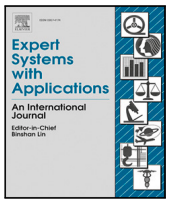
Expert Systems With Applications 282 (2025) 127612

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Fact retrieval from knowledge graphs through semantic and contextual attention

Akhil Chaudhary [a,*], Enayat Rajabi [a], Somayeh Kafaie [b], Evangelos Milios [c]

[a] Shannon School of Business, Cape Breton University, Grand Lake Dr., Sydney, B1M 1A2, NS, Canada
[b] Department of Mathematics and Computing Science, Saint Mary's University, 923 Robie St, Sydney, B3H 3C3, NS, Canada
[c] Faculty of Computer Science, Dalhousie University, 6299 South St, Halifax, B3H 4R2, Nova Scotia, Canada

ARTICLE INFO

MSC:
68T30
68T35
68T50

Keywords:
Querying
Knowledge graphs
Fact retrieval
Information extraction
BERT
Zero shot

ABSTRACT

Knowledge Graphs (KGs), such as DBpedia and ConceptNet, enhance Natural Language Processing (NLP) applications by providing structured information. However, extracting accurate data from KGs is challenging due to issues in entity detection, disambiguation, and relation classification, which often lead to errors and inefficiencies. We introduce **Attention2Query (A2Q)**, an attention-driven approach that directly ranks and selects the most relevant facts, thus minimizing error propagation. A2Q centres on three key contributions: (1) *Focused Node Selection*, which streamlines graph traversal; (2) *Global Attention Alignment*, improving retrieval by comparing facts against the query text; and (3) *Contextual Re-ranking*, enabling on-the-fly adjustments of fact importance based on evolving query context. Experimental results across multiple tasks and datasets show that A2Q substantially outperforms baseline methods, including those in zero-shot settings, achieving higher retrieval accuracy with reduced computational overhead.

1. Introduction

Knowledge graphs (KGs) (Bollacker et al., 2008; Lehmann et al., 2015; Vrandečić & Krötzsch, 2014) offer a structured approach to amalgamate data from varied sources through a semantic framework conducive to inference and computational processing. These graphs encapsulate facts as triplets (entity, relation, entity) and can represent extensive global knowledge. Their design allows for the incremental addition of entities and relationships in various ways – manually, semi-automatically, or automatically – without disrupting existing functionalities. This adaptability aligns with the semi-structured data model, supporting regular updates to KGs. However, this schema design flexibility brings challenges in KG management and querying. In the realm of natural language processing, language models (LMs) play a pivotal role (Brown et al., 2020; Devlin et al., 2019). However, these models may contain knowledge that is incomplete, outdated, or inaccurate. Recent research, Galetzka et al. (2021), Kang et al. (2022), Ma et al. (2022) and Oguz et al. (2022), highlights the potential of enhancing LMs with KG-sourced facts to improve outcomes in tasks such as question answering and dialogue generation.

Despite the wide-ranging applications of KGs, the current methods for fact retrieval from them tend to be overly complex. Current strategies, Baek et al. (2023), Lan et al. (2021) and Wang et al. (2021) and traditional methods (Fu et al., 2020; Lan et al., 2021), involve three main steps: identifying the relevant span within the text, disambiguating this span to link it to a specific entity in the KG, and then classifying the relationship that pertains to the query. For instance, when querying "Who is Jason Bourne?", the process begins with identifying "Jason Bourne" as the entity, linking this to its corresponding entity ID in the KG, and then determining the applicable relationship, such as "Who is", among potentially dozens associated with that entity. This series of steps is generally referred to as entity linking. Moreover, algorithms to initiate searches from specific entities within a subgraph based on a target condition to extract relevant subgraphs have been developed (Tu et al., 2023). In contrast, LM models synthesize information post-extraction from KGs or differentiate definitions in a more structured manner (Chen et al., 2021).

The current methods for retrieving information from KGs face several significant challenges. Firstly, they depend on complex processes for exploring graphs and resolving paths, which is problematic due to the scarcity of high-quality graph data. This scarcity, compounded by the necessity for data cleaning and path resolution, results in minimal usable data. Secondly, the step-by-step nature of these methods introduces the risk of error propagation (Singh et al., 2020). Errors

* Correspondence to: 1250 Grand Lake Rd, Sydney, B1M 1A2, Nova Scotia, Canada.
E-mail addresses: akhil_chaudhary@cbu.ca (A. Chaudhary), enayat_rajabi@cbu.ca (E. Rajabi), somayeh.kafaie@smu.ca (S. Kafaie), emilios@dal.ca (E. Milios).

https://doi.org/10.1016/j.eswa.2025.127612
Received 28 October 2024; Received in revised form 16 March 2025; Accepted 4 April 2025
Available online 18 April 2025

in early steps, like span detection, can reduce the accuracy of subsequent steps, such as relation classification or path mapping, which may result in incorrect outcomes. Thirdly, the algorithms used to link query entities to KG entities or to predict relationships are often not adaptable to new entities and relationships, limiting their applicability across different KGs. A more flexible approach that avoids KG-specific extractions would be beneficial. Lastly, these methods are inherently time-consuming, requiring the initial extraction of all potential neighbouring paths into a subgraph, followed by extensive pruning and cleaning to isolate the relevant subgraph or path. This process not only adds to the computational burden but also the overall complexity of the task.

To address these challenges, our approach introduces on-the-go retrieval of relevant triplets for natural language queries by calculating attention scores within a shared representation space of contextual and graph embeddings, facilitating focused exploration of each entity in the neighbourhood. This method draws inspiration from Transformers (Vaswani et al., 2017), which introduced the attention mechanism as a means to selectively concentrate on the most pertinent segments of the query for optimizing relation estimation between tokens. Unlike the original application of attention in transformer models, which primarily enhances the interaction between encoder and decoder modules, our implementation leverages attention to establish a direct contextual link between the input query and the graph's nodes, including their relationships. This strategy not only streamlines the retrieval process but also dynamically adapts to the specific requirements of the query, thereby mitigating the limitations associated with traditional KG exploration methods.

To develop our fact retrieval system, we begin by tokenizing the input query and initially mapping all tokens against the KG and Taxonomies to capture preliminary information. Following the embedding of this information using graph embeddings and transformer embeddings, we employ a triple-pass attention mechanism. The first pass involves identifying a focused path through the graph and pinpointing nodes relevant to the current token. We pinpoint the most appropriate node at this juncture in the second pass, progressively deepening our search. Subsequently, we evaluate these paths in the third pass against any available supplementary information, such as multiple-choice options in question-answering systems. To further refine the search for relevant triplets, we approximate similarity assessments using a relevance classifier at the end of each pass that leverages a specially fine-tuned model. This model evaluates the relevance between the input token from the query and the extracted triplet, ranking them according to their pertinence to the query. Importantly, because our system embeds triplets using a language model, it can adapt to various KGs without retraining to accommodate new schemas or types of entities and relations. This capability distinguishes our method from traditional retrieval systems, which often require extensive retraining. Our framework is named Attention2Query (A2Q), reflecting its core operational mechanism.

We assessed our Attention2Query (A2Q) framework on various fact retrieval tasks within the question-answering domain, aiming to efficiently retrieve relevant triplets for given queries. Our experimental findings show that A2Q outperforms existing baseline methods that rely on traditional fact retrieval pipelines from KGs. Furthermore, our reranking approach notably enhances retrieval performance, as supported by comprehensive analyses demonstrating the effectiveness and simplicity of the A2Q framework.

This work contributes the following advancements:

- We introduce a novel, on-the-go fact retrieval framework for KGs, employing an attention mechanism to refine neighbourhood and path selection. This approach uses representational similarities between the query and triplets to streamline the processes of entity detection, disambiguation, and relation classification into a single step.
- We develop a reranking strategy to address the challenge of minimal context in on-the-go fact retrieval. This strategy involves successive attention calculations to improve the accuracy of selection.
- We propose a versatile strategy to mitigate the issue of retrieving non-relevant data from KGs. This includes integrating additional knowledge sources, such as taxonomies, and leveraging optional information, like Multiple-Choice Question (MCQ) data.
- We demonstrate the superiority of our A2Q framework through validation on fact retrieval tasks, highlighting its advantages over conventional baselines in both the quality of data extracted and the efficiency of retrieval across unsupervised and supervised settings.

2. Related work

The pursuit of information retrieval focuses on sourcing documents or KG entries that align with user queries. Information retrieval has evolved significantly, starting with lexical matching methods like TF-IDF, which often encounter vocabulary mismatches (Jeong et al., 2021; Nogueira et al., 2019). Recent advancements have embraced the use of language models for query-document processing, showcasing a shift toward understanding queries and documents through latent semantic representations (Devlin et al., 2019; Karpukhin et al., 2020; Xiong et al., 2020). This evolution demonstrates the growing ability of models to capture document meaning, though challenges persist in extracting succinct facts from KGs.

In the realm of knowledge representation, KGs stand out as critical infrastructures for storing factual data in structured formats, specifically as triplets (entity-relation-entity). These triplets serve various domains, including KG question answering (KGQA) and improving the factual accuracy of language model responses (Bollacker et al., 2008; Chakraborty et al., 2019; Kang et al., 2022; Lukovnikov et al., 2017; Lv et al., 2019a; Sha et al., 2023; Vrandečić & Krötzsch, 2014; Zhang et al., 2019). The integration of structured and unstructured data, as seen in methods that combine knowledge bases with external databases or inferential graphs for reasoning, underscores the dynamic potentials of KGs in enhancing information retrieval and reasoning processes (Lin et al., 2019; Lv et al., 2019a; Min et al., 2019; Sap et al., 2019). Additionally, approaches like schema graphs utilize a knowledge-based symbolic domain and path-based LSTM for scoring (Lin et al., 2019). In contrast, others blend structured knowledge bases with unstructured sources for commonsense reasoning (Lv et al., 2019a).

Extracting relevant facts efficiently from KGs continues to be challenging. Traditional methods like neural semantic parsing, which translates queries into executable languages on KGs, require extensive annotations and deep logical structure understanding (Bakhshi et al., 2020; Bakken & Soylu, 2023; Dong & Lapata, 2016; Liang, 2013; Luo et al., 2018). Sequential methods of entity identification, resolution, and relationship categorization often accumulate errors, leading to inefficient outcomes (Bordes et al., 2014; Chen et al., 2019; Han et al., 2020; Hao et al., 2017; Singh et al., 2020; Wang et al., 2021). Recent approaches attempt to align textual triplets with input text for retrieval, yet they still rely on preliminary entity linking (Oguz et al., 2022). Addressing these limitations, our approach simplifies the process by directly harnessing representational similarities for fact retrieval, thus streamlining the retrieval into a single, efficient step (Ma et al., 2022). Our approach bypasses traditional complexities by using representational similarities for direct fact retrieval, thereby reducing the process to a single, streamlined step.

In this study, we introduce a novel methodology, the Attention2Query (A2Q) method, designed to optimize fact retrieval from KGs by employing a machine learning approach that uses attention mechanisms to identify and evaluate relevant nodes directly related to each token in the input text. This innovative approach not only simplifies the retrieval process but also significantly enhances the accuracy and efficiency of extracting information from KGs, making it highly effective for various NLP tasks.

3. Background

In this section, we briefly introduce the foundational concepts that underpin our proposed methodology 4. Specifically, we highlight (1) transformer-based embeddings for text representation, (2) attention mechanisms, and (3) KG structures. By covering these foundational concepts here, we establish the necessary background for understanding our approach. This allows the Methodology section to focus exclusively on our novel contributions, such as [mention key novel aspects], without reiterating fundamental details.

3.1. Transformer-based embeddings

Transformer-based language models, such as BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019), have become standard for encoding textual data into dense vector representations. They leverage self-attention layers to capture contextual information from both left and right contexts. This bidirectional encoding renders them particularly effective for downstream tasks like fact retrieval, where nuanced semantic similarities between queries and KG entries are crucial. In our work, these embeddings form the backbone of the *Relevance Classifier*, enabling contextual alignment between query text and KG triplets.

3.2. Attention mechanisms

Attention mechanisms (Vaswani et al., 2017) allow models to focus on the most relevant parts of a sequence by calculating compatibility scores between elements (e.g., query-key–value). Originally introduced for machine translation, attention has since been applied to numerous NLP tasks, from summarization to reasoning. In the *Attention2Query (A2Q)* framework, we extend this concept to identify and prioritize key nodes in the KG that align semantically with specific tokens in the user query. By repeatedly recalculating attention scores, we minimize the need for exhaustive graph traversal and help mitigate error propagation through selective focus. As the attention is not directly designed to work on KG, we have created our own variant based on a self-attention mechanism and designed additional modules to perform similar tasks.

