Applied Intelligence (2025) 55:1102
https://doi.org/10.1007/s10489-025-06885-5

Knowledge graph-extended retrieval augmented generation for question answering

Jasper Linders[1] · **Jakub M. Tomczak**[1]

Received: 14 April 2025 / Accepted: 29 August 2025
© The Author(s) 2025

Abstract

Large Language Models (LLMs) and Knowledge Graphs (KGs) offer a promising approach to robust and explainable Question Answering (QA). While LLMs excel at natural language understanding, they suffer from knowledge gaps and hallucinations. KGs provide structured knowledge but lack natural language interaction. Ideally, an AI system should be both robust to missing facts as well as easy to communicate with. This paper proposes such a system that integrates LLMs and KGs without requiring training, ensuring adaptability across different KGs with minimal human effort. The resulting approach can be classified as a specific form of a Retrieval Augmented Generation (RAG) with a KG, thus, it is dubbed Knowledge Graph-extended Retrieval Augmented Generation (KG-RAG). It includes a question decomposition module to enhance multi-hop information retrieval and answer explainability. Using In-Context Learning (ICL) and Chain-of-Thought (CoT) prompting, it generates explicit reasoning chains processed separately to improve truthfulness. Experiments on the MetaQA benchmark show increased accuracy for multi-hop questions, though with a slight trade-off in single-hop performance compared to LLM with KG baselines. These findings demonstrate KG-RAG's potential to improve transparency in QA by bridging unstructured language understanding with structured knowledge retrieval.

Keywords Knowledge graphs · Large language models · Retrieval-augmented generation · Question answering

1 Introduction

As our world becomes increasingly digital and information is more widely available than ever before, technologies that enable information retrieval and processing have become indispensable in both our personal and professional lives. The advent of Large Language Models (LLMs) has had a great impact, by changing the way many internet users interact with information, through models like ChatGPT[1]. This has arguably played a large role in sparking an immense interest in solutions that build on artificial intelligence.

The rapid adoption of LLMs has transformed the fields of natural language processing (NLP) and information retrieval (IR). Understanding natural language, with its long-range dependencies and contextual meanings, as well as human-like text generation capabilities, allows these models to be applied to a wide variety of tasks. For instance, Question Answering (QA) is one of the main NLP tasks where a system is given a question (in natural language) and is expected to generate or extract the correct answer from a given context [1, 2]. There is a rich body of work on using Knowledge Bases (KBs) [3–5] or hybrids [6, 7] for QA, however, they all require costly and time-consuming updates to keep up-to-date with new facts. On the other hand, LLMs have proven to be few-shot learners, meaning that they have the ability to perform unseen tasks with only a couple of examples [8]. Unfortunately, the benefits of LLMs come at the cost of characteristic downsides, which are important to consider.

LLMs can hallucinate, generating untruthful or incoherent outputs [9, 10]. They also miss knowledge not present

---

[1] https://chatgpt.com/

Currently working at Chan Zuckerberg Initiative: jtomczak@chanzuckerberg.com.

✉ Jakub M. Tomczak
jmk.tomczak@gmail.com

Jasper Linders
jasper.linders@gmail.com

[1] Department of Mathematics and Computer Science, Eindhoven University of Technology, De Zaale, Eindhoven 5600 MB, the Netherlands

Published online: 19 November 2025

during training, leading to knowledge cutoff, and cannot guarantee that certain training data is remembered [11]. Because of their massive size and data requirements, LLMs are expensive to train, deploy, and maintain [12]. Thus, smaller models or those needing only fine-tuning can be more practical for many use cases.

By contrast, Knowledge Graphs (KGs) store information explicitly as entities and relationships, allowing symbolic reasoning and accurate answers [10]. Even if a direct link between entities is missing, inferences can be drawn from their shared associations. KGs may also recall underrepresented knowledge better than LLMs [11]. However, they are costly to build, specialized to a domain, and typically require querying languages rather than natural language [13]. They also do not easily generalize to other domains [10].

Retrieval-Augmented Generation (RAG) [14] addresses LLMs' lack of external knowledge by augmenting them with a text document database. Text documents are split into chunks, embedded, and stored in a vector database; the most similar chunks to an input query are retrieved and added to a prompt so the LLM can generate an answer based on this external information [15]. However, relying on unstructured text can miss comprehensive entity data and even introduce distracting misinformation [15].

To overcome these limitations, RAGs can utilize KGs [16–19]. The resulting system integrates structured data from Knowledge Graphs in a RAG, enabling precise retrieval and complex reasoning. For example, KAPING [20] performs Knowledge Graph Question Answering (KGQA) without requiring any training. When training is needed for KG-enhanced LLMs, issues arise such as limited training data, domain specificity, and the need for frequent retraining as KGs evolve [21, 22]. In short, while RAG enhances LLMs by providing explainable, natural language outputs, incorporating structured Knowledge Graphs may offer improved reasoning and domain adaptability.

In this paper, we propose the Knowledge Graph-extended Retrieval Augmented Generation (KG-RAG) system, which combines the reliability of Retrieval Augmented Generation (RAG) with the high precision of Knowledge Graphs (KGs) and operates without any training or fine-tuning. We focus on the task of Knowledge Graph Question Answering; although this focus is narrow, our findings may have broader implications. For instance, certain insights could be applied to the development of other systems that utilize KG-based information retrieval, such as chatbots. The primary objective of this work is to investigate how LLMs can be enhanced through the integration of KGs. Since the term "enhance" can encompass various improvements, we define it as follows. First, we aim to enable LLMs to be more readily applied across different domains requiring specialized or proprietary knowledge. Second, we seek to improve answer explainability, thereby assisting end users in validating LLM outputs. Eventually, we aim to answer the following research questions:

1. How can Large Language Models be enhanced with Knowledge Graphs without requiring any training?
2. How can answer explainability be improved with the use of Knowledge Graph-extended Retrieval Augmented Generation systems?

As a result, the contribution of the paper is threefold:

- We present an overview of SOTA QA methods that combine KGs and LLMs.
- We propose a new method for QA that combines a KG-extended RAG for retrieving facts with CoT for decomposing questions into sub-questions, and eventually an LLM for answering sub-questions and synthesizing them into the final answer. The method is explainable due to its question decomposition module and using KGs instead of a vector database. In our approach, we do not carry out any form of training of any of the modules.
- We provide an empirical validation of our approach on a well-known benchmark for QA, MetaQA. We indicate the importance of the question decomposition and the use of KG by analyzing results for 1-hop, 2-hop, and 3-hop questions.

2 Related work

2.1 Overview

Knowledge Graphs *Knowledge Graphs (KGs) are structured databases that model real-world entities and their relationships as graphs, which makes them highly amenable to machine processing. They enable efficient querying to retrieve all entities related to a given entity, a task that would be significantly more challenging with unstructured text databases. Complex queries are executed using specialized languages such as SPARQL [23]. As noted in recent research, "the success of KGs can largely be attributed to their ability to provide factual information about entities with high accuracy" [11]. Typically, the information in KGs is stored as triples, i.e. (subject, relation, object).*

Large Language Models *Large Language Models (LLMs) learn natural language patterns from extensive text data, enabling various NLP tasks such as text generation and sentiment classification. Their emergence was enabled by the Transformer architecture, introduced in Attention Is All You Need [24], which efficiently models sequential data via* *attention mechanisms. Scaling these models—by increasing compute, dataset size, and parameter count—yields performance improvements following a power law [25], with LLMs typically comprising hundreds of millions to hundreds of billions of parameters.*

LLMs generate text in an autoregressive manner. Given a sequence $x_{1:t}$, the model produces a probability distribution $p(x_{t+1}|x_{1:t}) = \mathrm{softmax}(z/T)$ over its vocabulary, where z are the raw logits and T is a temperature parameter that controls randomness. Instead of selecting tokens via simple argmax*, more sophisticated sampling methods are employed (see Section 3.4) to generate coherent and diverse output consistent with the input context [26].*

In-Context Learning & Chain-of-Thought *In-Context Learning (ICL) improves LLM performance by providing few-shot examples instead of zero-shot queries. This method boosts task performance through prompt engineering without altering model parameters [27]. It is often combined with Chain-of-Thought (CoT) that can significantly enhance performance without modifying the model's parameters or incurring the high cost of fine-tuning [28]. A CoT prompt instructs the model to generate intermediate reasoning steps that culminate in the final answer, rather than directly mapping a query to an answer [28]. This approach naturally decomposes complex queries into simpler steps, yielding more interpretable results.*

Knowledge Graph Question Answering *Knowledge Graph Question Answering (KGQA) is the task of answering questions using a specific knowledge graph (KG). Benchmarks such as Mintaka [29], WebQuestionsSP [30], and MetaQA [31] provide datasets where each row includes a question, its associated entity/entities, and the answer entity/entities, along with the corresponding KG (provided as a file of triples or accessible via an API). In these benchmarks, the question entity is pre-identified (avoiding the need for entity matching or linking), and performance is evaluated using the binary Hit@1 metric.*

KGQA systems are typically classified into three categories [20]:

- Neural Semantic Parsing-Based Methods: These map a question to a KG query (e.g., in SPARQL), reducing the search space between question and answer entities. Although effective [30], they require labor-intensive semantic parse labels.
- Differentiable KG-Based Methods: These employ differentiable representations of the KG (using sparse matrices for subjects, objects, and relations) to perform query execution in the embedding space. They enable end-to-end training on question-answer pairs [32, 33], but necessitate ample training data and may not generalize across different KGs.
- Information Retrieval-Based Methods: These combine KGs with LLMs by retrieving relevant facts—which are then injected into the prompt—to generate answers [20]. Although they leverage off-the-shelf components, they often require fine-tuning on KG-specific datasets [22].

Knowledge Graph-extended Retrieval Augmented Generation *Information retrieval-based KGQA (IR-KGQA) systems differ from neural semantic parsing and differentiable KG methods by delegating part of the reasoning over triples to the LLM. The process is split into retrieving candidate triples and then having the LLM reason over them to formulate an answer, whereas the other methods map directly from the question to the answer entities [32, 34].*

KG-RAG is defined as an IR-KGQA system that employs a similarity-based retrieval mechanism using off-the-shelf text embedding models, akin to the original RAG system [14]. In KG-RAG (exemplified by the KAPING system [20]), candidate triples are retrieved up to N hops from the question entity/entities, verbalized, and embedded alongside the question. Their similarity is computed via dot or cosine product, and the Top-K similar triples are passed to an answer generation LLM, which then outputs the answer.

