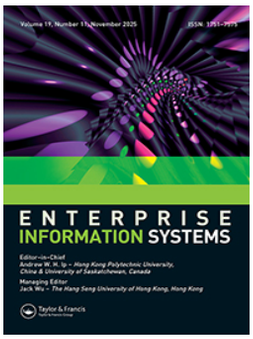


Enterprise Information Systems

ISSN: 1751-7575 (Print) 1751-7583 (Online) Journal homepage: www.tandfonline.com/journals/teis20

Hierarchical knowledge graph-based QA systems with retrieval-augmented generation

Wan-Chi Yang, Xuan Li, Chih-Yung Chang & Diptendu Sinha Roy

To cite this article: Wan-Chi Yang, Xuan Li, Chih-Yung Chang & Diptendu Sinha Roy (19 Nov 2025): Hierarchical knowledge graph-based QA systems with retrieval-augmented generation, Enterprise Information Systems, DOI: 10.1080/17517575.2025.2580477

To link to this article: https://doi.org/10.1080/17517575.2025.2580477

ENTERPRISE INFORMATION SYSTEMS
https://doi.org/10.1080/17517575.2025.2580477

Taylor & Francis
Taylor & Francis Group

RESEARCH ARTICLE

Hierarchical knowledge graph-based QA systems with retrieval-augmented generation

Wan-Chi Yang[a], Xuan Li[b], Chih-Yung Chang[c] and Diptendu Sinha Roy[d]

[a]General Education Center, National Taipei University of Nursing and Health Sciences, Taipei, Taiwan; [b]Frontier Interdisciplinary Science Research Center, Purple Mountain Laboratories, Nanjing, China; [c]Department of Computer Science and Information Engineering, Tamkang University, Taipei, Taiwan; [d]Department of Computer Science and Engineering, National Institute of Technology, Shillong, India

ABSTRACT

Hierarchical knowledge graphs (KGs) are vital to question-answering (QA) systems for complex queries, integrating structured and unstructured knowledge. This study introduces a QA system combining a hierarchical KG, graph convolutional networks (GCNs), and retrieval-augmented generation (RAG) to enhance reasoning, retrieval, and response generation. The KG organises information into title, subtitle, and content layers for structured, efficient retrieval; GCNs aggregate local and global relations across layers; RAG incorporates external sources (e.g., Wikipedia) for contextually accurate answers. On standard benchmarks, the system outperformed strong baselines in precision, recall, and F1-score, offering an effective solution for complex queries and advancing QA design.

ARTICLE HISTORY
Received 3 February 2025
Accepted 21 October 2025

KEYWORDS
Information and knowledge management models; hierarchical knowledge graphs; question-answering systems; retrieval-augmented generation

1. Introduction

In recent years, the development of Question-Answering (QA) systems has been instrumental in advancing accurate and efficient information retrieval across various domains. As QA systems evolve, there is a growing demand for handling complex and context-dependent queries that require deeper reasoning and external knowledge sources (Khushhal et al. 2020; Noraset, Lowphansirikul, and Tuarob 2021). Existing QA approaches generally fall into three categories: semantic-based QA, knowledge graph (KG)-based QA, and QA systems that integrate large language models (LLMs) with KGs (Khushhal et al. 2020; Sun et al. 2018; Zafar et al. 2024).

Semantic-based QA models often employ traditional retrieval methods, such as BM25 for lexical matching and representation-based techniques that leverage embeddings for semantic similarity (Ren et al. 2021). Although these methods improved retrieval accuracy, they often fail to provide a sufficient understanding of complex contextual relationships, particularly for queries requiring advanced reasoning or external knowledge (Devlin et al. 2019; Hui et al. 2017). In contrast, KG-based QA systems utilised structured knowledge bases to support multi-hop reasoning and address intricate queries. By constructing semantic graphs and encoding relationships, these systems can effectively represent entities and connections, enabling more sophisticated reasoning capabilities. However, KG-based models face challenges with scalability and integration of unstructured data, which limits their applicability in diverse areas (Guo et al. 2023; J. Zhang et al. 2020).

Recently, hybrid QA systems that integrated KGs with LLMs (KGs-LLMs) have shown potential in bridging structured and unstructured knowledge (Lai and Qiu 2025; Mohamed et al. 2023). These systems created dynamic interaction between language models and knowledge bases, which makes it possible to handle complex queries with deeper contextual understanding (M. Zhang, He, and Dong 2024). While advancements have improved domain-specific accuracy and multi-hop reasoning, challenges remained in achieving fine-grained query interpretation and fully resolving complex reasoning tasks.

To overcome these limitations, this paper proposes a novel QA mechanism that combines a hierarchical KG structure with deep learning techniques and Retrieval-Augmented Generation

CONTACT Chih-Yung Chang cychang@mail.tku.edu.tw Department of Computer Science and Information Engineering, Tamkang University, No. 151, Yingzhuan Rd., Tamsui Dist., New Taipei City, 251301, Taiwan

(RAG). This approach features a multi-layered KG that stores entities and relationships across three levels – titles, subtitles, and content nodes – providing a comprehensive organisation of knowledge. Through this structured representation, each vertex in the KG stores essential information and a vector representation to enable effective similarity-based retrieval. When a user query is issued, it is transformed into a vector and matched it with the KG to retrieve relevant content. The retrieved content is then processed by LLM using RAG to achieve a deeper understanding and deliver accurate responses. The contributions of this study include:

(1) Multi-layered KG structure: A hierarchical KG is proposed that organises information across title, subtitle, and content layers, to address limitations in previous studies that often rely on single-layer or flat structures (Guo et al. 2023; J. Zhang et al. 2020). The layered design improves scalability and ensures precise information retrieval and interpretation.
(2) GCN integration: In this paper, GCNs are employed to propagate and aggregate features within our multi-layered KG, capturing both local and global relationships across layers. Compared to other studies (Fang et al. 2023; Lv et al. 2020), which primarily focus on simpler graph structure or limited relational contexts, our approach leverages this hierarchical KG to enhance multi-hop reasoning and provide deeper contextual understanding across interconnected data layers.
(3) RAG for answer generation: Integrating RAG enables our system to effectively use KG-retrieved content, allowing the LLM to generate precise, contextually relevant answers for complex queries. Previous studies have explored retrieval-augmented methods, such as RALM for interpretability (Guu et al. 2020), RPT for handling long texts (Rubin and Berant 2023), and In-Context RALM for efficient retrieval (Ram et al. 2023). Our approach builds on these RAG methods and addresses common issues such as hallucination in LLMs.

The remainder of this paper is organised as follows. Section 2 reviews related work, categorising prior studies in semantic-based QA, KG-based QA, and QA with KG-LLM integration. Section 3 presents the notations, assumptions, problem descriptions and objective of this paper. Section 4 introduces our proposed method in detail, including KG construction, GCN training, and RAG integration. Section 5 presents experimental results and analysis to demonstrate the effectiveness of our approach. Finally, Section 6 concludes with a discussion on future research directions.

2. Related work

In the context of QA systems, the nature of data, whether structured or unstructured, plays a fundamental role in shaping how queries are processed and how answers are generated. Structured data typically refers to machine-readable formats with explicitly defined semantics, such as relational databases (Yu et al. 2021). Unstructured data, by contrast, encompasses human-expressed formats like natural language text, which lack inherent structure interpretable by machines (Yu et al. 2021). This classification provides a conceptual basis for understanding the progression of QA system architectures. Initial approaches primarily handled unstructured data through semantic-based retrieval techniques. As the limitations of such methods became evident, structured knowledge sources, particularly knowledge graphs, were introduced to enhance reasoning capabilities. More recently, hybrid models have emerged, integrating KGs with LLMs to combine structural accuracy with contextual fluency.

