} + K + 2$, which is independent of N_{pth} and $L_{\max}$ for fixed K. With KV-cached autoregressive decoding of N_{out} tokens, the total online inference cost is bounded by $\mathcal{O}\big(|\mathcal{Q}_{\text{test}}|\left[C_{\text{pref}} + N_{\text{layer}}d_t\left(N_{\text{seq}}^2 + N_{\text{seq}}N_{\text{out}} + N_{\text{out}}^2 + (N_{\text{seq}} + N_{\text{out}})(d_t + d_{\text{ffn}})\right)\right]\big)$. Thus, additional structural paths increase graph-encoding cost without enlarging the LLM input sequence. Figure 7 reports the measured online inference time across methods on dataset FB15K-237.

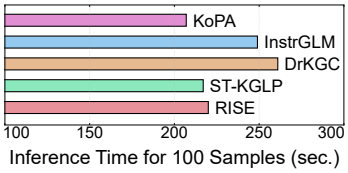


Figure 7: Online inference time.

Trajectory Map Storage. The optimization trajectory map is constructed offline and stores only the local parameter updates associated with each training triple. Specifically, as shown in Table 7, for each triple $\xi = (h, r, t) \in \mathcal{T}_{\text{train}}$, we record $\Delta_\xi = [\Delta_{\xi,h}; \Delta_{\xi,r}; \Delta_{\xi,t}]$. Its storage complexity is therefore $\mathcal{O}(|\mathcal{T}_{\text{train}}|(2d_e + d_r))$, where d_e and d_r denote the numbers of stored real-valued components for entity and relation updates, respectively. Since the trajectory map is constructed only over training triples, its storage scales with $|\mathcal{T}_{\text{train}}|$ rather than the full KG triple set $|\mathcal{T}|$. For RotatE, its dimensions are $d_e = d_r = 400$. Table 8 reports the actual on-disk footprint of our JSON-serialized trajectory maps, including decimal-string serialization, triple keys, and formatting overhead. For fixed embedding dimensions, storage remains linear in $|\mathcal{T}_{\text{train}}|$ despite this implementation-dependent constant factor. After KGE training, the serialized trajectory map remains fixed and is reused for subsequent subgraph construction; its storage footprint does not grow during LLM training or inference.

Table 7: Example of an optimization trajectory record for triple (*Jennifer*, *sister of*, *James*).

Triple component	Update	Example	Dim.
Head: *Jennifer*	$\Delta_{\xi,h}$	[0.00430, −0.01318, ...]	400
Relation: *sister of*	$\Delta_{\xi,r}$	[−0.01045, −0.00188, ...]	400
Tail: *James*	$\Delta_{\xi,t}$	[−0.00180, 0.00270, ...]	400

Table 8: Storage overhead of the trajectory map.

Dataset	Storage
Kinship-MH	3.53 MB
UMLS	154 MB
FB15K-237	7.90 GB
CoDEx-L	15.80 GB

F Analysis of Rollback Sensitivity

For a query $(h, r, ?)$, the normalized KGE prediction is

$$\log P_\Theta(c_i \mid h, r) = f_\Theta(h, r, c_i) - \log \sum\nolimits_{e \in \mathcal{E}} \exp f_\Theta(h, r, e). \tag{16}$$

For an intermediate entity $e_j \neq h, c_i$, its embedding affects the target log-probability through the normalization term:

$$\frac{\partial \log P_\Theta(c_i \mid h, r)}{\partial \mathbf{e}_j} = -P_\Theta(e_j \mid h, r) \frac{\partial f_\Theta(h, r, e_j)}{\partial \mathbf{e}_j}. \tag{17}$$

Thus, intermediate entities generally affect the target prediction indirectly, while the query head, query relation, and target candidate directly participate in the target score. Likewise, an intermediate path relation does not directly affect the prediction unless it coincides with the query relation. Consequently, $\mathcal{S}_{\text{roll}}$ may place greater emphasis on query- and target-related updates and is therefore used as a path-level sensitivity signal for prioritization rather than fine-grained attribution over individual hops. The internal quality of candidate paths is first controlled by $\mathcal{S}_{\text{struct}}$ and $\mathcal{S}_{\text{trace}}$, while rollback sensitivity subsequently provides target-specific prioritization.

