Enhancing Knowledge Base Question Answering with AMR-Driven Subgraph Retrieval

Anonymous ACL submission

Abstract

Knowledge Base Question Answering (KBQA) seeks to provide answers to natural language questions by utilizing pertinent triples from knowledge graphs (KGs). The mainstream methods of KBQA involve the use of graph neural networks for the reasoning and rely on subgraph retrieval to reduce the complexity. However, current retrieval methods predominantly align question text with graph relations, leading to inconsistent subgraph quality and limited interpretability, thereby impeding QA performance. Here, we proposed a subgraph retrieval method based on Abstract Meaning Representation (AMR) to captures deep semantic structures, enhance retrieval precision and optimize the reasoning by leveraging the structural similarity of AMR to KGs. Additionally, we construct reasoning chains in AMR form to enhance interpretability. Experiments on the WebQSP and CWQ datasets demonstrated that the integrating of AMR enhances retrieval performance, improves the subgraph quality, and achieves competitive KBQA performance and interpretable reasoning.

1 Introduction

Knowledge graphs (KGs) store factual knowledge in a structured format of triples (*head*, *relation*, *tail*), to represent entities and their relations (Paulheim, 2016). Knowledge Base Question Answering (KBQA) seeks to identify answer entities in a KG based on a given natural language question. This task can be considered as a node classification problem, where KG entities are classified as answers vs. non-answers for a given question (Mavromatis and Karypis, 2024). Although large language models (LLMs) perform well in many natural language processing (NLP) tasks, they are face with challenges in handling complex problems and specialized domains, often generating hallucinated or inaccurate results (Zhang et al., 2023). In contrast,

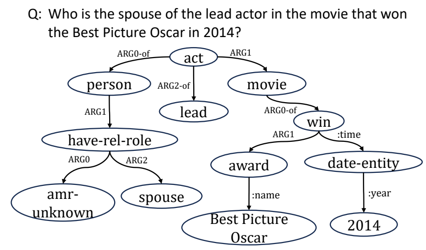

Figure 1: An example of utilizing AMR to extract semantic information. AMR captures the core semantics of a sentence and represents the relations and entity types in a concise graph structure

graph neural networks (GNNs), owing to their ability to process intricate graph structures, have been widely adopted in KBQA systems (Schlichtkrull et al., 2018). However, reasoning over an entire large-scale KG is computationally impractical (Ji et al., 2021), necessitating subgraph retrieval as a preprocessing step.

Existing subgraph retrieval methods primarily rely on text embedding to measure the similarity between a question and KG relations, to select entities and paths accordingly. For instance, PullNet (Sun et al., 2019) uses an LSTM -based retriever to iteratively select relations based on the question, followed by a GNN-based reasoner to identify the tail entities. Similarly, SR (Zhang et al., 2022) decouples the retriever and reasoner, and constructs subgraphs from the top-k retrieved paths. However, these embedding-based methods often fail to capture the semantic structure of the question, resulting in subgraphs with noisy nodes and irrelevant relations (Jain et al., 2021).

A natural language question often consists of multiple relations that can serve as explicit guidance for subgraph retrieval. Abstract Meaning Representation (AMR) provides a structured representation of a question's semantic information, stripping away syntactic variations while preserving the core relations (Banarescu et al., 2013). As illustrated in Figure 1, to solve this problem, the first step is to identify the film that won the Best Picture Oscar in 2014. The next step is to determine the lead actors of the film, followed by identifying the name of the spouse as the final answer. This process corresponds to the predicates in the AMR graph, such as *win*, *act*, and *spouse*. This clear graph structure enhances the quality of subgraph retrieval and provides interpretability for the reasoning process.

Motivated by these advantages of AMR, we propose an AMR-based subgraph retrieval method for KBQA with the following objectives: (a) Transform the question into an AMR graph to capture richer semantic information. (b) Extract fine-grained relations based on nodes and edges in the AMR graph. (c) Leverage the extracted relations to guide subgraph retrieval, to ensure compact and precise subgraphs. (d) Integrate relations extracted from AMR into the adjacency matrix for improving GNN-based reasoning. Subsequently, a subgraph-oriented reasoning model is employed to identify answer entities while ensuring an interpretable reasoning process.

Our key contributions can be summarized as:

- We introduce AMR as a semantic representation for KBQA, transforming textual questions into structured semantic graphs to capture richer fine-grained relations.

- We propose three relation construction patterns based on AMR nodes and edges, and the constructed relations can effectively align with KG structures to improve subgraph retrieval.