This chapter details the development of QA systems across three primary paradigms: semantic-based QA, KG-based QA, and QA systems that integrate KGs with LLMs. Each represents a step forward in improving answer precision, contextual interpretation, and reasoning depth.

2.1. Semantic-based QA

Early development in semantic-based QA systems primarily rely on two types of retrieval matching techniques: traditional probabilistic methods and representation-based embeddings. Here, semantics refers to the relationship between symbols and their meanings, as opposed to syntax, which concerns how symbols are

structured regardless of their meaning (ANSI/IEEE 1983). This distinction is key to semantic-based QA, which aims to go beyond surface-level matching and capture the deeper meaning of language.

Traditional methods, such as BM25, utilise probabilistic scoring to determine query-document relevance (Ren et al. 2021). It has also been effectively incorporated with other methods to improve QA systems. For instance, Noraset, Lowphansirikul, and Tuarob (2021) developed WabiQA, integrating BM25F for structure-sensitive retrieval in complex documents, while Khushhal et al. (2020) combined BM25 with index-based methods to improve community question retrieval. As QA requirements grew more complex, representation-based approaches emerged, leveraging embeddings to capture semantic relationships. These methods allowed for efficient similarity-based retrieval and a better understanding of contextual dependencies (Ren et al. 2021). Techniques such as PACRR (Hui et al. 2017) and BERT-based models (Devlin et al. 2019) encode complex contextual dependencies to improve retrieval precision. Despite these advances, semantic-based QA approaches faced challenges when dealing with highly intricate queries that required external or specialised knowledge sources. This limitation highlighted the need for structured knowledge sources, which led to the development of KG-based QA systems.

2.2. KG-based QA

KG-based QA models aimed to enhance reasoning by using structured knowledge bases. Knowledge is a representation of instructions in a formalised manner suitable for communication, interpretation, or processing by human or automatic means. It is closely related to information, which, according to W. Zhang (1994), can be understood as an interpreted or contextualised form of data. In this view, data encodes the information about what to be, while knowledge encodes the information about how to do (W. Zhang 1994). Knowledge graphs organise such representations into entities and relationships, enabling multi-hop reasoning and deeper understanding.

Sorokin and Gurevych (2018) demonstrated an early application of KG-based QA by using Wiki data to construct semantic graphs that represent entities, relationships, and constraints in query-specific subgraphs. This setup allowed for multi-hop reasoning by connecting query entities through a network of relationships, which was further improved using Gated Graph Neural Networks (GGNNs) to encode structural dependencies.

Similarly, Sun et al. (2018) extended this approach by combining Freebase with Wikipedia to incorporate both structured and unstructured data. Their model built subgraphs from Freebase and used supporting text from Wikipedia, providing additional context for complex queries. This integration of structured KG data with unstructured text significantly improved query interpretation and highlight the potential of mixed data sources in KG-based QA.

In the area of embeddings, foundational models such as TransE (Bordes et al. 2013) and its successor ComplEx (Saxena, Tripathi, and Talukdar 2020) further advanced KG-based QA. TransE used a simple translational approach, representing relationships as translations in the vector space, while ComplEx enhanced this by capturing asymmetric relations in complex vector spaces. Han, Wang, and Wang (2024) expanded on these methods by employing enhanced multi-relational embeddings to handle more intricate relational dependencies. To further improving reasoning with KG-based QA, Graph Neural Networks (GNNs), particularly Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs), have been introduced as mechanisms for aggregating information from neighbours and refine node representations (Fang et al. 2023). Lv et al. (2020) employed GCNs to capture neighbour information while Guo et al. (2023) improved upon this by using neighbour interaction networks, a variant of GATs, to assign weights based on neighbour relevance to the query. These advancements enabled KG-based QA systems to process layered and relational data more effectively, although challenges still existed in scaling and in effectively integrating unstructured data. These challenges resulted in hybrid approaches that combine the robustness of KGs with the flexibility of LLMs, leading to significant improvement in the capabilities of QA systems.

2.3. QA systems integrating KGs and LLMs

One major motivation for integrating KGs with LLMs is to address a limitation of LLM-based QA systems – hallucination (Hao, Yu, and You 2025; Simhi et al. 2025). Hallucination refers to the generation of content that

is nonsensical or unfaithful to the provided source content (Ji et al. 2023). It is generally categorised into intrinsic hallucinations, where the output contradicts the input, and extrinsic hallucinations, where the output cannot be verified from the input (Ji et al. 2023). These issues raise serious concerns in knowledge-intensive tasks such as QA, where factual accuracy is essential.

To mitigate hallucination and improve answer reliability, recent QA systems have increasingly adopted hybrid architectures that combine the structured accuracy of KGs with the linguistic fluency and contextual strength of LLMs (Lang, Liu, and Jia 2023). These systems use structured triples to ground answers, reduce errors, and improve reasoning.

The integration of KGs with LLMs represents a key advancement in this direction by leveraging the benefits of both structured and unstructured data to improve reasoning and accuracy. JointLK (Sun et al. 2022) exemplifies this by combining RoBERTa with a relational graph attention mechanism, enabling commonsense reasoning for complex QA tasks.

Additionally, KI-MAG (Knowledge-Infused Medical Abstractive Generator) (Zafar et al. 2024) uniquely incorporates relevant medical entities from a knowledge graph directly into the answer generation process. The model introduced Q2K (Question-to-Knowledge) and K2Q (Knowledge-to-Question) attention mechanisms to dynamically align KG information with the question context. Through the use of pre-trained language models, KI-MAG generated responses that are contextually accurate and tailored to medical domains. This approach ensures the precision and correctness, which is crucial in medical applications. Building on these advancements, M. Zhang, He, and Dong (2024) proposed a Meta-path Reasoning Graph Neural Network (MRGNN), an advanced model that captures complex KG relationships using meta-paths. Meta-paths allow MRGNN to address multi-hop reasoning more effectively than prior models and improve answer accuracy on complex questions.

Alongside these KG – LLM approaches, retrieval-augmented models have also been developed as another complementary line of research. A representative model is REALM (Guu et al. 2020), which incorporates a neural retriever into the pre-training of language models. In this framework, the retriever selects relevant passages from large-scale unstructured corpora such as Wikipedia, and both the retriever and the encoder are optimised together in an end-to-end manner. This design enables the language model to access external knowledge beyond its parameters and improves performance on open-domain QA benchmarks. Unlike REALM, the proposed system emphasises structured reasoning through a hierarchical KG, where title, subtitle, and content layers are integrated using GCN-based feature aggregation. An external RAG component further supplements internal knowledge with Wikipedia documents, supported by a domain-specific lexicon and validated fuzzy matching. This illustrates a different integration strategy, which complements retrieval-augmented approaches such as REALM and shows how these methods can be extended towards domain-specific QA settings.

This hybrid use of structured and unstructured sources is not unique to QA. In business process analysis and process mining, De Nicola et al. (2024) found that traditional NLP pipelines, which combine rule-based parsing, dependency analysis, and statistical matching, produce structured and interpretable outputs. In contrast, embedding-based and LLM-driven approaches can better handle domain-specific language but need mechanisms to maintain semantic rigour. Similarly, Van Woensel and Motie (2024) reported that in rule-based, machine learning, and deep learning methods for process extraction, there is a recurring trade-off between precise semantic parsing and flexibility in handling diverse, domain-specific process descriptions. These cross-domain observations reflect the same motivation behind integrating KGs with LLMs in QA: bringing together the reliability of structured knowledge and the expressive power of LLMs to ensure both accuracy and contextual depth.