3.3. Knowledge graphs and graph embeddings

A KG stores facts in the form of *triplets* (subject, relation, object). Popular KGs, such as ConceptNet (Speer et al., 2018), DBpedia (Lehmann et al., 2015), and Wikidata (Vrandečić & Krötzsch, 2014), offer large-scale structured information. Many approaches represent KG entities and relations through specialized graph embeddings (e.g., Numberbatch Speer et al., 2018 for ConceptNet), which place related nodes closer in a continuous vector space. We leverage these embeddings alongside transformer-based text embeddings to measure semantic relevance between query terms and candidate triplets. Unlike traditional KG pipelines (which typically apply distinct modules for entity linking, relation classification, and disambiguation for triplet extraction), our proposed KG pipeline uses representational similarities and attention queries to unify fact retrieval in a single pass.

3.4. Taxonomies, MCQ context, and optional information

In some tasks, additional structured data (e.g., taxonomies or MCQ answer choices) can guide fact retrieval, especially if the query is under-specified. We incorporate these additional structured datasets to the input to improve the performance on domain-specific evaluations. This modular design allows new or domain-specific taxonomies to be plugged in without altering the underlying retrieval architecture.

Algorithm 1: Overall A2Q Pipeline. A high-level procedure orchestrating subgraph extraction, optional path generation, and triplet selection.

Input: **Query** Q, **Knowledge Graph** KG, **Max Triplets** K, **(Optional) Extra Info** e.g., MCQs
Output: **Top-**K **Relevant Triplets** T^*
▷ 1) Subgraph Extraction (Algorithm 3)
$G \leftarrow \text{ExtractSubgraph}(Q, KG)$
▷ 2) Relevance classification (model-based, not shown as a separate algorithm)
$G^* \leftarrow \text{RelevanceClassifier}(G^*, Q)$; ▷ Scores or filters triplets
▷ 3) Final selection
$T^* \leftarrow \text{SelectTopK}(G^*, K)$
▷ 4) (Optional) Path Generation to merge disjoint regions (Algorithm 4)
$G^* \leftarrow \text{PathGeneration}(G, KG)$
return T^*

4. Methodology

In this section, we define the problem and outline the proposed framework to address it. Our main goal is to extract relevant information from KGs (Lehmann et al., 2015; Speer et al., 2018) in the most efficient way possible in terms of speed and computation without degrading quality. An overview of the proposed framework is shown in Fig. 1 and Algorithm 1.

The Attention2Query (A2Q) framework consists of four parts, as shown in Fig. 1: the neighbourhood extractor, sub-graph extractor, relevance classifier, and path generator.

4.1. Task formulation

Given a question q and a set of candidate answers, such as in MCQs $\{a_1, a_2, \ldots, a_q\}$, our goal is to identify a subset $S \subseteq T$ of triplets from a KG such that $|S| = n$, where n is a predefined number. Each triplet in S should maximize relevance and plausibility concerning the question. This can be mathematically represented as:

$$S = \arg \max_{\substack{S' \subseteq T \\ |S'|=n}} \sum_{t \in S'} \text{score}(t, q)$$

where $\text{score}(t, q)$ measures the relevance and plausibility of triplet t in answering the question q.

To illustrate, consider the question "Canada is a member of ?" and the set of candidate answers:

{NATO, USA, Europe, EU}

Here, q represents the question and $\{a_1, a_2, a_3, a_4\}$ are the candidate answers. The KG contains triplets such as:

{(Canada, isA, Country),
(Canada, PartOf, NATO),
(NATO, isA, Group)}

The goal is to select the triplets that best support the question, such as (Canada, PartOf, NATO).

Our proposed method takes the question q and an external KG as input. The KG is represented as a collection of triplets:

$$T = \{(e_i, r_{ij}, e_j) \mid e_i, e_j \in E, r_{ij} \in R\}$$

where E is the set of entities and R is the set of possible relations. Each triplet (e_i, r_{ij}, e_j) consists of a pair of entities e_i, e_j connected by a relation r_{ij}.

In the example above, selecting the relevant triplet (Canada, PartOf, NATO) helps correctly identify the answer "NATO" as the most relevant option.

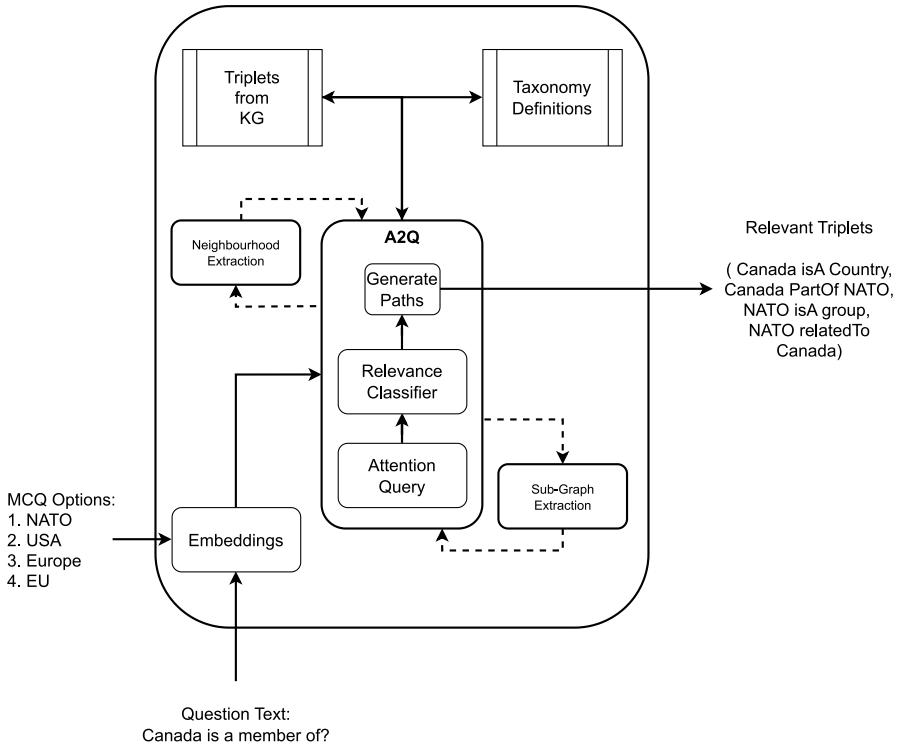


Fig. 1. Overview of the Proposed Information Extraction Approach: **Input 1:** The Question includes a query text to identify relevant information. **Input 2:** MCQ Options provide additional context for multiple-choice questions. **Triplets from KG:** Search within the KG to locate and extract relevant entity neighbourhoods and triplets. **Taxonomy Definitions:** Retrieve definitions for unfamiliar and medical terminology. **A2Q Process:** This multi-step procedure begins by identifying relevant neighbourhoods for each entity mentioned in the question. Next, the most relevant nodes are selected from all neighbourhoods during the sub-graph extraction phase. Triplets are then ranked according to their relevance to the question using a fine-trained Relevance Classifier. Finally, the Generate Paths utility reconstructs broken pathways during selection.

4.2. Relevance classifier

We have chosen BERT (Devlin et al., 2019) to perform triplet classification on the extracted triplets from KG, allowing for a simplified triplet selection process.

Our fine-tuned BERT language model (LM) is specifically trained on a dataset of $1K$ samples to rank triplets from a KG on a scale between 1 and 10.

1. **Input Formation:** For an input question q, and a triplet t_n, the input L for the LM is formed by:

   $$L = [\text{CLS}]; q; [\text{SEP}]; t_n$$

   where [CLS] and [SEP] are special tokens used by BERT and similar pre-trained models.
2. **Model Processing:** The concatenated input L is processed by the fine-tuned BERT LM to produce a rank:

   $$\text{Rank} = \text{BERT\_LM}(L)$$

   The output, Rank, assigns a score between 1 and 10, where each score r indicates the level of relevance, with $r = 1$ being the least relevant and $r = 10$ the most relevant.
3. **Attention Mechanism:** The ranking scores are further refined by an attention module:

   $$\text{Final Score} = \text{Attention\_Module}(\text{Rank})$$

   This mechanism computes a weighted average where weights are adjusted according to the relevance scores, prioritizing triplets with higher relevance.

This method allows for a dynamic and nuanced assessment of triplet relevance, leveraging semantic analysis and contextual alignment enabled by the BERT LM.

Algorithm 2: AttentionNeighborhood. Discovers relevant neighbours for a single query token using attention-based scoring.

Input: **Token** q_i, **Knowledge Graph** KG
Output: **Neighbourhood** N_i
▷ 1) Retrieve immediate neighbours from KG
neighbours ← KGLookup(q_i, KG)
▷ 2) Apply attention-based scoring to each neighbour
foreach *node* $n \in$ neighbours **do**
  score(n) ← ComputeAttention(q_i, n)
end
▷ 3) Select top relevant neighbours based on attention scores
$N_i \leftarrow \text{TopNodes}(\text{neighbours}, \text{score})$
return N_i

4.3. Neighbourhood extractor

In this section, we extract the neighbourhood of each token. We take a tokenized input query and process each token in KG to extract all the relevant information related to the query in the KG for further information refinement. An overview can be seen in Algorithm 2. As per the structure of the KGs, the idea here is that the relevant information lies within the same vicinity in the KG for similar information (Speer et al., 2018). For the KG, we utilize ConceptNet (Speer et al., 2018) and DBPedia (Lehmann et al., 2015) KGs.

To extract the neighbourhood for each sample, we use the following approach. We start by querying each token in the question into the KG and then generating an embedding of the token. We expand the neighbourhood by calculating attention between each neighbourhood

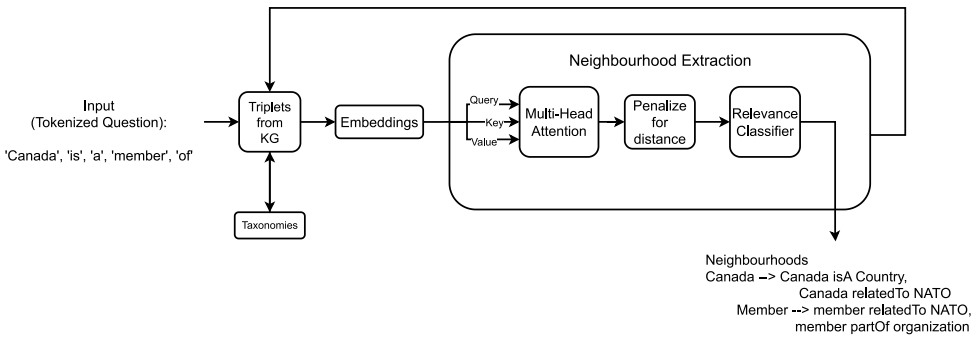


Fig. 2. This figure illustrates the comprehensive workflow for neighbourhood extraction associated with each entity in the question. Initially, we identify and map the neighbours of each entity within the graph. These neighbours are then embedded, and a novel attention query is employed to assess and score their relevance based on their proximity to the entity and their relation to the input question. We apply a penalty to any triplet farther from the entity under the hypothesis that the most relevant nodes are typically closer together. Finally, a novel relevance classifier evaluates and ranks all considered triplets, selecting the top-ranked triplets based on a predefined threshold.

token and the current token. Then, we penalize the scored triplets for distance from the original token, select the node with the highest score as a current token, and continue expanding the neighbourhood until we reach the specified threshold or set hyperparameter. Then, we pass all these tokens through the relevance classifier to perform triplet ranking based on semantic similarity compared to the input query and select the triplets that scored highest compared to the input question in context and relevance with the triplets. Finally, we selected triplets based on ranking. The process can be seen in Fig. 2.

For the edge cases where the input token is not part of the KG, we use semantic similarity to select relevant words to get the neighbourhood for the missing token.

To extract the neighbourhood for each token within the question from the KG, the process is outlined as follows:

1. **Query and Embedding:** For a given token t_i from the input question q, search it in the knowledge graph G and generate an embedding $\mathbf{e}(t_i)$ for token and all its neighbours $Nb\_i = \{n_{i,1}, n_{i,2}, \ldots, n_{i,N}\}$.

   $$\mathbf{e}(t_i) = \text{Embed}(t_i)$$

   $$\mathbf{e}(n_{i,j}) = \text{Embed}(n_{i,j}) \; \forall n_{i,j} \in Nb_i$$

2. **Neighbourhood Expansion:** Calculate the attention scores α_{ij} between each token t_i and its neighbours in N of Nb_i where i represents the token and j represents neighbour of token.

   $$\alpha_{ij} = \text{Attention}(\mathbf{e}_{t_i}, \mathbf{e}_{n_{i,j}}) \; \forall n_{i,j} \in Nb_i$$

3. **Scoring and Penalization:** Apply a distance penalty to the attention scores based on the distance from t_i to each neighbour node Nb_i where Nb is a subset of selected neighbours based on semantic similarity, adjusting the score s_{Nb_i}.

   $$s_{Nb_i} = \alpha_{ij} - \lambda \cdot \text{Dist}(t_i, Nb_i)$$

   where λ is a scaling factor that modulates the extent of the distance penalty applied to the attention score. A higher value of λ increases the penalty for greater distances, thus diminishing the score more substantially for nodes further away from t_i. Conversely, a smaller λ value reduces the impact of distance on penalizing the score, enabling distant nodes to maintain higher scores. The choice of λ is a hyperparameter based on the desired balance between proximity influence and other factors influencing node interaction.