- We employ a semantic matching mechanism to retrieve relevant subgraphs and utilize the AMR graph structure to refine the subgraph's adjacency matrix, thereby enhancing both its quality and reasoning accuracy.

- We leverage AMR-based semantic information to construct reasoning paths from the topic entity to the answer entity, enhancing the interpretability of the reasoning process.

2 Related Work

Subgraph Retrieval Direct reasoning on the entire KG is often inefficient. In mainstream information retrieval-based KBQA, only relevant nodes and relations are retained to form a subgraph, on which reasoning is performed to obtain the answer. However, determining the specific nodes and relations that constitute the subgraph remains a significant challenge. GraftNet (Sun et al., 2018) employs a heuristic approach that retrieves entities within two hops of the topic entity and ranks them based on their personalized PageRank scores to control the size of the final subgraph. However, this method overlooks the semantics of the question, which limits the accuracy of subsequent reasoning. Recent studies have introduced neural models to solve this issue by retrieving relevant subgraphs specific to the questions. PullNet (Sun et al., 2019) proposes a framework that iteratively expands the subgraph, utilizing an LSTM-based retriever to select relations at each hop through semantic matching and a GCN-based reasoner (Kipf and Welling, 2016) to identify the tail entities of these relations. Similarly, SR (Zhang et al., 2022) employs a bidirectional encoder to develop a trainable retriever, decoupling the retriever from the reasoner and enabling integration with any subgraph-oriented reasoning model. Despite these advancements, subgraph retrieval methods heavily rely on black-box neural models to interpret question semantics. Consequently, they often fail to preserve the detailed semantic structure of the question, resulting in subgraphs containing unnecessary noisy nodes, which adversely affect the reasoning process and limit the overall effectiveness of the approach.

Abstract Meaning Representation Abstract Meaning Representation (AMR) encodes the entity types and relations in a question as nodes and edges, stripping away the syntactic variations while preserving the semantic structure. Recent studies have leveraged the explicit graph structure of AMR for semantic parsing and reasoning tasks. AMR-SG (Xu et al., 2021) constructs AMR-based semantic graphs from relevant evidence and performs reasoning over them to explain the answers. Similarly, NSQA (Kapanipathi et al., 2020) transforms questions into query graphs resembling KGs by utilizing AMR, generating logical forms and employing a neuro-symbolic reasoner to predict the final answers. In contrast, QDAMR (Deng et al., 2022) utilizes AMR to convert multi-hop questions into symbolic forms, facilitating the decomposition of complex queries and identification of intermediate unknowns. Inspired by these studies, we aim to leverage the semantic structure provided by AMR to enhance subgraph retrieval.