In other inference domains, multi-source and multi-expert integration has also been explored. For example, Modi et al. (2011) proposed a socially inspired framework for human state inference that integrates physiological and contextual signals with multiple experts, including rule-based fuzzy systems, Bayesian networks, statistical models, and neural networks, through structured opinion fusion. Their study demonstrated that combining diverse experts improves performance in tasks such as classification and prediction, as illustrated in their fatigue detection experiments. While effective in its domain, the reliance on rule-based and expert-driven components constrains reasoning depth and limits generalisation across broader tasks. In contrast, our approach applies this integration concept to QA by leveraging hierarchical KGs, GCN-based reasoning, and internal and external RAG, where layered neural architectures capture knowledge

representations. This design enables richer reasoning and broader applicability beyond domain-specific classification problems.

While previous studies have made significant progress in areas such as context extraction, embedding methods, scalability, multi-hop reasoning, and external knowledge integration, certain issues persist. Traditional approaches often struggled with effectively extracting context, handling diverse embeddings, scaling with large datasets, supporting complex multi-hop reasoning, and integrating unstructured data or external sources seamlessly. LLM-based QA systems also face hallucination issues, where generated answers are unfaithful or unverifiable (Hao, Yu, and You 2025; Simhi et al. 2025).

To highlight how the proposed QA system addressed these challenges, Table 1 presents a comparison between the limitations observed in related work and the solutions introduced in our study.

3. Notations, assumption and problem descriptions

This section presents the notations, assumptions, problem descriptions and objective of this paper.

3.1. Notations and assumptions

In this study, data refers to a representation of facts, concepts, or instructions in a formalised manner suitable for communication, interpretation, or processing by human or automatic means (ANSI/IEEE 1983). This definition aligns with the structured nature of the input used to construct the hierarchical knowledge graph.

Let D^{web} represent the existing set of data collected from websites, which consists of multiple sentences. The dataset D^{web} can be further partitioned into $D^{web} = \left\{s_1^{web}, s_2^{web}, \ldots, s_{|D^{web}|}^{web}\right\}$, where $\left|D^{web}\right|$ denotes the total number of sentences in the dataset. Each sentence S_i^{web} is composed of multiple words, denoted as $S_i^{web} = \left\{w_{i,1}^{web}, w_{i,2}^{web}, \ldots, w_{i,\left|S_i^{web}\right|}^{web}\right\}$, where $w_{i,k}^{web}$ represents the k-th word of the i-th sentence.

Let D^{wiki} represent the existing set of data collected from Wiki sources, consisting of multiple sentences. The dataset D^{wiki} can be further partitioned into $D^{wiki} = \left\{s_1^{wiki}, s_2^{wiki}, \ldots, s_{|D^{wiki}|}^{wiki}\right\}$, where $\left|D^{wiki}\right|$ denotes the total number of sentences in the dataset. Each sentence s_1^{wiki} is composed of multiple words, denoted as $S_i^{wiki} = \left\{w_{i,1}^{wiki}, w_{i,2}^{wiki}, \ldots, w_{i,\left|S_i^{wiki}\right|}^{wiki}\right\}$, where $w_{i,j}^{wiki}$ represents the k-th word of the i-th sentence.

Assume a correct answer $A_i = \left\{s_{i,1}, s_{i,2}, \ldots, s_{i,|A_i|}\right\}$ is composed of multiple sentences. Each sentence $s_{i,j} = \left\{w_{i,j,1}, w_{i,j,2}, \ldots, w_{i,j,\left|s_{i,j}\right|}\right\}$ in answer A_i is composed of multiple words $w_{i,j,k}$, where $\left|s_{i,j}\right|$ is the number of words in the sentence $s_{i,j}$. Similarly, let $\hat{A}_i = \left\{\hat{s}_{i,1}, \hat{s}_{i,2}, \ldots, \hat{s}_{i,\left|\hat{A}_i\right|}\right\}$ denote a predicted answer, which is also

Table 1. Comparison of the proposed QA system with related work.

Aspect	Related Work Limitations	Proposed QA Study	Advantages
Context Extraction	Relies on Bag-of-Words, limited context understanding (Khushhal et al. 2020; Noraset, Lowphansirikul, and Tuarob 2021)	Multi-layered KG with RAG	Better context relevance and interpretation
Embedding Method	Homogeneous embeddings, limited flexibility (Hui et al. 2017; Ren et al. 2021)	Diverse KG embeddings with GCN	Captures local and global relationships, enriching feature representations
Scalability	Difficulty scaling single-layer or flat KGs with diverse data (Guo et al. 2023; J. Zhang et al. 2020)	Hierarchical KG for layered data	Enhanced scalability, effective handling of structured and unstructured data
Multi-hop Reasoning	Limited multi-hop reasoning, restricting complex queries (Sorokin and Gurevych 2018; Sun, Bedrax-Weiss, and Cohen 2019)	GCN integration in hierarchical KG	Improved reasoning across layers for complex, relational queries
External Knowledge Integration	Lacks seamless integration with external knowledge for complex queries (Ram et al. 2023)	RAG to include external knowledge when KG data is insufficient	More accurate and complete answers by adding relevant external information
Hallucination	Prone to generating unfaithful or unverifiable answers in QA tasks (Hao, Yu, and You 2025; Simhi et al. 2025)	Constrains LLM outputs to KG and Wiki content; uses multi-LLM verification	Enhances answer reliability through knowledge-grounded and cross-verified generation.

composed of multiple sentences. Each sentence $\hat{s}_{i,j} = \left\{\hat{w}_{i,j,1}, \hat{w}_{i,j,2}, \ldots, \hat{w}_{i,j,|\hat{s}_{i,j}|}\right\}$ in the predicted answer, is composed of multiple words $\hat{w}_{i,j,k,}$.

For model M, let Boolean variable $\delta^M_{i,j,k}$ represent whether or not the k-th word $w_{i,j,k}$ in the j-th sentence is in A_i. The value of $\delta^M_{i,j,k}$ can be derived from Equation 1

$$\delta^M_{i,j,k} = \begin{cases} 1 if w_{i,j,k} \in A_i \\ 0 if w_{i,j,k} \notin A_i \end{cases} \tag{1}$$

Furthermore, let Boolean variable $\rho^M_{i,j,k}$ represent whether or not the k-th word $w_{i,j,k}$ in the j-th sentence is in $\hat{A}_i$. The value of $\rho^M_{i,j,k}$ can be derived from Equation 2

$$\rho^M_{i,j,k} = \begin{cases} 1 if w_{i,j,k} \in \hat{A}_i \\ 0 if w_{i,j,k} \notin \hat{A}_i \end{cases} \tag{2}$$

3.2. Problem descriptions

Let C denote the confusion matrix for mechanism $\mathcal{M}$, which predicts a set of y answers $\hat{A}=\{\hat{A}_i|1 \le i \le y\}$ for a set of y questions $Q=\{Q_i|1 \le i \le y\}$. Each element $\mathcal{C}_{i,j,k}$ in the matrix C indicates whether the word $w_{j,k} \in D_j$ is correctly or incorrectly predicted as part of the predicted answer $\hat{A}_i$ to the question Q_i.