2.2 State-of-the-art

Recent advances in question answering have seen the development of several state-of-the-art methods that leverage a diverse array of Large Language Models alongside innovative baseline strategies. For instance, one method employs multiple scales of models such as T5, T0, OPT, and GPT-3, while experimenting with baselines ranging from no knowledge to generated knowledge on datasets like WebQSP [30] and Mintaka [29]. Another approach expands this exploration by integrating Llama-2, Flan-T5, and ChatGPT, and introducing baselines that utilize triple-form knowledge and alternative KG-to-Text techniques, evaluated on datasets that include WebQSP, MetaQA [31], and even a Chinese benchmark, ZJQA [22]. Additionally, methods centered on ChatGPT are further compared with systems like StructGPT and KB-BINDER across varying complexities of MetaQA and WebQSP. The overview of the SOTA methods is presented in Table 1.

Table 1 Comparison of the question answering LLMs, baselines and benchmark datasets that were used for the different models

Model	QA LMs	Baselines	Datasets
KAPING [20]	T5 (0.8B, 3B, 11B) T0 (3B, 11B) OPT (2.7B, 6.7B) GPT-3 (6.7B, 175B)	No knowledge Random knowledge Popular knowledge Generated knowledge	WebQSP (w/ 2 KGs) Mintaka
Retrieve-Rewrite-Answer [22]	Llama-2 (7B, 13B) T5 (0.8B, 3B, 11B) Flan-T5 (80M, 3B, 11B) T0 (3B, 11B) ChatGPT	No knowledge Triple-form knowledge 2x Alternative KG-to-Text 2x Rival model	WebQSP WebQ MetaQA ZJQA (Chinese)
Keqing [21]	ChatGPT	ChatGPT StructGPT KB-BINDER	WebQSP MetaQA-1hop MetaQA-2hop MetaQA-3hop

The full set of QA LMs is as follows: T0 [35], T5 [36], Flan-T5 [37], OPT [38], GPT-3 [8], ChatGPT, AlexaTM [39], and Llama-2 [40]. The full set of datasets is as follows: WebQuestions [41], WebQSP [30], ComplexWebQuestions [42], MetaQA [31], Mintaka [29], LC-QuAD [43], and ZJQA [22]

2.2.1 KAPING

KAPING [20] is one of the best IR-KGQA models that requires no training. For example, due to the large number of candidate triples–27% of entities in WebQSP [30] have more than 1000 triples–a text embedding-based selection mechanism is employed, typically using cosine similarity [44], instead of appending all triples directly to the prompt. KAPING outperforms many baselines presented in Table 1 in terms of Hit@1, especially those with smaller LLMs, suggesting that external knowledge compensates for the limited parameter space. Notably, using 2-hop triples degrades performance, so only 1-hop triples are selected; when retrieval fails to fetch relevant triples, performance drops below a no-knowledge baseline. An additional finding is that triple-form text outperforms free-form text for retrieval, as converting triples to free-form via a KG-to-Text model often leads to semantic incoherence, and using free-form text in prompts does not improve answer generation.

2.2.2 Retrieve-rewrite-answer

Motivated by KAPING's limitations, the Retrieve-Rewrite-Answer (RRA) architecture was developed for KGQA [22]. Unlike KAPING, which overlooked the impact of triple formatting, RRA introduces a novel triple verbalization module, among other changes. Specifically, question entities are extracted from annotated datasets (with entity matching deferred). The retrieval process consists of three steps: (i) a hop number is predicted via a classification task on the question embedding; (ii) relation paths–sequences of KG relationships–are predicted by sampling and selecting the top-K candidates based on total probability; (iii) selected relation paths are transformed into free-form text using a fine-tuned LLM. This verbalized output, together with the question, is fed to a QA LLM via a prompt template.

For training, the hop number and relation path classifiers, as well as the KG-to-Text LLM, are tuned on each benchmark. Due to the lack of relation path labels and subgraph-text pairs in most benchmarks, the authors employ various data construction techniques, limiting the model's generalizability across domains and KGs.

As detailed in Table 1, evaluations were carried out using QA LLM, baselines (no knowledge, triple-form knowledge and two standard KG-to-Text models), and benchmark datasets, compared with models from [20] and [33] on WebQ [41] and WebQSP [30] using the Hit@1 metric. The main results show that RRA significantly outperforms rival models, achieving an improvement of 1–8% over triple-form text and 1–5% over the best standard KG-to-Text model. Moreover, RRA is about 100× more likely to produce a correct answer when the no-knowledge baseline fails, confirming the added value of IR-based KGQA models over vanilla LLMs.

2.2.3 Keqing

Keqing, proposed in [21], is the third SOTA model that is positioned as an alternative to SQL-based retrieval systems. Its key innovation is a question decomposition module that uses a fine-tuned LLM to break a question into sub-questions. These subquestions are matched to predefined templates via cosine similarity, with each template linked to specific KG relation paths. Candidate triples are retrieved based on these relation paths, and sub-questions are answered sequentially–the answer to one sub-question seeds the next. The triples obtained are verbalized and processed through a prompt template by a Quality Assurance LLM, ultimately generating a final answer that reflects the model's reasoning chain.

In this approach, only the question decomposition LLM is trained using LoRA [45], which adds only a small fraction of trainable weights. However, the construction of sub-question templates and the acquisition of relation path labels are not clearly detailed, which may limit the system's scalability.

According to Table 1, Keqing outperforms vanilla ChatGPT and two rival models, achieving Hit@1 scores of 98.4% to 99.9% on the MetaQA benchmark and superior performance on the WebQSP benchmark. Its ability to clearly explain its reasoning through sub-question chains further underscores its contribution to answer explainability.

3 Methodology

3.1 Problem statement

Let G be a knowledge graph, defined as a set of triples of the form (s, r, o) where:

- Each triple $(s, r, o) \in G \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ represents a fact;
- $s, o \in \mathcal{E}$ are entities from the set of all entities $\mathcal{E}$;
- $r \in \mathcal{R}$ is a relation from the set of all relations $\mathcal{R}$.

We assume that the following objects are given:

- A question q that can be answered using facts from G
- The question entity/entities part of that question $e_q \in \mathcal{E}$

Moreover, let us introduce the following variables:

- a denotes a natural language answer that can be derived from the facts in G;
- c is a reasoning chain in natural language, explaining the *logical* steps from q and e_q to a

Our **objective** is to develop a function f that maps given object to both an answer and the reasoning chain, namely:

$$f : q \times e_q \times G \to (a, c).$$

Ideally, the function f should fulfill the following requirement:

- It should have **high answer accuracy**, e.g., as evaluated by the Hit@1 metric;
- For each answer a generated by the function f, the reasoning chain c should provide a clear logical **explanation** of how the answer was derived, so that it is more easily verifiable by the user;
- The function f should operate **without training** or **finetuning** on specific Knowledge Graphs, using only In-Context Learning examples. The Knowledge Graphs must include sufficient amounts of natural language information, as the system relies on natural language-based methods.

The degree to which the function f achieves these requirements is evaluated using both quantitative and qualitative methods, based on experiments with a KGQA benchmark, namely:

- Quantitative evaluation of answer accuracy, based on the Hit@1 metric.
- Qualitative analysis of reasoning chain clarity and logical soundness, as judged by a human evaluator on a sample of results.

3.1.1 Current state-of-the-art: a research gap

After KAPING was introduced as the first KG-Augmented LLM for KGQA, RRA [22] and Keqing [21] followed, each employing different triple retrieval methods. Although all three use an LLM for question answering, KAPING relies on an untrained similarity-based retriever, while RRA and Keqing develop trainable retrieval modules, improving performance at the cost of significant engineering. Specifically, RRA trains separate modules (hop number classifier, relation path classifier, and KG-to-Text LLM) for each benchmark, requiring two custom training datasets (one for questions with relation path labels and one for triples with free-form text labels). The need for KG-specific techniques limits generalizability and raises concerns about the extra labor required when no Q&A dataset is available. Keqing fine-tunes an LLM for question decomposition to enhance answer interpretability and triple retrieval. This approach also demands a training dataset with sub-question templates and relation path labels, though the methods for constructing these remain unclear. Consequently, it is debatable whether the performance gains justify the additional engineering effort.

In summary, these shortcomings reveal a gap for models that are both as generalizable as KAPING and as explainable as Keqing. KAPING's training-free design allows minimal human intervention across diverse KGs and domains, even in the absence of benchmark datasets. For this reason, we propose an improvement to the KAPING model by introducing a question decomposition module.

3.2 Our approach

KAPING, a SOTA method combinining KGs and LLMs, outperforms many zero-shot baselines. However, its retrieval process, a vital process for accurate answer generation, can benefit from reducing irrelevant triple inclusion [20]. Therefore, we build on top of the KAPING model and propose to enhance it by integrating a question decomposition module to improve triple retrieval, answer accuracy, and explainability while maintaining application generalizability.

The proposed question decomposition module decomposes complex, multi-hop questions into simpler sub-questions. This allows the similarity-based retriever to focus on smaller, manageable pieces of information, thereby improving retrieval precision and yielding a more interpretable reasoning chain. Unlike conventional Chain-of-Thought prompting, which may induce hallucinated reasoning [46], decomposing the question forces the LLM to independently resolve each sub-question, ensuring fidelity to the stated reasoning. Our question decomposition module uses manually curated in-context learning examples for the KGQA benchmark, obviating the need for additional training and minimizing human labor. As a result, our approach aligns well with the goals of enhanced generalizability and answer explainability while potentially outperforming KAPING for multi-hop questions. The following section details the overall system architecture and the roles of its individual components.

3.3 System architecture

Our system comprises multiple components, each executing a specific role in answering KG-based questions. The overall process involves four primary steps, with the first two being non-sequential:

1. **Question Decomposition:** The decomposition module splits the question into sub-questions. For simple queries, it avoids unnecessary decomposition.
2. **Candidate Triple Retrieval:** Given the question entity, the system retrieves all triples up to N hops from the KG. Each triple is verbalized into text for subsequent selection via a sentence embedding model.
3. **Sub-Question Answering:** This sequential step answers each sub-question using the candidate triples. The process involves embedding the candidate triples to form a vector database, selecting the Top-K similar triples for the sub-question, and reformulating subsequent sub-questions based on prior sub-answers.
4. **Answer Synthesis:** Finally, the system synthesizes the final answer from the sub-questions and their corresponding answers. The output also includes the chain-of-thought from the decomposition stage, enhancing interpretability.

Figure 1 illustrates the system architecture, highlighting the data structures and interactions between components. The diagram shows how the question reformulation module, which processes all previous sub-answers, enables the sequential resolution of sub-questions until the final answer is generated by the answer synthesis module.

Different components utilize distinct data sources and models. The candidate triple retriever directly accesses the KG, while the similarity-based triple selection leverages an off-the-shelf sentence embedding model trained on question-answer pairs. The remaining modules—the decomposition module, sub-answer generator, question reformulator, and final answer generator—are implemented using a LLM.