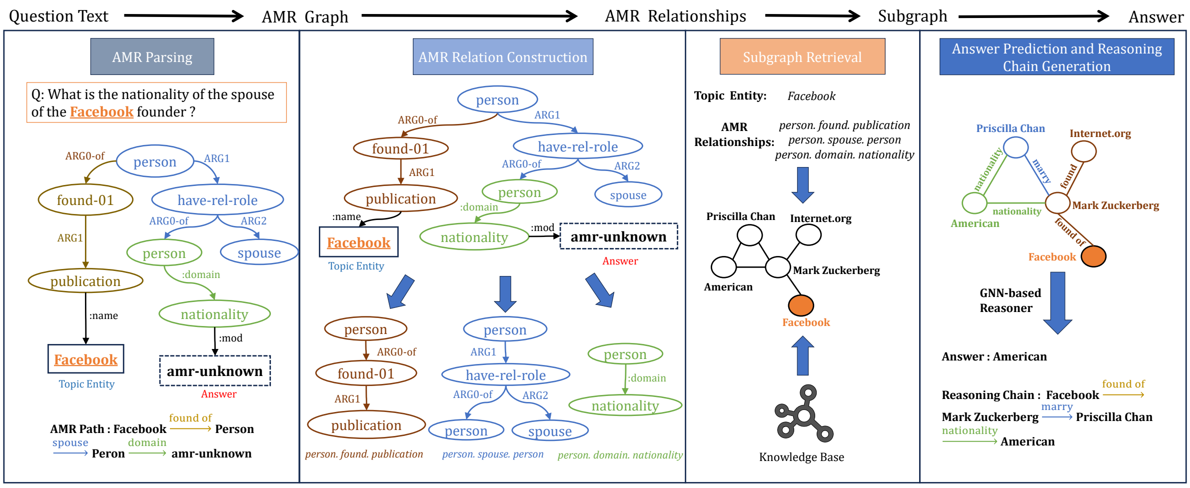

Figure 2: An overview of our KBQA framework. Firstly, the question Q: "What is the nationality of the spouse of the Facebook founder?" is parsed using an AMR parser, converting it into an AMR graph. The path from the topic entity to the *amr-unknown* node is then explored, and the AMR relations are constructed based on the predicate nodes and semantic edges encountered along the path. Subsequently, semantic matching is performed with the relations in the knowledge graph to select the corresponding subgraph. Finally, the adjacency matrix of the subgraph is optimized using the AMR graph structure, and an AMR-based GNN reasoner is employed to predict the answer, replacing the corresponding nodes along the AMR path to obtain the reasoning chain.

3 Method

Figure 2 illustrates the pipeline of our AMR-based KBQA system, which consists of four modules: *i*) AMR Parsing (§3.1), which transforms question text into AMR graphs to obtain structured semantic information. *ii*) AMR Relation Construction (§3.2), which leverages the semantic structure of the AMR graph to extract the relations needed to answer the question. *iii*) Subgraph Retrieval (§3.3), which performs semantic matching between extracted relations and relations in the KG to retrieve the relevant subgraph. *iv*) Answer Prediction and Reasoning Chain Generation, whichpredicts the answer and generates reasoning chains by reasoning over the subgraph. Additionally, specific entities along the path to the answer are used to replace the corresponding nodes in the AMR graph, thereby constructing the reasoning chain.

3.1 AMR Parsing

An AMR graph is a rooted, directed, and acyclical structure. The nodes in an AMR graph represent different concepts, i.e., named entities, quantities, dates, and other phenomena (Banarescu et al., 2013); Edges represent relations between these concepts, such as *:domain* and *:name* (Kapanipathi et al., 2020). An AMR path can be obtained from the topic entity to the answer entity, where the nodes and edges along this path capture the semantic information necessary to answer the question. This semantic information is then used for the construction of AMR relations. The following is the path we extract based on AMR parsing:

$$\text{Facebook} \xrightarrow{\text{found\_of}} \text{Person} \xrightarrow{\text{spouse}} \text{Person} \xrightarrow{\text{domain}} \text{unknown}$$

We leverage AMR parsing to capture the semantic structure of questions. Figure 2 illustrates the AMR graph for the question: "*What is the nationality of the spouse of the Facebook founder?*". *amr-unknown* and *Facebook* are two key entities involved in answering the question, where *amr-unknown* represents the unknown answer entity and *Facebook* serves as the topic entity extracted from the question. These two entities correspond to the starting point (topic entity) and final target (answer entity) in the KBQA reasoning process.

For AMR parsing, we adopt SPRING (Bevilacqua et al., 2021), a widely used method for AMR parsing. A key advantage of SPRING is its seamless integration with BLINK (Wu et al., 2019), a highly effective entity linking tool. BLINK directly annotates topic entities in the AMR graph, ensuring a one-to-one correspondence with their KG counterparts, thereby eliminating the need for separate entity linking.

3.2 AMR Relation Construction

A high-quality subgraph must contain a path from the topic entity to the correct answer entity, while minimizing extraneous noisy paths that may mislead the reasoner into incorrect answers. Since qestions often consist of multiple relations, an intuitive approach is to extract the relations involved in the question and use them as a basis to filter the relations that constitute the subgraph. The core of constructing relations is the identification of predicates and the nodes involved in these predicates. We developed an algorithm to construct relevant relations from AMR graphs while annotating the types of entities involved in these relations. Specifically, the algorithm first extract the path from the topic entity, obtained through entity linking, to the *amr-unknown* node (lines 1–7 in the 1), and then identify PropBank predicates (e.g., *have-rel-role-91* and *found-01*) along the path and their surrounding nodes and edges to construct the relations. Figure 2 illustrates the three main patterns of relations construction. The colored sections in the figure represent the core nodes or edges involved in construction of the relations, including predicates, role predicates, and semantic edges. The remaining parts serve as participants in the relations or determinants of the direction of the relations. The following sections will provide a detailed explanation of these three construction patterns.