Let $TP^{\mathcal{M}}_{i,j,k}$ denote the true positives in $\mathcal{C}_{i,j,k}$. The value of $TP^{\mathcal{M}}_{i,j,k}$ can be calculated using Equation 3.

$$TP^{\mathcal{M}}_{i,j,k} = 1 if \delta^M_{i,j,k} \cdot \rho^M_{i,j,k} \tag{3}$$

Let $FP^{\mathcal{M}}_{i,j,k}$ and $FN^{\mathcal{M}}_{i,j,k}$ represent the false positive and false negative in $\mathcal{C}_{i,j,k}$ respectively. The values of $FP^{\mathcal{M}}_{i,j,k}$ and $FN^{\mathcal{M}}_{i,j,k}$ can be computed using Equation 4 and Equation 5.

$$FP^{\mathcal{M}}_{i,j,k} = 1 if \left(1 - \delta^M_{i,j,k}\right) \cdot \rho^M_{i,j,k} \tag{4}$$

$$FN^{\mathcal{M}}_{i,j,k} = 1 if \left(1 - \rho^M_{i,j,k}\right) \cdot \delta^M_{i,j,k} \tag{5}$$

Let $TN^{\mathcal{M}}_{i,j,k}$ denote the true negatives in $\mathcal{C}_{i,j,k}$. The value of $TN^{\mathcal{M}}_{i,j,k}$ can be calculated using Equation 6

$$TN^{\mathcal{M}}_{i,j,k} = 1 if \left(1 - \delta^M_{i,j,k}\right) \cdot \left(1 - \rho^M_{i,j,k}\right) \tag{6}$$

Let $\mathcal{C}_i$ be the union of all $\mathcal{C}_{i,j,k}$, for $1 \le j \le m$. That is:

$$\mathcal{C}_i = \bigcup_{j=1}^{m} \mathcal{C}_{i,j,k}$$

Let $TP^{\mathcal{M}}_i$, $FP^{\mathcal{M}}_i$, $FN^{\mathcal{M}}_i$, and $TN^{\mathcal{M}}_i$ denote true positive, false positives, false negatives, and true negatives of all $\mathcal{C}_i$ respectively. The values of $TP^{\mathcal{M}}_i$, $FP^{\mathcal{M}}_i FN^{\mathcal{M}}_i$, and $TN^{\mathcal{M}}_i$ can be measured by applying Equation 7, Equation 8, Equation 9, and Equation 10:

$$TP^{\mathcal{M}}_i = \sum_{i=1}^{|Q_i|} \sum_{j=1}^{|A_i|} \sum_{k=1}^{|s_{i,j}|} TP^{\mathcal{M}}_{i,j,k} \tag{7}$$

$$FP^{\mathcal{M}}_i = \sum_{i=1}^{|Q_i|} \sum_{j=1}^{|A_i|} \sum_{k=1}^{|s_{i,j}|} FP^{\mathcal{M}}_i \tag{8}$$

$$FN^{\mathcal{M}}_i = \sum_{i=1}^{|Q_i|} \sum_{j=1}^{|A_i|} \sum_{k=1}^{|s_{i,j}|} FN^{\mathcal{M}}_i \tag{9}$$

$$TN_i^{\mathcal{M}} = \sum_{i=1}^{|Q_i|} \sum_{j=1}^{|A_i|} \sum_{k=1}^{|s_{i,j}|} TN_i^{\mathcal{M}} \tag{10}$$

Let $\mathcal{P}^{\mathcal{M}}$ and $\mathcal{R}^{\mathcal{M}}$ represent the precision and recall of $\mathcal{C}$, respectively. The values can be calculated using Equation 11 and Equation 12.

$$\mathcal{P}^{\mathcal{M}} = \frac{TP^{\mathcal{M}}}{TP^{\mathcal{M}} + FP^{\mathcal{M}}} \tag{11}$$

$$\mathcal{R}^{\mathcal{M}} = \frac{TP^{\mathcal{M}}}{TP^{\mathcal{M}} + FN^{\mathcal{M}}} \tag{12}$$

Let $\mathcal{F}1^{\mathcal{M}}$ represent F1-score of $\mathcal{C}$ predicted by mechanism $\mathcal{M}$. The value of $\mathcal{F}1^{\mathcal{M}}$ can be computed using Equation 13.

$$\mathcal{F}1^{\mathcal{M}} = \frac{2 * \mathcal{P}^{\mathcal{M}} * \mathcal{R}^{\mathcal{M}}}{\mathcal{P}^{\mathcal{M}} + \mathcal{R}^{\mathcal{M}}} \tag{13}$$

3.3. Objective

Assume there is a model M which takes a query Q_i as its input and generates the answer $\hat{A}_i$. That is, $M(Q_i) = \hat{A}_i$, where $\hat{A}_i$ is the predicted answer for the question Q_i. Assume that the true answer A_i belongs to the set of the web documents D^{web}. That is, $A_i \in D^{web}$.

Assume there is a scoring mechanism for evaluating the accuracy of the predicted answers. The scoring mechanism has two components: a hit score S_M^{hit} and a semantic score $S_M^{semantic}$. The total score S_M for the model M is a weighted combination of these two scores, given by:

$$S_M = \alpha S_M^{hit} + (1 - \alpha) S_M^{semantic} \tag{14}$$

The hit score S_M^{hit} for the model M is defined by the Equation 15.

$$S_M^{hit} = \beta\left(Precision^M\right) + (1 - \beta)\left(Recall^M\right) \tag{15}$$

The semantic score $S_M^{semantic}$ for an individual question Q_i and its predicted answer $\hat{A}_i^{\,M}$ is evaluated by *ChatGPT*[4], which provides a score that reflects how well the predicted answer matches the true meaning or context of the question, as denoted by the Equation 16

$$S_{M,i}^{semantic} = ChapGPT^4\left(Q_i, \hat{A}_i^{\,M}\right) \tag{16}$$

The overall semantic score $S_M^{semantic}$ for the model is computed by summing the semantic scores for all individual questions, as defined by the Equation 17.

$$S_M^{semantic} = \sum_{i=1}^{|Q_i|} S_{M,i}^{semantic} \tag{17}$$

Let Ω be the set of all possible models for solving the QA problem. The goal of this paper is to design the best model $\mathcal{M}^{best}$ that achieves the highest score among all possible models in Ω. Equation 18 reflects the objective of this paper.

$$\mathcal{M}^{best} = arg \max_{\mathcal{M} \in \Omega} S_M \tag{18}$$

4. The proposed QA system

This paper introduces a QA system designed to handle user inquires by integrating hierarchical KG, deep learning techniques, and RAG strategies. The core innovation of our proposed system lies in a multi-layered KG architecture, specifically designed to enrich RAG in QA tasks. Each layer in the KG architecture serves distinct roles, storing essential content in vertices along with corresponding vectors. When queried, the

proposed QA system converts the user query into a vector, retrieves similar content for augmentation, and uses RAG techniques to enable the LLM to read and comprehend the retrieved content before answering the question.

As shown in Figure 1, the proposed QA system consists of three main stages: Stage 1 Data Preprocessing and KG Construction, Stage 2 GCN Training for KG, and Stage 3 Internal and External RAG Model. In the initial stage, data is gathered from extensive web-based content, and advanced natural language processing (NLP) techniques are used to extract crucial relationships and entities. These extracted elements are organised into structured triples that capture the semantics of each piece of information. To optimise query responses, these triples are arranged into a multi-layered, hierarchical KG, where each layer represents different levels of granularity, from fine-grained details to broader context. This hierarchical design enhances the KG's ability to store, interlink, and retrieve content with contextual depth, establishing a robust foundation for subsequent QA tasks. In the second stage, the QA mechanism utilises GCN to propagate and aggregate features across the KG. The GCN performs both vertical and horizontal feature convolution to enhance the relation between edges and nodes for effective query matching in the future. In the final stage, the system analyses the question and retrieves answer-related content from the KG. When relevant information cannot be found internally, the system dynamically shifts to an external retrieval module, which accesses updated knowledge from external sources such as Wikipedia. This dynamic retrieval mechanism ensures that the system can supplement missing information and provide accurate, contextually appropriate answers.

