RSF-GLLM: Bridging the Semantic Gap in Multi-Hop Knowledge Graph QA via Recurrent Soft-Flow and Decoupled LLM Generation

Sambaran Bandyopadhyay [1] **Ananth Muppidi** [2]

Abstract

Multi-hop Question Answering over Knowledge Graphs faces a critical challenge: traditional retrieve-then-read pipelines break differentiability, preventing the retriever from learning to bridge the semantic gap where intermediate nodes lack lexical overlap with the query. To address this, we propose RSF-GLLM, a framework decoupling differentiable graph reasoning from answer generation. Our Recurrent Soft-Flow (RSF) module employs a GRU-guided query updater to propagate continuous relevance scores, utilizing a dynamic gating mechanism to traverse semantically dissimilar bridge nodes via structural cues. We introduce flow sparsity regularization to theoretically guarantee convergence from soft probabilities to discrete reasoning paths. These paths are extracted and textualized to fine-tune a Large Language Model (LLM), ensuring generation is grounded in factual topology. Experiments on WebQSP and CWQ demonstrate that RSF-GLLM achieves competitive performance with superior inference efficiency compared to LLM based computationally expensive approaches.

1. Introduction

Reasoning over structured knowledge bases to answer complex, multi-hop queries is a fundamental challenge in artificial intelligence, essential for applications requiring high factual precision such as biomedical discovery and financial auditing (Wang et al., 2021). While Large Language Models (LLMs) have demonstrated exceptional fluency in open-domain tasks (Brown et al., 2020), they frequently exhibit hallucinations and unfaithful reasoning when operating over long-tail knowledge or complex logical chains (Ji et al., 2023; Nandy & Bandyopadhyay, 2025). Consequently, Knowledge Graphs (KGs) remain indispensable for grounding generation in verifiable facts (Hogan et al., 2021; Nandy & Bandyopadhyay, 2025). However, effective retrieval over KGs is impeded by the *semantic gap*. Consider the query: *"What awards did the director of Inception win?"*. To answer this, a system must traverse from the topic entity *"Inception"* to the intermediate node *"Christopher Nolan"* (the director), and finally to *"Academy Award"*. Crucially, the intermediate bridge entity *"Christopher Nolan"* shares no lexical overlap with the query terms "awards" or "win", rendering standard similarity-based traversal ineffective (Sun et al., 2018; Xiong et al., 2017).

Current approaches to Knowledge Graph Question Answering (KGQA) effectively bifurcate into two paradigms, each utilizing distinct inductive biases yet suffering from complementary limitations. The first category comprises *iterative subgraph retrieval* methods, such as GraftNet (Sun et al., 2018), PullNet (Sun et al., 2019), and NSM (He et al., 2021). These architectures typically treat reasoning as a discrete *Retrieve-then-Read* process (Karpukhin et al., 2020). However, the discrete selection of nodes at each hop breaks end-to-end differentiability, preventing the retriever from adapting to downstream errors (Kim et al., 2023). Furthermore, rigid entity-linking requirements often cause these models to fail when the requisite *bridge* nodes are semantically dissimilar to the query text (Liang et al., 2024).

The second, more recent paradigm integrates the *generative capabilities of LLMs* directly. Frameworks such as Graph of Thoughts (Besta et al., 2024) and RoG (Luo et al., 2024) attempt to improve expressivity by structuring LLM reasoning as an iterative graph traversal. However, these agentic approaches incur prohibitive computational costs, often requiring multiple passes through billion-parameter models to answer a single query (Sun et al., 2024). Furthermore, while methods like Self-RAG (Asai et al., 2024) and CRAG (Yan et al., 2024) have introduced critique-based loops to detect hallucinations, they still struggle with the *reasoning shortcut* problem: the generator frequently ignores retrieved structural constraints in favor of its pre-trained parametric memory, compromising factual grounding (Li et al., 2024; Gao et al., 2023).

To reconcile the structural rigor of differentiable traversal

[1]Adobe Research [2]Adobe Systems. Correspondence to: Sambaran Bandyopadhyay <samb.bandyo@gmail.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

with the efficiency required for scalable deployment, we propose **RSF-GLLM** (Recurrent Soft-Flow Graph-to-LLM). Unlike monolithic end-to-end architectures, our framework decouples reasoning from generation. We first introduce a *Recurrent Soft-Flow (RSF)* module, a lightweight differentiable reasoned that propagates continuous probability mass guided by a Recurrent Query Updater (Hudson & Manning, 2019). To bridge the semantic gap, we devise a *Dynamic Gating Mechanism* that modulates the influence of node content versus graph topology, allowing the model to traverse semantically disjoint nodes purely via valid structural relations. Crucially, we impose a *Flow Sparsity Regularization* (Jang et al., 2017) to force these continuous distributions to converge to discrete reasoning paths, ensuring interpretability.

Following are the **contributions** we made in this paper:

Semantic Gap in Differentiable KGQA: We identify and formalize a critical failure mode in existing differentiable multi-hop KGQA models: the semantic gap, where enforcing semantic similarity at intermediate hops suppresses valid reasoning paths through semantically disjoint bridge entities.

Dynamic Structure–Semantics Decoupling: We propose Recurrent Soft-Flow (RSF), a differentiable graph reasoning module that introduces a dynamic gating mechanism to adaptively decouple structural propagation from semantic matching. This enables the model to rely purely on graph topology when semantic signals are misleading, and to re-introduce content bias only when disambiguation is required.

Principled Sparsity for Interpretable and Faithful Reasoning Paths: We introduce an entropy-based flow sparsity regularization and provide a theoretical analysis showing that it incentivizes convergence from soft relevance distributions to discrete, interpretable reasoning paths, enabling *provably* faithful path extraction without sacrificing differentiability.

Decoupled Graph-to-LLM Architecture for Efficient Grounded QA: We propose a two-stage Graph-to-LLM framework that cleanly decouples graph reasoning from answer generation. A lightweight differentiable graph module performs all structural inference, while a large language model is used solely as a grounded conditional generator over extracted reasoning paths—avoiding expensive agentic reasoning loops.

Empirical Validation Across Accuracy, Robustness, and Efficiency: Extensive experiments on WebQSP and CWQ demonstrate that RSF-GLLM achieves competitive accuracy, robust performance on semantic-gap-heavy queries, and orders-of-magnitude inference efficiency improvements over recent LLM-centric and agentic KGQA baselines.

2. Methodology

2.1. Problem Formulation

We formulate Multi-Hop Question Answering (MHQA) over Knowledge Graphs as a path-grounded generation task. Let $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$ be a Knowledge Graph, where $\mathcal{V}$ is the set of entities, $\mathcal{R}$ is the set of relation types, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is the set of factual triplets. Given a natural language query Q, the objective is to generate an answer A derived from a reasoning chain within $\mathcal{G}$.

Unlike standard retrieval-augmented generation (RAG) which retrieves unstructured text passages, we aim to discover a structural reasoning path $P = (v_0, r_1, v_1, \ldots, r_T, v_T)$ with v_0 as a topic entity identified in Q, such that:

1. Each step $(v_{t-1}, r_t, v_t) \in \mathcal{E}$ is a valid triplet in $\mathcal{G}$.
2. The path P semantically bridges the gap between the query intent and the answer node v_T.

The final answer A is generated by maximizing the probability $P(A \mid Q, P)$, ensuring the generation is hallucination-free and strictly grounded in the graph topology. We address the core challenge where intermediate nodes v_t (for $0 < t < T$) may share no lexical similarity with Q (the *Semantic Gap*), requiring a differentiable traversal mechanism that learns to propagate probability mass $P(v_T \mid Q, \mathcal{G})$ via structural cues rather than surface matching.

Now, we discuss the details of our proposed solution **RSF-GLLM**. As shown in Figure 1, it operates in a decoupled two-stage manner: first, we optimize a differentiable graph traverser (RSF Module) to identify reasoning paths; second, we fine-tune an LLM to generate answers conditioned on the textualized paths retrieved by the RSF module.

2.2. Phase 1: Anchor Identification & Initialization

Since processing the entire KG is intractable, we first identify a computational workspace. Unlike older methods that rely on exact string matching, we use dense retrieval to handle synonyms and ambiguity.

2.2.1. Dense Anchor Retrieval

We employ a bi-encoder architecture (similar to DPR (Karpukhin et al., 2020)). We pre-compute embeddings for all entities $e \in \mathcal{V}$ using a BERT-based encoder E_{Ent}. Given question Q, we encode it into $\mathbf{v}_Q = E_Q(Q)$. We retrieve the topic entity v_{topic} via Maximum Inner Product Search (MIPS):

$$v_{topic} = \underset{e \in \mathcal{V}}{\text{argmax}}(\mathbf{v}_Q^\top E_{Ent}(e)) \tag{1}$$

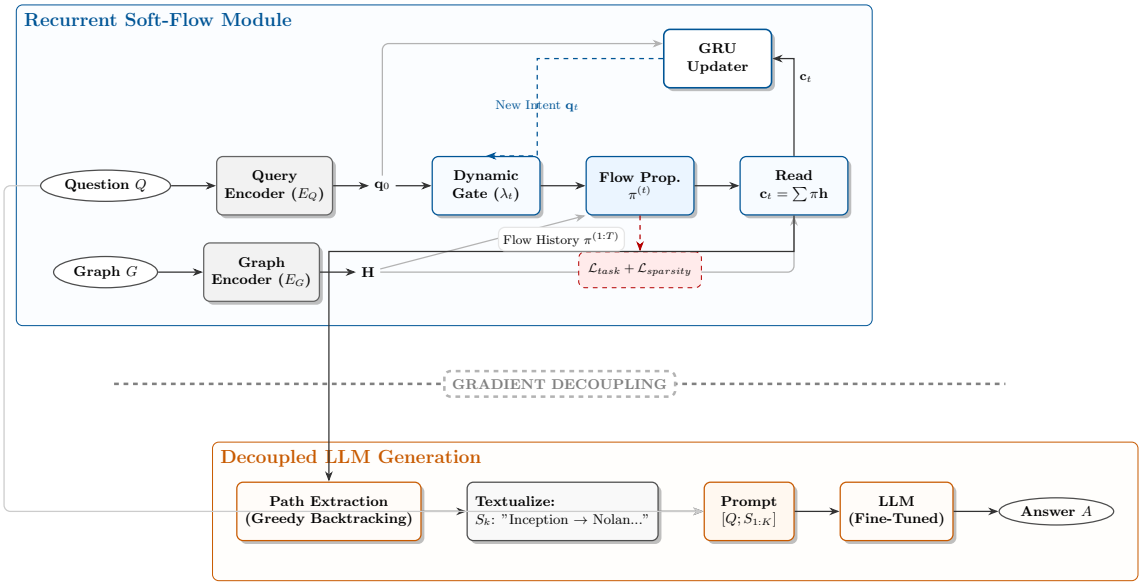

Figure 1. Overall flow diagram of RSF-GLLM

Example: For $Q =$ *"What awards did the director of Inception win?"*, the model retrieves $v_{topic} =$ Inception. We utilize a FAISS index (Johnson et al., 2021) for efficient retrieval.