3.4 System components

3.4.1 Question decomposition

Overview *The question decomposition module splits a complex question into simpler sub-questions while generating an explicit reasoning chain, thereby enhancing both triple retrieval and answer explainability (Section 3.2). Inspired by Chain-of-Thought and In-Context Learning techniques [46], the module uses manually constructed ICL examples from the benchmark (Section 4.1). The prompt is designed to first elicit the reasoning chain (CoT) followed by the sub-questions, aligning with the natural text-based reasoning of LLMs.*

Inputs and Outputs *As illustrated in Fig. 1, the module takes a natural language question as input and outputs a string containing the reasoning chain and sub-questions. This output is post-processed to extract the CoT and store the sub-questions in a list.*

Depending on the complexity of a question, it requires answering multiple sub-questions. For instance, for the question: "When were the films written by David Morlet released", it is first needed to query about films directed by David Morlet, and then, for each film, a release date must be indicated. A few examples of such question decomposition is provided in Table 2.

Techniques *The decomposition prompt instructs the LLM to decide if a question requires decomposition. If so, it generates a CoT followed by sub-questions, strictly adhering to a specified format and avoiding irrelevant content. In-context examples–covering three question types from the MetaQA benchmark–guide the LLM, with the stop token "<END>" marking completion.*

Implementation Details *Here, we use a 4-bit quantized version of Mistral-7B-Instruct-v0.2 [40, 47], originally a 7.24B-parameter model that outperforms Llama 2 and Llama 1 in reasoning, mathematics, and code generation. The quantized model, sized at 4.37 GB[2], is compatible with consumer-grade hardware (e.g., NVIDIA RTX 3060 12GB[3]). Fast inference is achieved using the* `llama.cpp` *package[4], and prompts are designed with LM Studio[5].*

---

[2] https://huggingface.co/TheBloke/Mistral-7B-Instruct-v0.2-GGUF

[3] https://www.msi.com/Graphics-Card/GeForce-RTX-3060-VENTUS-2X-12G-OC

[4] https://github.com/ggerganov/llama.cpp

[5] https://lmstudio.ai/

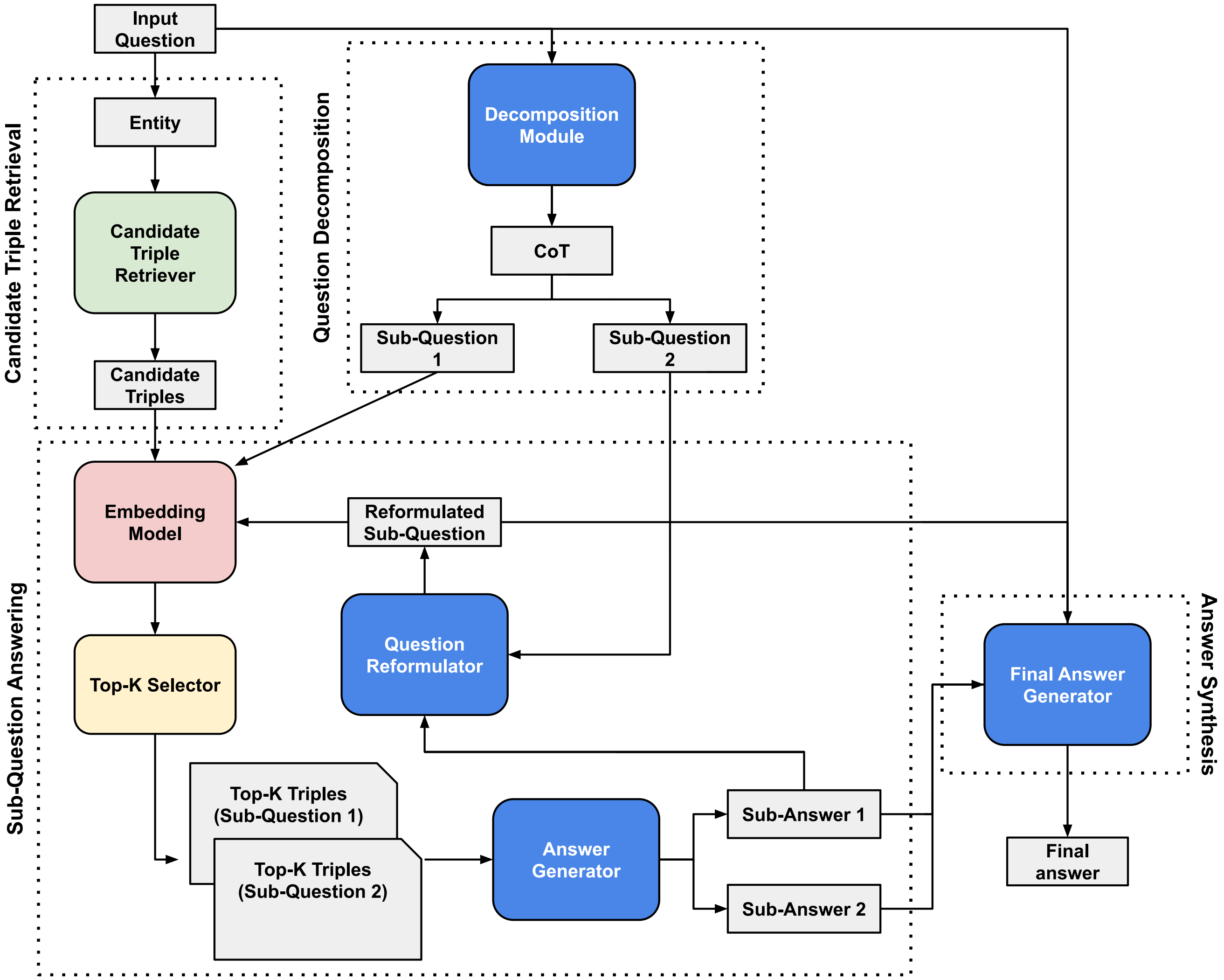


Fig. 1 The architecture of the proposed system. An example of a 2-hop question is included, to give an idea of the data structures that are involved in the end-to-end process. The green color indicates processing with the KG; the red block shows the embedding model and the blue modules utilize an LLM

Table 2 Examples of question decomposition for various types of questions

Type	Question	Sub-Questions
1-hop	Can you describe movie The Place Promised in Our Early Days in a few words?	**1.** What is the synopsis or plot summary of the movie ”The Place Promised in Our Early Days”?
2-hop	When were the films written by David Morlet released?	**1.** Which films were written by David Morlet? **2.** What is the release date for each of these films?
3-hop	What are the languages spoken in the movies starred by Presenting Lily Mars actors?	**1.** Who are the actors in the movie ”Presenting Lily Mars”?, **2.** What films have these actors acted in? **3.** In which languages are these films spoken?

Table 3 The inference parameters that were used for the question decomposition LLM

Parameter	Value
Max Tokens	256
Temperature	0.3
Min-p	0.05
Top-k	40

Inference parameters (see Table 3) include a max tokens limit (256) to prevent runaway generation, a temperature of 0.3 to reduce randomness, and top-k (40) and min-p (0.05) settings to ensure controlled token sampling [48].

3.4.2 Candidate triple retrieval

Overview *Candidate triple retrieval collects all triples up to N hops from a given question entity in the KG, converting each triple into a text string of the form (subject, relation, object). Although the worst-case complexity is exponential in the number of hops—approximately* $\Theta(d^N)$ *for an undirected KG with average degree d—real-world KGs are sparse, making the average or median complexity more relevant (Section 4.1). The value of N is treated as a hyperparameter.*

Inputs and Outputs *This component accepts the question entity/entities as a natural language string and retrieves candidate triples from the KG. The output is a list of lists, where each sub-list corresponds to the candidate triples for each hop up to N. Each triple is stored as a formatted text string, with underscores replaced by spaces (e.g., "acted_in" becomes "acted in").*

Techniques *Candidate triple retrieval employs a breadth-first search strategy. In the MetaQA benchmark, which uses a directed KG, retrieval can be unidirectional (considering only outgoing edges) or bidirectional (including both outgoing and incoming edges). For example, as illustrated in Fig. 2, unidirectional retrieval from the Inception entity would only yield entities like 2010, Christopher Nolan, and Tom Hardy, whereas bidirectional retrieval expands the search across successive hops. This example underscores the impact of retrieval direction on both the candidate set and computational load.*

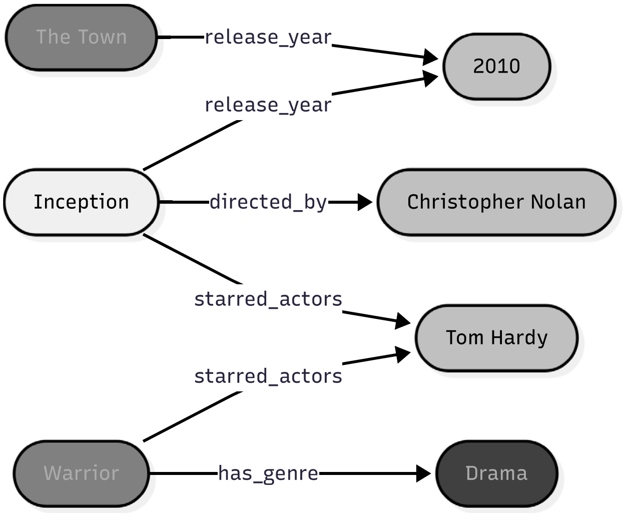


Fig. 2 A simple subgraph of triples from MetaQA [31]. As indicated by the arrows, this KG is a directed graph, which has implications for candidate triple retrieval. If *Inception* were the entity we were retrieving for, each darker tint of gray shows the entities that would be reached for a hop deeper

Implementation Details *The MetaQA benchmark provides the KG as a text file with one triple per row. This file is preprocessed into a compressed KG with indexed entities and relationships to streamline retrieval and minimize memory usage. Each triple is embedded using a sentence embedding model (introduced in Section 3.4.3), forming a dictionary of embeddings that enhances retrieval efficiency by avoiding redundant computations. Retrieval is performed bidirectionally up to 3 hops, i.e.,* $N \in \{1, 2, 3\}$.