4. **Selection and Expansion:** Select the token t^* with the highest adjusted score and expand the neighbourhood. Repeat until a threshold or a set hyperparameter Θ is reached.

   $$t^* = \arg\max_{t_i \in N}(s_{t_i}), \quad \text{until } \Theta \text{ is met}$$

Algorithm 3: ExtractSubgraph. Aggregates token-level neighbourhoods into a single subgraph.

Input: Query Q, Knowledge Graph KG
Output: Combined Subgraph G
▷ `Initialize an empty collection of nodes/edges`
$G \leftarrow \emptyset$
foreach *token* $q_i \in Q$ **do**
  ▷ `Extract local neighbourhood for each token (Algorithm 2)`
  $N_i \leftarrow \text{AttentionNeighborhood}(q_i, KG)$
  ▷ `Merge into combined subgraph`
  $G \leftarrow G \cup N_i$
end
return G

5. **Relevance Classification and Ranking:** Pass all tokens through a relevance classifier to rank the triplets based on their contextual relevance and similarity to q, selecting the top n triplets.

   $$\{T_1, T_2, \ldots, T_n\} = \text{Top-Rank-Relevance}(q, \{s_{t_i}\})$$

6. **Missing Nodes:** There exists an edge case where the current approach will return an empty response. This occurs when the searched token is missing from the KG itself. To handle this case, we employ semantic similarity using cosine similarity in the embedding space of KG using its graph embeddings (e.g., Numberbatch Speer et al., 2018 in the case of ConceptNet) to extract all the nodes that have the highest relevance to the currently searched node and extract those neighbourhoods. This process will provide us with enough data to work with to generate a relevant response.

4.4. Graph extractor

In this section, we extract the combined neighbourhood sub-graph from KGs based on all the neighbourhoods we have extracted for each token. Here, we combine all the extracted neighbourhoods for all tokens to extract a collective fully connected sub-graph from KG, which contains all the most relevant information possible about the input query from the KGs. An overview can be seen in Algorithm 3.

To extract the sub-graph for the input query, we follow the following approach. We start by embedding the input query question. Then, we feed the question into multi-head attention to generate the corresponding value matrix. We take this value matrix along with embedded neighbourhoods generated by the neighbourhood Extractor, explained in Section 4.3, feed them to another multi-head attention block, and calculate the attention scores. Based on calculated attention scores,

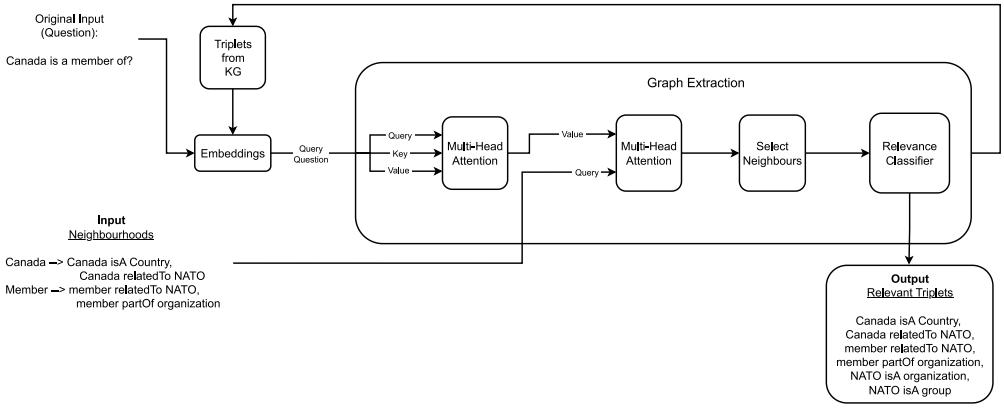


Fig. 3. This figure depicts the comprehensive workflow for graph extraction. The process begins by refining the neighbourhoods to select the most relevant triplets. It extracts a combined sub-graph by combining all the neighbourhoods of all the tokens in the question. First, it combines all the neighbourhoods and then performs pruning to select the most optimal sub-graph based on the attention query.

we select a symmetrically similar neighbourhood. Then, we process this interim neighbourhood and pass it to the Relevance Classifier, described in Section 4.2. Finally, we select the highest-rated triples outputted from the Relevance Classifier as our final sub-graph. The process is shown in Fig. 3.

To extract the sub-graph for input questions from the KG, the process is outlined as follows:

1. **Query and Embedding:** For a given original question q from the input, and for all extracted neighbourhood triplets $T_1, T_2, \ldots, T_n$ for each token from question from the neighbourhood Extractor 4.3, compute their embeddings:

$$\mathbf{e}(q) = \text{Embed}(q)$$

$$\mathbf{e}(T_i) = \text{Embed}(T_i), \quad i = 1, 2, \ldots, n$$

2. **Sub-Graph Provisioning:** Aggregate the neighbourhoods of each token into a single graph, merging all connected components:

$$G = \bigcup_{i=1}^{n} \text{Neighbourhood}(T_i)$$

3. **Attention Scoring:** Calculate the attention scores for each node in the sub-graph G against the query question q, based on relevance and semantic similarity:

$$\alpha_v = \text{Attention}(\mathbf{e}(q), \mathbf{e}(v)), \quad \text{for each } v \in G$$

4. **Node Selection:** Select the top n nodes with the highest attention scores from the collective sub-graph, where n is a hyperparameter:

$$V^* = \{\, v \mid v \text{ is among the top } n \text{ scoring nodes in } G \,\}$$

5. **Relevance Classifier:** Rank the triplets based on their contextual relevance and similarity to the input question q, using a relevance classifier, and select the top n triplets:

$$\{T_1^*, T_2^*, \ldots, T_n^*\} = \text{Top-Rank-Relevance}(q, \{T_i\})$$

4.5. MCQ graph extractor

We use the following approach to extract the MCQ sub-graph for the MCQ options. The process begins by embedding the input MCQ options. These are then processed through the neighbourhood Extractor to retrieve neighbourhood triplets for each option. Following this, all neighbourhoods are passed through a multi-head attention mechanism to produce a value matrix. This matrix is further processed with another multi-head attention, incorporating relevant triplets from the Graph Extractor as indicated in Graph Extractor 4.4, to compute the final attention scores. The attention is calculated on a sub-graph consisting of triplets from Graph Extractor 4.4 and the new triplets we received from the neighbourhood Extractor 4.3 for each MCQ option. Then, we select top-scoring triplets as our current sub-graph (neighbourhood) and pass them to the Relevance Classifier 4.2. Finally, we select the highest-rated triplets outputted from the Relevance Classifier as our final sub-graph in the form of a set of triplets. The process can be seen in Fig. 4.

To extract the MCQ sub-graph for MCQ options for the KG, the process is outlined as follows:

1. **Embedding MCQ Options:** Embed each MCQ option to obtain initial semantic representations:

$$\mathbf{e}(\text{MCQ}_i) = \text{Embed}(\text{MCQ}_i), \quad i = 1, 2, \ldots, n$$

2. **neighbourhood Extraction:** For each embedded MCQ option, retrieve the associated neighbourhood triplets using the neighbourhood Extractor, see Section 4.3:

$$N_{\text{MCQ}_i} = \text{neighbourhood Extractor}(\mathbf{e}(\text{MCQ}_i))$$

3. **Combining neighbourhoods:** Aggregate the neighbourhoods of all MCQ options into a single set to form a composite neighbourhood:

$$G_{\text{MCQ}} = \bigcup_{i=1}^{n} N_{\text{MCQ}_i}$$

4. **Calculating Attention Scores:** Calculating attention scores for the combined sub-graph:

$$\alpha_v = \text{Attention}(\textit{Attention}(G_{\text{MCQ}}), \text{Graph Extractor})$$

5. **Selecting Top-Scoring Triplets:** Select the top n nodes with the highest attention scores from the collective sub-graph, where n is a hyperparameter:

$$V^* = \{\, v \mid v \text{ is among the top } n \text{ scoring nodes in } G \,\}$$

6. **Relevance Classification:** Rank the triplets based on their contextual relevance and similarity to the input question q, using a relevance classifier, and select the top n triplets:

$$\{T_1^*, T_2^*, \ldots, T_n^*\} = \text{Top-Rank-Relevance}(q, V^*)$$

$$\{R^*\} = \{T_1^*, T_2^*, \ldots, T_n^*\}$$

The final set R^* represents the selected sub-graph of triplets tailored for the MCQ context, which is visualized in Fig. 4.

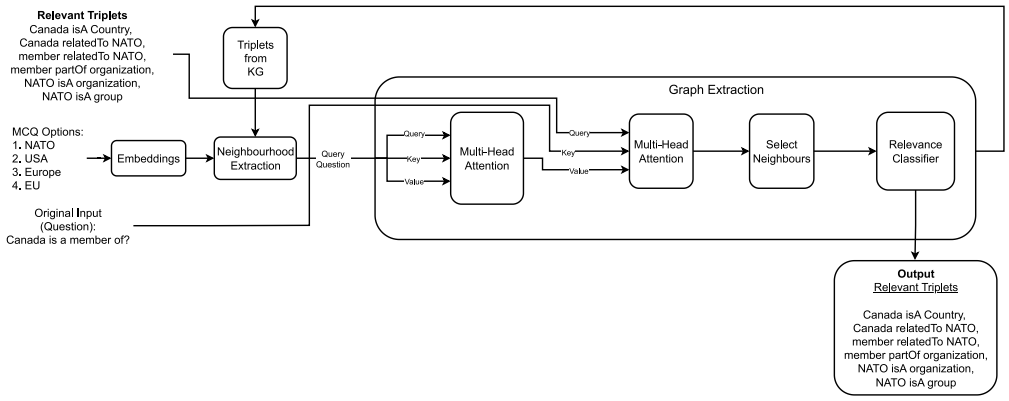


Fig. 4. This figure depicts the additional workflow performed for MCQ options if the question is of MCQ type. The process begins with the refined sub-graph extracted in Fig. 2. It first generates neighbourhoods for all MCQ options and then combines all those new neighbourhoods into subgraphs from Fig. 2. Then, it performs pruning to select the most optimal sub-graph based on the attention query.

Algorithm 4: PathGeneration. Merges disjoint components within the subgraph by bridging partial paths.

Input: **Subgraph** G, **Knowledge Graph** KG
Output: **Updated Subgraph** G^*
▷ 1) Identify disconnected regions or partial paths
$C \leftarrow \text{FindDisconnected}(G)$
▷ 2) Attempt to bridge each disconnected component
foreach *component* $C_j \in C$ **do**
  $C'_j \leftarrow \text{BridgePaths}(C_j, KG)$ $G \leftarrow G \cup C'_j$
end
▷ 3) optionally prune low-scoring or extraneous links
$G^* \leftarrow \text{PruneSubgraph}(G)$
return G^*

4.6. Path generation

In this section, we try to merge the extracted subgraphs by creating paths between the subgraphs during the attention query process. This additional step is crucial because the attention query chooses only one neighbour at each node to determine the path, which is further refined to optimize the route selection. Due to node selection, some paths get misaligned in the query refinement process inside Graph Extractor Section 4.4 and MCQ Graph Extractor Section 4.5. This misalignment causes the breaking of a subgraph into multiple subgraphs due to the deletion of a junction node. In the path generation process, we try to reintroduce that junction node to merge those broken subgraphs during the selection process.

To repair these interrupted pathways, the attention query is executed one last time using the neighbourhood Extractor Section 4.3 module. It processes the final set of triplets extracted either from the Graph Extractor Section 4.4 or the MCQ Graph Extractor Section 4.5, depending on whether the input question includes MCQs. The procedure involves expanding each node within the current subgraph. This expansion continues until it either connects with another node in the subgraph, establishing a clear path or reaches a set limit if a path cannot be formed in that direction. This expansion is repeated for every node in the subgraph. Subsequently, all potential paths are examined to identify the shortest possible path among the newly discovered paths. This entire process is illustrated in Fig. 5 and Algorithm 4.

To complete the disjoint paths in the sub-graph, the process is outlined as follows:

1. **Re-Executing Attention Query:** Run the neighbourhood Extractor to expand the nodes in the vicinity of the current node on the extracted triplets to amend paths that were previously incomplete or misaligned:

   $$P = \text{neighbourhoodExtractor}(E)$$

   where E represents the set of triplets from the respective extractor.
2. **Path Expansion:** For each node v in the current sub-graph G, expand the search to connect disjoint paths or extend existing ones until another node within G is encountered or a predefined threshold is reached:

   $$\text{Expand}(v, G) = \begin{cases} v \to w & \text{if } w \in G \\ & \text{and the path is valid,} \\ \text{threshold reached} & \text{otherwise.} \end{cases}$$

   This function continues until all feasible paths are evaluated.
3. **Path Traversal and Shortest Path Selection:** Traverse all newly discovered or amended paths to identify the shortest possible paths among them:

   $$S = \min_{\text{paths } p \in P} \{ \text{length}(p) \}$$

   This step involves calculating the length of each path and selecting the minimum.
4. **Path Finalization:** Finalize the set of shortest paths for inclusion in the refined sub-graph:

   $$G^* = \{ p \mid p \text{ is among the shortest in } S \}$$

This refined approach ensures that the paths in the extracted subgraph are complete and form an optimal neighbourhood in regards to graph and semantic similarity, facilitating more accurate and reliable knowledge retrieval. The entire process can be visualized in Fig. 5.