2.2.2. Subgraph Construction

We extract the T-hop neighborhood around v_{topic} to form the subgraph $\mathcal{G}_{sub} = (\mathcal{V}_{sub}, \mathcal{E}_{sub})$. To ensure alignment between the text question and graph nodes, we re-initialize all node features in $\mathcal{G}_{sub}$ using the **same** encoder used for the question.

$$\mathbf{h}_v^{(0)} = \mathbf{W}_{\mathcal{V}} \cdot E_Q(\text{text}(v)), \quad \mathbf{e}_r = \mathbf{W}_{\mathcal{R}} \cdot E_Q(\text{text}(r)) \quad (2)$$

where $\mathbf{W}_{\mathcal{V}}, \mathbf{W}_{\mathcal{R}}$ are learnable projection matrices.

2.3. Phase 2: Recurrent Graph Reasoning (The RSF Module)

This module performs T steps of differentiable reasoning ($t = 1 \dots T$). At each step, the model updates a probability distribution over nodes (The Flow) (Sun et al., 2018) and evolves the question vector (The Intent) (Hudson & Manning, 2019).

Initialization: The flow $\boldsymbol{\pi}^{(0)}$ is a one-hot vector at v_{topic}. The query state is $\mathbf{q}_0 = \mathbf{v}_Q$.

2.3.1. Step 2.1: Dynamic Relation Attention (The Compass)

At step t, the model decides which edge types are valid. We compute an attention vector $\boldsymbol{\beta}^{(t)}$ over all relations using a Sigmoid activation to allow multi-path traversal:

$$\beta_r^{(t)} = \sigma\left(\mathbf{q}_{t-1}^\top \mathbf{W}_{att} \mathbf{e}_r\right) \quad (3)$$

Example: $\beta^{(1)}_{\texttt{directed\_by}} \approx 1.0$, while $\beta^{(1)}_{\texttt{released\_in}} \approx 0.0$.

2.3.2. Step 2.2: Relation-Guided Flow Propagation

This step updates the node relevance distribution by propagating flow from the previous hop. We decompose this process into structural propagation and conditional semantic refinement.

1. Structural Flow Propagation. Following standard message-passing neural networks (MPNNs) (Gilmer et al., 2017) and previous KGQA frameworks like GraftNet (Sun et al., 2018), we first calculate the probability mass arriving at a node v purely based on graph connectivity and relation validity:

$$\Phi_v^{(t)} = \sum_{(u,r,v) \in \mathcal{N}_{in}(v)} \pi_u^{(t-1)} \cdot \beta_r^{(t)} \quad (4)$$

where $\mathcal{N}_{in}(v)$ denotes the incoming neighborhood. $\Phi_v^{(t)}$ represents the structural likelihood of v being the next step in the reasoning chain.

2. Semantic Disambiguation via Content Bias. Structural flow alone suffers from the *fan-out dilution* problem (Sun et al., 2019) when multiple neighbors are connected via the same valid relation. Consider a modified query: *"Did the director of Inception also direct Interstellar?"*. At hop $t = 2$, the relation $\texttt{directed\_by}^{-1}$ is valid for all movies

directed by Christopher Nolan. Without node content awareness, the structural flow $\Phi_v^{(t)}$ is uniformly diluted across *Interstellar*, *Dunkirk*, and *Tenet*, making the correct target indistinguishable from noise. To resolve this, we introduce a content matching score $\rho_v^{(t)}$:

$$\rho_v^{(t)} = \mathbf{q}_{t-1}^\top \mathbf{W}_{node} \mathbf{h}_v \tag{5}$$

This score measures the alignment between the node and the current query intent, allowing the model to highlight "Interstellar" specifically.

3. Adaptive Gating Mechanism. Indiscriminately applying content bias is detrimental due to the **Semantic Gap**: intermediate reasoning nodes often share no lexical overlap with the query. In our running example *"What awards..."*, the intermediate node "Christopher Nolan" does not semantically match the query word "Awards." Enforcing content bias here would erroneously suppress the correct bridge node. We therefore employ a dynamic gating mechanism to modulate the influence of the content bias:

$$\lambda^{(t)} = \sigma(\mathbf{w}_\lambda^\top \mathbf{q}_{t-1} + b_\lambda) \tag{6}$$

Here, $\lambda^{(t)} \to 1$ when the query seeks specific named entities (resolving fan-out), and $\lambda^{(t)} \to 0$ when the query focuses on structural traversal (bridging the semantic gap).

4. Probabilistic Aggregation. We combine the structural and semantic signals into a unified probability distribution. The aggregation models a *conditional* conjunction: a node must be structurally valid, and—*only if* the gate $\lambda^{(t)}$ is active—semantically relevant. We implement this via summation in log-space:

$$\pi_v^{(t)} = \frac{\exp\left(\log(\Phi_v^{(t)} + \epsilon) + \lambda^{(t)} \cdot \rho_v^{(t)}\right)}{\sum_{z \in \mathcal{V}_{sub}} \exp\left(\log(\Phi_z^{(t)} + \epsilon) + \lambda^{(t)} \cdot \rho_z^{(t)}\right)} \tag{7}$$

where ϵ is a smoothing constant. When $\lambda^{(t)} \approx 0$, the term $\lambda^{(t)} \cdot \rho_v^{(t)}$ vanishes, and the selection relies purely on structural validity $\Phi_v^{(t)}$. When $\lambda^{(t)} \approx 1$, the selection requires both structure and content alignment.

Remark on Re-entrant Paths. Contrary to acyclic traversal constraints often imposed in symbolic reasoning, RSF-GLLM explicitly permits re-entrant paths where flow returns to a previously visited node. This is essential for solving *circular filtering* queries, such as *"Which movies starring Leonardo DiCaprio were directed by the director of Inception?"*. To answer this, the reasoning path must traverse $Inception \xrightarrow{directed\_by} Nolan \xrightarrow{directed} Inception$, returning to the start node to verify the second constraint ("starring DiCaprio"). Strict acyclicity would discard the correct answer ($Inception$) at step $t = 2$. Instead of hard blocking, we rely on the **Query Update** mechanism (Section 2.3.4) to prevent redundant oscillations.

2.3.3. Step 2.3: Context Aggregation (The *Read* Operation)

Once the flow distribution $\boldsymbol{\pi}^{(t)}$ is computed, we summarize the retrieved information into a single differentiable context vector $\mathbf{c}_t$. This operation functions as a *soft read* from the graph memory. We compute $\mathbf{c}_t$ as the expected value of the node embeddings under the current flow distribution:

$$\mathbf{c}_t = \sum_{v \in \mathcal{V}_{sub}} \pi_v^{(t)} \cdot \mathbf{h}_v \tag{8}$$

where $\mathbf{h}_v$ represents the static node embedding initialized in Phase 1. This vector $\mathbf{c}_t$ effectively compresses the semantic content of the *active* region of the graph at hop t into a fixed-size representation, which is then passed to the Recurrent Query Updater to determine the next reasoning step. In our example, $\mathbf{c}_1 \approx$ embedding("Christopher Nolan").

2.3.4. Step 2.4: Differentiable Query Update (The *Write* Operation)

To enable multi-hop reasoning, the model must handle *intent shifting*: resolving one constraint (e.g., finding the "Director") and updating the query to focus on the next (e.g., finding "Awards") (Hudson & Manning, 2019). We model this evolution using a Gated Recurrent Unit (GRU) that acts as a differentiable instruction pointer.

At each hop t, the query vector $\mathbf{q}_t$ is updated based on the retrieved context $\mathbf{c}_t$:

$$\mathbf{q}_t = \text{GRU}(\mathbf{c}_t, \mathbf{q}_{t-1}) \tag{9}$$

where $\mathbf{c}_t$ serves as the input (observation) and $\mathbf{q}_{t-1}$ as the hidden state (intent). Mathematically, the GRU's reset gate $\mathbf{r}_t$ detects features in $\mathbf{c}_t$ that satisfy current query constraints (e.g., identifying "Christopher Nolan"), while the update gate $\mathbf{z}_t$ rotates the vector in semantic space to point towards the next logical hop ("Awards"). This effectively performs *differentiable instruction subtraction*, removing satisfied constraints from the query representation.

We employ a GRU rather than a Transformer or LSTM for this module due to *sample efficiency* and *inductive bias*. Reasoning chains in KGQA are typically short ($T \leq 4$). Transformers often require large-scale data to learn positional dependencies that GRUs capture natively in sequential recurrence. Furthermore, the gating mechanism of the GRU provides a robust *forgetting* bias, which is structurally aligned with the task of discarding resolved query parts, preventing information overload in later hops.

2.4. Phase 3: Path Extraction & Textualization

A critical design choice in RSF-GLLM is the decoupling of graph reasoning from answer generation. It ensures that the LLM's high-variance autoregressive gradients do not

backpropagate into the GNN, allowing the RSF module to learn stable structural priors independently of the generator's linguistic biases.

Path Decoding Mechanism: We employ a *Greedy Backtracking* strategy to ensure that extracted paths strictly explain the model's most confident final predictions. We first identify the top-K answer candidates $\{v_T^{(1)}, \dots, v_T^{(K)}\}$ corresponding to the K highest probability mass values in the final distribution $\boldsymbol{\pi}^{(T)}$.

For each candidate target $v_T^{(k)}$, we reconstruct its causal reasoning chain $P_k = (v_0, \dots, v_T^{(k)})$ in reverse, tracing the flow from the answer back to the source. Specifically, for a node v_t at hop t, we select its predecessor v_{t-1} by choosing the neighbor in the previous distribution $\boldsymbol{\pi}^{(t-1)}$ with the highest probability:

$$v_{t-1} = \underset{u \in \mathcal{N}_{in}(v_t)}{\text{argmax}} \ \pi_u^{(t-1)} \tag{10}$$

We repeat this process recursively from $t = T$ down to $t = 1$ until we reach the topic entity v_{topic} (where $\pi^{(0)} = 1$). This yields a set of top-K paths $\mathcal{P}_{top}$ that represent the most likely structural trajectories leading to the predicted answers.

Textualization: We convert each structured path P_k into a natural language string S_k using a template-based linearizer. For example: *"Inception (film)* $\xrightarrow{directed\_by}$ *Christopher Nolan (director)* $\xrightarrow{won\_award}$ *Academy Award."*

This two-stage approach prevents the high-variance gradients of the LLM from destabilizing the structural learning of the GNN, ensuring robust optimization for both modules. By standardizing the interface as text, the RSF module becomes a universal plugin compatible with any LLM, including proprietary black-box models (e.g., GPT-4) where gradient access is unavailable. Furthermore, explicit textual paths allow for human verification of the reasoning logic prior to generation, eliminating hallucinated retrieval while significantly reducing computational costs compared to end-to-end inference.