4.1. Stage 1: data preprocessing and KG construction

4.1.1. Phase A: data preprocessing

This stage mainly consists of two tasks: preprocessing the collected data and constructing the KG, with a focus on preserving the data's structure and completeness. In the preprocessing task, the collected data T undergoes segmentation, hierarchical identification and analysis. Specifically, T, captured by the web

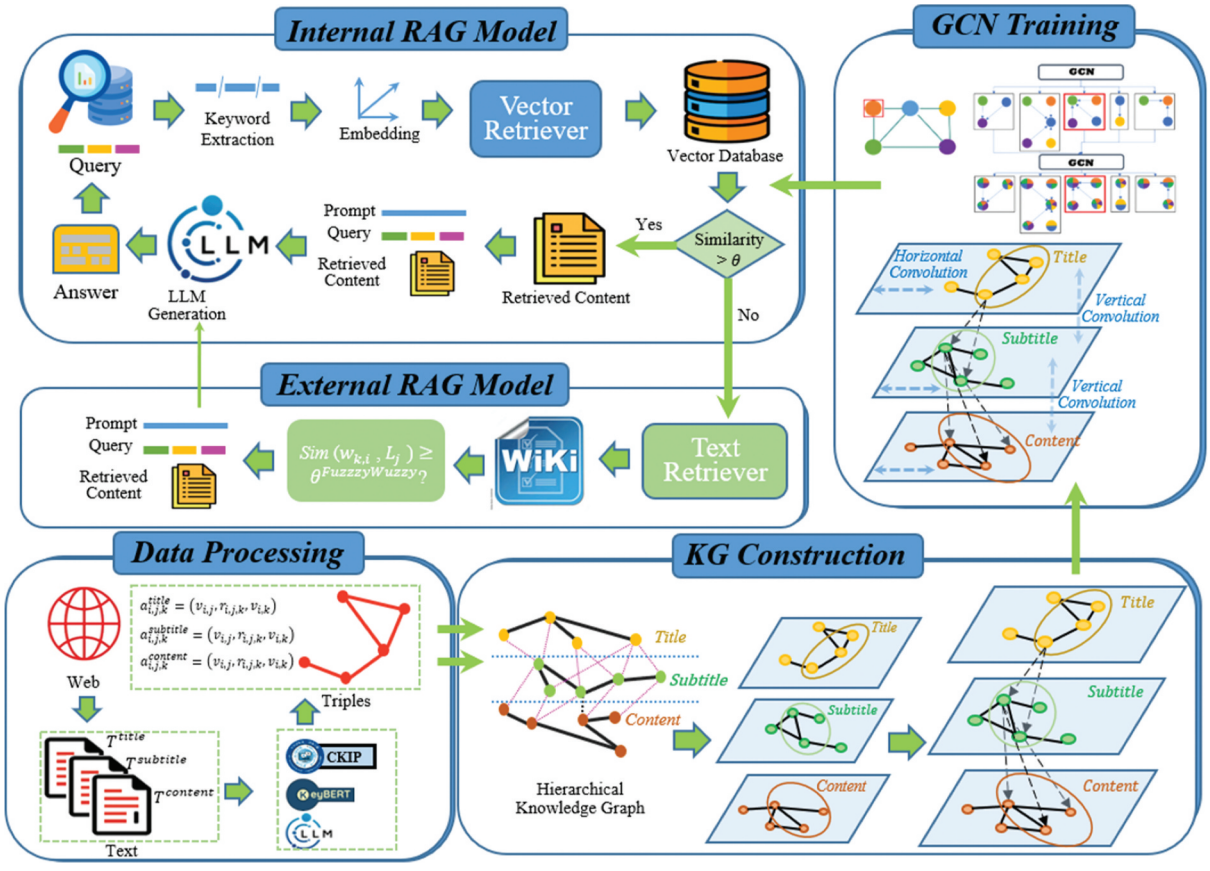


Figure 1. Comprehensive framework for knowledge graph QA with RAG integration.

crawler, is divided into three subsets based on their roles within the text: titles, subtitles, and content. These subsets are represented as T^{title}, $T^{subtitle}$ and $T^{content}$, respectively, and can be further detailed as follows:

$$T^{title} = \left\{T_1^{title}, \ldots, T_a^{title}\right\},$$

$T^{subtitle} = \left\{T_1^{subtitle}, \ldots, T_b^{subtitle}\right\}$, and

$$T^{content} = \left\{T_1^{content}, \ldots, T_c^{content}\right\},$$

where a, b and c denote the number of elements in T^{title}, $T^{subtitle}$, and $T^{content}$, respectively.

To establish a hierarchical relationship, each paragraph c_j in $T^{content}$ is paired with its corresponding title t_i from T^{title} to form clear content boundaries and ensure an organised structure. Subtitles from $T^{subtitle}$ are linked to both their corresponding titles and relevant content, reinforcing the hierarchy. This process facilitates the KG construction by defining clear relationships between titles, subtitles, and content.

For visual data, chart elements are translated into text and incorporated into the KG, with subordinate relationship established between hyperlinked sections and main content. Additionally, linguistic analysis is conducted to refine segmentation accuracy and improve understanding. The resulting data is formatted in a standardised structure, ready for integration into the KG.

In the KG generation task, the ChatGPT is applied to deeply analyse the text, aiming to extract meaningful relationships between entities in the content. Based on this analysis, triples are automatically generated by identifying key entities, their relationships, and relevant attributes. The hierarchical knowledge graph is then further constructed using data collected from web sources. The resulting KG combines both internal and external structures.

The internal part is organised as a three-layer architecture. The first layer consists of title that serve as the primary nodes, representing high-level thematic categories. Each node $v_{i,j} = \left(w_{i,j}^{title}, \vec{v}_{i,j}\right)$ in title layer includes title text $w_{i,j}^{title}$, which provides a broad thematic context, helping to identify core categories and concepts within the KG. The second layer includes subtitles, which refines and classifies the focus of each title node $w_{i,j}^{title}$. Each node $v_{i,j} = \left(w_{i,j}^{subtitle}, \vec{v}_{i,j}\right)$ contains subtitle text $w_{i,j}^{subtitle}$, establishing hierarchical connections to the *title layer* nodes. This layer further organises the KG into subtopics, clarifying the relationships between concepts. The third layer includes nodes for individual paragraphs or content sections, represented by $v_{i,j} = \left(p_{i,j}^{content}, \vec{v}_{i,j}\right)$. Each node contains specific phrases or sentences $p_{i,j}^{content}$ providing detailed information and examples to reinforce understanding of higher-level themes introduced in the *title* and *subtitle layers*.

The external part comprises additional data from external sources to complement the first three layers. This addition makes the KG more comprehensive and enriched it with supplemental information to address potential data gaps.

The KG construction is built layer by layer to represent hierarchical relationships among titles, subtitles, and content. Each layer organises data into a structured graph, which improves the system's retrieval and interpretive capabilities.