3.4.3 Sub-question answering

Overview *Once the question is decomposed into sub-questions and candidate triples are retrieved for the given entity/entities, the sub-question answering process begins. Iteratively, the sub-question and candidate triples are embedded using a sentence embedding model, and the top-K similar triples are selected to generate a sub-answer via an LLM. This sub-answer is then used to reformulate subsequent sub-questions if needed (see Fig. 1), continuing until all sub-questions are answered.*

Inputs and Outputs *Inputs include candidate triples (a list of strings, pre-embedded from the MetaQA KG) and a list of sub-questions. The output comprises two lists of strings: one containing the sub-answers and another with the reformulated sub-questions, both of which contribute to the final answer synthesis.*

Techniques *The process employs similarity-based retrieval where both the sub-question and candidate triples are embedded with the same model, and their dot-product similarity is computed. The top-K triples are then passed to a zero-shot LLM answer generator along with the sub-question. Unlike Keqing's multiple-choice approach [21] (Section 2.2.3), this method allows the LLM to reason over the context. A zero-shot LLM also performs question reformulation.*

Implementation Details *The similarity-based triple selection uses the* `multi-qa-mpnet-base-dot-v1`[6] *model from the* `sentence_transformers`[7] *package, which embeds text into 768-dimensional vectors. Similarity is computed as the dot product between these vectors, and the model is run locally on the GPU. Both the sub-question answering and question reformulation LLMs use parameters from Table 3 with minor adjustments: the sub-question answering LLM employs a* `repeat_penalty` *of 1.1 to mitigate repetitive output, while the reformulation module*

---

[6] https://huggingface.co/sentence-transformers/multi-qa-mpnet-base-dot-v1

[7] https://www.sbert.net/

uses "?" as the stop token to restrict its output to a properly reformulated question.

3.4.4 Answer Synthesis

Overview *The final step synthesizes an answer to the original question using the generated reasoning chain, sub-questions, and sub-answers. This output, which includes the reasoning chain, provides transparency into the system's decision-making process.*

Inputs and Outputs *Inputs comprise the main question, reasoning chain, sub-questions (reformulated if applicable), and sub-answers—all as strings. The output is a single natural language string that integrates both the final answer and the reasoning chain.*

Techniques *A custom zero-shot prompt instructs the LLM to formulate the final answer from the provided context. The prompt template merges the main question, sub-questions, and sub-answers, and subsequently incorporates the reasoning chain into the final output. This straightforward zero-shot approach was preferred over ICL due to the simplicity of the final synthesis task compared to the more complex decomposition step.*

Implementation Details *The LLM parameters mirror those in Table 3, with the exception of* `max_tokens`*, which is increased to 512 to accommodate the typically more complex final answers.*

4 Experiments

The goal of our experiments is check whether the usefulness of a KG in question answering and whether our approach, i.e., using an additional question decomposition module, results in a better performance. For this purpose, we use a widely-used Knowledge Graph Question Answering (KGQA) benchmark called MetaQA [31]. In order to verify whether we achieved our objectives, we assess three baselines: a stand-alone LLM, an LLM with an LLM-based question-answering module, and an LLM with a KG (i.e., KAPING). Eventually, the experimental results are presented and discussed.

4.1 Dataset

The MetaQA benchmark, introduced in 2017, addresses the need for KGQA benchmarks featuring multi-hop questions over large-scale KGs, extending the original WikiMovies benchmark with movie-domain questions of varying hop counts [31].

Several factors motivated the selection of MetaQA for this research. First, its questions are categorized by hop count, enabling detailed analysis of multi-hop performance, a key area for improvement via question decomposition. Second, each question includes an entity label, avoiding the complexities of entity linking; many benchmarks, which focus on neural semantic parsing for SPARQL query generation, lack such labels [49]. Third, MetaQA's simplicity and locally processable KG make it ideal for studies with limited resources, in contrast to highly complex KGs like Wikidata (over 130 GB, 1.57 billion triples, 12,175 relation types[8]).

Data *MetaQA consists of three datasets (1-hop, 2-hop, and 3-hop), each split into train, validation, and test sets, and further divided into three components: vanilla, NTM text data, and audio data [31]. This research utilizes only the vanilla data, where the 1-hop dataset contains original WikiMovies questions and the 2-hop and 3-hop datasets are generated using predefined templates. Each dataset row includes a question, its associated entity, and answer entities.*

Knowledge Graph *The MetaQA benchmark provides a KG as a text file with each row representing a triple. The KG comprises 43,234 entities and 9 relation types, with movie titles as subjects. Figure 3 illustrates the degree distribution: most entities have few associated triples (median of 4), while the long-tailed distribution includes entities with up to 4431 triples.*

4.2 Experimental design

In this study, we carry out two experiments:

1. The goal of experiment 1 is to find out how the model parameters impact performance, in order to find a parameter configuration that leads to consistent performance over the different question types. The chosen parameter configuration can then be used to compare the system to baselines in the second experiment.
2. The main goal of the second experiment is to find out how different components of the system impact performance and overall behavior. This is achieved by comparing the performance of the system with specific baselines, which are essentially made up of combinations of system components.

---

[8] https://www.wikidata.org/wiki/Wikidata:Main_Page

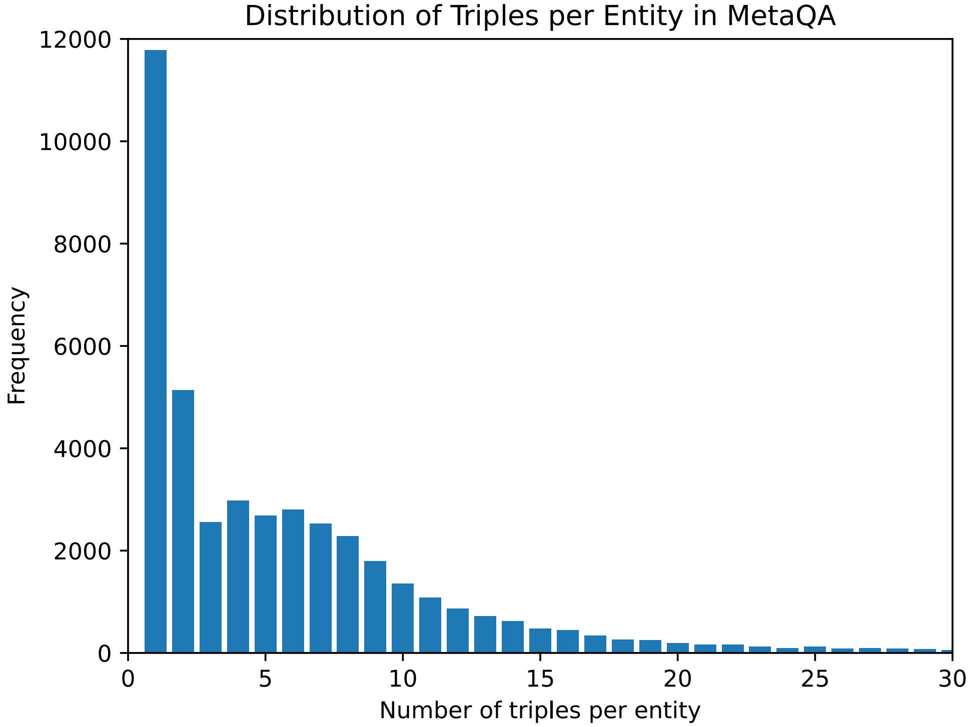


Fig. 3 The distribution of degrees (triples per entity) in the MetaQA KG. (Note that the distribution is long-tailed, so the cut-off at the value of 30 is for the purpose of visualization)

4.2.1 Experiment 1: model selection

Experiment 1 investigates the effect of model parameters on performance to determine a configuration that yields consistent results across different question types. The parameters under examination are the number of hops N for candidate triple retrieval (tested with values 1, 2, 3) and the number of top triples K selected for each sub-question (tested with values 10, 20, 30), consistent with values reported in the literature (Section 2).

For each MetaQA test dataset, 100 questions are sampled using a fixed seed, and the system is evaluated across all parameter combinations. This process is repeated with 10 different seeds (0–9) to capture performance variability, and all LLM components use the same seed for inference to ensure reproducibility.

Performance is measured using the Hit@1 metric, which checks if the generated answer exactly matches any of the label answer entities (after lowercasing and stripping). For example, if the label is "Brad Pitt" and the generated answer is "Pitt is the actor in question," the response is deemed incorrect. The final score for each dataset sample is the average Hit@1.

4.2.2 Experiment 2: a comparative analysis with baselines

Experiment 2 serves a purpose of assessing how individual system components influence overall performance by comparing the full system to three baselines:

1. **LLM:** Uses only an LLM with a simple zero-shot prompt to directly answer the question.
2. **LLM+QD:** Incorporates the question decomposition module to split questions and reformulate sub-questions before answering with the same zero-shot prompt as the LLM baseline.
3. **LLM+KG:** Functions as the full system without the question decomposition component, which is equivalent to KAPING [20] by employing candidate triple retrieval, top-K triple selection, and the sub-question answering module.

Both the full system and the LLM+KG baseline use the parameter configuration selected in Section 4.3.1. As in Experiment 1, 500 questions are sampled per MetaQA dataset using 8 different seeds (0–7) to ensure consistency. Performance is quantitatively evaluated using the Hit@1 metric to determine the impact of different components, and results are qualitatively analyzed for error insights and to assess accuracy, explainability, and generalizability as outlined in Section 3.1.

4.3 Results and discussion

4.3.1 Experiment 1: quantitative analysis

The results of Experiment 1 (Fig. 4) indicate high overall performance that decreases with increasing