Algorithm 1 Relation Construction based on AMR

1: **Input:** Question text Q
2: **Output:** AMR relation Set R
3: AMR Graph G:=**AMR Parsing**(Q)
4: Topic Entitis E:=**Entity linking**(G)
5: **if** E not null and *amr-unknown* in G **then**
6:     **for** e in E **do**
7:         P_e=**getShortestPath**(G,e,*amr-unknown*)
8:     **end for**
9:     **for** v in V_p **do**
10:         **if** isVerb(v) **then**
11:             e_1, e_2=**getNeighborNodes(v)**
12:             **buildRelation**(e_1,e_2,v)
13:         **else**
14:             **if** isRoleVerb(v) **then**
15:                 e_1, e_2=**getNeighborNodes(v)**
16:                 e_3=**getRole(v)**
17:                 **buildRelation**(e_1,e_2,e_3)
18:             **end if**
19:         **end if**
20:     **end for**
21:     **for** r in R_p **do**
22:         **if isRelation**(r) **then**
23:             e_1, e_2=**getEntity**(r)
24:             **buildRelation**(e_1,e_2,r)
25:         **end if**
26:     **end for**
27: **end if**
28: **return** R

A Predicate With Two Entities. The presence of predicate nodes typically indicates the occurrence of an action, while their adjacent attribute edges, such as *ARGx*, represent the primary or secondary participants in the action. According to the explanation provided by PropBank framesets (Kingsbury and Palmer, 2002), the entity connected by the edge with a smaller parameter label is considered as the agent of the action, whereas the other entity serves as the recipient of the action. For example, the *found-01* predicate in Figure2 represents the relation *person.found.publication*.

Predicates With Auxiliary Attribute. Some predicate nodes may be simultaneously connect to three parameterized edge labels. The additional entity connected via the *ARG2* edge represents an auxiliary attribute of the relation. For instance, the predicate *org* typically links to auxiliary attributes that indicate positional roles, such as director or president, while the predicate *rel* often connects to auxiliary attributes representing interpersonal relations, such as parent or spouse. As shown in Figure 2, by replacing the intermediate predicate entity with the auxiliary attribute entity, this relation construction pattern can be transformed into the first construction pattern.

Two Directly Connected Entities. In addition to predicate nodes, edges in an AMR graph can also represent relations, such as the *:domain* edge shown in Figure 2. These edges connect two entities, with one entity acting as an attribute of the other to exemplify relations such as time, location, or quantity. We use edge labels and neighboring entities to construct relations. When the guiding predicate parameter *':arg'* is absent, we utilize the the direction of the path from the topic entity to the answer entity to intuitively determine the relation's direction, as the answer always starts from the topic entity. Specifically, the nodes closer to the topic entity and *amr-unknown* are selected as the head and tail entity of the relation, respectively.

The relations construction process may also involve the *amr-unknown* nodes, which can be replaced with the answer type corresponding to the question. We focus on the entity type represented by the node. Specifically, the entity type *time*, *place*, and *person* are used when the question begins with 'when', 'where', and 'who', respectively.

3.3 Subgraph Retrieval

In this section, we use AMR relations obtained at the previous step to guide subgraph retrieval. Since AMR graphs are fine-grained representations of question text transformations, we can perform subgraph retrieval on any subgraph to select the relations and entities relevant to the question. For any subgraph containing several triples $\mathcal{G} = (h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the entity set and the relation set, respectively. Let $\mathcal{E}_r = \{e_1, e_2, \ldots, e_n\}$, which denotes the n entity types available in AMR, and let $\mathcal{R}_i = \{r_i^1, r_i^2, \ldots, r_i^m\}$, which denotes the set of relations where e_i serves as the head entity. For each entity $e \in \mathcal{E}$ in the graph $\mathcal{G}$, we first determine whether its entity type exists in $\mathcal{E}_r$. If the corresponding entity type e_i is present, we retain the entity and then determine whether the entity's relations should be preserved; otherwise, the entity is excluded.

For each relation $r_i \in \mathcal{R}_i$, we compute its correlation with the relation of the retained entity e (denoted as r'), obtain its embedding using the pretrained model BERT (Devlin, 2018), and measure the correlation using cosine similarity:

$$\text{Similarity}(r, r') = \frac{h(r) \cdot h(r')}{|h(r)||h(r')|} \tag{1}$$

where $h(\cdot)$ denotes the embedding function instantiated by BERT (Devlin, 2018), and the parameter δ is the threshold for subgraph retrieval. The above operations for each entity will result in a subgraph that includes topic entities, type-specific entities, and question-related relations.

3.4 Reasoning Over Subgraphs

After retrieval of the subgraph using our AMR-based method, we introduce an AMR-informed structural enhancement to incorporate the explicit semantic structure of AMR graphs, thereby improving reasoning efficiency.