5. Experiments

This section outlines our study's datasets, evaluation metrics, and baselines. The evaluation focuses on assessing the quality of triplets (facts) retrieved directly from KGs. Our approach bypasses traditional preprocessing steps such as entity span detection, entity disambiguation, and relation classification. We evaluate how our method performs compared to contemporary and earlier methods of fact retrieval from KGs.

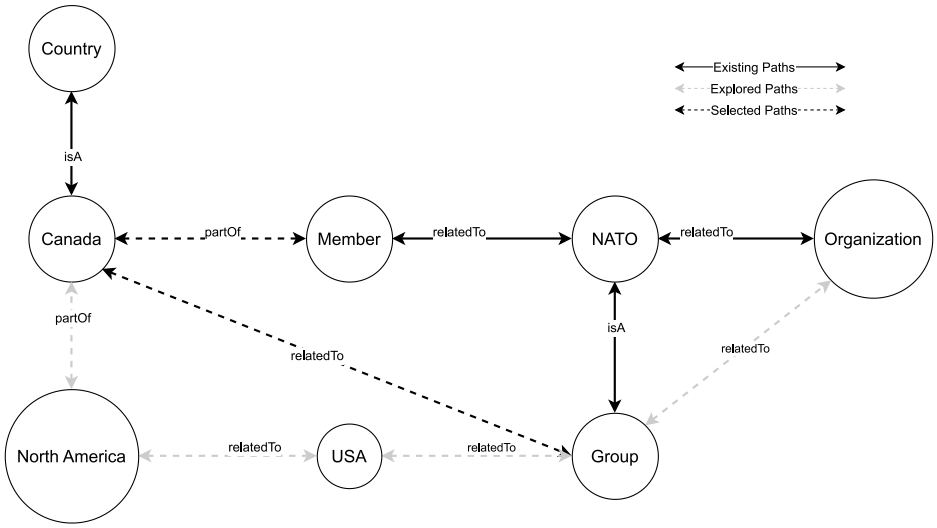


Fig. 5. This figure demonstrates the operation of the path generation utility. The objective is to construct paths between nodes to establish a connected graph. The process begins by extending the current nodes through a novel attention query, selecting only one neighbour at a time until overlapping nodes are identified. Finally, the shortest path is selected among all paths explored by the attention query.

5.1. Datasets

We evaluate the performance of our approach on fact retrieval tasks whose goal is to retrieve relevant triplets over KGs given a query and answer questions given a query. We used several datasets for the question-answering task.

In question-answering tasks, we aim to predict the factual triplets, which we consider direct answers. (See Table 1.)

- SimpleQuestions[1] (Bordes et al., 2015) is a question answering dataset which is designed with Freebase KG (Bollacker et al., 2008).
- Commonsense_QA[2] (Talmor et al., 2019) is an MCQ question-answering dataset that requires different types of commonsense knowledge to predict the correct answers. It is designed with ConceptNet KG (Speer et al., 2018).
- OpenBook_QA[3] (Mihaylov et al., 2018) contains questions that require multi-step reasoning, the use of additional common and common sense knowledge, and rich text comprehension. OpenBookQA is a new kind of question-answering dataset modelled after open-book exams for assessing human understanding of a subject.
- PubMed_QA[4] (Jin et al., 2019) is a medical dataset where research questions are answered with yes/no/maybe (e.g., Do preoperative statins reduce atrial fibrillation after coronary artery bypass grafting?) using the corresponding abstracts.
- Head_QA[5] (Vilares & Gómez-Rodríguez, 2019) is a multi-choice healthcare dataset. The questions come from exams to access a specialized position in the Spanish healthcare system and are challenging even for highly specialized humans. The dataset contains questions about the following topics: Medicine, Nursing, Psychology, Chemistry, Pharmacology and Biology.
- Mintaka[6] (Sen et al., 2022) is a question answering dataset designed with the wikidata KG (Vrandečić & Krötzsch, 2014).

Table 1
Statistics of the selected Datasets. The table provides an overview of the datasets used in the study.

Dataset	Type	Samples
SimpleQuestions	Triplet	21,687
Commonsense_QA	MCQ	12,102
OpenBook_QA	MCQ	35,928
PubMed_QA	True/False	2,39,668
Head_QA	MCQ	97,432
Mintaka	Direct Answer	36000

5.2. Evaluation

1. **BERTScore**: We utilize BERTScore[7] (Zhang et al., 2020), a metric that calculates the similarity between predicted and ground-truth sequences using contextual embeddings. BERTScore has demonstrated a strong correlation with human evaluations, as verified by the original BERTScore paper (Zhang et al., 2020). To calculate the BERTScore for a given query question q and the corresponding extracted triples from KGs $(t_1, \ldots, t_n)$, where t is the element index of word in the query, the following formula is applied:

$$\text{score\_topic}_t = \max_{[1,\ldots,n]} \text{BERTScore}(q_t, t_{ti}) \tag{1}$$

The overall evaluation score for the model is computed as the average BERTScore across all triplets:

$$\text{score\_model} = \frac{1}{T}\sum_{t=1}^{T} \text{score\_triplet}_t \tag{2}$$

2. **Coverage**: Coverage is defined as the ratio of tokens from the input query that appeared in the retrieved triplets (facts) from KG.

$$\text{Coverage} = \frac{|Q \cap T|}{|Q|} \tag{3}$$

where:
Q = set of tokens in the input query.
T = set of tokens from retrieved triplets in the KG.

[1] https://huggingface.co/datasets/fbougares/simple_questions_v2
[2] https://huggingface.co/datasets/tau/commonsense_qa
[3] https://huggingface.co/datasets/allenai/openbookqa
[4] https://huggingface.co/datasets/qiaojin/PubMedQA
[5] https://huggingface.co/datasets/dvilares/head_qa
[6] https://huggingface.co/datasets/AmazonScience/mintaka

[7] Results are computed using the official implementation: https://github.com/Tiiiger/bert_score

3. **Hits@K**: Measures, whether retrieved Top-K triplets, include a correct answer or not.

5.3. Baselines

1. **Linked Entity Retrieval**: It predicts relationships among candidate triplets connected to identified entities using different entity linking methods. These include **spaCy** (Honnibal et al., 2020), **GENRE** (Cao et al., 2021), **BLINK** (Wu et al., 2020), and **ReFinED** (Ayoola et al., 2022) for Wikidata, as well as **GrailQA** (Gu et al., 2021) for Freebase.
2. **Factoid QA**: It retrieves entities based on their similarities with input query (Lukovnikov et al., 2017).
3. **DiFaR**: Direct knowledge retrieval directly retrieves the nearest triplets to the input text on the latent space (Baek et al., 2023).
4. **Roberta-large**: Is a common language model baseline for multi-chose question answering tasks (Liu et al., 2019).
5. **Howard** (Howard et al., 2022): This method aimed to extract a domain-focused subgraph from KG.
6. **Shangwen** (Lv et al., 2019b): This method aimed to select and extract a relevant neighbourhood from the KG.
7. **Sha** (Sha et al., 2023): This method aimed to reason using a density matrix along the knowledge path, extract sub-graphs, and fuse graph entities with a bidirectional attention strategy.
8. **Our Methods**: Our **A2Q** directly retrieves all the relevant triplets to the input text from the KG based on semantic and contextual similarity. **A2Q-vanilla** does not include a fine-tuned relevance classifier and tries to use the base model as it is.

5.4. Implementation details

We use our novel Attention Query method for fact retrieval from KG and a lightweight MiniLM[8] as a relevance classifier. The relevance classifier is pre-trained on the MS MARCO dataset (Dong & Lapata, 2016). We then further tuned a manually curated reranking triplet dataset based on MS MARCO datasets (Bajaj et al., 2018) with concepts from ConceptNet KG. The relevance classifier model is fine-tuned by Pytorch. The batch size was set to 16, and the maximum input sequence defaulted to the original. The learning rate of the model is set to 10^{-5}. The model was trained on a single GPU (V100) with 8 GB VRAM and lots of optimizations like gradient accumulation, etc, to optimally fit in smaller VRAM.

5.4.1. Hyperparameter discussion and sensitivity

In our approach, two key hyperparameters govern the fact retrieval process: (1) the *threshold* for neighbourhood expansion (Θ), which controls how many layers of neighbours are explored around each token, and (2) the *scaling factor* λ, which penalizes more distant nodes during expansion. Below, we discuss these hyperparameters and how they influence performance:

- **Threshold Θ for Neighbourhood Expansion:** This value determines how far we explore the graph around each query token. A small Θ (e.g., 2) restricts the search to immediate neighbours and speeds up retrieval but risks missing relevant nodes farther away. A large Θ (e.g., 5 or more) can improve coverage but increases computational cost and may introduce noisy or redundant nodes. In our experiments, we found that $\Theta = 3$ struck a balance between coverage and efficiency on most datasets.

[8] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

Table 2
Evaluation on SimpleQuestions and Mintaka datasets based on Hits@N. This table presents the performance of various methods on the SimpleQuestions and Mintaka datasets, measured by Hits@1 and Hits@10 metrics. The methods are compared to highlight the effectiveness of the proposed A2Q model, which achieves the highest Hits@1 and Hits@10 scores on both datasets. Hit@K measures whether the retrieved Top-K triplets include a correct answer or not.

Method	SimpleQuestions		Mintaka	
	Hits@1	Hits@10	Hits@1	Hits@10
Retrieval with Gold Entities	0.59	0.94	0.09	0.29
Retrieval with spaCy	0.29	0.44	0.05	0.16
Retrieval with GENRE	0.13	0.22	0.06	0.15
Retrieval with BLINK	0.42	0.67	0.08	0.24
Retrieval with ReFinED	0.41	0.62	0.08	0.23
Factoid QA by Retrieval	0.69	0.93	0.08	0.23
DiFaR	0.76	0.94	0.13	0.36
Howard	0.53	0.71	0.05	0.17
Shangwen	0.64	0.67	0.04	0.15
A2Q-vanilla	0.71	0.86	0.15	0.32
A2Q	**0.78**	**0.94**	**0.15**	**0.37**

- **Scaling Factor λ in Distance Penalty:** We apply a distance-based penalty to attenuate the scores of nodes located several hops away from the starting token. A higher λ heavily penalizes distant nodes, reducing noise but sometimes discarding potentially useful contexts. A lower λ keeps more long-range nodes in play but can introduce spurious connections. Empirically, we observed that λ values in the range of 0.1–0.3 typically yielded a good trade-off across diverse queries in our evaluation.

6. Results and discussion

In this section, we evaluate the performance of our approach against baseline models across six datasets. We analyse the components and features of our model and compare them to the baselines. The results are detailed in Tables 2, 3, and 4.

Our A2Q framework demonstrates superior performance against all baselines, except when compared to incomparable models such as Retrieval with Gold Entities, which utilize labelled entities.

6.1. Main results

First, we conducted experiments on question-answering datasets, and the results are presented in Table 2. Our A2Q framework significantly outperforms all baselines on both datasets, SimpleQuestions and Mintaka, in unsupervised experimental settings with substantial margins.

We further experimented with MCQ-based datasets, and the results are shown in Table 3. As displayed, our A2Q framework significantly outperforms all baselines with substantial margins, similar to the results on question-answering datasets, demonstrating that our framework is highly effective in fact retrieval tasks.

Further, we tested our approach with the True/False dataset Head_QA shown in Fig. 9, which does not have direct answers to retrieve but instead relies on verifying the correctness of the information, akin to fact verification. Our A2Q framework outperforms all baselines, similar to the results on question-answering and MCQ datasets, as shown in Table 4.

To evaluate the performance gains from our fine-tuned relevance classifier in improving the triplet ranking strategy, we compared the performances of two model variants: A2Q and A2Q-Vanilla. The enhanced relevance classifier brings significant performance improvements, particularly when relevant data is sparse or missing, as shown in Tables 2 and 3.

Finally, the error bar analysis (Fig. 9) provides insight into the variability of the method's performances, with smaller error bars indicating

Table 3
Evaluation on MCQ-based CommonsenseQA and OpenbookQA datasets. This table presents the performance of various methods on the CommonsenseQA and OpenbookQA datasets, evaluated across Hits@1, Hits@10, BERTScore, and Coverage metrics. The A2Q model demonstrates superior performance, achieving the highest scores in most metrics for both datasets.