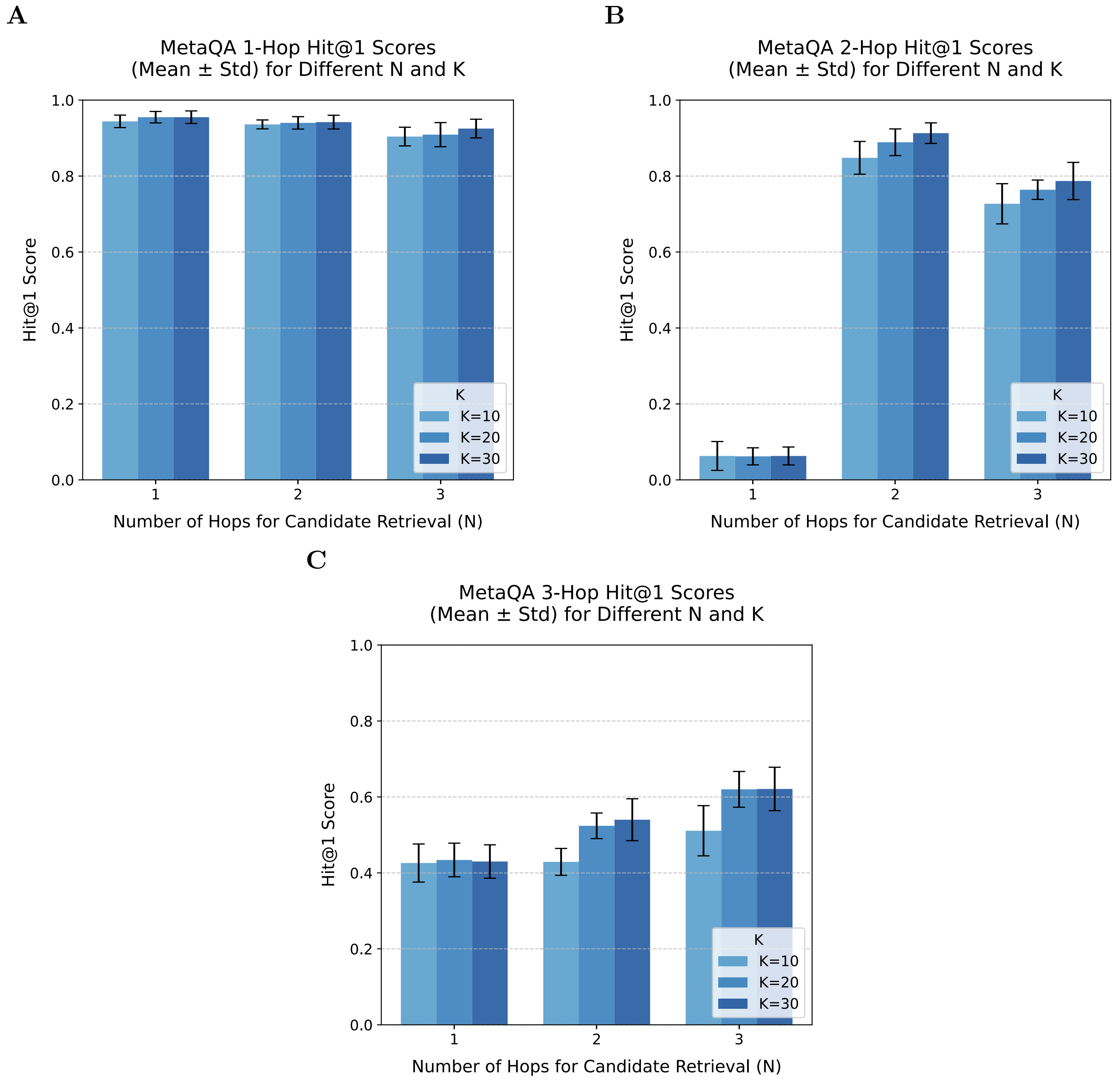


Fig. 4 MetaQA performance results for experiment 1, over 10 samples of 100 questions for each of the three datasets. The bars show the mean Hit@1 for different parameter configurations; the error bars show the standard deviation

question complexity, with standard deviations remaining low (≤ 0.063) across samples.

Performance is highest when the parameter N equals the actual number of hops in the questions. As expected, for the 2-hop dataset, $N = 1$ yields poor results; however, for the 3-hop dataset, performance with $N < 3$ is unexpectedly high due to MetaQA's question templates–for instance, some 3-hop questions (e.g., *"Who are the directors of the films written by the writer of Blue Collar?"*) can be answered with $N = 1$ triples. This represents a limitation of the MetaQA benchmark.

When holding the dataset and N constant, increasing K (the number of top triples selected) from 10 to 30 shows minimal effect on the 1-hop dataset, with slight improvements observed for the 2-hop and 3-hop datasets. Given that a higher K is unlikely to reduce performance and is more likely to include the necessary triples, $K = 30$ is chosen.

Considering the trade-offs across datasets, a balanced configuration is selected. Since $N = 1$ is unacceptable for 2-hop questions and improved performance on 3-hop questions likely requires all candidate triples up to 3 hops, $N = 3$ is deemed the best choice despite a minor reduction

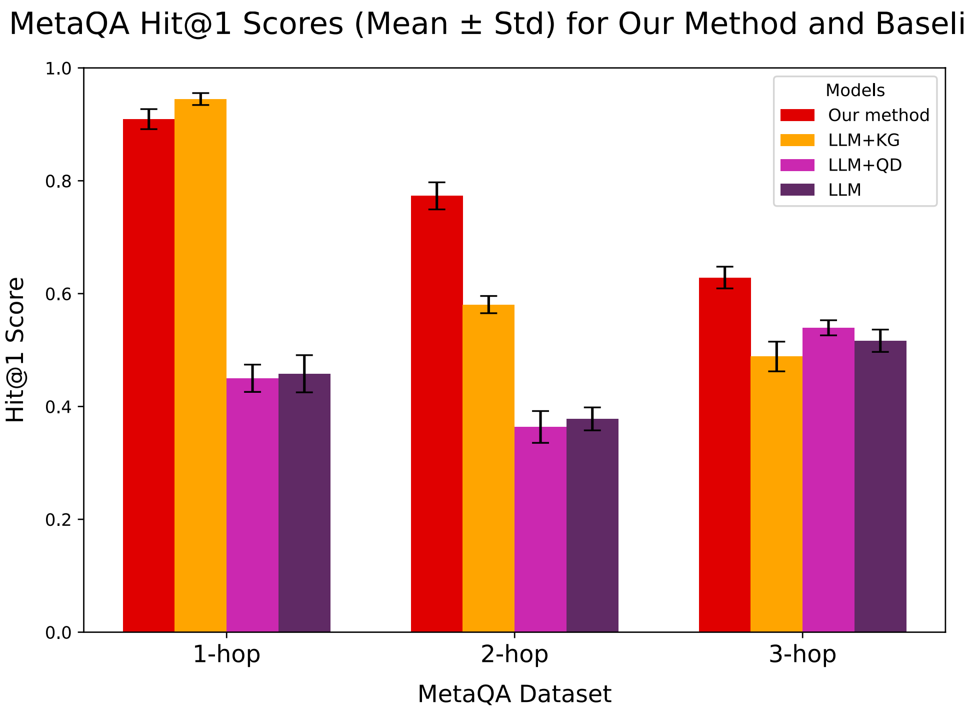


Fig. 5 MetaQA performance results for experiment 2, over 8 samples of 500 questions for each of the three datasets. The bars show the mean Hit@1, and the error bars show the standard deviation. The results for both the system and the baselines are shown

in 2-hop performance (0.787 ± 0.046). Consequently, the optimal parameter configuration for MetaQA is $N = 3$ and $K = 30$.

4.3.2 Experiment 2: quantitative analysis

The goal of our Experiment 2 is to assess how individual components influence the overall performance of the proposed system. To verify that, we assess a zero-shot prompt LLM, an LLM with a question decomposition component (LLM+QD), an LLM with a knowledge graph (LLM+KG; an equivalent approach to KAPING), and our approach. Figure 5 presents the performance results for Experiment 2 across 8 samples of 500 questions per MetaQA dataset, for 1-hop, 2-hop and 3-hop. First of all, we note that the zero-shot prompt LLM performs similarly to LLM+QD, indicating there is no clear gain from having the QD component alone. There is a visible advantage of adding a knowledge graph component on 1-hop and 2-hop cases. However, there is no difference between LLM+KG and the other two baselines on 3-hop. Interestingly, adding the QD component to LLM+KG, which is our method, allows to achieve a significant improvement of the performance in all three cases, and outperforming LLM+KG on 2-hop and 3-hop.

Comparing the baselines, the advantage of the KG retrieval module is most pronounced for 1-hop questions, but diminishes for 2-hop questions and disappears for 3-hop questions—likely because complex queries increase the difficulty of retrieving relevant triples. The integration of question decomposition in our system, however, maintains the benefits of KG retrieval for multi-hop questions while also enhancing answer explainability.

Table 4 The datasets that were analyzed for the qualitative analysis

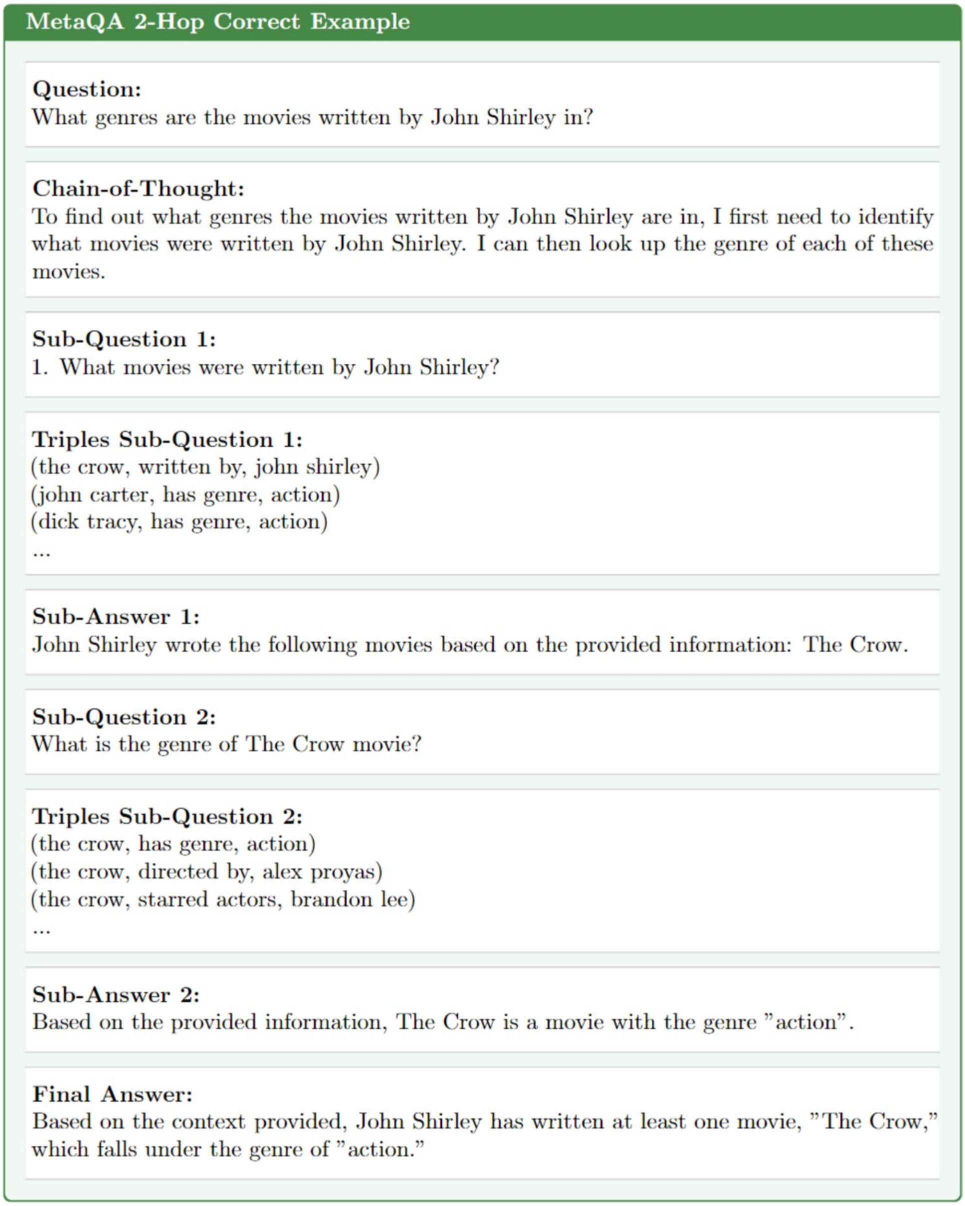
Dataset	Model	Experiment	Seed	Parameters
MetaQA 1-hop	KG-RAG	1	0	N=3, K=30
MetaQA 2-hop	KG-RAG	1	0	N=3, K=30
MetaQA 3-hop	KG-RAG	1	0	N=3, K=30

In summary, our system achieves improved performance on multi-hop questions with only a minor loss for 1-hop queries compared to the LLM+KG baseline. Although the relative and absolute advantage decreases as the number of hops increases, these quantitative results, combined with a forthcoming qualitative analysis (Section 4.4), support the effectiveness of our approach.