4.1.2. Phase B: the construction task of title layer

A hierarchical KG is constructed using three interconnected layers: *title*, *subtitle*, and *content*. These layers organise data from high-level themes to detained content, ensuring a structured framework for reasoning and retrieval. The foundation of the KG is the *title layer*. The set of all titles is denoted as:

$$T^{title} = \left\{T_1^{title}, \ldots T_{|T^{title}|}^{title}\right\},$$

where $\left|T^{title}\right|$ denotes the number of titles in T^{title}. Each title T_i^{title} consists of several words and can be further decomposed into a set of words. That is,

$$T_i^{title} = \left\{w_{i,1}^{title}, , \ldots, w_{i,\left|T_i^{title}\right|}^{title}\right\},$$

where $|T_i|$ denotes the number of words in T_i^{title}. During this phase, a prompt is prepared to instruct LLM to take T_i^{title} as input and output a set of triplets:

$$a_{i,j,k}^{title} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right),$$

where $v_{i,j} \in T_i^{title}$, $v_{i,k} \in T_i^{title}$ and $r_{i,j,k}$ represents the relationship between words of $v_{i,j}$ and $v_{i,k}$. Let A_i^{title} denote the set of all triplets $a_{i,j,k}^{title}$, such that:

$$A_i^{title} = \left\{ a_{i,j,k}^{title} | v_{i,j} \in T_i^{title}, v_{i,k} \in T_i^{title} \right\}.$$

Each vertex $v_{i,j} \in T_i^{title}$ stores rich information represented as:

$$v_{i,j} = \left(w_{i,j}^{title}, \vec{v}_{i,j}\right),$$

where $w_{i,j}^{title}$ is the textual information and $\vec{v}_{i,j}$ is its vector representation. This is achieved by inputting the text of $v_{i,j}$ into the ADA model developed by Open AI. The vector $\vec{v}_{i,j}$ is stored alongside other attributes of $v_{i,j}$ and is used in tasks such as node similarity search. The graph G_i^{title} is defined as:

$$G_i^{title} = \left(V_i^{title}, E_i^{title}\right),$$

where V_i^{title} consists of all entities $v_{i,j}, v_{i,k} \in a_{i,j,k}$ and R_i^{title} consists of all relations $r_{i,j,k} \in a_{i,j,k}$. That is: $V_i^{title} = \{v_{i,j}, v_{i,k} | v_{i,j}, v_{i,k} \in a_{i,j,k}\}$, and

$$E_i^{title} = \{r_{i,j,k} | r_{i,j,k} \in a_{i,j,k}\}.$$

To illustrate, consider $Title_1$ = '*Tamkang is a university*'., as shown in Figure 2. Using the LLM, triplets are extracted from the title, such as:

$$a_{1,1,2}^{title} = \left(v_{1,1}, r_{1,1,2}, v_{1,2}\right),$$

where $v_{1,1} = \left(Tamkang, \vec{v}_{1,1}\right)$ and $v_{1,2} = \left(Univ., \vec{v}_{1,2}\right)$. The relationship $r_{1,1,2} =$' *isa*' connects these vertices. This forms the graph G_1^{Title}

$$G_1^{title} = \left(V_1^{title}, E_1^{title}\right),$$

where $V_1^{Title} = \{v_{1,1}, v_{1,2}\}$ and $E_1^{Title} = \{r_{1,1,2}\}$.

The unified KG for the *title layer* is constructed by combining all individual graphs G_i^{Title}:

$$G^{title} = \bigcup_{i=1}^{|T|} G_i^{title}$$

where G^{title} is represented as:

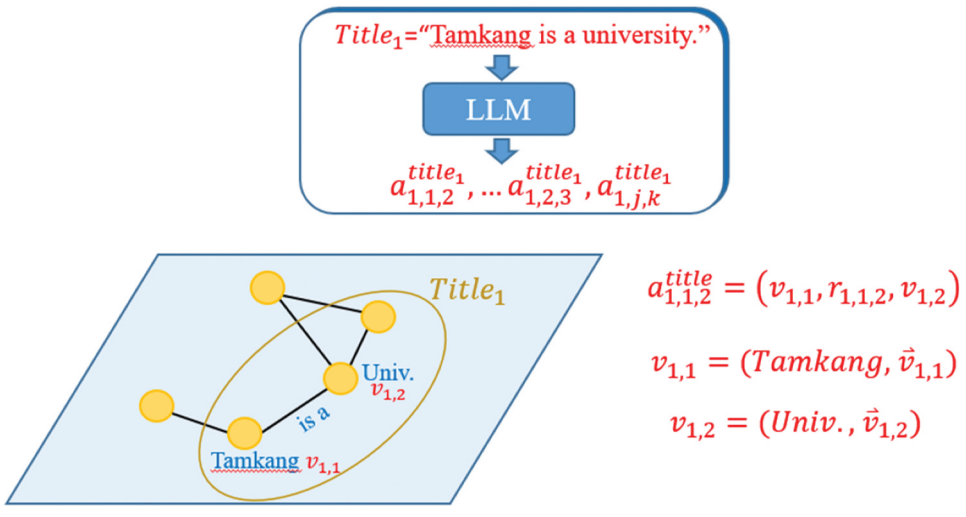


Figure 2. Illustration of title layer construction.

$$G^{title} = \left(V^{title}, E^{title}\right),$$

with:

$$V^{title} = \bigcup_{i=1}^{\left|v_i^{title}\right|} V_i^{title}, E^{title} = \bigcup_{i=1}^{\left|E_i^{title}\right|} E_i^{title}$$

Thus, the KG $G^{title} = \left(V^{title}, E^{title}\right)$ consists of both the entity set V^{title}, containing vertices $v_{i,j}$ and their vectors $\vec{v}_{i,j}$, and the relation set E^{title}, representing the directed edges between vertices in V^{title}.

4.1.3. Phase C: the construction task of subtitle layer

Building on the *title layer*, the *subtitle layer* is constructed similarly, refining and organising the thematic context into a KG $G^{subtitle}$. The set of subtitles is denoted as:

$$T^{subtitle} = \left\{T_1^{subtitle}, \ldots T_{\left|T^{subtitle}\right|}^{subtitle}\right\},$$

where $\left|T^{subtitle}\right|$ represents the number of subtitles. Each subtitle $T_i^{subtitle}$ consists of words:

$$T_i^{subtitle} = \left\{w_{i,1}^{subtitle}, , \ldots, w_{i,\left|T_i^{subtitle}\right|}^{subtitle}, \right\}.$$

Similar to the title layer, relationships within subtitles are captured as triplets:

$$a_{i,j,k}^{subtitle} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right),$$

where $v_{i,j}$ and $v_{i,k}$ are entities, and $r_{i,j,k}$ represents their relationship. Let $A_i^{subtitle}$ denote the set of all triplets $a_{i,j,k}^{subtitle}$, such that:

$$A_i^{subtitle} = \left\{a_{i,j,k}^{subtitle} | v_{i,j} \in T_i^{subtitle}, v_{i,k} \in T_i^{subtitle}\right\}.$$

Each subtitle graph $G_i^{subtitle}$ is defined as:

$$G_i^{subtitle} = \left(V_i^{subtitle}, E_i^{subtitle}\right),$$

where $V_i^{subtitle}$ consists of the vertices and $E_i^{subtitle}$ consists of the relations.