2.5. Phase 4: LLM Fine-Tuning

We utilize a pre-trained Large Language Model (LLM) to synthesize the final answer. The LLM is fine-tuned to condition its generation on the extracted paths.

Input Construction: We concatenate the original question Q with the textualized reasoning paths $S_{1:K}$:

$$\mathbf{X}_{input} = \texttt{Context:} \quad [S_1; \dots; S_K] \quad \texttt{Question:} Q$$
$$\mathbf{Y}_{output} = \texttt{Answer:} \quad A$$

Fine-Tuning: The LLM is trained to generate the ground truth answer A autoregressively, minimizing the negative log-likelihood of the answer tokens.

2.6. Training Protocol

We adopt a two-stage training strategy. This decoupling prevents the *gradient noise* from the LLM from destabilizing the graph reasoning process and allows for lighter, more efficient optimization.

2.6.1. Stage 1: RSF Module Training

In this stage, the LLM is frozen (or not present). We train the GNN encoders, Attention weights, and Query GRU to maximize the probability of the correct answer node at the final hop T.

Task Loss ($\mathcal{L}_{task}$): Let v_{ans} be the ground truth answer node. We construct a target one-hot distribution $\mathbf{y}_{gt} \in \{0, 1\}^{|V_{sub}|}$ where $\mathbf{y}_{gt}[v_{ans}] = 1$. We minimize the KL Divergence between the predicted final flow $\boldsymbol{\pi}^{(T)}$ and the ground truth:

$$\mathcal{L}_{task} = \text{KL}(\mathbf{y}_{gt} \,||\, \boldsymbol{\pi}^{(T)}) = -\sum_{v \in \mathcal{V}_{sub}} \mathbf{y}_{gt}[v] \log \boldsymbol{\pi}_v^{(T)} \tag{11}$$

This is equivalent to the Cross-Entropy loss on the final node distribution.

Flow Sparsity Regularization ($\mathcal{L}_{flow}$): A common failure mode in differentiable graph traversal is *flooding*, where the model minimizes risk by assigning uniform non-zero probability to the entire subgraph. This dilutes the context vector $\mathbf{c}_t$ with irrelevant noise. To enforce focused reasoning, we minimize the entropy of the node relevance distribution $\boldsymbol{\pi}^{(t)}$ at each hop t:

$$\mathcal{L}_{flow} = \frac{1}{T} \sum_{t=1}^{T} \left(-\sum_{v \in \mathcal{V}_{sub}} \pi_v^{(t)} \log(\pi_v^{(t)} + \epsilon) \right) \tag{12}$$

where ϵ is a small constant for numerical stability. Minimizing entropy encourages the distribution to be *spiky*, forcing the model to select a distinct reasoning path (e.g., exclusively highlighting "Christopher Nolan") rather than smearing probability mass across all neighbors. This mimics discrete logical steps while retaining differentiability.

Total Stage 1 Loss: $\mathcal{L}_{Stage1} = \mathcal{L}_{task} + \lambda_1 \mathcal{L}_{flow}$.

2.6.2. Stage 2: LLM Optimization

After Stage 1 converges, we freeze the RSF module. We run the RSF module on the training set to extract the top-K paths and generate the prompt $\mathbf{X}_{input}$. We then fine-tune the LLM using standard Causal Language Modeling (CLM)

loss:

$$\mathcal{L}_{Stage2} = -\sum_{j=1}^{|A|} \log P_\theta(a_j \mid a_{<j}, \mathbf{X}_{input}) \quad (13)$$

2.7. Theoretical Analysis

We analyze three critical theoretical properties of RSF-GLLM: its ability to converge to crisp reasoning paths (Sparsity), its capacity to traverse semantically dissimilar nodes (Expressivity), and its capability to ensure structurally faithful reasoning. Detailed proofs of all the theorems are provided in Appendix A.

Theorem 2.1 (Sparsity Convergence)**.** *The global minima of the flow sparsity loss $\mathcal{L}_{flow}(\boldsymbol{\pi}) = -\sum \pi_i \log \pi_i$ lie exclusively at the vertices of the probability simplex Δ^{N-1}.*

The above property ensures that the model avoids flooding the graph with uniform probability, which would dilute the reasoning signal with noise. Instead, the optimization landscape mathematically incentivizes the model to make decisive, unambiguous choices at each hop, mimicking discrete logical steps.

Theorem 2.2 (Semantic Gap Bridging)**.** *Let $P = (v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_K} v_K)$ be a valid reasoning path satisfying:* (i) *intermediate nodes possess zero semantic similarity to the question (i.e., $\mathbf{h}_{v_t}^\top \mathbf{q}_0 \approx 0$ for $0 < t < K$), and* (ii) *for each hop $t \in \{1, \ldots, K\}$, v_t is the unique r_t-neighbor of v_{t-1} in $\mathcal{G}_{sub}$ (no fan-out). There exists a parameter configuration for the Recurrent Query Updater and Relation Attention such that $\pi_{v_t}^{(t)} \approx 1$ at each hop $t \in \{1, \ldots, K\}$, solely via structural propagation.*

This theorem proves that RSF-GLLM can traverse structural bridges (like *Christopher Nolan*) purely based on relation semantics. This allows it to solve complex queries where the intermediate evidence has no lexical overlap with the original question, effectively bypassing the *semantic gap*.

Theorem 2.3 (Guaranteed Structural Faithfulness)**.** *Let v_T be a target node at hop T with non-zero structural flow, i.e., $\Phi_{v_T}^{(T)} > 0$. The greedy backtracking procedure, defined by $v_{t-1} = \text{argmax}_{u \in \mathcal{N}_{in}(v_t)} \pi_u^{(t-1)}$, is guaranteed to reconstruct a continuous, valid path $P = (v_0, \ldots, v_T)$ connecting the source entity v_{topic} to v_T within T steps.*

This theorem provides a critical guaranty for factual grounding. It ensures that every answer generated by our pipeline is supported by an explicit, verifiable reasoning chain existing in the knowledge graph. Unlike heuristic search methods that may fail to connect endpoints or LLM based approaches with the risk of hallucinations, **our flow-based backtracking mathematically ensures connectivity, thereby enforcing structural faithfulness by design**.

3. Experiments

We evaluate RSF-GLLM on two benchmarks requiring multi-hop reasoning over Knowledge Graphs. Our experiments are designed to verify: (1) The superiority of the decoupled *Graph-to-LLM* approach over rigid retrievers, hallucination-prone LLMs and extremely expensive agentic approaches; (2) The efficiency gains from our lightweight RSF module; and (3) The empirical validity of our theoretical claims and model ablations.

3.1. Experimental Setup

Datasets. We utilize two standard KGQA benchmarks:

WebQSP (Yih et al., 2016): Contains approx. 4,700 queries derived from Google Search. It requires up to 2 hops of reasoning and is characterized by high semantic variety.

CWQ (Complex WebQuestions) (Talmor & Berant, 2018): A dataset focusing on complex, compositional queries requiring up to 4 hops. It serves as a stress test for the Recurrent Query Updater's ability to maintain state over long chains.

Baselines: All baselines are listed in Table 1 and discussed in Appendix C. We have only selected baselines that are using open source LLMs with a similar number of model parameters as in Qwen3-8B to have a fair comparison. The reported numbers are mostly taken from the source papers.

Implementation Details. RSF-GLLM is implemented in PyTorch using PyTorch Geometric. Following RoG (Luo et al., 2024), we use preprocessed WebQSP and CWQ datasets with entities pre-linked to Freebase. We extract K-hop subgraphs via BFS ($K = 2$ for WebQSP, $K = 4$ for CWQ). Node and relation embeddings are obtained from Qwen3-Embedding.. The RSF module is trained for up to 10 epochs using AdamW (lr=5×10^{-5}, weight decay 0.01, $\lambda_1 = 0.1$). For the LLM reader, we use Qwen3-8B with extracted reasoning paths. We report Hit@1 and F1 following standard evaluation metrics. All experiments use a single NVIDIA A100 GPU. Baseline details are in Appendix C.

3.2. Main Performance Results

Table 1 presents the performance comparison on WebQSP and CWQ. Our proposed RSF-GLLM achieves state-of-the-art results in Hit@1 on WebQSP and competitive performance on the complex CWQ benchmark, validating the effectiveness of decoupling differentiable graph reasoning from LLM generation.

Comparison with State-of-the-Art LLM+KG Methods: RSF-GLLM demonstrates robust superiority in answer accuracy (Hit@1) over leading baselines in the *LLM + KG* category. On WebQSP, we achieve **90.45%** Hit@1, signif-

Table 1. Performance comparison on WebQSP and CWQ benchmarks. Best results are in **bold**, second-best are underlined. The "Agentic Search" category (shaded) relies on computationally expensive iterative LLM calls, serving as an oracle-style upper bound. RSF-GLLM uses Qwen3-8B or LLaMa-2-7B as the LLM answer generator.

Category	Method	WebQSP		CWQ	
		Hit@1	F1	Hit@1	F1
Embedding-based	KV-Mem (Miller et al., 2016)	46.7	34.5	18.4	15.7
	EmbedKGQA (Saxena et al., 2020)	66.6	–	45.9	–
	NSM (He et al., 2021)	68.7	62.8	47.6	42.4
	TransferNet (Shi et al., 2021)	71.4	–	48.6	–
Graph Retrieval	GraftNet (Sun et al., 2018)	66.4	60.4	36.8	32.7
	PullNet (Sun et al., 2019)	68.1	–	45.9	–
	SR+NSM (Zhang et al., 2022)	68.9	64.1	50.2	47.1
	ReaRev (Mavromatis & Karypis, 2022)	76.4	70.9	52.9	47.8
	UniKGQA (Jiang et al., 2023)	77.2	72.2	51.2	49.1
LLM Reasoning	Qwen3-8B (Zero-shot)	50.1	34.0	27.5	25.8
	LLaMA3.1-8B (Zero-shot)	55.1	35.6	27.7	22.8
LLM + KG	KD-CoT (Wang et al., 2023)	68.6	52.5	55.7	–
	DECAF (FiD-3B) (Yu et al., 2023)	82.1	78.8	70.4	–
	RoG (LLaMA2-7B) (Luo et al., 2024)	85.7	70.8	62.6	56.2
	G-Retreiver (He et al., 2024)	70.1	–	–	–
	GNN-RAG (LLaMA2-7B) (Mavromatis & Karypis, 2025)	80.6	71.3	61.7	59.4
	GNN-RAG + RA (LLaMA2-7B) (Mavromatis & Karypis, 2025)	82.8	73.5	62.8	60.4
Ours	**RSF-GLLM (LLaMA2-7B)**	89.50	78.93	66.44	60.80
	RSF-GLLM (Qwen3-8B)	**90.45**	**79.15**	**67.39**	**61.87**
Agentic Search	EffiQA (Llama3.1-8B) (Dong et al., 2025)	82.9	–	69.5	–
	FD-PORT (LLaMA3-8B) (Shen et al., 2025)	89.2	–	74.5	–

icantly outperforming *RoG* (85.7%), *GNN-RAG* (80.6%), and *DECAF* (82.1%). RSF-GLLM also surpasses *DECAF* on F1 (79.15% vs. 78.8%), demonstrating that our framework achieves superior performance on both Hit@1 and F1 metrics simultaneously.