G Candidate Recall and Size Selection

Table 9 reports RotatE Recall@K under different candidate budgets, measuring the coverage of ground-truth entities among the retrieved candidates. For reranking models restricted to these candidates, Recall@K therefore defines the upper bound of attainable Hits@1. We use $K = 30$ to balance candidate coverage against subgraph construction and reranking costs. Increasing K from 30 to 50 yields at most a 4.6-point gain in recall across datasets, despite increasing the candidate budget by 66.7%.

Table 9: Recall of RotatE on each dataset.

Dataset	Recall@K (%)				
	$\mathbf{K = 10}$	$\mathbf{K = 20}$	$\boxed{\mathbf{K = 30}}$	$\mathbf{K = 40}$	$\mathbf{K = 50}$
Kinship-MH	87.0	97.8	100.0	100.0	100.0
UMLS	91.8	95.2	97.0	98.0	98.8
FB15K-237	54.5	62.5	68.7	71.6	73.2
CoDEx-L	64.5	73.2	77.5	80.3	82.1

H Analysis Experiments

Influence of Loss Weight λ. Figure 8 examines the sensitivity of RISE to the loss weight λ. Performance follows an inverted-U trend on both datasets: increasing λ initially improves ranking quality, while large values cause degradation. The best performance is achieved at $\lambda = 0.8$ on UMLS, with an MRR of 90.9 and H@1 of 89.1, and at $\lambda = 0.9$ on FB15K-237, reaching 47.8 MRR and 41.5 H@1. Moreover, performance remains relatively stable around $\lambda = 0.8$–0.9, indicating limited sensitivity within this range. These results suggest that an appropriate balance between the training objectives is important, whereas either insufficient or excessive weighting can weaken the learned representations.

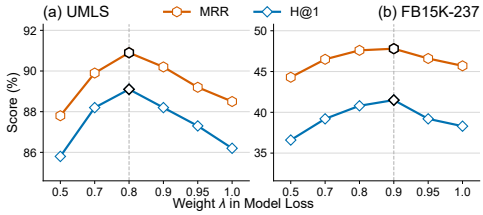


Figure 8: Effects of λ value in model loss.

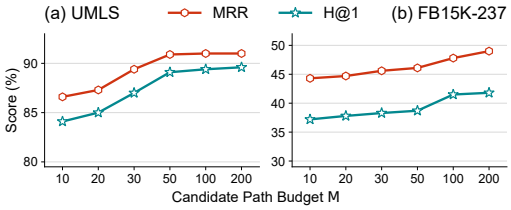


Figure 9: Impact of candidate path budget.

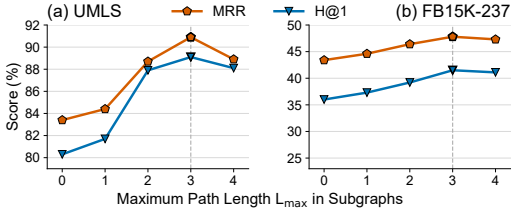


Figure 10: Influence of max path length.

Effects of Candidate Path Budget M. Figure 9 examines the impact of the candidate path budget M. Increasing M consistently improves ranking performance on both datasets, confirming the benefit of retaining a broader set of candidate paths. On UMLS, both MRR and H@1 improve substantially up to $M = 50$ and then largely saturate, indicating that a moderate path budget already captures most useful structural evidence. In contrast, FB15K-237 continues to benefit from larger budgets, although the gains become more moderate beyond $M = 100$. This difference suggests that larger and more diverse graphs require a broader search space to capture sufficient structural evidence. Considering

both computational cost and marginal performance gains, we therefore set $M = 50$ for UMLS and $M = 100$ for FB15K-237.

Impact of Max Path Length $L_{\max}$ in Subgraphs. Figure 10 examines the effect of the maximum path length $L_{\max}$. Performance improves consistently as $L_{\max}$ increases from 0 to 3 on both datasets, highlighting the importance of incorporating progressively richer multi-hop structural evidence. In particular, moving from $L_{\max} = 0$ to 3 raises MRR from 83.4 to 90.9 on UMLS and from 43.4 to 47.8 on FB15K-237, with similar gains in H@1. Performance peaks at $L_{\max} = 3$, while extending paths to length 4 leads to a slight decline on both datasets. This suggests that sufficiently deep paths improve structural reasoning, whereas excessively long paths begin to introduce less relevant topology and dilute useful evidence. We therefore set $L_{\max} = 3$ to balance structural coverage and noise.