GNN models are well-suited for handling multi-hop reasoning in KGs, where entities must be contextually linked. However, standard KGs may lack explicit links between semantically related entities, which may cause longer reasoning chains and less efficient reasoning. Since AMR provides a structured semantic representation, it can reveal implicit relations that are not directly encoded in the KG. To integrate this additional information, we modify the adjacency matrix of the retrieved subgraph $\mathcal{G}_Q = (\mathcal{E}_Q, \mathcal{R}_Q)$ as follows:

1. **Identifying AMR-Linked Nodes:** Given an AMR graph $\mathcal{A}_Q$ derived from the question, we identify pairs of nodes (e_i, e_j) that are connected in $\mathcal{A}_Q$ but not in $\mathcal{G}_Q$.

2. **Graph Structure Modification:** If e_i and e_j are not adjacent in $\mathcal{G}_Q$, we add an edge (e_i, e_j) to the adjacency matrix.

3. **Semantic Edge Labeling:** The newly added edge is assigned with a special relation label `[amr]`, ensuring its differentiation from standard KG relations and enabling the model to recognize AMR-derived semantic dependencies.

Formally, we update the adjacency matrix A of the subgraph as:

$$A'_{i,j} = \begin{cases} 1, & \text{if } (e_i, e_j) \in \mathcal{G}_Q \text{ or } \in \mathcal{A}_Q, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Similarly, we modify the relation matrix R by setting:

$$R'_{i,j} = \begin{cases} \texttt{[amr]}, & \text{if} (e_i, e_j) \in \mathcal{A}_Q \text{ and } \notin \mathcal{G}_Q, \\ R_{i,j}, & \text{otherwise.} \end{cases} \tag{3}$$

During message passing, the feature update equation follows the standard GNN formulation but now includes the newly introduced `[amr]` edges:

$$\mathbf{h}_e^{(l)} = \sigma\Bigg(\sum_{r \in \mathcal{R}_Q \cup \{\texttt{[amr]}\}} \sum_{e' \in \mathcal{N}_r(e)} W_r^{(l)} \mathbf{h}_{e'}^{(l-1)} + W_0^{(l)} \mathbf{h}_e^{(l-1)} \Bigg). \tag{4}$$

where $\mathcal{N}_r(e)$ now includes both standard KG relations and AMR-derived relations, and $W_r^{(l)}$ represents the transformation matrix for the type of relation r at layer l.

This modification enhances the capability of the GNN model to propagate information efficiently by explicitly integrating AMR-informed semantic structures. The updated adjacency matrix allows the model to directly attend to semantically related entities that would otherwise require multi-hop reasoning.

3.5 Answer Prediction and Reasoning Chain Generation

Unlike traditional embedding-based approaches that only produce answer entities, our method utilizes AMR to provide a structured representation of the question. We replace the entity types in the AMR graph with specific entities from the retrieved subgraph, generating an interpretable reasoning path from the topic entity to the answer entity.

A straightforward approach to constructing a reasoning chain is to extract the shortest path from the topic entity to the answer entity in the retrieved subgraph. However, the shortest path does not always align with the correct reasoning process. For the question: *"Where does the spouse of the founder of Facebook live?"*, an ideal reasoning path should be:

$$\text{Facebook} \rightarrow \text{founder} \rightarrow \text{spouse} \rightarrow \text{residence}$$

However, due to structural constraints in the KG, the shortest path might be:

$$\text{Facebook} \rightarrow \text{founder} \rightarrow \text{residence}$$

which incorrectly skips the spouse relation, leading to an invalid reasoning chain. To address this issue, we propose a heuristic search constrained by entity types to construct the reasoning paths.

Rather than enforcing strict relation consistency between AMR and the KG, we prioritize entity type consistency when selecting the reasoning path. Given an AMR-derived entity sequence $T_A = \{e_1, e_2, \ldots, e_m\}$, and a candidate reasoning path in the retrieved subgraph $P = \{p_1, p_2, \ldots, p_n\}$, we define a path score $S(P)$ based on the number of matched entity types:

$$S(P) = \frac{|T_A \cap P|}{|T_A|} \quad (5)$$

where $|T_A \cap P|$ represents the number of matched entity types. A higher $S(P)$ indicates greater alignment between the path and the entity types in the ideal reasoning path.