Crucially, our performance is achieved with superior efficiency. While methods like *GNN-RAG* and *DECAF* require heavy retrieval-augmented generation pipelines or multiple LLM calls, RSF-GLLM employs a lightweight, differentiable RSF module for path extraction. This allows us to use a **single LLM call** for the final answer generation, drastically reducing computational overhead while maintaining high structural grounding. For example, the average inference time per question by RSF-GLLM was around 0.25 sec on both data sets on an A100 80GB machine, which is significantly faster than other competitive baselines.

Comparison with Agentic Search (Oracle Baselines): The *Agentic Search* category (shaded in Table 1) treats KG traversal as a sequential decision process involving iterative LLM invocations (e.g., FD-PORT requires 50+ calls). We consider these methods as high-cost *oracles*. Remarkably, RSF-GLLM surpasses these computationally expensive frameworks on WebQSP (90.45% vs. 89.2% for FD-PORT). This indicates that our differentiable Recurrent Soft-Flow mechanism effectively captures complex reasoning structures akin to agentic planning, but at a fraction of the latency and cost.

Note on the WebQSP–CWQ Hit@1 gap. The raw Hit@1 gap between WebQSP (90.45%) and CWQ (67.39%) partially reflects a structural property of the datasets rather than a reasoning limitation. Under the standard K-hop BFS subgraph construction adopted by all retrieval-based baselines, 19.3% of CWQ test questions have their gold answer entity *outside* the extracted subgraph, imposing a theoretical Hit@1 ceiling of approximately 80.7% on CWQ. WebQSP's coverage under $K{=}2$ is 95.7%, yielding a naturally higher ceiling. A substantial portion of the observed cross-dataset Hit@1 gap is therefore attributable to subgraph coverage limits, not model failure; the K-hop bottleneck this exposes motivates the Dynamic Beam Expansion extension discussed in Section 4.

3.3. Retrieval Performance and Efficiency

Table 2 presents a comprehensive evaluation of retrieval quality versus computational cost. The results highlight distinct limitations in existing baselines that RSF-GLLM effectively overcomes.

Limitations of LLM-Heavy Retrievers: Methodologies like *RoG* rely on fine-tuning large language models (6.7B parameters) to generate reasoning paths directly. Despite its massive size and computational demand (>64GB memory, ∼10k seconds/epoch), RoG performs significantly worse

Table 2. Comparison of retrieval efficiency and performance. Efficiency columns (Params, Time, Mem) are measured for 1 epoch training on CWQ. **#LLM** denotes the number of LLM calls required strictly during the path retrieval phase. H@k measures whether the ground truth answer appears in the top-k candidates provided to the LLM reader.

Method	Params	Time (s)	Mem (MiB)	#LLM	WebQSP			CWQ		
					H@1	H@5	H@10	H@1	H@5	H@10
RoG	6.7B	9671	64,568	1	59.3	73.5	77.7	26.1	46.4	54.1
GNN-RAG	**2.9M**	**737**	**156**	**0**	76.2	85.8	88.0	52.9	64.2	67.0
GNN-RAG+RA	6.7B	10406	64,579	1	**76.3**	87.6	91.0	**55.1**	**69.3**	72.1
RSF (Ours)	176M	993	2,770	**0**	73.9	**89.4**	**93.3**	51.3	67.7	**73.1**

than our approach, achieving only 54.1% Hit@10 compared to RSF's 72.1%. This indicates that generative LLMs, while fluent, often hallucinate relations or lose track of structural constraints in purely parametric generation. Similarly, *GNN-RAG+RA*, an agentic extension of GNN-RAG that integrates an LLM reasoner, improves retrieval to 71.1%. However, this performance gain comes at the cost of extreme complexity—matching RoG's prohibitive resource footprint and requiring slow, iterative LLM calls during retrieval. Even with this agentic overhead, it fails to match the retrieval precision of our lightweight module.

Limitations of Pure GNNs: At the other end of the spectrum, the standard *GNN-RAG* retriever is extremely lightweight (2.9M parameters) and fast. However, its performance (67.0% Hit@10) is inadequate for complex multi-hop QA. Relying primarily on static embedding similarity, it lacks the capacity to model the dynamic, state-dependent reasoning required for compositional queries.

The RSF Advantage: RSF-GLLM occupies a unique *sweet spot*. By designing a specialized differentiable reasoner, we reduce the model size to **176M parameters** ($38\times$ smaller than RoG/GNN-RAG+RA) and eliminate the need for LLM calls during retrieval. Unlike the static GNN-RAG, our Recurrent Soft-Flow mechanism is expressive enough to model complex agentic-like planning steps (Section E), allowing it to achieve state-of-the-art retrieval performance (**72.1% Hit@10**). This demonstrates that a structure-aware module can outperform massive general-purpose LLMs in graph navigation tasks while remaining efficient enough for training on consumer-grade hardware.

3.4. Ablation Study

We validate the contribution of each RSF component through systematic ablations on WebQSP (Table 3).

Content Bias (-6.4%) is the most critical component. Without the content score ρ_v, the model cannot disambiguate fan-out scenarios where multiple neighbors share the same valid relation. When querying about a specific movie by a prolific director, structural flow Φ_v distributes uniformly across all films, making the correct target indistinguishable.

Table 3. Ablation study of RSF module on WebQSP. Δ indicates change from the unablated RSF module performance on the Hit@10 metric.

Configuration	Hit@10	F1	Δ
Full RSF	**89.2**	**52.4**	—
w/o Query Update	83.6	46.2	-5.6
w/o Content Bias	82.8	45.1	-6.4
w/o Dynamic Gate	86.8	49.8	-2.4
w/o Flow Sparsity	85.6	47.5	-3.6

Table 4. Effect of RSF paths vs. QA-pair memorization. "No RSF" fine-tunes the LLM on (question, answer) pairs only, without reasoning paths.

Condition	WebQSP		CWQ	
	H@1	F1	H@1	F1
LLaMA-2-7B (no RSF)	73.16	55.65	46.42	40.34
Qwen3-8B (no RSF)	71.94	54.27	48.21	43.19
LLaMA-2-7B (w/ RSF)	**89.50**	**78.93**	**66.44**	**60.80**
Δ (LLaMA)	+16.34	+23.28	+20.02	+20.46

Query Update (-5.6%) disables the GRU-based intent evolution. Without it, the query vector $\mathbf{q}$ remains frozen across hops, preventing "differentiable instruction subtraction"—the model cannot shift attention from resolving one constraint (e.g., "find the director") to the next (e.g., "find awards").

Dynamic Gate (-2.4%) validates the Semantic Gap hypothesis. Fixing $\lambda \equiv 1$ (see Eq. 6) forces content matching at every hop, suppressing valid bridge entities like "Christopher Nolan" that share no lexical overlap with query terms like "awards."

Flow Sparsity by setting $\lambda_1 = 0$ in Total Stage 1 Loss (-3.6%) prevents "flooding" where probability spreads uniformly, degrading the GRU's ability to track reasoning progress.

Effect of λ_1 Figure 2 reveals an inverted U-curve analogous to *temperature in softmax*: $\lambda_1 \to 0$ yields diffuse distributions ($\sim$11 candidates), while $\lambda_1 \to 0.5$ causes premature commitment ($\sim$2 candidates) where early-hop errors propagate irrecoverably. The optimal $\lambda_1 = 0.1$ achieves $\sim$4 effective candidates, validating Theorem 2.1.

Table 5. Sensitivity to subgraph radius K on CWQ (RSF retrieval Hit@1).

K-hop	1	2	3	4
Hit@1	44.35	45.77	48.34	**51.30**

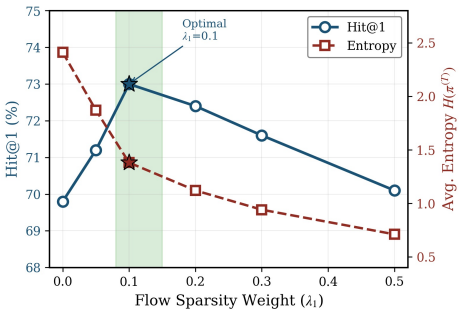

Figure 2. Effect of λ_1 on Hit@1 and entropy. Optimal at $\lambda_1 = 0.1$.

More experiments on dynamic gating (including a CWQ-specific gate ablation in Table 6) and case studies are presented in Appendix D and E respectively.

Isolating RSF Path Contribution: To verify that performance gains stem from RSF-extracted reasoning paths rather than QA-pair memorization, we trained an identical LLM on (question, answer) pairs without any RSF paths (Table 4). The +16–23pt gap across all metrics confirms that the LLM cannot recover structural and factual information from questions alone; RSF paths provide essential grounding that drives the performance improvement.

Sensitivity to Subgraph Radius: We conducted a sensitivity study varying the K-hop subgraph radius on CWQ (Table 5). Results show a consistent improvement in RSF retrieval Hit@1 with increased hop budget (44.35% at K=1 to 51.30% at K=4), highlighting the importance of multi-hop reasoning capacity and our design choice to include self-loops.

4. Conclusion and Future Work

We introduced RSF-GLLM, a framework that decouples differentiable graph reasoning from LLM generation to bridge the semantic gap in KGQA. Our Recurrent Soft-Flow module employs dynamic gating and flow sparsity regularization to extract interpretable reasoning paths, avoiding the computational cost of iterative agentic loops. Experiments demonstrate that RSF-GLLM achieves state-of-the-art retrieval accuracy and a 21× inference speedup over baselines by requiring only a single LLM call.