Method	CommonsenseQA				OpenbookQA			
	Hits@1	Hits@10	BERTScore	Coverage	Hits@1	Hits@10	BERTScore	Coverage
Retrieval with spaCy	0.33	0.47	0.35	0.41	0.35	0.44	0.32	0.44
Retrieval with BLINK	0.45	0.69	0.37	0.42	0.44	0.67	0.36	0.46
Roberta-Large	0.67	0.91	0.41	0.47	0.65	0.76	0.41	0.48
Sha	0.72	0.91	0.44	0.52	0.71	0.89	0.47	0.51
Factoid QA by Retrieval	0.72	0.94	0.41	0.44	0.74	0.92	0.43	0.47
Howard	0.55	0.71	0.36	0.39	0.57	0.67	0.37	0.41
Shangwen	0.65	0.68	0.39	0.46	0.67	0.65	0.41	0.47
A2Q-vanilla	0.74	0.88	0.45	0.47	0.75	0.84	0.44	0.49
A2Q	**0.79**	**0.95**	**0.49**	**0.59**	**0.81**	**0.92**	**0.48**	**0.58**

Table 4
Evaluation on PubMed_QA and Head_QA medical datasets. This table compares the performance of various methods on the PubMed_QA and Head_QA datasets, measured by Hits@1, Hits@10, BERTScore, and Coverage metrics. The A2Q model outperforms other methods, particularly in Hits@1 and Coverage, demonstrating its effectiveness in these medical and academic QA contexts.

Method	PubMed_QA			Head_QA			
	Hits@1	Hits@10	Coverage	Hits@1	Hits@10	BERTScore	Coverage
Retrieval with spaCy	0.29	0.44	0.39	0.26	0.41	0.35	0.32
Roberta-Large	0.39	0.61	0.34	0.41	0.62	0.36	0.34
Factoid QA by Retrieval	0.65	0.69	0.37	0.61	0.71	0.39	0.35
Howard	0.51	0.68	0.35	0.49	0.67	0.31	0.34
Shangwen	0.61	0.65	0.42	0.59	0.61	0.35	0.32
A2Q-vanilla	0.66	0.65	0.44	0.65	0.66	0.41	0.36
A2Q	**0.72**	**0.67**	**0.45**	**0.69**	**0.71**	**0.42**	**0.42**

Table 5
Sample queries and corresponding extracted information by the model. This table presents examples of queries, the relevant triplets retrieved by the model, and the original answer. The examples illustrate the model's ability to retrieve pertinent facts which consists of accurate responses.

Question: Who was the president of the USA in 1963? **Triplets Retrieved:** (Robert F. Kennedy, relatedTo USA), (Robert F. Kennedy, president, USA), (president, relatedTo, Kennedy) **Answer:** Kennedy
Question: Who commanded Germany during WW2 and founded the National Socialist Party? **Triplets Retrieved:** (Hitler, relatedTo, WW2), (National Socialist Party, founder, Adolf Hitler), (Hitler, commander, Germany) **Answer:** Hitler
Question: What happens when mercury is placed in water? **Triplets Retrieved:** (Mercury, isA, heavy metal), (Mercury, sinks, water) **Answer:** it sinks

higher consistency, again showcasing the A2Q model's stability and effectiveness.

We have also done an extended comprehensive evaluation of the model's performance across multiple dimensions, as detailed in Appendix. Our evaluation indicates that the A2Q model has higher consistency, reliability, and performance based on our evaluation. Altogether our analysis shows A2Q's superiority and better performance on different evaluations across various datasets while performing in a zero-shot setting.

6.2. Analysis on medical datasets

To evaluate whether our A2Q framework can handle challenging retrieval tasks on medical datasets, which are not fully covered by general-purpose KGs such as ConceptNet, DBPedia, or WikiData, we tested two types of medical datasets: PubMed_QA (True/False questions) and Head_QA (MCQ type questions).

Our A2Q framework significantly outperforms all baselines, as shown in Table 4. Although the results are not as precise in terms of BERTScore accuracy, our model surpasses the baselines even with significantly constrained information. Our relevance classifier strategy also brings substantial performance improvements, especially on medical datasets.

6.3. Retrieved samples

Our A2Q model demonstrates strong performance when examining samples from the extracted results, as shown in Table 5. Despite not being explicitly trained to predict entities mentioned in the input query, the model retrieves information highly relevant to the query in terms of both contextual and semantic similarities. This capability likely arises from the framework's ability to differentiate between relevant and irrelevant parts of the graph.

6.4. Impact of the relevance classifier

To assess the impact of our fine-tuned relevance classifier, we compared the performance of the A2Q model with the classifier against

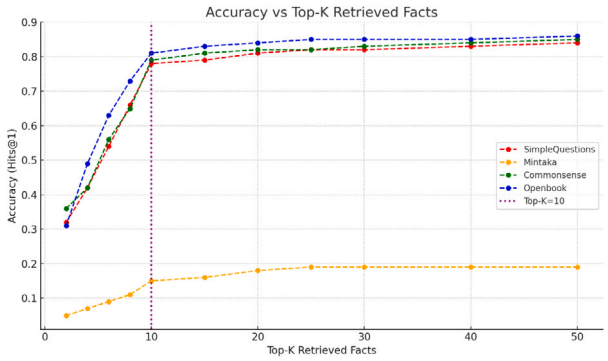


Fig. 6. Performance and efficiency of A2Q with varying top-K retrieved facts. The plots illustrate how the accuracy (Hits@1) changes as the number of retrieved triplets increases. The results are shown for the SimpleQuestions, Mintaka, Commonsense, and Openbook datasets. The results are based on 10 random runs for a randomly selected subset of 1000 samples.

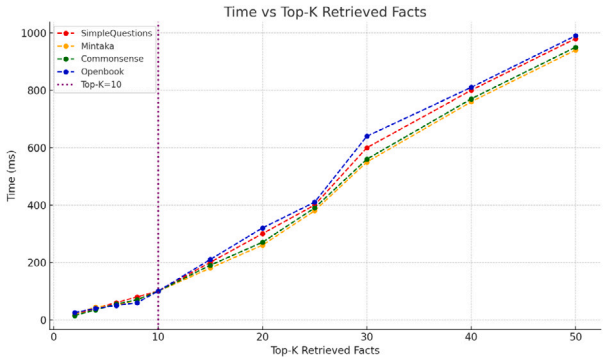


Fig. 7. Performance and efficiency of A2Q with varying top-K retrieved facts. The plots illustrate how the time taken (in milliseconds) changes as the number of retrieved triplets increases. The results are shown for the SimpleQuestions, Mintaka, Commonsense, and Openbook datasets. The results are based on 10 random runs for a randomly selected subset of 1000 samples.

Table 6
Performance comparison of different relevance classifiers. This table evaluates various models based on Hits@1 and Hits@10 metrics, highlighting the effectiveness of the MS-MARCO-MiniLM-Tuned model, which achieves the highest scores in both metrics.

Model	Hits@1	Hits@10
MiniLM	0.59	0.79
MS-MARCO-TinyBERT	0.64	0.81
MS-MARCO-MiniLM	0.65	0.83
MS-MARCO-MiniLM-Tuned	**0.67**	**0.84**

a baseline A2Q model without the classifier, which we refer to as "A2Q-vanilla." The A2Q-vanilla model does not include the fine-tuning process and serves as a benchmark for comparison. The results of this comparison are presented in Tables 2, 3, and 4.

In most cases, the inclusion of the fine-tuned relevance classifier led to significant performance improvements, allowing our model to outperform all baselines by substantial margins.

The pre-trained MS MARCO model using MiniLM demonstrated the best performance when employed as a relevance classifier, as detailed in Table 6. Further fine-tuning of this model resulted in substantial additional gains, leading us to adopt it as the primary base for our relevance classifier.

6.5. Ablation study

6.5.1. Analysing results with varying number of triplets K

The performance of our approach completely depends on the number of triplets it has extracted. Therefore, to further evaluate it, we have varied the value of K, which represents the number of outputted triplets, and reported performance in Figs. 6 and 7. The performance rapidly increases until ten triplets (k=10 for how many triplets the method should return as output) and then saturates afterwards, as shown in Figs. 6 and 7. Also, the time for extracting the triplets linearly increases till twenty and then explodes further near exponential. Although the time taken between the value of k from two to ten is relatively less, it increases suddenly when the graph expands due to an increase in the number of neighbours and then the corresponding increase in the number of sub-queries to evaluate each relation.

We have also completed a statistical analysis to check the consistency of our approach with other baselines, as shown in Fig. 8. In the plot, we have shown the analysis of ten different runs for each method with different random initializations on a random sample of one thousand samples from the CommonSense QA dataset. We can see in Fig. 8 that A2Q is performing comparative, if not better, than most baselines.

Table 7
Ablation study on the Commonsense_QA dataset evaluating the impact of different A2Q components. The table presents the BERTScore and Coverage metrics for various configurations of the A2Q model, demonstrating the contribution of each component to the overall performance. The full A2Q model achieves the highest scores, indicating the effectiveness of the complete architecture.

Component	BERTScore	Coverage
A2Q	**0.49**	**0.59**
A2Q-vanilla	0.45	0.47
A2Q-No-Taxonomy	0.42	0.45
A2Q-No-Classifier	0.39	0.44
A2Q-GraphEmbed	0.32	0.38
A2Q-Word2Vec	0.29	0.32
A2Q-Glove	0.28	0.29
A2Q-No-Path-Gen	0.41	0.45
A2Q-No-MCQ-Sub	0.34	0.35

6.5.2. Impact of components

This section discusses the impact of various components of the A2Q model based on an ablation study conducted on the CommonsenseQA dataset, as presented in Table 7.

First, we evaluated the model without the fine-tuned relevance classifier, referred to as "A2Q-vanilla." This modification led to a nearly ten percent decrease in both BERTScore and Coverage metrics, highlighting the critical role of selecting the top K triplets for high relevance.

Next, we tested the model without the taxonomy component, labelled "A2Q-No-Taxonomy." Although this also caused a performance drop, the impact was less severe. This is likely due to the robust coverage of the ConceptNet KG within this dataset, making the taxonomy less essential compared to other datasets with lower coverage.

We then examined the effect of removing the relevance classifier altogether, resulting in the "A2Q-No-Classifier" model. This change caused a significant performance decline, reinforcing the importance of triplet ranking, similar to the impact observed in the "A2Q-vanilla" model.

Additionally, we tested different embedding types, including Word2Vec ("A2Q-Word2Vec"), GloVe ("A2Q-Glove"), and ConceptNet Numberbatch embeddings ("A2Q-GraphEmbed"). After these evaluations, we opted for a combination of graph and transformer embeddings, as detailed in the methodology section.

Finally, we assessed the importance of path generation ("A2Q-No-Path-Gen"). Although its impact was less pronounced compared to the relevance classifier, removing path generation led to a noticeable performance drop. This component aids in recovering information lost during triplet ranking and path disruption. We also evaluated the significance of MCQ sub-graph extraction ("A2Q-No-MCQ-Sub"), finding

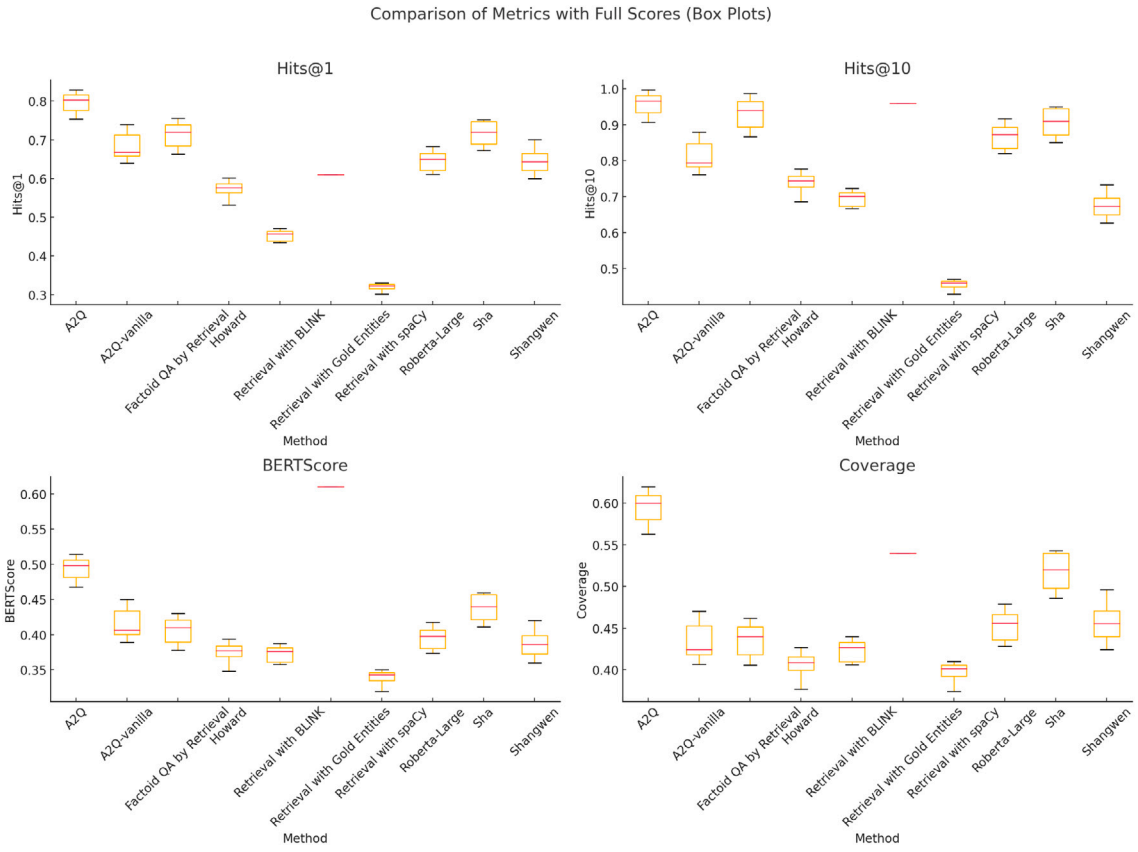


Fig. 8. Statistical analysis of A2Q compared to other baseline methods. This figure shows box plots for Hits@1, Hits@10, BERTScore, and Coverage metrics across various methods. The analysis was conducted using random samples of 1000 inputs from the CommonSense QA dataset, with each method evaluated 10 times with varying initializations and hyperparameters.

that its removal caused a substantial performance decline, particularly in MCQ datasets like CommonsenseQA, where the MCQ graph provides essential triplets for accurate answers.