For instance, consider $Subtitle_1$ = *'Tamkang University is located in Tamsui'.*, as shown in Figure 3. Using LLM, a triplet is extracted: $a_{1,1,2}^{subtitle} = \left(v_{1,1}, r_{1,1,2}, v_{1,2}\right)$, where

$$v_{1,1} = \left(Tamkang\ University, \vec{v}_{1,1}\right),$$

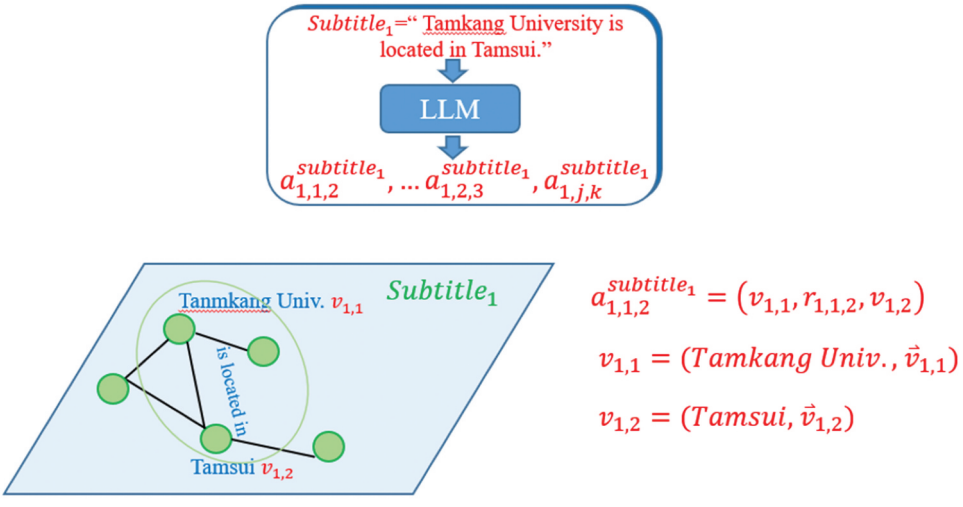


Figure 3. Illustration of subtitle layer construction.

$v_{1,2} = \left(Tamsui, \vec{v}_{1,2}\right)$, and $r_{1,1,2} = is\ located\ in$.

This forms the subgraph:

$$G_1^{Subtitle} = \left(V_1^{Subtitle}, E_1^{Subtitle}\right),$$

with $V_1^{Subtitle} = \{v_{1,1}, v_{1,2}\}$ and $E_1^{Subtitle} = \{r_{1,1,2}\}$.

As in the *title layer*, each vertex $v_{i,j}$ is transformed into a vector $\vec{v}_{i,j}$ using the ADA model. The unified KG for the *subtitle layer* is:

$$G^{subtitle} = \bigcup_{i=1}^{|T^{subtitle}|} G_i^{subtitle}$$

The *subtitle layer* bridges the broad concepts of the *title layer* with the detailed information in the *content layer*, ensuring a smooth transition between themes and their contexts.

4.1.4. Phase D: the construction task of content layers

Extending from the *title* and *subtitle layers*, the *content layer* captures detailed information within content sections to complete hierarchical KG. The set of all content sections is denoted as:

$$T^{content} = \left\{T_1^{content}, \ldots T_{|T^{content}|}^{content}\right\},$$

where $|T^{content}|$ represents the number of content sections, and each section consists of phrases or sentences:

$$T_i^{content} = \left\{p_{i,1}^{content}, , \ldots, p_{i,|T_i^{content}|}^{content}\right\}.$$

Similar to the *title* and *subtitle layers*, the triplets of the *content layer* are defined as:

$$a_{i,j,k}^{content} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right),$$

where $v_{i,j}, v_{i,k} \in T_i^{content}$, and $r_{i,j,k}$ represents their relationship. Let $A_i^{content}$ denote the set of all triplets $a_{i,j,k}^{content}$, such that:

$$A_i^{content} = \left\{a_{i,j,k}^{content} | v_{i,j} \in T_i^{content}, v_{i,k} \in T_i^{content}\right\}$$

Each content graph $G_i^{content}$ is defined as:

$$G_i^{content} = \left(V_i^{cotent}, E_i^{content}\right),$$

where $V_i^{content}$ consists of the vertices and $E_i^{content}$ consists of the relations.

As shown in Figure 4, consider the content section $Content_1$ = *'Tamkang Univ, was established in 1950. It has three main campuses'*. This section can be represented by two triplets that capture the key relationships within the content. The first triplet is defined as:

$$a_{1,1,2}^{content} = \left(p_{1,1}, r_{1,1,2}, p_{1,2}\right),$$

where $p_{1,1} = \left(Tamkang\ Univ., \vec{v}_{1,1}\right)$, $p_{1,2} = \left(1950, \vec{v}_{1,2}\right)$, and $r_{1,1,2}$ = was established in. The second triplet is defined as:

$$a_{1,1,3}^{content} = \left(p_{1,1}, r_{1,1,3}, p_{1,3}\right),$$

where $p_{1,3} = \left(three\ main\ campuses, \vec{v}_{1,3}\right)$, and $r_{1,1,3} = has$. Together, these triplets form the subgraph

$$G_1^{content} = \left(V_1^{content}, E_1^{Content}\right),$$

with $V_1^{Content} = \{p_{1,1}, p_{1,2}, p_{1,3}\}$ and $E_1^{content} = \{r_{1,1,2}, r_{1,1,3}\}$.

Each vertex $v_{i,j}$ stores textual information $p_{i,j}^{content}$ and its vector representation $\vec{v}_{i,j}$, generated using the ADA model. The KG for the *content layer* is:

$$G^{content} = \bigcup_{i=1}^{|T^{content}|} G_i^{content} = \left(V^{content}, E^{content}\right),$$

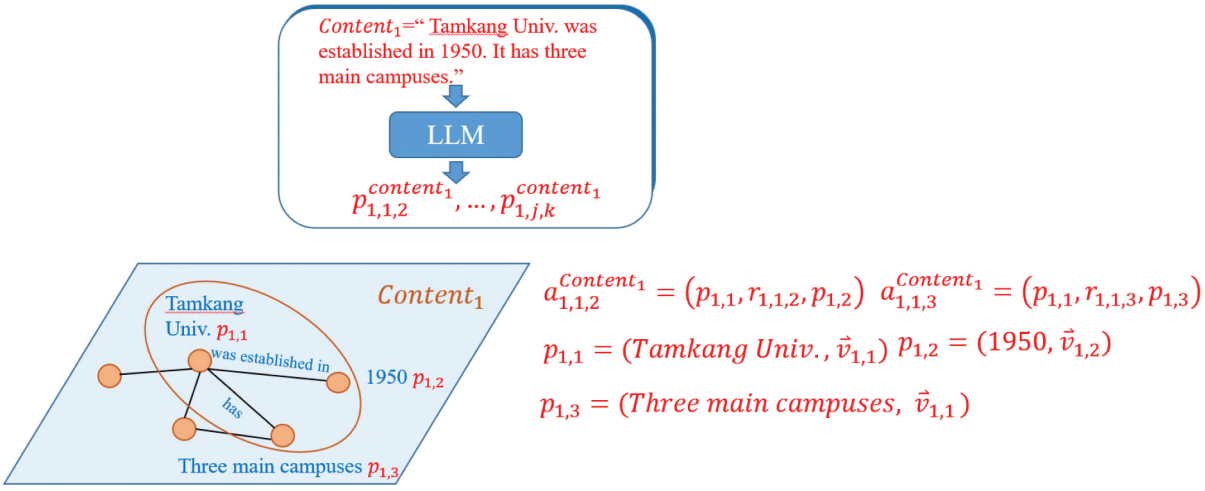


Figure 4. Illustration of content layer construction.

where $V^{content}$ and $E^{content}$ are the unified sets of vertices and edges from all content sections. This layer refines the structure established in the *title* and *subtitle layers*, enabling detailed reasoning and retrieval.

The organisation of the KG which connects the *title*, *subtitle* as well as *content layers* will be presented in the next subsection.