Sensitivity to Path Count N_{pth} in Subgraphs. Figure 11 examines the number of paths retained after rollback prioritization. Performance improves sharply when a small number of paths are introduced, but additional paths yield diminishing returns and eventually slight degradation. Notably, the optimal N_{pth} is much smaller than the preselection budget M, indicating that the two budgets serve different roles: a relatively large M preserves sufficient path diversity for target-specific rollback scoring, whereas a small N_{pth} keeps the final subgraph focused by discarding lower-ranked or redundant paths. This supports a broad-preselection, sparse-retention design rather than directly setting $M = N_{\text{pth}}$.

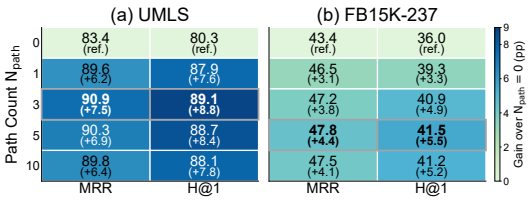


Figure 11: Impact of path count in subgraphs.

Effect of Trajectory Reservation Ratio ρ. The ratio ρ controls the fraction of the path-preselection budget reserved for high-$\mathcal{S}_{\text{trace}}$ paths, with $\rho = 0$ relying entirely on structural plausibility. As shown in Figure 12, introducing a small trajectory reservation consistently improves performance on both datasets, confirming that optimization traces provide complementary information beyond $\mathcal{S}_{\text{struct}}$. The improvement is more pronounced on UMLS, whereas FB15K-237 benefits mainly from a modest amount of trajectory-based reservation and then quickly saturates. Further increasing ρ brings little additional gain and may slightly degrade performance, as more structurally plausible paths are displaced by paths selected primarily for large update magnitudes. Overall, these results support combining the two preselection signals rather than relying exclusively on either one, with a moderate ρ providing a stable balance between structural plausibility and trajectory diversity.

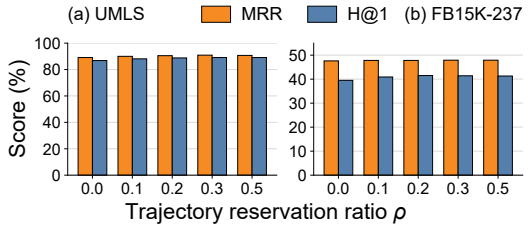


Figure 12: Effect of trajectory reservation ratio ρ.

Sensitivity to Training Data Removal. Since all candidate-specific subgraphs are constructed strictly from $\mathcal{T}_{\text{train}}$, we examine how RISE behaves when different proportions of training triples are removed. As shown in Figure 13, performance decreases monotonically with the deletion ratio, but remains relatively stable under mild removal; for example, deleting 5% of the training triples reduces MRR only from 90.9 to 88.7. The degradation becomes much more pronounced beyond 10%, with MRR falling to 75.9 at 20% deletion and 64.0 at 30%. A sparser training graph provides fewer complete multi-hop connections and less trajectory information for identifying informative candidate-specific subgraphs, making structural augmentation increasingly difficult. H@1 exhibits a similar but sharper decline, suggesting that precise top-ranked prediction is particularly sensitive to missing structural evidence. Overall, RISE tolerates modest training-data reduction, whereas substantial deletion exposes its dependence on sufficient training-graph coverage for both structural representation learning and subgraph construction.

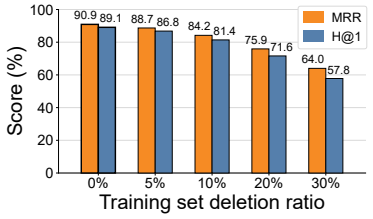


Figure 13: Sensitivity to training-set deletion on UMLS dataset.