6.5.3. Correlation and robustness

Informal sensitivity checks across different datasets (e.g., SimpleQuestions vs. Commonsense QA) showed that while optimal Θ and λ values differed slightly, our chosen defaults performed reasonably well overall. For instance, doubling Θ or λ had only a moderate effect on retrieval scores, indicating that the model is not critically sensitive to small hyperparameter shifts (Sections Section 5.4.1).

6.5.4. Error propagation from neighbourhood extraction

Although our approach organizes fact retrieval into sequential steps – neighbourhood extraction, triplet ranking, and re-ranking – some inaccuracies inevitably arise if neighbourhood extraction omits relevant nodes or includes tangential ones. Based on our existing evaluation data (Section 6), we note instances where the extracted subgraph was either incomplete or slightly noisy. Yet, overall performance metrics (e.g., Hits@1, Coverage) remained near their averages, indicating that subsequent modules can compensate for modest errors upstream. Three main factors contribute to this resilience:

1. **Global Attention Alignment (**Section 4**):** This directly compares each candidate triplet with the query, down-weighting or excluding facts that are only loosely connected to the input text.
2. **Relevance Classifier and Contextual Re-ranking (**Section 4**):** By systematically scoring and reevaluating triplets based on both semantic closeness and evolving context, these components help offset mistakes from earlier extraction stages.
3. **Path Generation (**Section 4**):** When initial neighbourhood extraction overlooks critical connections or forms disjoint subgraphs, path generation can reconnect relevant nodes by bridging partial subgraphs. This step effectively recovers certain relationships that might otherwise be lost due to early omissions or missteps.

These modules collectively ensure that errors in neighbourhood extraction typically exert only a minimal effect on the final retrieval quality for the datasets evaluated. As we expand our methodology to more specialized or large-scale KGs, a deeper look at how frequently misalignment occurs and how effectively path generation and re-ranking correct it will be central to confirming our system's robustness across diverse contexts.

6.6. Discussion of pre-trained model dependencies and knowledge base biases

We acknowledge the potential dependencies introduced by leveraging both pre-trained embeddings (e.g., BERT) and external KGs such as ConceptNet. Nonetheless, like any curated resource, ConceptNet can exhibit coverage gaps and biases. Our modular design addresses these risks by allowing the substitution of alternative or domain-specific KGs as they become available, thereby reducing over-reliance on any single

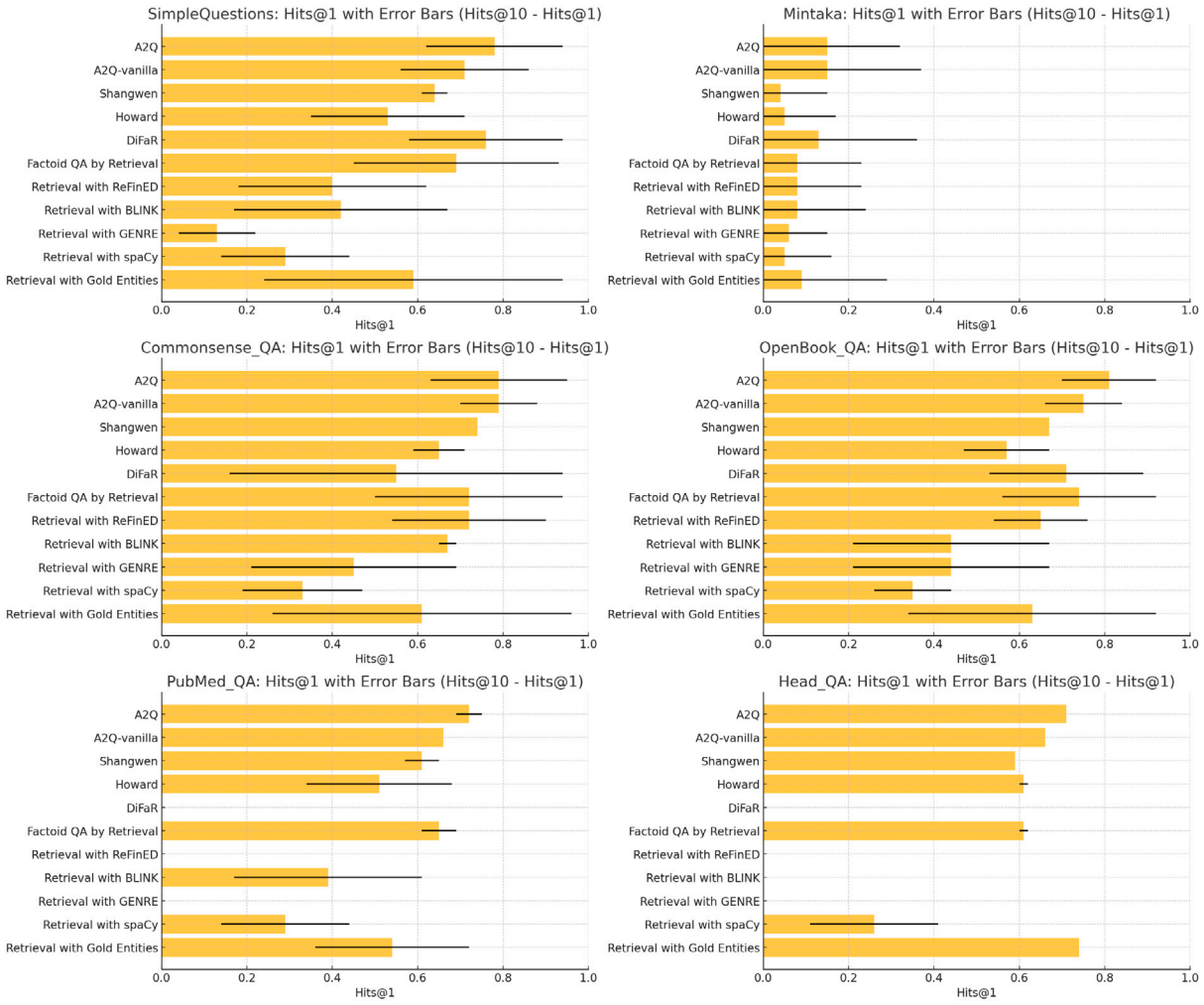


Fig. 9. Comparison of Hits@1 across multiple datasets with error bars. The plots depict the performance of various methods on the SimpleQuestions, Mintaka, Commonsense_QA, OpenBook_QA, PubMed_QA, and Head_QA datasets. Error bars represent the difference between Hits@10 and Hits@1 for each method, indicating variability in the model's ranking accuracy. A smaller error bar signifies that the method consistently ranks the correct answer higher among the top-K predictions, while a larger error bar suggests greater variability in ranking accuracy. If Hits@1 and Hits@10 scores are similar, the difference (Hits@10 - Hits@1) is minimal, resulting in a small or non-visible error bar, which indicates that the method consistently ranks correct answers very high. Larger error bars indicate a more significant difference between Hits@1 and Hits@10, meaning the method might not always rank the correct answer at the very top but frequently within the top 10.

source. Similarly, pre-trained language models inherit biases from their training corpora, which may influence retrieval outcomes. While our architecture currently utilizes BERT-based embeddings, future work will explore multiple embeddings and knowledge bases to broaden coverage and minimize the impact of such biases, all without necessitating structural changes to our retrieval pipeline.

6.7. Zero-shot generalization and threshold rationale

While our approach is presented as zero-shot-capable, we acknowledge that highly heterogeneous KGs may introduce domain-specific challenges. The key idea behind our zero-shot design is to embed the query text and triplets in a shared semantic space (using transformer-based embeddings), bypassing explicit training or fine-tuning for each new KG schema. This strategy has shown promise across multiple tested domains (Section 6), but we have not yet systematically assessed its limits on very large or highly specialized KGs. The thresholding steps in our triplet ranking (Sections 5.4.1 and 4) are primarily guided by empirical observations that show balancing coverage (i.e., extracting sufficient context) against precision (i.e., filtering noise) yields favourable results. In practice, thresholds are set to moderate levels to avoid overfitting to a single domain. Though these choices may seem ad hoc, they reflect a minimal calibration process based on the dataset sizes and complexity we evaluated. Future work will formalize these thresholds by examining broader parameter sweeps and leveraging optimization techniques such as Bayesian optimization or reinforcement learning to determine optimal values systematically. Additionally, we aim to derive theoretical guarantees for zero-shot retrieval under extreme domain shifts (e.g., very sparse or unusually large KGs) by analysing retrieval stability and generalization bounds. We also plan to explore adaptive thresholding strategies that dynamically adjust retrieval depth based on feedback signals, such as confidence scores or user interactions, potentially using self-supervised learning or reinforcement learning to reduce reliance on domain-specific heuristics.

7. Conclusion

This paper proposes a novel direct fact retrieval framework from KGs, which retrieves triplets from external knowledge sources based on

natural language queries. The main applications of this research are in information retrieval and information extraction. This method can be used with any application of KG, such as Question-Answering Systems, Chatbots, Fact-Checking Systems, and many more, to efficiently and accurately extract information from KGs for any downstream task. Our key innovations are as follows: (i) a novel use of an attention-based matrix to identify relevant neighbourhoods in the KGs and remove irrelevant nodes from a combined neighbourhood, and (ii) a novel relevance classifier strategy to improve triplet selection based on both the query and KG information. The combined A2Q framework can process multiple KGs and handle missing information links. A2Q performs on-the-go triplet selection without being computationally expensive while achieving similar or better results than conventional methods.

CRediT authorship contribution statement

Akhil Chaudhary: Conceptualization, Methodology, Data curation, Writing, Visualization, Investigation, Software. **Enayat Rajabi:** Supervision, Reviewing, Editing, Validation, Funding acquisition, Project administration. **Somayeh Kafaie:** Supervision, Reviewing, Editing, Validation, Funding acquisition. **Evangelos Milios:** Supervision, Reviewing, Editing, Validation.

Funding

This research has been funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant (RGPIN-2020-05869) and SMU Internal Research Grant 2023: Strategic Research Grant (201690 6400 FGSR Internal Grant Fund).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Performance

In this section, we have done an extended performance analysis of the performance of A2Q across various datasets using numerous evaluation techniques. The heatmap (Fig. A.10) provides a detailed view of the performance metrics across different methods on the Commonsense_QA dataset. It clearly illustrates the superiority of the A2Q model, particularly in the Hits@1 and Hits@10 metrics, while also showing its robust performance in BERTScore and Coverage. Complementing this, the spider chart (Fig. A.13) visualizes the comparative strengths and weaknesses of the top three and bottom three methods across all datasets, highlighting the A2Q model's consistent excellence across all metrics. The parallel coordinates plots (Figs. A.11 and A.12) offer a simultaneous comparison of Hits@1 and BERTScore across multiple datasets, further reinforcing A2Q's reliability and robustness in diverse scenarios. Together, these visualizations offer a holistic view of the models' performance, underscoring A2Q's superiority and consistent high performance across different evaluation metrics and datasets.

Data availability

Data will be made available on request.