4.4 Qualitative analysis

This section examines the model outputs to identify recurring behaviors, strengths, and weaknesses, and to suggest directions for future improvements. Given the inherent limitations of a small, quantized LLM, our focus is on common patterns rather than isolated errors.

Table 4 lists the generated outputs used in this analysis. First, we present an example output from the MetaQA 2-hop dataset (Fig. 6). This example demonstrates how the system correctly decomposes a question into coherent subquestions, substitutes sub-answers appropriately, and synthesizes a final answer that accurately addresses the main question while maintaining a logical chain-of-thought.



MetaQA 2-Hop Correct Example

Question:
What genres are the movies written by John Shirley in?

Chain-of-Thought:
To find out what genres the movies written by John Shirley are in, I first need to identify what movies were written by John Shirley. I can then look up the genre of each of these movies.

Sub-Question 1:
1. What movies were written by John Shirley?

Triples Sub-Question 1:
(the crow, written by, john shirley)
(john carter, has genre, action)
(dick tracy, has genre, action)
...

Sub-Answer 1:
John Shirley wrote the following movies based on the provided information: The Crow.

Sub-Question 2:
What is the genre of The Crow movie?

Triples Sub-Question 2:
(the crow, has genre, action)
(the crow, directed by, alex proyas)
(the crow, starred actors, brandon lee)
...

Sub-Answer 2:
Based on the provided information, The Crow is a movie with the genre "action".

Final Answer:
Based on the context provided, John Shirley has written at least one movie, "The Crow," which falls under the genre of "action."

Fig. 6 An example of the system's intermediate outputs, which lead to the final answer. The example was taken from the MetaQA 2-hop sample that was analyzed for the qualitative analysis

4.4.1 A qualitative comparison to the LLM baseline

To further verify the advantage of adding a QD component and a KG component in our approach, we compare two queries for which the difference is most profound. In Table 5, we indicate that by introducing subquestions, our method is capable of providing correct answer while a baseline LLM tends to present generic answers. There are a few of such cases in the considered dataset. Nevertheless, they clearly highlight the importance of combining LLM+QD and LLM+KG in comparison to the vanilla LLM.

4.4.2 Question decomposition

By analyzing the distribution of the number of generated sub-questions per dataset (Fig. 7), we observe that the model generally recognizes the appropriate complexity of MetaQA multi-hop questions. For 1-hop questions, the model typically avoids decomposition, though ambiguous queries (e.g. asking for a movie description) sometimes lead to unnecessary sub-questions. For 2-hop and 3-hop questions, the model usually generates the expected number of sub-questions, although there are occasional cases of under-decomposition.

Table 5 Two examples of incorrect answers provided by an LLM and correct answers, together with proper subquestions, generated by our method

No.	Question	Answer	Model	Subquestions	Answer
1	Which person directed the films acted by Gloria DeHaven?	Jack Donohue, Charles Walters, Roy Del Ruth, Peter Yates	LLM	N/A	Gloria DeHaven was an actress who appeared in numerous films between the 1940s and 1960s. She worked with various directors throughout her career. Some of the films she starred in include "Thousands Cheer" directed by Mack Sandrich, "Anchors Aweigh" directed by George Sidney, "Neptune's Daughter" directed by Edward Buzzell, and "It Started with a Kiss" directed by George Cukor. It's important to note that she worked with many other directors as well.
			Ours	1. What films did Gloria DeHaven act in? 2. Who was the director of each of the following films: summer holiday, summer stock, the yellow cab man, Broadway rhythm, irma la douce, dedée d'anvers, miranda, b.f.'s daughter, and that lady in ermine (different from the first instance)?	Peter Yates, Charles Walters, and Jack Donohue are the directors of the films "Summer Holiday", "Summer Stock", and "The Yellow Cab Man", respectively, that Gloria DeHaven acted in.
2	Who appeared in the same movie with Essence Atkins?	LL Cool J, Gabrielle Union, Duane Martin	LLM	N/A	Essence Atkins has appeared in various movies with many actors and actresses. Some notable ones include: * Boris Kudjoe in "Love & Basketball" (2001) * Morris Chestnut in "The Game Plan" (2007) * Terry Crews in "Lottery Ticket" (2010) * Dwayne "The Rock" Johnson in "Be Cool" (2005) * Reginald Hudlin in "Daddy Day Camp" (2007). These are just a few examples. If you have a specific movie in mind, feel free to ask!
			Ours	1. What are the movies that Essence Atkins has acted in? 2. Who were the co-stars of Essence Atkins in the movies "Deliver Us from Eva" and "Intermission"?	Based on the context provided, Essence Atkins acted in the movie "Deliver Us from Eva" with Duane Martin and Gabrielle Union. Therefore, those three actors appeared in the same movie with Essence Atkins.

4.4.3 Qualitative performance

Overall, the system effectively distinguishes question complexity, but several systematic errors were identified:

- **Over-decomposition:** In approximately 16% of 1-hop cases, ambiguous questions lead to extra sub-questions, resulting in longer, sometimes overcomplicated answers.
- **Under-decomposition:** For 2-hop and 3-hop datasets, the system occasionally fails to generate enough sub-questions, sometimes producing only a 1-hop and a 2-hop question instead of the full decomposition.
- **Sub-answer Inconsistencies:** The LLM sometimes produces sub-answers that do not align with the provided triples, either by overlooking relevant data or by incorporating its own external knowledge.
- **Final Answer Synthesis:** While the final synthesis step generally succeeds, it occasionally yields overly long answers that may exceed token limits or include unwarranted information.

Despite these issues, the generated reasoning chains remain logical and coherent, allowing users to trace and verify the main answer. Many of the observed errors can be attributed to the limitations of the quantized LLM, and it is expected that a more sophisticated model or refined prompting strategies (potentially using ICL) could mitigate these problems.

Overall, there are only a few cases for which our approach misspelled names, or it did not provide a name in the answer. In other words, the Hit@1 metric cannot be necessarily at fault here. We present such examples in Table 6.

In conclusion, while triple selection remains robust when question decomposition is successful, the identified issues in decomposition, sub-answer generation, and answer synthesis indicate clear avenues for future research and improvements.

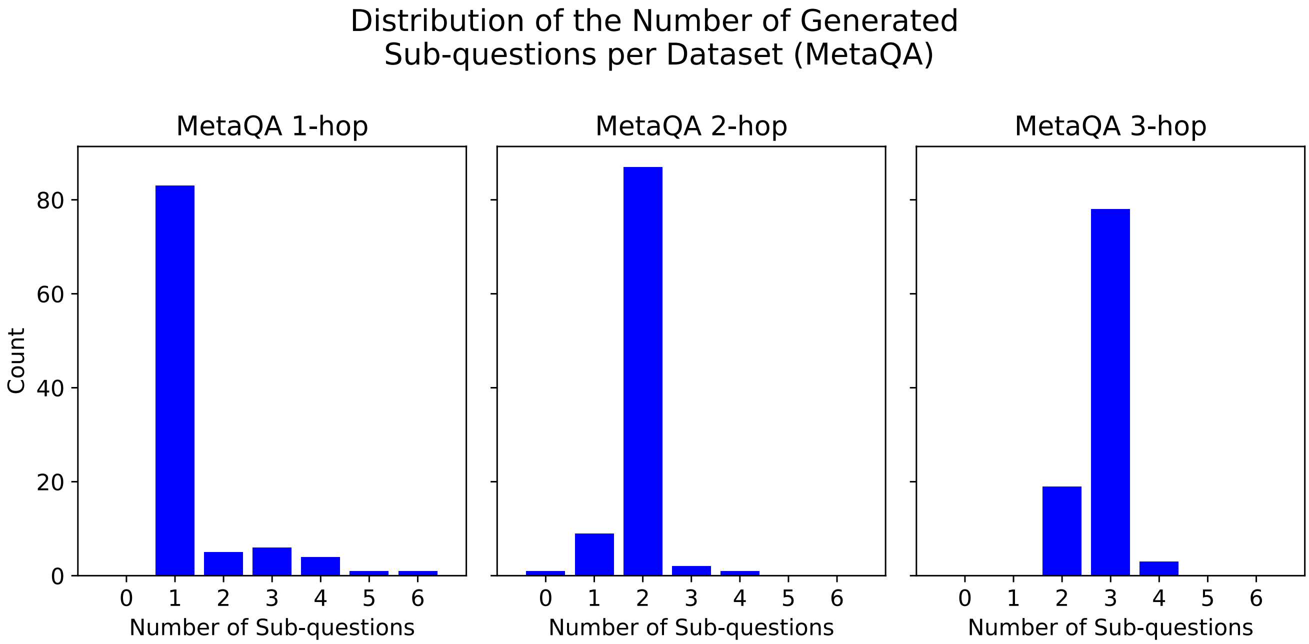


Fig. 7 The distribution of the number of sub-questions that were generated, for each of the MetaQA samples that was analyzed for the qualitative analysis (see Table 4 for more details)

4.5 Discussion: limitations

Here, we outline key limitations of the carried out research, which subsequently allow us to formulate future work.

First, constrained computational resources forced the use of a quantized, relatively small LLM, significantly impacting absolute performance—despite potentially preserving relative improvements over baselines. These constraints also necessitated random sampling of test subsets rather than evaluating on full datasets.

Second, the MetaQA benchmark is relatively simple, with a narrow domain and exclusively multi-hop questions. As noted in Section 4.3.1, some 3-hop questions are answerable using only 1-hop triples, which may skew performance evaluations compared to more complex benchmarks.

In summary, key limitations include the restricted LLM size, the simplicity and flaws of the MetaQA benchmark, and the inadequacy of the Hit@1 metric for modern KGQA systems.