Impact of Soft Prompt Position. Table 10 shows that *Prefix* achieves the best performance on both datasets. Moving the soft prompt to *Infix* reduces MRR from 90.9 to 88.3 on UMLS and from 47.8 to 45.7 on FB15K-237, while *Suffix* lowers it to 69.0 and 31.3, respectively. Notably, *Suffix* also underperforms *None* across all metrics, indicating that inappropriate placement can negate the benefits of structural prompts. This pattern is consistent with causal attention: prefix prompts can condition subsequent textual representations, whereas suffix prompts cannot influence preceding text. Although generated output tokens can still attend to suffix prompts, the results suggest that incorporating structural information during text processing is more effective than making it available only afterward.

Table 10: Impact of soft prompt position.

Position	UMLS		FB15K-237	
	MRR	H@1	MRR	H@1
Prefix	**90.9**	**89.1**	**47.8**	**41.5**
Infix	88.3	86.1	45.7	38.5
Suffix	69.0	63.2	31.3	22.3
None	75.1	71.1	33.7	27.8

Structural Encoding vs. Subgraph Text. Table 11 compares RISE with variant *Subgraph Text*, which directly textualizes the same influential subgraphs as LLM prompts without structural soft prefixes. Replacing latent structural encoding with subgraph text substantially degrades performance, with MRR/H@1 dropping from 90.9/89.1 to 69.8/64.3 on UMLS and from 47.8/41.5 to 29.2/23.2 on FB15K-237. The larger gap on FB15K-237 further highlights the advantage of latent structural encoding over direct textualization. These results favor compact structural representations for incorporating selected subgraphs into LLM-based reranking.

Table 11: Structural vs. textual encoding

Methods	UMLS		FB15K-237	
	MRR	H@1	MRR	H@1
Our RISE	**90.9**	**89.1**	**47.8**	**41.5**
Subgraph Text	69.8	64.3	29.2	23.2

I Overall Algorithm

Algorithm 1 outlines RISE's training. We first utilize pre-trained KGEs to retrieve top-K candidates and construct their corresponding influential subgraphs. Then we encode these subgraphs and frozen KGEs into a compact structural soft prefix. By prepending this prefix to the textual prompt, the framework jointly optimizes the structural modules and the LLM (via LoRA) end-to-end using a composite generative ($\mathcal{L}_{\text{gen}}$) and contrastive auxiliary ($\mathcal{L}_{\text{sub}}$) loss.

J LLM Prompt Template

Table 12 presents the prompt template and output constraints for LLMs in the KGC task. The template first prepends **continuous graph embedding vectors** (`KG Soft Prefix`) to the natural language sequence to implicitly inject structured prior knowledge. Next, utilizing **a standardized triple query instruction**, it prompts the model to reorder **a given candidate set of tail entities**. Finally, to facilitate seamless automated evaluation metrics (such as HR and MRR), the template strictly restricts the model to outputting only **the reordered list format** (e.g., `[<C`$_1^*$`>, ..., <C`$_K^*$`>]`).

Algorithm 1: Optimization of RISE

Input: Training graph $\mathcal{G}_{\text{train}}$, training triples $\mathcal{T}_{\text{train}}$, pretrained LLM, candidate budget K.
Output: Optimized RISE model $\mathcal{M}_{\text{RISE}}$.

/* KGE Pre-training and Subgraph Preparation */
$\Theta^*_{\text{KGE}}, \Delta \leftarrow \text{TrainKGE}(\mathcal{G}_{\text{train}})$ // record embedding-update traces
$\mathcal{D} \leftarrow \emptyset$
foreach $(h, r, t) \in \mathcal{T}_{\text{train}}$ **do**
  $q \leftarrow (h, r, ?)$
  $\mathcal{C}_q \leftarrow \text{RetrieveTopK}(q; \Theta^*_{\text{KGE}})$
  Ensure $t \in \mathcal{C}_q$ by replacing the last candidate if necessary
  $\{\mathcal{G}_i\}_{c_i \in \mathcal{C}_q} \leftarrow \text{ConstructSubgraphs}(q, \mathcal{C}_q; \Theta^*_{\text{KGE}}, \Delta)$ // Path pruning and rollback ranking
  $y \leftarrow \text{TargetRanking}(\mathcal{C}_q, t)$
  Add $(q, t, y, \mathcal{C}_q, \{\mathcal{G}_i\})$ to $\mathcal{D}$