Future work will address two limitations of the current pipeline through extensions naturally enabled by the decoupled RSF–LLM design. To scale beyond the pre-extracted K-hop subgraph regime—which can bottleneck recall on massive real-world KGs—we plan to introduce *Dynamic Beam Expansion*, a reasoning-step-driven traversal in which the model dynamically queries the backend graph database for the 1-hop neighbors of *active* nodes ($\pi_v^{(t)} > \theta$), guided by the Flow Sparsity regularizer to keep the active set small and by an adaptive GRU-based halting criterion that eliminates the fixed hop count T. To mitigate the popularity bias of the LLM generator (Appendix E, Failure Mode 2), we plan to integrate *Context-Aware Contrastive Decoding* at inference: by contrasting LLM predictions conditioned on the question alone (capturing the parametric prior) against predictions conditioned on the question together with the extracted reasoning paths, we can dynamically penalize tokens favored purely by the internal prior and amplify predictions supported by retrieved structural context. Because RSF cleanly decouples reasoning from generation, this contrastive correction can be applied without retraining the underlying LLM. Beyond these, we plan to extend RSF-GLLM to temporal and multi-modal knowledge graphs and to investigate joint RSF–LLM optimization.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning, specifically in interpretable and efficient Question Answering. Our proposed RSF-GLLM framework explicitly addresses the critical issue of hallucination in Large Language Models by grounding generation in verifiable Knowledge Graph paths, thereby contributing to more reliable and trustworthy AI systems. Furthermore, by decoupling reasoning from generation, our approach significantly reduces the computational resources required for multi-hop inference compared to iterative agentic baselines, promoting more energy-efficient and environmentally sustainable AI deployment.

References

Asai, A., Wu, Z., Wang, Y., Sil, A., and Hajishirzi, H. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., and Hoefler, T. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17682–17690, 2024. doi: 10.1609/aaai.v38i16.29720.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G.,

Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Dong, Z., Peng, B., Wang, Y., Fu, J., Wang, X., Zhou, X., Shan, Y., Zhu, K., and Chen, W. EffiQA: Efficient question-answering with strategic multi-model collaboration on knowledge graphs. In Rambow, O., Wanner, L., Apidianaki, M., Al-Khalifa, H., Eugenio, B. D., and Schockaert, S. (eds.), *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 7180–7194, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.479/.

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., and Wang, H. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pp. 1263–1272. PMLR, 2017.

He, G., Lan, Y., Jiang, J., Zhao, W. X., and Wen, J.-R. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 553–561. ACM, 2021. doi: 10.1145/3437963.3441753.

He, X., Tian, Y., Sun, Y., Chawla, N. V., Laurent, T., LeCun, Y., Bresson, X., and Hooi, B. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://openreview.net/forum?id=MPJ3oXtTZl.

Hernandez, E., Sharma, A. S., Haklay, T., Meng, K., Wattenberg, M., Andreas, J., Belinkov, Y., and Bau, D. Linearity of relation decoding in transformer language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=w7LU2s14kE.

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., et al. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37, 2021.

Hudson, D. A. and Manning, C. D. Learning by abstraction: The neural state machine. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y., Madotto, A., and Fung, P. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Jiang, J., Zhou, K., Zhao, W. X., and Wen, J.-R. UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Z63RvyAZ2Vh.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2021.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

Kim, G., Kim, S., Jeon, B., Park, J., and Kang, J. Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 996–1009, 2023.

Li, X., Zhao, R., Chia, Y. K., Ding, B., Joty, S., Poria, S., and Bing, L. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *ICLR*, 2024.

Liang, K., Meng, L., Liu, M., Liu, Y., Tu, W., Wang, S., Zhou, S., Liu, X., Sun, F., and He, K. A survey of knowledge graph reasoning on graph types: Static, dynamic, and multi-modal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9456–9478, 2024.

Luo, L., Li, Y.-F., Haffari, G., and Pan, S. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ZGNWW7xZ6Q.

Mavromatis, C. and Karypis, G. ReaRev: Adaptive reasoning for question answering over knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 2447–2458, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.181.

Mavromatis, C. and Karypis, G. GNN-RAG: Graph neural retrieval for efficient large language model reasoning on knowledge graphs. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 16682–16699, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.856. URL https://aclanthology.org/2025.findings-acl.856/.

Miller, A., Fisch, A., Dodge, J., Karimi, A.-H., Bordes, A., and Weston, J. Key-value memory networks for directly reading documents. In Su, J., Duh, K., and Carreras, X. (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1400–1409, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1147. URL https://aclanthology.org/D16-1147/.

Nandy, A. and Bandyopadhyay, S. Language models of code are few-shot planners and reasoners for multi-document summarization with attribution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24930–24938, 2025.

Park, K., Choe, Y. J., and Veitch, V. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39643–39666. PMLR, 2024. URL https://proceedings.mlr.press/v235/park24c.html.

Saxena, A., Tripathi, A., and Talukdar, P. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4498–4507, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.412. URL https://aclanthology.org/2020.acl-main.412/.

Shen, T., Mao, R., Wang, J., Zhang, X., and Cambria, E. Flow-guided direct preference optimization for knowledge graph reasoning with trees. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, pp. 1165–1175, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715921. doi: 10.1145/3726302.3729980. URL https://doi.org/10.1145/3726302.3729980.

Shi, J., Cao, S., Hou, L., Li, J., and Zhang, H. TransferNet: An effective and transparent framework for multi-hop question answering over relation graph. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 4149–4158, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.341. URL https://aclanthology.org/2021.emnlp-main.341/.

Sun, H., Dhingra, B., Zaheer, M., Mazaitis, K., Salakhutdinov, R., and Cohen, W. W. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4231–4242, 2018.

Sun, H., Bedrax-Weiss, T., and Cohen, W. W. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. 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)*, pp. 2380–2390, Hong Kong, China, 2019. Association for Computational Linguistics.

Sun, J., Xu, C., Tang, L., Wang, S., Lin, C., Gong, Y., Ni, L. M., Shum, H.-Y., and Guo, J. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nnVO1PvbTv.

Talmor, A. and Berant, J. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 641–651, 2018.

Wang, K., Duan, F., Wang, S., Li, P., Xian, Y., Yin, C., Rong, W., and Xiong, Z. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering, 2023. URL https://arxiv.org/abs/2308.13259.

Wang, M., Qiu, L., and Wang, X. A survey on knowledge graph embeddings for link prediction. *Symmetry*, 13(3): 485, 2021.

Xiong, W., Hoang, T., and Wang, W. Y. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 564–573, 2017.

Yan, S.-Q., Gu, J.-C., Zhu, Y., and Ling, Z.-H. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*, 2024.

Yih, W.-t., Richardson, M., Meek, C., Chang, M.-W., and Suh, J. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 201–206, 2016.

Yu, D., Zhang, S., Ng, P., Zhu, H., Li, A. H., Wang, J., Hu, Y., Wang, W. Y., Wang, Z., and Xiang, B. DecAF: Joint decoding of answers and logical forms for question answering over knowledge bases. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XHc5zRPxqV9.

Zhang, J., Zhang, X., Yu, J., Tang, J., Tang, J., Li, C., and Chen, H. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5773–5784, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.396. URL https://aclanthology.org/2022.acl-long.396/.

Appendix

A. Theoretical Proofs

A.1. Proof of Theorem 2.1 (Sparsity Convergence)

Statement: *The global minima of the flow sparsity loss $\mathcal{L}_{flow}(\boldsymbol{\pi})$ lie exclusively at the vertices of the simplex Δ^{N-1}.*

Proof. Let $\boldsymbol{\pi} \in \mathbb{R}^N$ be the relevance distribution vector satisfying the simplex constraints: $\sum_{i=1}^{N} \pi_i = 1$ and $\pi_i \geq 0$. The Flow Sparsity Regularization term is defined as the entropy of the distribution:

$$\mathcal{L}_{flow}(\boldsymbol{\pi}) = H(\boldsymbol{\pi}) = -\sum_{i=1}^{N} \pi_i \log \pi_i \tag{14}$$

To analyze the minima, we inspect the curvature of the function. The Hessian of $\mathcal{L}_{flow}$ with respect to $\boldsymbol{\pi}$ is a diagonal matrix with elements:

$$\frac{\partial^2 \mathcal{L}_{flow}}{\partial \pi_i^2} = -\frac{1}{\pi_i} \tag{15}$$

Since $\pi_i > 0$ in the interior of the simplex, the second derivative is strictly negative. Thus, $\mathcal{L}_{flow}(\boldsymbol{\pi})$ is a **strictly concave** function over the domain.

A fundamental result in convex optimization states that the minimum of a strictly concave function over a convex polytope (such as the simplex Δ^{N-1}) must occur at one of the extreme points (vertices) of the polytope. The vertices of the probability simplex are the standard basis vectors $\{\mathbf{e}_1, \ldots, \mathbf{e}_N\}$, which correspond to one-hot distributions (i.e., $\pi_k = 1$ for some k, and 0 otherwise).

Evaluating the loss at a vertex $\mathbf{e}_k$:

$$\mathcal{L}_{flow}(\mathbf{e}_k) = -(1 \log 1 + \sum_{j \neq k} 0 \log 0) = 0 \tag{16}$$

(taking $\lim_{x \to 0} x \log x = 0$). Since entropy is non-negative $H(\boldsymbol{\pi}) \geq 0$, the value 0 is the global minimum. Therefore, minimizing $\mathcal{L}_{flow}$ forces the model to select a single crisp path node at each hop, mimicking discrete reasoning. □

A.2. Proof of Theorem 2.2 (Semantic Gap Bridging)

Statement: *Let $P = (v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_K} v_K)$ be a valid reasoning path satisfying: (i) intermediate nodes possess zero semantic similarity to the question (i.e., $\mathbf{h}_{v_t}^\top \mathbf{q}_0 \approx 0$ for $0 < t < K$), and (ii) for each hop $t \in \{1, \ldots, K\}$, v_t is the unique r_t-neighbor of v_{t-1} in $\mathcal{G}_{sub}$ (no fan-out). There exists a parameter configuration for the Recurrent Query Updater and Relation Attention such that $\pi_{v_t}^{(t)} \approx 1$ at each hop $t \in \{1, \ldots, K\}$, solely via structural propagation.*

Proof. We provide a constructive proof by induction on the hop index t. Let the path be $P = (v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \ldots \xrightarrow{r_K} v_K)$, and let the Semantic Gap assumption hold: $\text{sim}(\mathbf{h}_{v_t}, \mathbf{q}_0) = 0$ for all $t > 0$. We aim to show that for each hop $t \in \{1, \ldots, K\}$, there exists a parameter configuration such that $\pi_{v_t}^{(t)} \approx 1$.

Step 0: Gate Construction (Suppressing Content Bias). Set $b_\lambda \ll 0$ in Eq. 6. Then $\lambda^{(t)} = \sigma(\mathbf{w}_\lambda^\top \mathbf{q}_{t-1} + b_\lambda) \approx 0$ for all t, regardless of the query state. The probabilistic aggregation (Eq. 7) then simplifies. Since $\exp(\log(x)) = x$, it reduces directly to a ratio of structural flow values:

$$\pi_v^{(t)} \approx \frac{\Phi_v^{(t)}}{\sum_z \Phi_z^{(t)}} \tag{17}$$

(the additive ϵ terms cancel and are negligible in the regime $c \to \infty$ established in Step 1).