5 Conclusion

5.1 Contributions

Our study addressed two primary research questions. First, we investigated enhancing LLMs with knowledge graphs (KGs) without requiring any training. By leveraging the synergy between LLM reasoning and the structured knowledge in KGs, we identified a gap in creating KGQA models that are both generalizable (as in KAPING) and explainable (as in Keqing). To bridge this gap, we developed an improved, training-free version of KAPING.

Second, we explored methods to improve answer explainability using a KG-RAG system. Inspired by Keqing [21] and the work of [46], we designed a question decomposition module that first generates a chain-of-thought (CoT) followed by coherent sub-questions. This approach not only improved performance on multi-hop questions but also provided transparent reasoning chains, thereby enhancing answer explainability. Overall, the proposed solution achieved higher answer accuracy (as measured by the Hit@1 metric) and improved transparency, though further validation is needed to confirm its generalizability across different domains, KGs, and question types.

5.2 Future work

Future research should focus on deepening the investigation into application generalizability by employing benchmarks with KGs composed largely of natural language, ensuring the triple selection mechanism via text embeddings functions effectively. Given some limitations of the MetaQA benchmark, exploring alternative benchmarks with diverse question domains may yield more robust conclusions. However, additional experiments on other datasets, such as, WebQuestions [41], WebQSP [30], ComplexWebQuestions [42], Mintaka [29], LC-QuAD [43], and ZJQA [22], would be beneficial. Our initial attempt on the Mintaka benchmark revealed that there is a strong discrepancy between

Table 6 Examples of questions for which the given answer do not contain expected real answer or names are misspelled (in bold)

Question	Real Answer	Given Answer
Can you give a few words describing Just My Luck?	Lindsay Lohan or ”Kiss” or Chris Pine	Based on the context provided, ”Just My Luck” is a comedy film as indicated by the genre label associated with it in the sub-questions. The movie features a comedic tone and explores themes common to comedy films. The cast includes actors known for their roles in comedies, further supporting this characterization. However, without additional information, it is impossible to identify specific themes or genres beyond the genre label of comedy.
Describe Dorian Gray in a few words?	Colin Firth or Ben Barnes	Dorian Gray is a character from Oscar Wilde’s novel ”The Picture of Dorian Gray”. He is known for his youthful beauty and obsession with preserving his appearance. Dorian makes a Faustian deal, leading to his portrait aging instead of himself, enabling him to live a hedonistic lifestyle without facing the consequences. The character is typically described as handsome and is known for his increasing moral corruption over time.
Who wrote the screenplay for Frankenstein’s Army?	Richard Raaphorst	The screenplay for ”Frankenstein’s Army” was written and directed by **Richard Raaporst**.
Who stars in Max Keeble’s Big Move?	Alex D. Linz	**Alex D. Linden** is the actor who stars in the film ”Max Keeble’s Big Move”.

the Hit@1 score and qualitative analysis. The reason for that is due to how the system generated sub-answers and final answers to specific question types in that benchmark, which was not the case for MetaQA. For example, a question might be comparative, e.g., a question like *“Who is taller, A or B?”*. Since our system generates a full main answer rather than just mentioning the answer entity, the Hit@1 score might incorrectly attribute points where none are due. In other words, even though further testing with other KGQA benchmarks is desirable to confirm that the system architecture is generalizable across domains, KGs and question types, this would require better evaluation metrics and, potentially, modifications to the system. There is a common agreement in the literature that Hit@1 is not appropriate metric for tasks for assessing modern KGQA systems [20–22]. Recent research on automated evaluation of natural language outputs [50] may offer promising alternatives. We leave investigating this research direction for future.

Furthermore, the low hanging fruit is the use of a more sophisticated LLM, as the small, quantized LLM was presumably the cause of a variety of systematic errors in the different system components. Interestingly, recently proposed reasoning models [51–53] has proven their usefulness in multiple specific QA tasks like in biology [54] and medicine [55], and they seem to be more faithful [56]. As a result, it seems replacing the LLM used in this paper with recently trained reasoning LLMs may further improve question answering and also increase the performance of our approach. Additionally, LLM inference parameters could be explored further, although this is expected to be more of a nuance; the selection of a LLM that is up to the wide variety of tasks for a successful system, e.g., a reasoning LLM, is expected to be the most influential for better overall performance.

Additionally, the design of a more advanced triple retrieval mechanism might be attempted, although it will be challenging to do this in a manner that does not require training and that might be generalized over different benchmarks. Still, a more sophisticated way of retrieving candidate triples or selecting from candidate triples could improve performance and efficiency. For example, for KGs with a relatively low number of relationship types, a suitable LLM could be asked to select a select number of the most relevant relationship types, which could then be used for candidate triple retrieval. This could avoid the retrieval of irrelevant triples, thus possibly leading to enhanced performance and efficiency.

For real-world applications, it is recommended to investigate finetuning techniques, since these can considerably increase performance [21, 22]. For example, if both question and answer entities are known, all paths between them could be identified, the most relevant ones could be selected, and these could then be used to finetune the text embedding model for the triple selection mechanism. This would likely not be very computationally intensive, since the text embedding model that was used is relatively small. Alternatively, another text embedding model could be used, or inspiration could be taken from the related work in Section 2.2 regarding finetuning techniques.

Author Contributions **Conceptualization**: Jakub M. Tomczak, Jasper Linders; **Methodology**: Jasper Linders, Jakub M. Tomczak; **Formal analysis and investigation**: Jasper Linders, Jakub M. Tomczak; **Implementation**: Jasper Linders; **Writing - original draft preparation**: Jasper Linders; **Writing - review and editing**: Jasper Linders, Jakub M. Tomczak; **Supervision**: Jakub M. Tomczak.

Data Availability and Access The dataset MetaQA is freely available at https://github.com/yuyuz/MetaQA (Last accessed: April 14, 2025).

Declarations

Competing Interests The authors have no competing interests to declare that are relevant to the content of this article.

Ethical and Informed Consent for Data Used The data used in this study does not require any ethical approval and can be freely used under the Creative Commons 3.0 license.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. 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. arXiv preprint arXiv:2007.13069
2. Kwiatkowski T, Palomaki J, Redfield O, Collins M, Parikh A, Alberti C, Epstein D, Polosukhin I, Devlin J, Lee K et al (2019) Natural questions: a benchmark for question answering research. Trans Assoc Comput Linguis 7:453–466
3. Garcia N, Otani M, Chu C, Nakashima Y (2020) KnowIT VQA: Answering knowledge-based questions about videos. Proc AAAI Conf Artif Intell 34:10826–10834
4. Wu P, Zhang X, Feng Z (2019) A survey of question answering over knowledge base. In: Knowledge graph and semantic computing: knowledge computing and language understanding: 4th China conference, CCKS 2019, Hangzhou, China, August 24–27, 2019, revised selected papers 4, pp 86–97. Springer
5. Wu T, Garcia N, Otani M, Chu C, Nakashima Y, Takemura H (2021) Transferring domain-agnostic knowledge in video question answering. In: Proceedings of british machine vision conference (BMVC)
6. Esposito M, Damiano E, Minutolo A, De Pietro G, Fujita H (2020) Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering. Inf Sci 514:88–105
7. Zhang J, Huang B, Fujita H, Zeng G, Liu J (2023) Feqa: Fusion and enhancement of multi-source knowledge on question answering. Expert Syst Appl 227:120286
8. Brown TB, 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 DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D (2020) Language models are few-shot learners. Advances in Neural Information Processing Systems 2020-Decem
9. Ji Z, Lee N, Frieske R, Yu T, Su D, Xu Y, Ishii E, Bang YJ, Madotto A, Fung P (2023) Survey of Hallucination in Natural Language Generation. ACM Comput Surv 55(12). https://doi.org/10.1145/3571730
10. Pan S, Luo L, Wang Y, Chen C, Wang J, Wu X (2024) Unifying large language models and knowledge graphs: A roadmap. IEEE Trans Knowl Data Eng 36(7):3580–3599
11. Pan JZ, Razniewski S, Kalo J-C, Singhania S, Chen J, Dietze S, Jabeen H, Omeliyanenko J, Zhang W, Lissandrini M, Biswas R, Melo G, Bonifati A, Vakaj E, Dragoni M, Graux D (2023) Large Language Models and Knowledge Graphs: Opportunities and Challenges 000(111):1–30
12. Bender EM, Gebru T, McMillan-Major A, Shmitchell S (2021) On the dangers of stochastic parrots: Can language models be too big? FAccT 2021 - Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, pp 610–623. https://doi.org/10.1145/3442188.3445922
13. Yang L, Chen H, Li Z, Ding X, Wu X (2023) ChatGPT is not Enough: Enhancing Large Language Models with Knowledge Graphs for Fact-aware Language Modeling 14(8):1–20
14. Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, Küttler H, Lewis M, Yih WT, Rocktäschel T, Riedel S, Kiela D (2020) Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems 2020-Decem
15. Gao Y, Xiong Y, Gao X, Jia K, Pan J, Bi Y, Dai Y, Sun J, Wang M, Wang H (2023) Retrieval-Augmented Generation for Large Language Models: A Survey
16. Hussien MM, Melo AN, Ballardini AL, Maldonado CS, Izquierdo R, Sotelo MA (2025) Rag-based explainable prediction of road users behaviors for automated driving using knowledge graphs and large language models. Expert Syst Appl 265:125914
17. Matsumoto N, Moran J, Choi H, Hernandez ME, Venkatesan M, Wang P, Moore JH (2024) Kragen: a knowledge graph-enhanced rag framework for biomedical problem solving using large language models. Bioinformatics 40(6):353
18. Xu Z, Cruz MJ, Guevara M, Wang T, Deshpande M, Wang X, Li Z (2024) Retrieval-augmented generation with knowledge graphs for customer service question answering. In: Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval, pp 2905–2909
19. Zhao X, Liu S, Yang S-Y, Miao C (2025) Medrag: Enhancing retrieval-augmented generation with knowledge graph-elicited reasoning for healthcare copilot. Proceedings of the ACM on web conference 2025:4442–4457
20. Baek J, Aji A.F, Saffari A (2023) Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering. Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp 70–98. https://doi.org/10.18653/v1/2023.nlrse-1.7
21. Wang C, Xu Y, Peng Z, Zhang C, Chen B, Wang X, Feng L, An B (2023) keqing: knowledge-based question answering is a nature chain-of-thought mentor of LLM
22. Wu Y, Hu N, Bi S, Qi G, Ren J, Xie A, Song W (2023) Retrieve-Rewrite-Answer: A KG-to-Text Enhanced LLMs Framework for Knowledge Graph Question Answering
23. Arenas M, Perez J (2013) Querying Semantic Web Data with SPARQL. ACM
24. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A.N, Kaiser L, Polosukhin I (2017) Attention is all you need. Adv Neural Inf Process Syst 30
25. Kaplan J, McCandlish S, Henighan T, Brown TB, Chess B, Child R, Gray S, Radford A, Wu J, Amodei D (2020) Scaling Laws for Neural Language Models
26. Zhao WX, Zhou K, Li J, Tang T, Wang X, Hou Y, Min Y, Zhang B, Zhang J, Dong Z, Du Y, Yang C, Chen Y, Chen Z, Jiang J, Ren R, Li Y, Tang X, Liu Z, Liu P, Nie J-Y, Wen J-R (2023) A Survey of Large Language Models, 1–97
27. Chen J, Chen L, Zhu C, Zhou T (2023) How many demonstrations do you need for in-context learning? Findings of the association for computational linguistics: EMNLP 2023:11149–11159

28. Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E, Le Q, Zhou D (2022) Chain-of-Thought Prompting Elicits Reasoning in Large Language Models
29. Sen P, Aji AF, Saffari A (2022) Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering. Proceedings - International Conference on Computational Linguistics, COLING 29(1):1604–1619
30. Yih WT, Richardson M, Meek C, Chang MW, Suh J (2016) The value of semantic parse labeling for knowledge base question answering. 54th Annual meeting of the association for computational linguistics, ACL 2016 - short papers, pp 201–206. https://doi.org/10.18653/v1/p16-2033
31. Zhang Y, Dai H, Kozareva Z, Smola A, Song L (2018) Variational reasoning for question answering with knowledge graph. In: Proceedings of the AAAI conference on artificial intelligence, vol 32
32. Oliya A, Saffari A, Sen P, Ayoola T (2021) End-to-End Entity Resolution and Question Answering Using Differentiable Knowledge Graphs. In: EMNLP (1)
33. Sen P, Mavadia S, Saffari A (2023) Knowledge Graph-augmented Language Models for Complex Question Answering. Proceedings of the annual meeting of the association for computational linguistics, pp 1–8. https://doi.org/10.18653/v1/2023.nlrse-1.1
34. Gu Y, Pahuja V, Cheng G, Su Y (2022) Knowledge Base Question Answering: A Semantic Parsing Perspective
35. Sanh V, Webson A, Raffel C, Bach SH, Sutawika L, Alyafeai Z, Chaffin A, Stiegler A, Le Scao T, Raja A, Dey M, Bari MS, Xu C, Thakker U, Sharma S, Szczechla E, Kim T, Chhablani G, Nayak NV, Datta D, Chang J, Jiang MTJ, Wang H, Manica M, Shen S, Yong ZX, Pandey H, McKenna M, Bawden R, Wang T, Neeraj T, Rozen J, Sharma A, Santilli A, Fevry T, Fries JA, Teehan R, Bers T, Biderman S, Gao L, Wolf T, Rush AM (2022) Multitask Prompted Training Enables Zero-Shot Task Generalization. ICLR 2022 - 10th International Conference on Learning Representations
36. Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ (2020) Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res 21:1–67
37. Chung HW, Hou L, Longpre S, Zoph B, Tay Y, Fedus W, Li Y, Wang X, Dehghani M, Brahma S, Webson A, Gu SS, Dai Z, Suzgun M, Chen X, Chowdhery A, Castro-Ros A, Pellat M, Robinson K, Valter D, Narang S, Mishra G, Yu A, Zhao V, Huang Y, Dai A, Yu H, Petrov S, Chi EH, Dean J, Devlin J, Roberts A, Zhou D, Le QV, Wei J (2022) Scaling Instruction-Finetuned Language Models, 1–54
38. Zhang S, Roller S, Goyal N, Artetxe M, Chen M, Chen S, Dewan C, Diab M, Li X, Lin XV, Mihaylov T, Ott M, Shleifer S, Shuster K, Simig D, Koura PS, Sridhar A, Wang T, Zettlemoyer L (2022) OPT: Open Pre-trained Transformer Language Models
39. Fitzgerald J, Ananthakrishnan S, Arkoudas K, Bernardi D, Bhagia A, Delli Bovi C, Cao J, Chada R, Chauhan A, Chen L, Dwarakanath A, Dwivedi S, Gojayev T, Gopalakrishnan K, Gueudre T, Hakkani-Tur D, Hamza W, Hueser JJ, Jose KM, Khan H, Liu B, Lu J, Manzotti A, Natarajan P, Owczarzak K, Oz G, Palumbo E, Peris C, Prakash CS, Rawls S, Rosenbaum A, Shenoy A, Soltan S, Sridhar MH, Tan L, Triefenbach F, Wei P, Yu H, Zheng S, Tur G, Natarajan P (2022) Alexa Teacher Model: Pretraining and Distilling Multi-Billion-Parameter Encoders for Natural Language Understanding Systems. Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, pp 2893–2902. https://doi.org/10.1145/3534678.3539173
40. Touvron H, Martin L, Stone K, Albert P, Almahairi A, Babaei Y, Bashlykov N, Batra S, Bhargava P, Bhosale S, Bikel D, Blecher L, Ferrer CC, Chen M, Cucurull G, Esiobu D, Fernandes J, Fu J, Fu W, Fuller B, Gao C, Goswami V, Goyal N, Hartshorn A, Hosseini S, Hou R, Inan H, Kardas M, Kerkez V, Khabsa M, Kloumann I, Korenev A, Koura PS, Lachaux M-A, Lavril T, Lee J, Liskovich D, Lu Y, Mao Y, Martinet X, Mihaylov T, Mishra P, Molybog I, Nie Y, Poulton A, Reizenstein J, Rungta R, Saladi K, Schelten A, Silva R, Smith EM, Subramanian R, Tan XE, Tang B, Taylor R, Williams A, Kuan JX, Xu P, Yan Z, Zarov I, Zhang Y, Fan A, Kambadur M, Narang S, Rodriguez A, Stojnic R, Edunov S, Scialom T (2023) Llama 2: Open Foundation and Fine-Tuned Chat Models
41. Berant J (2013) Semantic Parsing on Freebase from Question-Answer Pairs (October), 1533–1544
42. Talmor A, Berant J (2013) The Web as a Knowledge-base for Answering Complex Questions
43. Dubey M, Banerjee D, Abdelkawi A, Lehmann J (2019) Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In: The semantic web–ISWC 2019: 18th international semantic web conference, Auckland, New Zealand, October 26–30, 2019, proceedings, part II 18, pp 69–78. Springer
44. Pedersen T, Patwardhan S, Michelizzi J et al (2004) WordNet: Similarity-Measuring the Relatedness of Concepts. In: AAAI, vol 4, pp 25–29
45. Hu EJ, Shen Y, Wallis P, Allen-Zhu Z, Li Y, Wang S, Wang L, Chen W (2021) LoRA: Low-Rank Adaptation of Large Language Models
46. Radhakrishnan A, Nguyen K, Chen A, Chen C, Denison C, Hernandez D, Durmus E, Hubinger E, Kernion J, Lukošiūtė K, Cheng N, Joseph N, Schiefer N, Rausch O, McCandlish S, Showk SE, Lanham T, Maxwell T, Chandrasekaran V, Hatfield-Dodds Z, Kaplan J, Brauner J, Bowman SR, Perez E (2023) Question Decomposition Improves the Faithfulness of Model-Generated Reasoning
47. Jiang AQ, Sablayrolles A, Mensch A, Bamford C, Chaplot DS, Casas Ddl, Bressand F, Lengyel G, Lample G, Saulnier L, Lavaud LR, Lachaux M-A, Stock P, Scao TL, Lavril T, Wang T, Lacroix T, Sayed WE (2023) Mistral 7B
48. Nguyen M, Baker A, Neo C, Roush A, Kirsch A, Shwartz-Ziv R (2024) Turning Up the Heat: Min-p Sampling for Creative and Coherent LLM Outputs
49. Steinmetz N, Sattler KU (2021) What is in the KGQA Benchmark Datasets? Survey on Challenges in Datasets for Question Answering on Knowledge Graphs. J Data Semantics 10(3–4):241–265. https://doi.org/10.1007/s13740-021-00128-9
50. Guo Z, Jin R, Liu C, Huang Y, Shi D, Supryadi, Yu L, Liu Y, Li J, Xiong B, Xiong D (2023) Evaluating Large Language Models: A Comprehensive Survey
51. Chen Y, Benton J, Radhakrishnan A, Uesato J, Denison C, Schulman J, Somani A, Hase P, Wagner M, Roger F et al (2025) Reasoning models don't always say what they think. arXiv preprint arXiv:2505.05410
52. Guo D, Yang D, Zhang H, Song J, Zhang R, Xu R, Zhu Q, Ma S, Wang P, Bi X et al (2025) Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948
53. OpenAI (2024) Learning to reason with LLMs. OpenAI Blog Post. https://openai.com/index/learning-to-reason-with-llms/
54. Shao L, Yu H, Huang W, Zhao H, Zhang L, Song J (2025) Deepseek-based multi-dimensional augmentation of short and highly domain-specific textual inquires for aquaculture question-answering framework. Aquacult Int 33(4):279
55. Luo P-W, Liu J-W, Xie X, Jiang J-W, Huo X-Y, Chen Z-L, Huang Z-C, Jiang S-Q, Li M-Q (2025) DeepSeek vs ChatGPT: a comparison study of their performance in answering prostate cancer radiotherapy questions in multiple languages. Am J Clin Exp Urol 13(2):176

56. Chua J, Evans O (2025) Are DeepSeek R1 And Other Reasoning Models More Faithful? In: ICLR 2025 workshop on foundation models in the wild

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Jasper Linders is a graduate of the MSc Data Science & Artificial Intelligenceprogram at TU Eindhoven, with a background in Mechanical Engineering. He has broad experience indata, analytics, and intelligent systems, ranging from building dashboards and data pipelines todeveloping AI-powered tools and research-driven solutions. Skilled in SQL, statistics, and big data,Jasper focuses on transforming complex data into clear insights and scalable systems. Beyond hisacademic and technical pursuits, he has worked at IKEA and Albert Heijn, gaining hands-on experiencein customer-oriented roles.

Jakub M. Tomczak is a leading researcher and practitioner in GenerativeAI with over 15 years of experience in machine learning and deep learning. He has led 50+ academicand 10+ industrial projects spanning computer vision, large language models, foundation models, andagentic AI. His work includes more than 45 peer-reviewed papers at top venues (e.g., NeurIPS, ICML,ICLR, CVPR, ICCV), two patents, and a landmark book Deep Generative Modeling.Jakub has raisedover €2.3M in research funding and held leadership roles in both academia and industry, includingpositions at Qualcomm, eBay, and multiple startups. He served as Program Chair of NeurIPS 2024 andis the founder of Amsterdam AI Solutions.