We employ a heuristic search strategy for efficient determination of the optimal reasoning path. First, we extract multiple candidate paths from the topic entity to the answer entity in the retrieved subgraph. Then, we compute the entity-type-based score $S(P)$ by evaluating each path's alignment with the AMR-derived entity sequence. Finally, we select the path with the highest score. If there are multiple paths sharing the highest score, we choose the shortest one to ensure minimal reasoning steps.

After obtaining the optimal path, similar to entity linking, the abstract entity types in the AMR graph are sequentially linked to specific entities on the path, resulting in an interpretable reasoning chain.

Finally, we generate the reasoning path by replacing nodes and edges we obtain in the AMR parsing step, as shown in Figure 2:

$$\text{Facebook} \xrightarrow{\text{found of}} \text{Mark Zuckerberg} \xrightarrow{\text{marry}} \text{Priscilla Chan} \xrightarrow{nationality} American$$

4 Experiments

In this section, we designed experiments to answer the following key questions: (1) Can AMR-extracted relations contribute to higher-quality subgraph retrieval? (2) To what extent does AMR-based semantic extraction improve QA performance? (3) How effective is graph structure optimization in enhancing the reasoning accuracy?

4.1 Dataset.

We evaluated our method on two widely used benchmarks: WebQuestionsSP (WebQSP) (Yih et al., 2015) and Complex WebQuestions 1.1 (CWQ) (Talmor and Berant, 2018). Both datasets are built on the Freebase knowledge base (Bollacker et al., 2008). The former mostly contains simple questions, while the latter includes more complex questions. Table 1 presents the statistics.

Dataset	#Train	#Val.	#Test
CWQ	27,639	3,519	3,531
WebQSP	2,848	250	1,639

Table 1: Statistics of the CWQ and WebQSPDataset.

4.2 Subgraph Retrieval

For the KBQA task, an ideal subgraph should include paths from the topic entities to the target answers, with minimal noisy paths that may lead to incorrect entities. Since the ground truth of the path is often difficult to obtain, we evaluated the subgraph quality using two metrics: the size of the retrieved subgraph and the answer coverage rate, which represents the proportion of questions for which the subgraph contains at least one answer entity.

Baselines We compare with two models that enhance subgraph retrieval, in which SR (Zhang et al., 2022) trains a subgraph retriever to expand relation paths via a sequential decision process, and GraftNet (Sun et al., 2018) extracts subgraphs by calculating PPR scores for topic entity neighborhoods.

Subgraph Quality Evaluation We compared three types of subgraphs: (a) Subgraphs retrieved by our method, (b) Subgraphs obtained by baseline methods, and (c) Subgraphs obtained by baseline methods and refined using AMR constraints.

Table 2 presents the answer coverage rate and the number of relation nodes in subgraphs retrieved from WebQSP. PPR represents the retriever used by Graft(Sun et al., 2018), while SR refers to the retriever employed by SR(Zhang et al., 2022). The experimental settings follow those in the original paper. We use AMR to refer to the retriever proposed in this paper, with a similarity threshold $\delta = 0.67$ determined through the development set.

Compared with other subgraph retrieval methods, our method significantly reduces the subgraph size while maintaining a high answer coverage rate. Furthermore, application of AMR constraints to the baseline methods further improves subgraph quality, demonstrating the effectiveness of AMR in enhancing subgraph retrieval.

Method	Answer Coverage	Nodes	Relations
AMR	92.37%	183423	7983
PPR	94.87%	1441420	6102
PPR+AMR	92.13%	581840	3222
SR	91.22%	183433	26282
SR+AMR	90.30%	108362	5630

Table 2: Answer Rate and Subgraph Size Comparison.

4.3 KBQA

For the reasoning performance, we follow the practice of recent studies (Sun et al., 2018, 2019), which consider reasoning as a ranking task. For each test question, we ranked all candidate entities based on the prediction probability of the evaluated model, and then determined whether the top-1 answer is correct using Hits@1. Considering that a question may have multiple answers, we also adopted the commonly used F1 metric.

Baselines We compare our framework with the following baselines, in which KV-Mem (Miller et al., 2016) stores triplets in a key-value memory network for reasoning. EmbedKGQA (Saxena et al., 2020) formulates answer reasoning as a link prediction task. GraftNet (Sun et al., 2018), PullNet (Sun et al., 2019) and NSM (He et al., 2021) are subgraph-oriented embedding models. SR (Zhang et al., 2022) decoupled from the subsequent reasoning process, which enables a plug-and-play framework to enhance any subgraph-oriented KBQA model. EPR (Ding et al., 2024) constructs evidence patterns from nodes in the KG and extracts subgraphs by combining these evidence patterns. UniKGQA (Jiang et al., 2022) proposes a unified framework that designs a shared pre-training task based on problem-relation matching, enabling parameter sharing between the subgraph retrieval module and the reasoning module.