Step 1: Relation Alignment (Existence of $\mathbf{W}_{att}$). Assume the relation embeddings $\{\mathbf{e}_r\}_{r \in \mathcal{R}}$ are linearly independent. Their dual basis $\{\mathbf{d}_r\}_{r \in \mathcal{R}}$ then exists, satisfying $\mathbf{d}_r^\top \mathbf{e}_{r'} = \delta_{rr'}$ (Kronecker delta). For a target relation r_t, define the direction:

$$\mathbf{f}_{r_t} = c \left(\mathbf{d}_{r_t} - \alpha \sum_{r'' \neq r_t} \mathbf{d}_{r''} \right), \quad c, \alpha > 0 \tag{18}$$

Then $\mathbf{f}_{r_t}^\top \mathbf{e}_{r_t} = c$ and $\mathbf{f}_{r_t}^\top \mathbf{e}_{r''} = -c\alpha$ for all $r'' \neq r_t$. We fix $\mathbf{W}_{att}$ as any full-rank matrix from the query space to the relation embedding space; its specific value does not affect the construction. Since $\mathbf{W}_{att}$ is full-rank, for each hop's target direction $\mathbf{f}_{r_t}$ there exists a query state satisfying $\mathbf{W}_{att}^\top \mathbf{q}_{t-1} = \mathbf{f}_{r_t}$; Step 2 below constructs GRU weights to produce these target query states at each hop. With $\mathbf{q}_{t-1}$ so positioned, the relation attention scores (Eq. 3) evaluate to $\mathbf{q}_{t-1}^\top \mathbf{W}_{att} \mathbf{e}_{r_t} = \mathbf{f}_{r_t}^\top \mathbf{e}_{r_t} = c$ and $\mathbf{q}_{t-1}^\top \mathbf{W}_{att} \mathbf{e}_{r''} = \mathbf{f}_{r_t}^\top \mathbf{e}_{r''} = -c\alpha$, yielding:

$$\beta_{r_t}^{(t)} = \sigma(c) \;\to\; 1 \qquad \text{and} \qquad \beta_{r''}^{(t)} = \sigma(-c\alpha) \;\to\; 0 \quad \forall\, r'' \neq r_t \tag{19}$$

as $c \to \infty$.

With $\beta_{r_t}^{(t)} \approx 1$ and $\beta_{r''}^{(t)} \approx 0$ for all $r'' \neq r_t$, the structural flow propagation concentrates on nodes reachable from v_{t-1} via r_t. Using the inductive hypothesis $\pi_{v_{t-1}}^{(t-1)} \approx 1$:

$$\Phi_{v_t}^{(t)} = \sum_{(u,\, r,\, v_t) \,\in\, \mathcal{N}_{in}(v_t)} \pi_u^{(t-1)} \cdot \beta_r^{(t)} \;\approx\; \pi_{v_{t-1}}^{(t-1)} \cdot \beta_{r_t}^{(t)} \;\approx\; 1 \tag{20}$$

$$\Phi_z^{(t)} \approx\; 0 \qquad \text{for all } z \text{ reachable from } v_{t-1} \text{ only via } r'' \neq r_t \tag{21}$$

By condition (ii) of the theorem, v_t is the unique r_t-neighbor of v_{t-1} in $\mathcal{G}_{sub}$, so no other node receives $\Phi \approx 1$. Substituting into the simplified softmax of Step 0, we obtain $\pi_{v_t}^{(t)} \approx 1$.

Step 2: Dynamic Update (Existence of GRU parameters). After traversing edge r_t, the query must update so that $\mathbf{W}_{att}^\top \mathbf{q}_t = \mathbf{f}_{r_{t+1}}$, aligning with the next target relation r_{t+1}. The update rule is $\mathbf{q}_t = \text{GRU}(\mathbf{c}_t, \mathbf{q}_{t-1})$, where $\mathbf{c}_t = \sum_v \pi_v^{(t)} \mathbf{h}_v \approx \mathbf{h}_{v_t}$ since $\pi_{v_t}^{(t)} \approx 1$. Since GRUs with sufficient hidden state dimension are universal approximators of dynamical systems, there exist weights implementing a state transition f such that:

$$\mathbf{q}_t = f(\mathbf{q}_{t-1},\, \mathbf{h}_{v_t}) \quad \text{s.t.} \quad \mathbf{W}_{att}^\top \mathbf{q}_t = \mathbf{f}_{r_{t+1}} \tag{22}$$

Even when $\mathbf{h}_{v_t}$ shares no lexical similarity with the query, it acts as a distinct structural key: the GRU learns the state transition "currently aligned with relation r_t; upon observing node v_t, realign with r_{t+1}."

Conclusion. The base case $\pi_{v_0}^{(0)} = 1$ holds by initialization. Steps 1–2 establish the inductive step: $\pi_{v_{t-1}}^{(t-1)} \approx 1$ implies $\pi_{v_t}^{(t)} \approx 1$. Therefore, at each hop $t \in \{1, \ldots, K\}$, the flow is concentrated on the correct path node v_t, confirming that the RSF architecture bridges the semantic gap using only structural cues (relation type attention $\beta_r^{(t)}$, Eq. 3) and state memory (GRU), without relying on node-question lexical similarity. □

Remark on the linear independence assumption. Step 1 assumes that the relation embeddings $\{\mathbf{e}_r\}_{r \in \mathcal{R}}$ are linearly independent, which guarantees the existence of a dual basis enabling simultaneous concentration ($\beta_{r_t} \to 1$) and suppression ($\beta_{r''} \to 0$) of relation attention scores. In practice, this condition is generically satisfied for high-dimensional embeddings: a set of $|\mathcal{R}|$ vectors in $\mathbb{R}^d$ with $d \gg |\mathcal{R}|$ is almost surely linearly independent under any continuous distribution. Furthermore, recent LLM probing studies show that relational structures in high-dimensional dense representations—such as those produced by Qwen3-Embedding—tend to emerge as approximately linearly independent subspaces (Hernandez et al., 2024; Park et al., 2024). The theorem should therefore be read as an *architectural-capacity* guarantee: it shows that the RSF parameterization is expressive enough to realize the required relation alignment whenever the linear independence condition is met by the underlying embedding space.

A.3. Proof of Theorem 2.3 (Guaranteed Structural Faithfulness)

Statement: Let v_T be a target node at hop T with non-zero structural flow, i.e., $\Phi_{v_T}^{(T)} > 0$. The greedy backtracking procedure, defined by $v_{t-1} = \text{argmax}_{u \in \mathcal{N}_{in}(v_t)} \pi_u^{(t-1)}$, is guaranteed to reconstruct a continuous, valid path $P = (v_0, \ldots, v_T)$ connecting the source entity v_{topic} to v_T within T steps.

Proof. Recall the structural flow propagation equation for a node v at hop t in the RSF module:

$$\Phi_v^{(t)} = \sum_{(u,r,v) \in \mathcal{E}} \pi_u^{(t-1)} \cdot \beta_r^{(t)} \tag{23}$$

where $\pi_u^{(t-1)}$ is the probability mass at predecessor node u, and $\beta_r^{(t)} \in [0, 1]$ is the relation attention score produced by the Dynamic Relation Attention module (Eq. 3). The greedy backtracking algorithm selects the predecessor at step t via:

$$v_{t-1}^* = \underset{u \in \mathcal{N}_{in}(v_t)}{\operatorname{argmax}} \ \pi_u^{(t-1)} \tag{24}$$

We assume the initialization $\boldsymbol{\pi}^{(0)}$ is a one-hot vector at the source node v_{topic}, such that $\pi_v^{(0)} = 1$ if $v = v_{topic}$ and 0 otherwise.

We prove the theorem by induction on the hop index t, proceeding backwards from the target hop T down to the source hop 0.

Base Assumption (Step T): The premise of the theorem states that the selected target node v_T has received non-zero structural flow: $\Phi_{v_T}^{(T)} > 0$.

Inductive Step: Consider an arbitrary node v_t selected at hop t (where $0 < t \leq T$) such that its structural flow $\Phi_{v_t}^{(t)} > 0$. By the definition of the flow summation, if $\Phi_{v_t}^{(t)} > 0$, the sum $\sum_u \pi_u^{(t-1)} \beta_r^{(t)}$ is strictly positive. Since probabilities π and attention scores β are non-negative, this implies that the set of contributing predecessors is non-empty. Specifically, there exists at least one neighbor $u' \in \mathcal{N}_{in}(v_t)$ such that:

$$\pi_{u'}^{(t-1)} \cdot \beta_{r'}^{(t)} > 0 \implies \pi_{u'}^{(t-1)} > 0 \tag{25}$$

The greedy backtracking algorithm selects the predecessor v_{t-1}^* that maximizes $\pi_u^{(t-1)}$. Since the set of neighbors with non-zero probability is non-empty (containing at least u'), the maximum value in this set must be strictly positive:

$$\pi_{v_{t-1}^*}^{(t-1)} \geq \pi_{u'}^{(t-1)} > 0 \tag{26}$$

Thus, the selected predecessor v_{t-1}^* has non-zero probability mass. If $t - 1 > 0$, this node v_{t-1}^* must have received flow from its own predecessors (since $\pi^{(t-1)}$ is derived from the flow $\Phi^{(t-1)}$). Therefore, $\Phi_{v_{t-1}^*}^{(t-1)} > 0$. This establishes the inductive step: if the node at t has valid flow, the selected node at $t - 1$ also has valid flow.

Termination (Step 0): The recursion proceeds until we reach $t = 0$, selecting a node v_0 with $\pi_{v_0}^{(0)} > 0$. By definition, the initial distribution $\boldsymbol{\pi}^{(0)}$ is a one-hot vector peaked at v_{topic}. The only node with non-zero probability at $t = 0$ is v_{topic}. Therefore, it must be that $v_0 = v_{topic}$.

Conclusion: The sequence of nodes $P = (v_0, v_1, \ldots, v_T)$ constructed by the backtracking procedure satisfies two conditions: 1. $v_0 = v_{topic}$. 2. For every step $t > 0$, v_{t-1} is a valid neighbor of v_t in $\mathcal{G}$ (specifically, the neighbor contributing maximal flow).

Consequently, P forms a valid, continuous reasoning path in the Knowledge Graph starting at the source entity and ending at the answer candidate. □

B. Related Work

We position our work within the broader landscape of Knowledge Graph Question Answering (KGQA), categorized into subgraph retrieval, LLM-integrated frameworks, and agentic reasoning.

Subgraph Retrieval and Reasoning. Early neural approaches to KGQA formulated reasoning as a path traversal or subgraph retrieval problem. Methods like **GraftNet** (Sun et al., 2018) and **PullNet** (Sun et al., 2019) employ a *Retrieve-then-Read* paradigm, identifying a discrete subgraph via heuristic or learned retrieval before applying a GNN reader. However, the discrete selection of nodes breaks end-to-end differentiability, preventing the retriever from adapting to downstream reasoning errors. While **NSM** (Neural State Machine) (He et al., 2021) introduced a differentiable instruction signal, it relies on fixed node embeddings, causing it to fail when intermediate *bridge* nodes lack lexical overlap with the query (the *Semantic Gap*). **EmbedKGQA** (Saxena et al., 2020) mitigates this via latent space matching but lacks the explicit multi-hop traversal required for complex compositional queries. *Our Contribution:* RSF-GLLM addresses the non-differentiability of retrieval via continuous soft-flow propagation and resolves the Semantic Gap using a novel Dynamic Gating Mechanism that prioritizes structural validity over semantic matching when necessary.