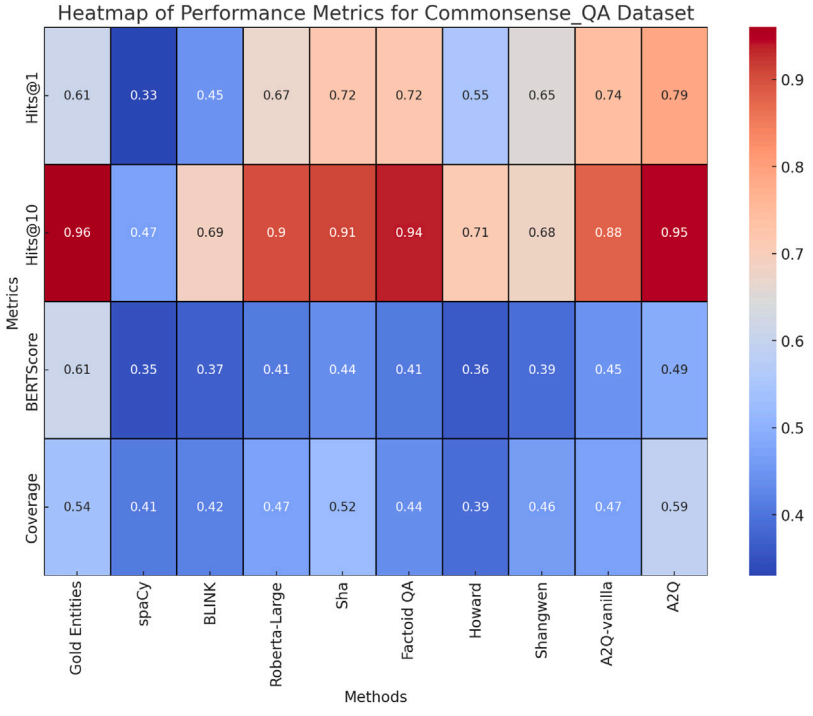


Fig. A.10. Heatmap of performance metrics for Commonsense_QA dataset. The heatmap illustrates the performance of different methods across Hits@1, Hits@10, BERTScore, and Coverage metrics. Each cell's colour intensity reflects the performance value, with darker shades indicating better performance. This visual representation enables a quick comparison of methods, showing how A2Q consistently outperforms others across all metrics.

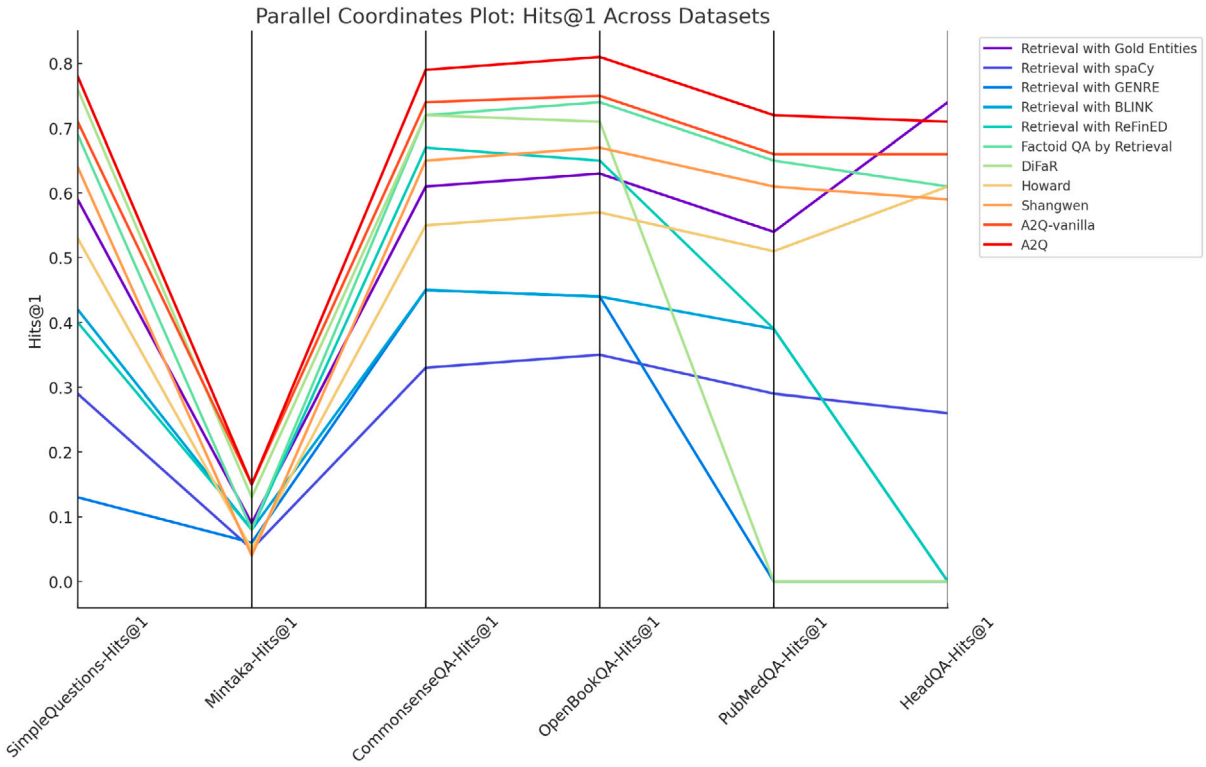


Fig. A.11. Parallel coordinates plot showing Hits@1 performance across multiple datasets. This plot simultaneously visualizes the Hits@1 scores of various methods on the SimpleQuestions, Mintaka, CommonsenseQA, OpenBookQA, PubMedQA, and HeadQA datasets. Each line represents a method, allowing for a direct comparison of performance across different datasets. The red line corresponding to A2Q indicates its consistently high performance relative to other methods.

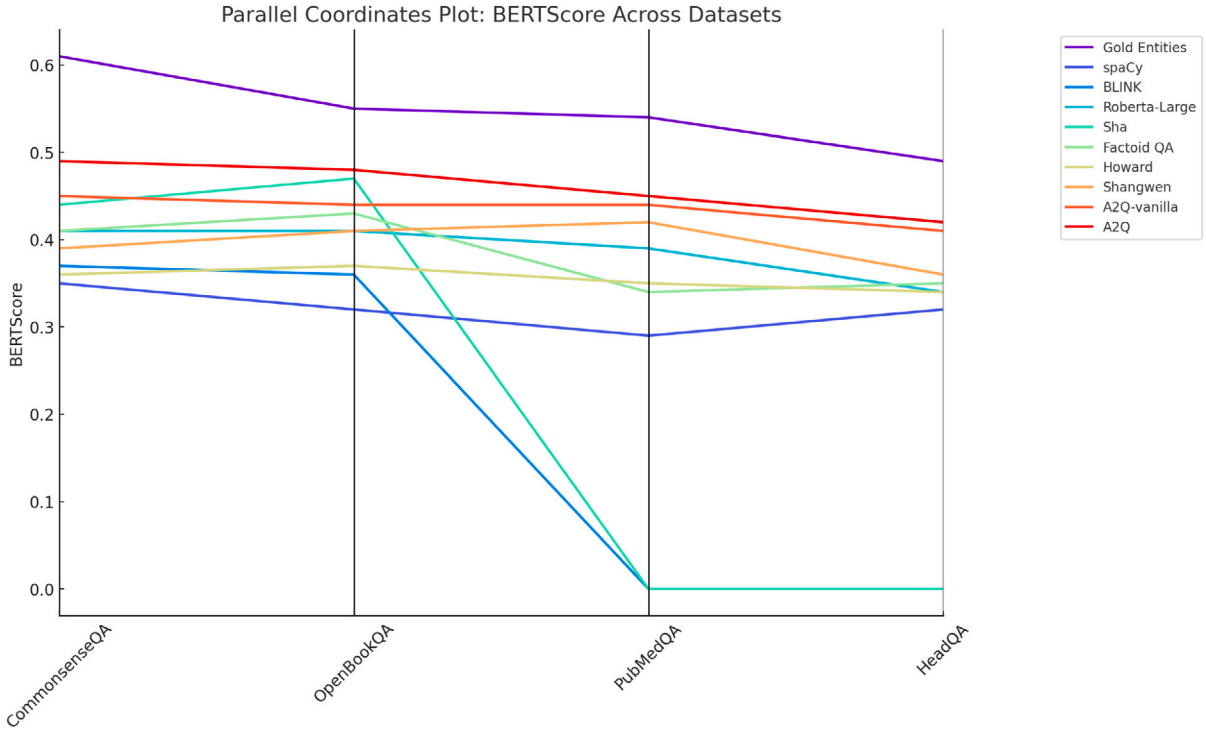


Fig. A.12. Parallel coordinates plot showing BERTScore performance across multiple datasets. This plot visualizes the BERTScore for various methods on the CommonsenseQA, OpenBookQA, PubMedQA, and HeadQA datasets. Each line represents a method, allowing for a direct comparison of semantic similarity performance across different datasets. The red line representing A2Q shows consistently high BERTScore values, indicating robust semantic relevance in its predictions compared to other methods.

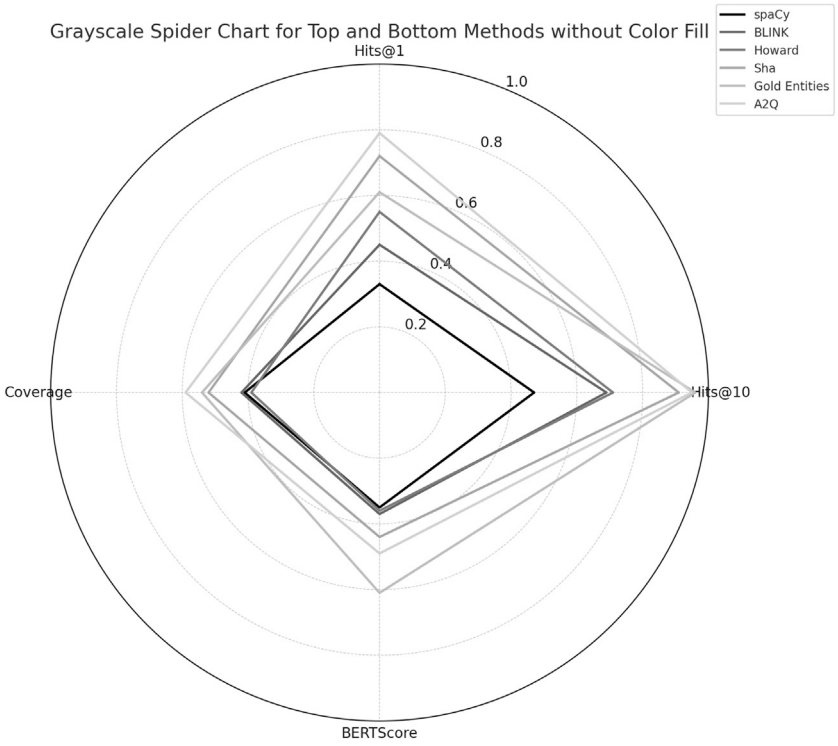


Fig. A.13. Spider chart comparing the top 3 and bottom 3 methods based on their average performance across all metrics. The chart displays Hits@1, Hits@10, BERTScore, and Coverage for each method, providing a visual summary of how the best and worst-performing methods differ in their capabilities. A2Q consistently shows strong performance across all metrics, as indicated by its position near the outer edge of the chart.

References

Ayoola, T., Tyagi, S., Fisher, J., Christodoulopoulos, C., & Pierleoni, A. (2022). ReFinED: An efficient zero-shot-capable approach to end-to-end entity linking. In A. Loukina, R. Gangadharaiah, & B. Min (Eds.), *Proceedings of the 2022 conference of the North American chapter of the association for computational linguistics: Human language technologies: Industry track* (pp. 209–220). Hybrid: Seattle, Washington + Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2022.naacl-industry.24, URL https://aclanthology.org/2022.naacl-industry.24.

Baek, J., Aji, A. F., Lehmann, J., & Hwang, S. J. (2023). Direct fact retrieval from knowledge graphs without entity linking. In A. Rogers, J. Boyd-Graber, & N. Okazaki (Eds.), *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 10038–10055). Toronto, Canada: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2023.acl-long.558, URL https://aclanthology.org/2023.acl-long.558/.

Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., Rosenberg, M., Song, X., Stoica, A., Tiwary, S., & Wang, T. (2018). MS MARCO: A human generated machine reading comprehension dataset. URL https://arxiv.org/abs/1611.09268. arXiv:1611.09268.

Bakhshi, M., Nematbakhsh, M., Mohsenzadeh, M., & Rahmani, A. M. (2020). Data-driven construction of SPARQL queries by approximate question graph alignment in question answering over knowledge graphs. *Expert Systems with Applications*, *146*, Article 113205. http://dx.doi.org/10.1016/j.eswa.2020.113205, URL https://www.sciencedirect.com/science/article/pii/S0957417420300312.

Bakken, M., & Soylu, A. (2023). Chrontext: Portable SPARQL queries over contextualised time series data in industrial settings. *Expert Systems with Applications*, *226*, Article 120149. http://dx.doi.org/10.1016/j.eswa.2023.120149, URL https://www.sciencedirect.com/science/article/pii/S0957417423006516.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *SIGMOD '08, Proceedings of the 2008 ACM SIGMOD international conference on management of data* (pp. 1247–1250). Association for Computing Machinery, http://dx.doi.org/10.1145/1376616.1376746, URL https://doi.org/10.1145/1376616.1376746.