QA Performance Evaluation We used existing subgraph-oriented reasoners for KBQA tasks and denoted models with * to indicate the application of our adjacency matrix modification strategy. For example, NSM* represents a modified version of NSM, where the adjacency matrix is adjusted to incorporate AMR-derived semantic information into the subgraphs. Additionally, we tune the similarity threshold δ on the development sets of WebQSP and CWQ to ensure sufficient answer coverage in the retrieved subgraphs. The final thresholds used for WebQSP and CWQ were set to 0.67 and 0.64, respectively.

Table 3 presents the main comparison results of different methods. Our method achieves competitive performance across both datasets. Compared with other NSM-based methods, our method achieves state-of-the-art results on WebQSP, with a +3.1 improvement in Hit@1 and a +6.8 increase in F1 relative to SR+NSM. In terms of CWQ, which consists of more complex multi-hop questions, our method also outperforms most of the baseline methods, demonstrating high effectiveness in handling complex reasoning tasks.

When the same reasoning model is used, our method significantly outperforms the state-of-the-art PLM-based retriever SR in QA performance, confirming that AMR-based subgraph retrieval can effectively capture richer semantic structures, enhances subgraph quality, and therefore improve the QA performance. Additionally, our method, along with EPR, exhibits notably better performance in answering complex questions compared with other methods. These results suggest that construction of subgraphs in a structured manner is more effective in eliminating noisy nodes, thereby enhancing the model robustness.

Method	CWQ		WebQSP	
	H@1	F1	H@1	F1
KV-Mem	18.4	15.7	46.6	34.5
EmbedKGQA	32.0	-	66.6	-
PullNet	45.9	-	68.1	-
GraftNet	36.8	32.7	66.4	60.4
NSM	47.6	42.4	68.5	62.8
SR + GCN	49.1	42.7	66.7	63.1
SR + NSM	49.3	46.3	69.5	64.1
UniKGQA+NSM	49.2	-	69.1	-
UniKGQA+UniKGQA	50.7	48.0	**75.1**	70.2
EPR+NSM	**60.6**	**61.2**	71.2	70.2
Our Method				
AMR + GCN*	52.6	50.5	69.8	65.6
AMR + NSM*	53.8	52.1	72.6	**71.3**

Table 3: Performance Comparison of Different Methods in Question Answering on CWQ and WebQSP.

4.4 Effect of Adjacency Matrix Modification

Table 4 demonstrates the impact of adjacency matrix modification on the on QA performance. Across all subgraph retrieval methods, this structural enhancement consistently improvement of reasoning accuracy. For SR-based subgraph retrieval, the modification increases H@1 and F1 by +1.1 and +0.8 on CWQ, and +0.7 and +0.7 on WebQSP, respectively.

The improvement is even more significant with AMR-based subgraph retrieval. Specifically, AMR+NSM* achieves a +2.1 gain in H@1 and a +2.3 gain in F1 on CWQ, and a +1.4 increase in H@1 and a +4.1 increase in F1 on WebQSP. These results suggest that AMR-informed structural enhancement significantly improves answer accuracy.

Similarly, EPR+NSM* outperforms EPR+NSM, despite with smaller gains, indicating that refining subgraph structure enhances multi-hop reasoning. Overall, these results confirm that incorporating AMR-derived connections into the graph structure can effectively capture implicit semantic relations, thereby improving both the precision and recall in KBQA.

Method	CWQ		WebQSP	
	H@1	F1	H@1	F1
SR+NSM	49.3	46.3	69.5	64.1
SR+NSM*	50.4	47.1	70.2	64.8
AMR+NSM	51.7	49.8	71.2	67.2
AMR+NSM*	53.8	52.1	72.6	71.3
EPR+NSM	60.6	61.2	71.2	70.2
EPR+NSM*	61.2	61.8	72.3	71.8

Table 4: Impact of Adjacency Matrix Modification on QA Performance across Different Retrieval Methods.

4.5 Case Study

This section illustrates the effectiveness of our approach through a case example using AMR. The upper part of Figure 3 shows subgraphs retrieved by SR's retriever for the question Where was Avril Lavigne born? The green node represents the correct answer, connected by the relation *people.person.place_of_birth*. Yellow nodes are relevant but non-optimal paths, gray nodes are irrelevant, and red nodes may lead to incorrect answers.