LLM-KG Integration. Recent paradigms leverage the generative power of LLMs to interpret graph structures. Frameworks like **RoG** (Reasoning on Graphs) (Luo et al., 2024) and **GNN-RAG** generate reasoning paths as planning steps or retrieve graph context to augment generation. While effective, these methods face two critical limitations: (1) **Computational Cost:** RoG requires fine-tuning large base models (e.g., LLaMA-2-7B), imposing significant memory and training overheads. (2) **Faithfulness:** As noted in recent studies (Gao et al., 2023; Li et al., 2024), retrieval-augmented LLMs are prone to *reasoning shortcuts*, ignoring retrieved structures in favor of parametric memory. *Our Contribution:* We propose a decoupled architecture where reasoning is handled by a lightweight ($\sim$40ms) GNN module. This avoids the cost of LLM fine-tuning for reasoning and ensures grounding by conditioning the frozen LLM strictly on extracted paths.

Agentic and Chain-of-Thought Reasoning. The state-of-the-art in complex reasoning involves treating LLMs as autonomous agents. Methods like **ToG** (Think-on-Graph) (Sun et al., 2024) and **Graph of Thoughts** (Besta et al., 2024) perform iterative beam search or Monte Carlo Tree Search (MCTS) over the KG, invoking the LLM at every hop to evaluate candidates. While achieving high accuracy, these *agentic* approaches are prohibitively slow, often requiring 10–50+ LLM calls per query (Sun et al., 2024; Besta et al., 2024). *Our Contribution:* RSF-GLLM achieves comparable multi-hop reasoning depth with a *single* LLM call. By offloading the iterative search to our Recurrent Soft-Flow module, we reduce inference latency by orders of magnitude compared to agentic baselines while maintaining theoretical guarantees on path sparsity and validity.

C. Baselines

We evaluate our proposed method against five categories of state-of-the-art baselines, ranging from pure parametric reasoning to specialized graph-agent frameworks.

1. **Embedding-based Methods:** This category includes traditional neural approaches that learn to propagate signals or match questions in a latent embedding space. We compare against *KV-Mem*, which uses key-value memory networks for reading documents; *EmbedKGQA*, which leverages knowledge graph embeddings for link prediction; *NSM* (Neural State Machine), which simulates sequential reasoning steps; and *TransferNet*, which utilizes a differentiable framework to propagate attention scores across graph relations.

2. **Graph Retrieval Methods:** These frameworks focus on identifying question-relevant subgraphs or using neural traversals to narrow the search space. This category includes *GraftNet*, which uses static embedding matching; *PullNet*, which iteratively retrieves relevant subgraphs; *SR+NSM*, a stepwise neural reasoner with subgraph retrieval; *ReaRev*, which adaptively revises reasoning instructions based on graph feedback; and *UniKGQA*, a unified architecture for retrieval and reasoning.

3. **LLM Reasoning:** These methods evaluate the parametric knowledge of Large Language Models (LLMs) without providing external Knowledge Graph (KG) context. We evaluate *Qwen3-8B* and *LLaMA3.1-8B* using zero-shot prompting to establish a baseline for pure LLM reasoning capability.

4. **LLM + KG Integration:** These methods leverage the reasoning power of LLMs by augmenting their context with retrieved KG facts. We compare against *KD-CoT*, an interactive framework for faithful reasoning; *DECAF*, which jointly decodes answers and logical forms using Fusion-in-Decoder; *RoG*, which generates grounded reasoning plans; *G-Retriever*, a retrieval-augmented generation method for graphs; and *GNN-RAG*, which uses Graph Neural Networks to retrieve paths for LLM reasoning.

5. **Agentic Search Frameworks:** The most recent frontier treats the LLM as an autonomous agent that explores the KG through sequential decision-making. This includes *EffiQA*, which employs strategic multi-model collaboration for efficient QA, and *FD-PORT*, which utilizes Monte Carlo Tree Search (MCTS) and flow-guided optimization to navigate complex multi-hop reasoning paths.

D. Dynamic Gating Analysis

To empirically validate the semantic gap bridging hypothesis (Theorem 2.2), we analyze the evolution of the dynamic gating weight $\lambda^{(t)}$ across reasoning hops. Recall that $\lambda^{(t)} \to 0$ indicates structure-focused propagation (traversing via graph topology), while $\lambda^{(t)} \to 1$ indicates content-focused selection (matching node semantics to the query).

Figure 3 presents the mean $\lambda^{(t)}$ values aggregated across all 4-hop questions in the CWQ test set. The results reveal an interesting pattern that validates our theoretical framework.

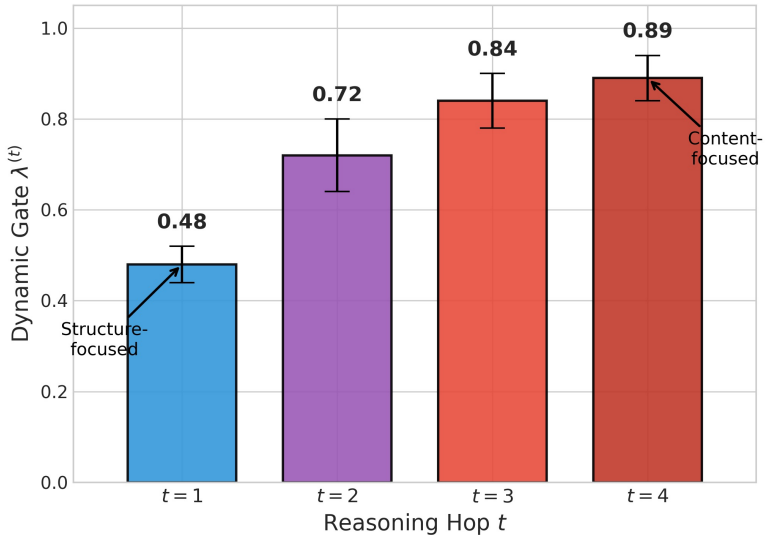

Figure 3. Evolution of dynamic gating weight $\lambda^{(t)}$ across reasoning hops on CWQ 4-hop questions. Error bars indicate standard deviation. The monotonic increase from structure-focused ($\lambda \approx 0.48$) to content-focused ($\lambda \approx 0.89$) validates the semantic gap bridging hypothesis.

Interpretation. The learned gating schedule demonstrates an adaptive transition from structure-focused to content-focused reasoning. At earlier hops, λ remains low (≈ 0.48), indicating that the model prioritizes graph topology over semantic matching. This is precisely the behavior required to bridge the semantic gap: when traversing from a topic entity like "Inception" to an intermediate node like "Christopher Nolan," lexical similarity to query terms such as "awards" would be misleading, so the model relies on valid relation types to guide propagation. As reasoning progresses, λ increases steadily, reflecting the model's growing confidence in using content signals for disambiguation—particularly important in fan-out scenarios where multiple neighbors share the same valid relation. At later hops ($\lambda \approx 0.89$), content matching dominates, ensuring the final answer aligns semantically with the original query intent. This monotonic increase emerges naturally from end-to-end training without explicit supervision on λ values, demonstrating that RSF learns to adaptively balance structure and semantics based on the reasoning stage.

We also remark that individual case by case analysis reveals that we notice samples with $\lambda < 0.5$ in the final hop, notably when the answer node itself is a generic entity (eg. "National Language").

CWQ Gate Ablation. To isolate the dynamic gate's effect on the harder CWQ benchmark, we trained RSF from scratch without the dynamic gate ($\lambda \equiv 1$). Table 6 reports the results. The gate provides a consistent benefit across all retrieval metrics, with the largest impact on H@1 (-4.0pt), substantially larger than the -2.4pt H@10 drop observed on WebQSP (Table 3). This confirms that CWQ's more complex compositional queries expose the gate's value more clearly. We note that CWQ does not provide explicit hop-depth annotations—questions are categorized by composition type, not reasoning depth—so a hop-stratified breakdown is not available from the dataset.

Table 6. Dynamic gate ablation on CWQ retrieval. Removing the gate ($\lambda \equiv 1$) causes a larger H@1 drop (-4.0pt) than on WebQSP (-2.4pt H@10, Table 3), confirming the gate's value scales with query complexity.

Model	**H@1**	**H@5**	**H@10**
RSF (with gate)	51.3	67.7	72.1
RSF (w/o gate, λ=1)	47.3	64.6	69.8
Δ	-4.0	-3.1	-2.3

E. Qualitative Analysis of Reasoning Chains

To provide deeper insight into RSF-GLLM's reasoning behavior, we present representative examples of successful and failed predictions from the CWQ and WebQSP test sets. These examples illustrate the model's ability to perform complex multi-hop reasoning while also revealing systematic failure modes.

E.1. Successful Multi-Hop Reasoning

The following examples demonstrate RSF-GLLM's ability to correctly navigate 3–4 hop reasoning chains, including cases requiring traversal through semantically dissimilar intermediate nodes.

Example 1: Geographic → Political Inference (CWQ, 4-hop)

Question: "What type of government is used in the country with Northern District?"
Topic Entity: Northern District **Ground Truth:** Parliamentary System
Prediction: Parliamentary System **F1:** 1.00

Northern District $\xrightarrow{\text{administrative\_division.country}}$ Israel $\xrightarrow{\text{governmental\_jurisdiction.governing\_officials}}$ [position_m.0j_m3_b] $\xrightarrow{\text{government\_position\_held.jurisdiction\_of\_office}}$ Israel $\xrightarrow{\text{location.country.form\_of\_government}}$ Parliamentary System

Analysis: This example demonstrates clean 4-hop reasoning through geographic and political knowledge. Starting from an obscure administrative district, the model correctly identifies Israel as the parent country, navigates through government official positions (CVT nodes), and extracts the governmental structure.

Example 2: Multi-Answer Language Retrieval (CWQ, 4-hop)

Question: "The people from the country that contains Nord-Ouest Department speak what languages today?"
Topic Entity: Nord-Ouest Department **Ground Truth:** Haitian Creole | French
Prediction: French, Haitian Creole **F1:** 1.00

Path 1: Nord-Ouest Department $\xrightarrow{\text{country}}$ Haiti $\xrightarrow{\text{governing\_officials}}$ [m.010g48js] $\xrightarrow{\text{jurisdiction\_of\_office}}$ Haiti $\xrightarrow{\text{official\_language}}$ French
Path 2: Nord-Ouest Department $\xrightarrow{\text{country}}$ Haiti $\xrightarrow{\text{governing\_officials}}$ [m.010g48js] $\xrightarrow{\text{jurisdiction\_of\_office}}$ Haiti $\xrightarrow{\text{official\_language}}$ Haitian Creole

Analysis: This example showcases RSF's ability to retrieve *multiple correct answers* through parallel reasoning paths. Both official languages of Haiti are successfully identified via paths that share the first three hops but diverge at the final relation. The flow sparsity regularization allows the model to maintain multiple high-probability candidates when the question semantics warrant it, rather than collapsing to a single answer prematurely.