Bordes, A., Chopra, S., & Weston, J. (2014). Question answering with subgraph embeddings. In A. Moschitti, B. Pang, & W. Daelemans (Eds.), *Proceedings of the 2014 conference on empirical methods in natural language processing* (pp. 615–620). Doha, Qatar: Association for Computational Linguistics, http://dx.doi.org/10.3115/v1/D14-1067, URL https://aclanthology.org/D14-1067.

Bordes, A., Usunier, N., Chopra, S., & Weston, J. (2015). Large-scale simple question answering with memory networks. ArXiv abs/1506.02075. URL https://api.semanticscholar.org/CorpusID:9605730.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ..., Hesse, C. et al. (2020). Language models are few-shot learners. In *Proceedings of the 34th international conference on neural information processing systems* (pp. 1–25). Red Hook, NY, USA: Curran Associates Inc..

Cao, N. D., Izacard, G., Riedel, S., & Petroni, F. (2021). Autoregressive entity retrieval. URL https://arxiv.org/abs/2010.00904. arXiv:2010.00904.

Chakraborty, N., Lukovnikov, D., Maheshwari, G., Trivedi, P., Lehmann, J., & Fischer, A. (2019). Introduction to neural network based approaches for question answering over knowledge graphs. arXiv:1907.09361.

Chen, Q., Ji, F., Zeng, X., Li, F.-L., Zhang, J., Chen, H., & Zhang, Y. (2021). KACE: Generating knowledge aware contrastive explanations for natural language inference. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)* (pp. 2516–2527). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.acl-long.196, URL https://aclanthology.org/2021.acl-long.196.

Chen, Y., Wu, L., & Zaki, M. J. (2019). Bidirectional attentive memory networks for question answering over knowledge bases. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Human language technologies, volume 1* (pp. 2913–2923). Minneapolis, Minnesota: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/N19-1299, URL https://aclanthology.org/N19-1299.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/N19-1423, URL https://aclanthology.org/N19-1423.

Dong, L., & Lapata, M. (2016). Language to logical form with neural attention. In K. Erk, & N. A. Smith (Eds.), *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 33–43). Berlin, Germany: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P16-1004, URL https://aclanthology.org/P16-1004.

Fu, B., Qiu, Y., Tang, C., Li, Y., Yu, H., & Sun, J. (2020). A survey on complex question answering over knowledge base: Recent advances and challenges. URL https://arxiv.org/abs/2007.13069. arXiv:2007.13069.

Galetzka, F., Rose, J., Schlangen, D., & Lehmann, J. (2021). Space Efficient Context Encoding for Non-Task-Oriented Dialogue Generation with Graph Attention Transformer. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)* (pp. 7028–7041). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.acl-long.546, URL https://aclanthology.org/2021.acl-long.546.

Gu, Y., Kase, S., Vanni, M., Sadler, B., Liang, P., Yan, X., & Su, Y. (2021). Beyond I.I.D.: Three levels of generalization for question answering on knowledge bases. In *WWW 201921*, *Proceedings of the web conference 2021* (pp. 0–1). ACM, http://dx.doi.org/10.1145/3442381.3449992.

Han, N., Topic, G., Noji, H., Takamura, H., & Miyao, Y. (2020). An empirical analysis of existing systems and datasets toward general simple question answering. In D. Scott, N. Bel, & C. Zong (Eds.), *Proceedings of the 28th international conference on computational linguistics* (pp. 5321–5334). International Committee on Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.coling-main.465, URL https://aclanthology.org/2020.coling-main.465.

Hao, Y., Zhang, Y., Liu, K., He, S., Liu, Z., Wu, H., & Zhao, J. (2017). An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In R. Barzilay, & M.-Y. Kan (Eds.), *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 221–231). Vancouver, Canada: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P17-1021, URL https://aclanthology.org/P17-1021.

Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). Spacy: Industrial-strength natural language processing in Python. In *SpaCy: Industrial-strength natural language processing in python* (pp. 0–1). http://dx.doi.org/10.5281/zenodo.1212303.

Howard, P., Ma, A., Lal, V., Simoes, A. P., Korat, D., Pereg, O., Wasserblat, M., & Singer, G. (2022). Cross-domain aspect extraction using transformers augmented with knowledge graphs. In *Association for computing machinery* (pp. 780–790). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3511808.3557275, URL https://doi.org/10.1145/3511808.3557275.

Jeong, S., Baek, J., Park, C., & Park, J. (2021). Unsupervised document expansion for information retrieval with stochastic text generation. In I. Beltagy, A. Cohan, G. Feigenblat, D. Freitag, T. Ghosal, K. Hall, D. Herrmannova, P. Knoth, K. Lo, P. Mayr, R. M. Patton, M. Shmueli-Scheuer, A. de Waard, K. Wang, & L. L. Wang (Eds.), *Proceedings of the second workshop on scholarly document processing* (pp. 7–17). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.sdp-1.2, URL https://aclanthology.org/2021.sdp-1.2.

Jin, Q., Dhingra, B., Liu, Z., Cohen, W., & Lu, X. (2019). PubMedQA: A dataset for biomedical research question answering. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *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. 2567–2577). Hong Kong, China: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D19-1259, URL https://aclanthology.org/D19-1259.

Kang, M., Kwak, J. M., Baek, J., & Hwang, S. J. (2022). Knowledge-consistent dialogue generation with knowledge graphs. In *Openreview.net* (pp. 1–23). URL https://api.semanticscholar.org/CorpusID:252760721.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)* (pp. 6769–6781). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.emnlp-main.550, URL https://aclanthology.org/2020.emnlp-main.550.

Lan, Y., He, G., Jiang, J., Jiang, J., Zhao, W. X., & Wen, J.-R. (2021). A survey on complex knowledge base question answering: methods, challenges and solutions. In *Twenty-ninth international joint conference on artificial intelligence, Vol. 5* (pp. 4483–4491). http://dx.doi.org/10.24963/ijcai.2021/611, URL https://www.ijcai.org/proceedings/2021/611.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., & Bizer, C. (2015). Dbpedia – a large-scale, multilingual knowledge base extracted from Wikipedia. In *IOS press, Vol. 6* (pp. 167–195). http://dx.doi.org/10.3233/SW-140134, URL https://doi.org/10.3233/SW-140134, 2.

Liang, P. (2013). Lambda dependency-based compositional semantics. arXiv:1309.4408.

Lin, B. Y., Chen, X., Chen, J., & Ren, X. (2019). KagNet: Knowledge-aware graph networks for commonsense reasoning. arXiv:1909.02151.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. URL https://arxiv.org/abs/1907.11692. arXiv:1907.11692.

Lukovnikov, D., Fischer, A., Lehmann, J., & Auer, S. (2017). Neural network-based question answering over knowledge graphs on word and character level. In R. Barrett, R. Cummings, E. Agichtein, & E. Gabrilovich (Eds.), *Proceedings of the 26th international conference on world wide web, WWW 2017, perth, Australia, April 3–7, 2017* (pp. 1211–1220). ACM, http://dx.doi.org/10.1145/3038912.3052675.

Luo, K., Lin, F., Luo, X., & Zhu, K. (2018). Knowledge base question answering via encoding of complex query graphs. In E. Riloff, D. Chiang, J. Hockenmaier, & J. Tsujii (Eds.), *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 2185–2194). Brussels, Belgium: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D18-1242, URL https://aclanthology.org/D18-1242.

Lv, S., Guo, D., Xu, J., Tang, D., Duan, N., Gong, M., Shou, L., Jiang, D., Cao, G., & Hu, S. (2019a). Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *AAAI conference on artificial intelligence* (pp. 1–9). URL https://api.semanticscholar.org/CorpusID:202565512.

Lv, S., Guo, D., Xu, J., Tang, D., Duan, N., Gong, M., Shou, L., Jiang, D., Cao, G., & Hu, S. (2019b). Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. (pp. 1–9). ArXiv abs/1909.05311. arXiv:1909.05311. URL https://api.semanticscholar.org/CorpusID:202565512.

Ma, K., Cheng, H., Liu, X., Nyberg, E., & Gao, J. (2022). Open domain question answering with a unified knowledge interface. In S. Muresan, P. Nakov, & A. Villavicencio (Eds.), *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)* (pp. 1605–1620). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2022.acl-long.113, URL https://aclanthology.org/2022.acl-long.113.

Mihaylov, T., Clark, P., Khot, T., & Sabharwal, A. (2018). Can a suit of armor conduct electricity? A new dataset for open book question answering. URL https://arxiv.org/abs/1809.02789. arXiv:1809.02789.

Min, S., Chen, D., Zettlemoyer, L., & Hajishirzi, H. (2019). Knowledge guided text retrieval and reading for open domain question answering. CoRR abs/1911.03868. URL http://arxiv.org/abs/1911.03868 arXiv:1911.03868.

Nogueira, R., Yang, W., Lin, J., & Cho, K. (2019). Document expansion by query prediction. arXiv:1904.08375.

Oguz, B., Chen, X., Karpukhin, V., Peshterliev, S., Okhonko, D., Schlichtkrull, M., Gupta, S., Mehdad, Y., & Yih, S. (2022). UniK-QA: Unified representations of structured and unstructured knowledge for open-domain question answering. In M. Carpuat (Ed.), *Findings of the association for computational linguistics: NAACL 2022* (pp. 1535–1546). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2022.findings-naacl.115, URL https://aclanthology.org/2022.findings-naacl.115.

Sap, M., Le Bras, R., Allaway, E., Bhagavatula, C., Lourie, N., Rashkin, H., Roof, B., Smith, N. A., & Choi, Y. (2019). ATOMIC: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence, 33* (01), (pp. 3027–3035). http://dx.doi.org/10.1609/aaai.v33i01.33013027, URL https://ojs.aaai.org/index.php/AAAI/article/view/4160.

Sen, P., Aji, A. F., & Saffari, A. (2022). Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering. URL https://arxiv.org/abs/2210.01613. arXiv:2210.01613.

Sha, Y., Feng, Y., He, M., Liu, S., & Ji, Y. (2023). Retrieval-augmented knowledge graph reasoning for commonsense question answering. *Mathematics*, *11*(15), 3269. http://dx.doi.org/10.3390/math11153269, URL https://www.mdpi.com/2227-7390/11/15/3269.

Singh, K., Lytra, I., Radhakrishna, A. S., Shekarpour, S., Vidal, M.-E., & Lehmann, J. (2020). No one is perfect: Analysing the performance of question answering components over the DBpedia knowledge graph. *Journal of Web Semantics*, *65*, Article 100594. http://dx.doi.org/10.1016/j.websem.2020.100594, URL https://www.sciencedirect.com/science/article/pii/S1570826820300342.

Speer, R., Chin, J., & Havasi, C. (2018). ConceptNet 5.5: An open multilingual graph of general knowledge. arXiv:1612.03975.

Talmor, A., Herzig, J., Lourie, N., & Berant, J. (2019). CommonsenseQA: A question answering challenge targeting commonsense knowledge. URL https://arxiv.org/abs/1811.00937. arXiv:1811.00937.

Tu, Y., Qiu, R., & Shen, H. (2023). SKG: A versatile information retrieval and analysis framework for academic papers with semantic knowledge graphs. ArXiv abs/2306.04758. URL https://api.semanticscholar.org/CorpusID:259108811 arXiv:2306.04758.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 6000–6010). Red Hook, NY, USA: Curran Associates Inc..

Vilares, D., & Gómez-Rodríguez, C. (2019). HEAD-QA: A healthcare dataset for complex reasoning. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 960–966). Florence, Italy: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P19-1092, URL https://www.aclweb.org/anthology/P19-1092.

Vrandečić, D., & Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, *57*(10), 78–85. http://dx.doi.org/10.1145/2629489, URL https://dl.acm.org/doi/10.1145/2629489.

Wang, Z., Ng, P., Nallapati, R., & Xiang, B. (2021). Retrieval, re-ranking and multi-task learning for knowledge-base question answering. In P. Merlo, J. Tiedemann, & R. Tsarfaty (Eds.), *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume* (pp. 347–357). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.eacl-main.26, URL https://aclanthology.org/2021.eacl-main.26.

Wu, L., Petroni, F., Josifoski, M., Riedel, S., & Zettlemoyer, L. (2020). Scalable zero-shot entity linking with dense entity retrieval. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing* (pp. 6397–6407). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.emnlp-main.519, URL https://aclanthology.org/2020.emnlp-main.519.

Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J., & Overwijk, A. (2020). Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv:2007.00808.

Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., & Liu, Q. (2019). ERNIE: Enhanced language representation with informative entities. In A. Korhonen, D. Traum, & L. Màrquez (Eds.), *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 1441–1451). Florence, Italy: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P19-1139, URL https://aclanthology.org/P19-1139.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). BERTScore: Evaluating text generation with BERT. (p. 14). ArXiv 0. arXiv:1904.09675.