The lower part of the figure shows the AMR graph for the question. Our method extracts the relation person.bear.place, excluding irrelevant or misleading paths, resulting in a more focused subgraph. The final reasoning chain is: Avril Lavigne $\xrightarrow{\text{bear}}$ Canada.

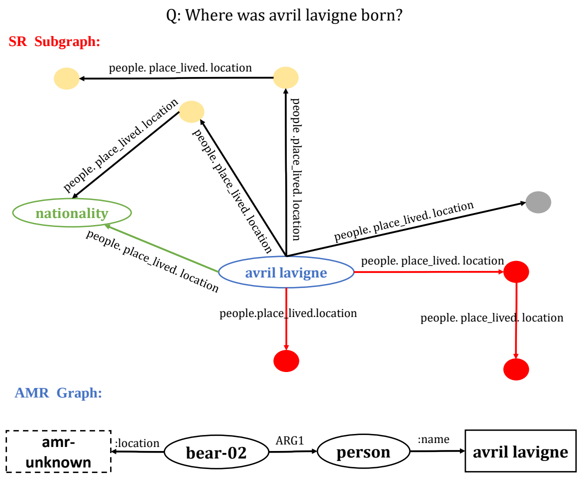

Figure 3: Subgraph of the question "Where was Avril Lavigne born?" obtained based on SR and its corresponding AMR graphs.

5 Conclusion

In this paper, we propose an AMR-driven subgraph retrieval method to enhance KBQA. Unlike traditional methods based on text similarity, our approach uses AMR to extract structured semantic information, improving subgraph retrieval precision and reasoning interpretability. By aligning AMR-derived relations with the knowledge graph, we filter noisy entities and ensure more accurate semantic alignment. Experimental results on WebQSP and CWQ show that our method outperforms existing approaches, especially for complex multi-hop questions. Additionally, the incorporation of AMR-derived semantic connections into the adjacency matrix boosts reasoning performance, while providing explicit and interpretable reasoning chains.

6 Limitations

Dependence on AMR Parsing Quality. Our approach heavily relies on AMR parsing quality to construct meaningful relations. Errors in predicate or entity identification can propagate through subsequent stages, affecting subgraph retrieval and reasoning. This limitation underscores the need for improved AMR parsing tools or robust strategies to mitigate parsing errors.

Handling Ambiguous or Incomplete Questions. Our method assumes that input questions are well-formed and contain sufficient semantic information for AMR parsing. However, in real-world scenarios, questions may be ambiguous, incomplete, or grammatically incorrect, which can degrade parsing quality and hinder subgraph retrieval. Addressing this limitation may require preprocessing techniques to refine input questions or error-tolerant AMR parsing models.

Interpretability of Reasoning Paths. Although our method improves interpretability by constructing reasoning chains from AMR graphs, the generated paths rely on predefined rules and heuristics. This limits their flexibility and may fail to capture nuanced semantic relations. A promising direction for improvement involves integrating learning-based approaches to dynamically generate more adaptive reasoning paths.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12564–12573.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250.

Zhenyun Deng, Yonghua Zhu, Yang Chen, Michael Witbrock, and Patricia Riddle. 2022. Interpretable amr-based question decomposition for multi-hop question answering. *arXiv preprint arXiv:2206.08486*.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Wentao Ding, Jinmao Li, Liangchuan Luo, and Yuzhong Qu. 2024. Enhancing complex question answering over knowledge graphs through evidence pattern retrieval. In *Proceedings of the ACM Web Conference 2024*, pages 2106–2115.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. 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*, pages 553–561.

Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. 2021. Do embeddings actually capture knowledge graph semantics? In *The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings 18*, pages 143–159. Springer.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graphs. *arXiv preprint arXiv:2212.00959*.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramon Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, et al. 2020. Leveraging abstract meaning representation for knowledge base question answering. *arXiv preprint arXiv:2012.01707*.

Paul R Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*, pages 1989–1993.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

Heiko Paulheim. 2016. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings*, pages 593–607. Springer.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2019. Scalable zero-shot entity linking with dense entity retrieval. *arXiv preprint arXiv:1911.03814*.

Weiwen Xu, Huihui Zhang, Deng Cai, and Wai Lam. 2021. Dynamic semantic graph construction and reasoning for explainable multi-hop science question answering. *arXiv preprint arXiv:2105.11776*.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. *arXiv preprint arXiv:2202.13296*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.