Example 3: Historical Multi-Answer Question (WebQSP, 3-hop)

Question: "What else did Ben Franklin invent?"
Topic Entity: Benjamin Franklin **Ground Truth:** Lightning rod | Glass harmonica | Bifocals | Franklin stove
Prediction: Franklin stove, Glass harmonica, Bifocals, Lightning rod **F1:** 1.00

Benjamin Franklin $\xrightarrow{\text{image.appears\_in\_topic}}$ Benjamin Franklin $\xrightarrow{\text{image.appears\_in\_topic}}$ Benjamin Franklin $\xrightarrow{\text{law.inventor.inventions}}$ [All 4 inventions retrieved]

Analysis: Despite the seemingly redundant intermediate hops through image relations, RSF correctly identifies all four inventions. This demonstrates robustness to noisy paths in the KG—the model learns that certain relation types (e.g., `image.appears_in_topic`) act as identity connectors and successfully propagates flow through them to reach the target `inventions` relation.

Example 4: Political Succession (WebQSP, 3-hop)

Question: "Who was vice president after Kennedy died?"
Topic Entity: John F. Kennedy **Ground Truth:** Lyndon B. Johnson
Prediction: Lyndon B. Johnson **F1:** 1.00

John F. Kennedy $\xrightarrow{\text{common.topic.webpage}}$ [m.0b48xzc] $\xrightarrow{\text{common.webpage.topic}}$ John F. Kennedy $\xrightarrow{\text{government.us\_president.vice\_president}}$ Lyndon B. Johnson

Analysis: The question contains implicit temporal reasoning ("after Kennedy died"), which the model handles by recognizing that JFK's vice president at the time of death would succeed him. The intermediate webpage node serves as a structural bridge, and the final hop directly extracts the vice-presidential relationship.

Example 5: Implicit Religious Context (CWQ, 4-hop)

Question: "What is the predominant religion where the leader is Ovadia Yosef?"
Topic Entity: Ovadia Yosef **Ground Truth:** Judaism
Prediction: Judaism **F1:** 1.00

Ovadia Yosef $\xrightarrow{\text{STAY}}$ Ovadia Yosef $\xrightarrow{\text{people.person.religion}}$ Judaism $\xrightarrow{\text{religion.founding\_figures}}$ Abraham $\xrightarrow{\text{religion.founding\_figure.religion\_founded}}$ Judaism

Analysis: This example demonstrates handling of implicit constraints. The phrase "where the leader is" does not explicitly mention religious leadership, yet the model correctly infers that Ovadia Yosef's domain of leadership is religious. The reasoning path exhibits a cyclic structure (Judaism $\rightarrow$ Abraham $\rightarrow$ Judaism), showing that RSF can handle re-entrant paths as discussed in Section 2.2.

E.2. Failure Mode Analysis

We identify three systematic failure modes through analysis of incorrect predictions. Understanding these limitations is crucial for guiding future improvements.

Failure Mode 1: Answer Entity Not in Subgraph (WebQSP)

Question: "Who plays Ken Barlow in Coronation Street?"
Topic Entity: Coronation Street **Ground Truth:** William Roache
Prediction: *"The provided reasoning paths do not contain information about the actor..."* **F1:** 0.00
Subgraph Reachability: No — Answer entity not present in extracted subgraph.

Path 1: Coronation Street $\xrightarrow{\text{program\_creator}}$ Tony Warren $\xrightarrow{\text{film.actor.film}}$ [m.0h0_mvx] $\xrightarrow{\text{performance.actor}}$ Tony Warren

Path 8: Coronation Street $\xrightarrow{\text{program\_creator}}$ Tony Warren $\xrightarrow{\text{fictional\_characters\_created}}$ Ken Barlow $\xrightarrow{\text{STAY}}$ Ken Barlow

Analysis: The answer entity **William Roache** is not present in the extracted subgraph. While the character "Ken Barlow" appears in Path 8, the actor-character relationship lies beyond the K-hop extraction radius.

Failure Mode 2: LLM Selects Wrong Candidate from Multiple Options (CWQ)

Question: "Which man is the leader of the country that uses 'Libya, Libya, Libya' as its national anthem?"
Topic Entity: Prime Minister of Libya **Ground Truth:** Abdullah al-Thani
Prediction: Muammar Gaddafi **F1:** 0.00
Subgraph Reachability: Yes — Correct answer is present in reasoning paths.

Path 1: Prime Minister of Libya $\xrightarrow{\text{office\_holders}}$ [m.011n4mp5] $\xrightarrow{\text{office\_holder}}$ Abdullah al-Thani $\xrightarrow{\text{positions\_held}}$ ...
Path 3: Prime Minister of Libya $\xrightarrow{\text{office\_holders}}$ [m.07lt4tb] $\xrightarrow{\text{office\_holder}}$ Muammar Gaddafi $\xrightarrow{\text{positions\_held}}$ ...
Path 4: Prime Minister of Libya $\xrightarrow{\text{office\_holders}}$ [m.0h89fd8] $\xrightarrow{\text{office\_holder}}$ Mahmoud Jibril $\xrightarrow{\text{positions\_held}}$ ...

Analysis: The reasoning paths correctly contain the ground truth answer (Abdullah al-Thani in Path 1), but also contain multiple other Libyan leaders. The LLM incorrectly selected Gaddafi due to popularity bias in its parametric knowledge. The KG lacks temporal annotations to disambiguate which leader is "current."

Failure Mode 3: Entity Alias Mismatch (CWQ)

Question: "What was the name of the team that won the 2008 FIFA Club World Cup Final championship?"
Topic Entity: 2008 FIFA Club World Cup Final **Ground Truth:** Newton Heath L&YR F.C.
Prediction: Manchester United F.C. **F1:** 0.00

Path 1: 2008 FIFA Club World Cup Final $\xrightarrow{\text{sports\_championship\_event.champion}}$ Manchester United F.C. $\xrightarrow{\text{STAY}}$ Manchester United F.C. ...
Path 2: 2008 FIFA Club World Cup Final $\xrightarrow{\text{champion}}$ Manchester United F.C. $\xrightarrow{\text{sports\_team.venue}}$ [m.0n4ykf9] $\xrightarrow{\text{team}}$ Manchester United F.C.

Analysis: All reasoning paths correctly lead to "Manchester United F.C.," and the LLM correctly outputs this entity. However, the ground truth annotation uses "Newton Heath L&YR F.C."—the historical name of Manchester United from 1878–1902. This is a dataset annotation artifact rather than a model failure; the prediction is factually correct but receives F1 of 0.00 due to string mismatch with an outdated alias. To mitigate such issues in production environments, we propose integrating an alias aware entity linker that normalizes both KG output and ground truth annotations to a unique uuid, ensuring the model is not penalized for nomenclature differences.

E.3. Summary of Failure Modes

Table 7 summarizes the three identified failure modes with their characteristics.

Table 7. Summary of systematic failure modes identified in qualitative analysis.

Failure Mode	Root Cause	Answer in Paths?
Answer Not in Subgraph	Answer entity lies outside K-hop subgraph	No
Wrong Candidate Selection	LLM popularity bias; no temporal info in KG	Yes
Entity Alias Mismatch	Dataset annotation uses outdated entity name	N/A (correct)

These qualitative examples validate that RSF-GLLM successfully performs complex multi-hop reasoning across diverse question types, while also revealing that remaining errors often stem from retrieval coverage limitations or LLM-side biases rather than fundamental reasoning failures in the RSF module.

F. Implementation Details and Reproducibility

We provide comprehensive implementation details to facilitate reproducibility of our results.

F.1. Compute Resources

All experiments were conducted on a single NVIDIA A100 GPU with 80GB HBM2e memory. No distributed training or multi-GPU setups were required, demonstrating the efficiency of our decoupled architecture.

F.2. Stage 1: RSF Module Training

Architecture. The RSF module consists of: (i) a projection layer mapping Qwen3 embeddings to lower hidden states, (ii) a GRU for query updating, (iii) attention layers for relation scoring and content bias, and (iv) a gating MLP with a single linear layer followed by sigmoid activation.

Embeddings. Node and relation embeddings are obtained using Qwen3-Embedding. Embeddings are pre-computed and cached for all entities and relations in the Freebase subset used by WebQSP and CWQ. Entity names and relation texts are used as input text.

Optimization. We use AdamW optimizer with learning rate 5×10^{-5}, weight decay 0.01, and linear warmup over the first 10% of training steps. Training runs for 10 epochs with early stopping based on validation Hit@1 (patience = 3 epochs). Batch size is 16 for WebQSP and 8 for CWQ (due to larger subgraphs). Gradient clipping is applied with max norm 1.0.

Hyperparameters. The flow sparsity coefficient $\lambda_1 = 0.1$ was selected via grid search over $\{0.01, 0.05, 0.1, 0.2, 0.5\}$. The number of reasoning hops T is set to 2 for WebQSP and 4 for CWQ, matching the dataset characteristics. The smoothing constant $\epsilon = 10^{-8}$ is used for numerical stability.

Subgraph Extraction. We perform K-hop BFS from the topic entity ($K = 2$ for WebQSP, $K = 4$ for CWQ).

F.3. Stage 2: LLM Fine-Tuning

Base Model. We report results with two answer-generation backbones: **LLaMA-2-7B** and **Qwen3-8B**. LLaMA-2-7B is included to enable an apples-to-apples comparison with prior LLM+KG baselines (e.g., RoG, GNN-RAG, GNN-RAG+RA), which use the same backbone. Qwen3-8B is reported as a stronger contemporary alternative of comparable parameter scale. Both backbones are trained and evaluated under identical hyperparameters and the same RSF-extracted reasoning paths; no other component of the pipeline changes between the two runs.

Fine-Tuning. We apply full parameter fine-tuning for both backbones, using a single combined checkpoint evaluated on both WebQSP and CWQ to match the protocol used by the LLaMA-2-7B baselines.

Optimization. We fine-tune using AdamW with learning rate 2×10^{-4}, cosine learning rate schedule, and batch size 1. Training runs for 3 epochs. Maximum sequence length is 2048 tokens.

F.4. Evaluation Metrics

We follow standard KGQA evaluation protocols. **Hit@1** measures exact match accuracy—whether the top-1 predicted answer matches any gold answer. **F1** computes token-level overlap between the predicted answer string and gold answers, averaged across all test samples.