/* Joint Training */
Initialize query encoder Ω_q, subgraph encoder Ω_{sub}, shared adapter Ω_{adp}, fusion gate Ω_{gate}, and LoRA parameters
Freeze KGE embeddings Θ^*_{KGE} and LLM base parameters
for *each training epoch* **do**
  foreach *batch* $\mathcal{B} \subset \mathcal{D}$ **do**
    $\mathbf{q} \leftarrow \Omega_q([\mathbf{e}_h; \mathbf{e}_r])$
    $\mathbf{v}_i \leftarrow \Omega_{\text{sub}}(\mathcal{G}_i, \mathbf{q}; h, c_i), \quad c_i \in \mathcal{C}_q$ // query-conditioned graph encoding and readout
    $\mathbf{u}_x \leftarrow \Omega_{\text{adp}}(\mathbf{e}_x), \quad x \in \{h, r\} \cup \mathcal{C}_q$
    $\tilde{\mathbf{v}}_i \leftarrow \Omega_{\text{adp}}(\mathbf{v}_i)$ // shared adaptation
    $\mathbf{g}_i \leftarrow \Omega_{\text{gate}}([\mathbf{u}_{c_i}; \tilde{\mathbf{v}}_i])$
    $\mathbf{z}_{c_i} \leftarrow \mathbf{u}_{c_i} + \mathbf{g}_i \odot \tilde{\mathbf{v}}_i$
    $\mathbf{S} \leftarrow [\mathbf{u}_h; \mathbf{u}_r; \mathbf{z}_{c_1}; \ldots; \mathbf{z}_{c_K}]$
    $\mathcal{P} \leftarrow \mathcal{D}_{\text{task}} \oplus \mathcal{D}_q \oplus \mathcal{D}_{\mathcal{C}}$ // path text excluded
    $\mathbf{T} \leftarrow \text{Embed}_{\text{LLM}}(\mathcal{P})$
    $\mathcal{L}_{\text{gen}} \leftarrow \text{LMLoss}([\mathbf{S}; \mathbf{T}], y)$
    $\mathcal{B}^+ \leftarrow \{b \in \mathcal{B} : \text{the correct candidate has valid paths}\}$
    $\mathcal{L} \leftarrow \mathcal{L}_{\text{gen}}$
    if $\mathcal{B}^+ \neq \emptyset$ **then**
      $\mathcal{G}^+, \mathbf{v}^+ \leftarrow$ correct-candidate subgraphs and their representations in $\mathcal{B}^+$
      $\{\mathcal{G}^-_j\} \leftarrow \text{CorruptRelationsOrDirections}(\mathcal{G}^+)$
      $\mathbf{v}^-_j \leftarrow \Omega_{\text{sub}}(\mathcal{G}^-_j, \mathbf{q}; h, t)$
      $\mathcal{L}_{\text{sub}} \leftarrow \text{ContrastiveLoss}(\mathbf{q}, \mathbf{v}^+, \{\mathbf{v}^-_j\}; \tau)$ // cosine similarity before adaptation
      $\widetilde{\mathcal{L}}_{\text{gen}}, \widetilde{\mathcal{L}}_{\text{sub}} \leftarrow \text{EMANormalize}(\mathcal{L}_{\text{gen}}, \mathcal{L}_{\text{sub}})$
      $\mathcal{L} \leftarrow \lambda \widetilde{\mathcal{L}}_{\text{gen}} + (1 - \lambda)\widetilde{\mathcal{L}}_{\text{sub}}$
    Update $\Omega_q, \Omega_{\text{sub}}, \Omega_{\text{adp}}, \Omega_{\text{gate}}$ and LoRA parameters using $\nabla\mathcal{L}$
Return $\mathcal{M}_{\text{RISE}}$

Table 12: LLM prompt template.

Prompt Template	<KG Soft Prefix> **### Instruction**: You are given a (head, relation, ?) query from a knowledge graph. Your task is to predict the most likely tail entity by reordering the candidate list. Output only the reordered list in the format [entity1, entity2, ...] with no additional text or explanations. Reorder candidates by likelihood: [<C$_1$>, <C$_2$>, ..., <C$_K$>] **### Query**: (<HEAD>, <RELATION>, ?) **### Response**:
LLM Response	[<C$^*_1$>, <C$^*_2$>, ..., <C$^*_K$>]