4.1.5. Phase E: the construction task of inter-layer connections

The hierarchical structure of the KG is built by connecting the three layers-title, subtitle, and content – ensuring that the relationships flow from broad concepts to detailed information. The overall multi-layer KG is denoted as the union of the entities and relations sets across all layers:

$$G^{multi-layer} = G^{title} \cup^{G^{subtitle}} \cup^{G^{content}}.$$

Recall that each individual layer is defined as:

$$G^{title} = \left(V^{title}, E^{title}\right),$$

$G^{subtitle} = \left(V^{subtitle}, E^{subtitle}\right)$, and

$$G^{content} = \left(V^{content}, E^{content}\right).$$

The multi-layer graph $G^{multi-layer}$ is expressed as $G^{multi-layer} = \left(V^{multi-layer}, E^{multi-layer}\right)$, where $V^{multi-layer}$ includes the set of all entities across the title, subtitle, and content layers, and $E^{multi-layer}$ contains the set of all relations connecting them. Specifically:

$$V^{multi-layer} = V^{title} \cup^{V^{subtitle}} \cup^{V^{content}} \text{ and}$$

$$E^{multi-layer} = E^{title} \cup^{E^{subtitle}} \cup^{E^{content}}.$$

Let V and E denote the sets of all nodes and relations in the entire KG. Therefore:

$$V = V^{title} \cup^{V^{subtitle}} \cup^{V^{content}} \cup^{V^{multi-layer}},$$

and

$$E = E^{title} \cup^{E^{subtitle}} \cup^{E^{content}} \cup^{E^{multi-layer}}.$$

Thus, the whole KG can be represented as $G = (V, E)$.

As visualised in Figure 5, the graph demonstrates how information flows across the layers. The title layer represents the broadest concepts (e.g. *Tamkang is a university*), while the subtitle layer narrows the context (e,g., *Tamkang University is located in Tamsui*) and connects to the title layer through semantic links such as is

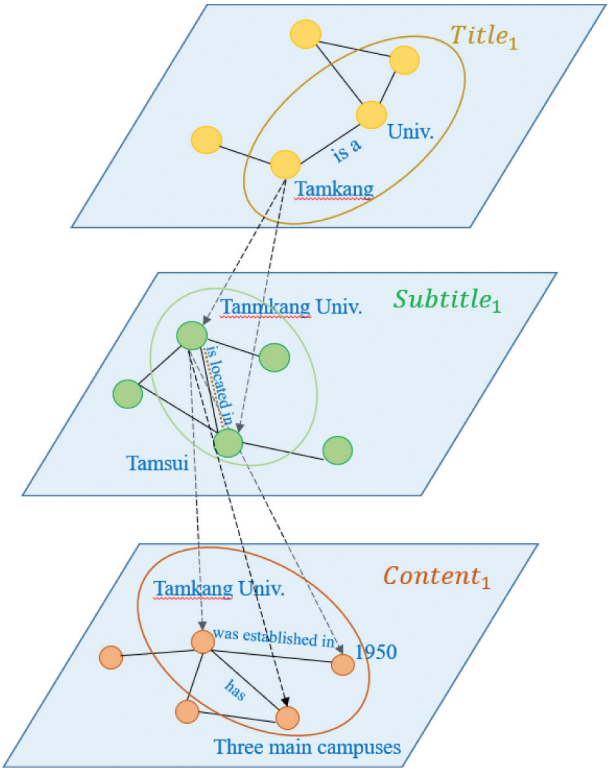


Figure 5. Illustration of inter-layer connections.

located in. The content layer captures detailed information (e.g. *Established in 1950 and has three main campuses*) to provide specifics.

In the figure, the node '*Tamkang*' in the title layer connects to '*Tamkang University*' in the subtitle layer via the relationship refines, indicating that subtitle layer provides a more detailed context of the title layer's concept. Similarly, '*Tamkang University*' in the subtitle layer connects to '*1950*' and '*three main campuses*' in the content layer through relations like was established in and has, demonstrating the flow of information from general to specific.

In this stage, the collected data was carefully preprocessed and structured into a hierarchical KG, representing key entities and relationships across title, subtitle, and content layers. This organisation ensures the information is clearly represented and ready for GCN training, as discussed in the next stage.

4.2. Stage 2: GCN training for KG

In the construction of the *title layer*, the key entities and their relations are organised as a structured KG $G^{title} = \left(V^{title}, E^{title}\right)$, where each node $v_{i,j} \in V^{title}$ is represented by a vector $\bar{v}_{i,j}$ capturing its features. This graph provides a high-level overview of the main concepts. However, simply constructing the hierarchy including G^{title}, $G^{subtitle}$ and $G^{content}$, is insufficient for capturing complex dependencies and interactions between nodes. In particular, relationships between nodes across different layers may contain latent information that cannot be fully exploited through traditional approaches to information aggregation.

To address the limitations of traditional methods, this stage aims to employ GCNs to propagate and aggregate node features both within individual layers and across different layers of the hierarchical KG. Let $v_{i,j}^{(l+1,k)}$ denote the node representation which reflects dual convolution operation, where the index $l+1$ denotes the updated feature representation after aggregating information from multi-hop neighbouring nodes within the same layer, and the index k further captures the vertical aggregation across adjacent layers (e.g. between G^{title}, $G^{subtitle}$ and $G^{content}$). This additional dimensionality k accounts for the integration of information from nodes in different layers, while $l+1$ represents the refinement of features through

horizontal GCN operations on nodes at different hops within the same layer. This comprehensive convolutional process allows for a more robust and hierarchical feature learning, enabling the model to capture both local and global relationships. As a result, the system enhances its capability in downstream tasks such as reasoning, question answering, and complex KG-based inference. In the following, the details of horizontal and vertical feature convolution operations will be introduced.

4.2.1. Phase A: horizontal GCN on multi-layer

The following outlines horizontal feature convolution, including the general operations of GCN, the specific application of GCN on each layer as well as the parameter optimisation. It enhances node representations and captures semantic relationships within each layer to strengthen the contextual understanding of the KG.

(1) General process of GCN

For simplicity, in layers title, subtitle, and content, nodes are represented as $v_{i,j}$, and relationships between nodes are denoted by triplets $a_{i,j,k} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right)$. The knowledge graph for three layers is represented as $G = (V, E)$, where V is the set of all entities (nodes) in G and E is the set of relations (edges).

To update the feature vector $\bar{v}_{i,j}$ of a node $v_{i,j}$ at the $(l+1)$-*th* horizontal GCN layer, features are aggregated from its horizontal set of neighbouring nodes $N_h\left(v_{i,j}\right)$ within the same layer. The operation is defined in Equation 19:

$$\bar{v}_{i,j}^{(l+1)} = \sigma\left(W^{(l)} \frac{1}{\mid N_h\left(v_{i,j}\right) \mid} \sum_{v_{i,k} \in N_h\left(v_{i,j}\right)} A_{i,j,k} \cdot \bar{v}_{i,k}^{(l)}\right) \tag{19}$$

where:

- $W^{(l)}$ is the trainable weight matrix at the l-th layer, responsible for linearly transforming node feature vectors,
- $A_{i,j,k}$ is the individual weight parameter for the edge between $v_{i,j}$ and $v_{i,k}$, adjusting the influence of neighbouring nodes,
- σ is the activation function, introducing non-linearity,
- $\mid N_h\left(v_{i,j}\right) \mid$ is the size of the neighbouring set $N_h\left(v_{i,j}\right)$.

(2) Application of GCN on each layer

After introducing the general process of GCN, this section applies it to the *title*, *subtitle*, and *content layers*. The following explains how nodes and relationships are defined and processed in each layer.

Let nodes $v_{i,j}^{title}$ represent the words in the title layer, and triplets $a_{i,j,k}^{title} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right)$ represent the relationships between nodes $v_{i,j}$ and $v_{j,k}$. The horizontal convolution mainly aggregates features from the set $N_h\left(v_{i,j}^{title}\right)$ of neighbouring nodes, aiming to update $\bar{v}_{i,j}^{title}$. This process further captures semantic relationships within title sentences and enhances the representation of high-level concepts.

Similarly, in the subtitle layer, let nodes $v_{i,j}^{subtitle}$ represent words in the subtitle layer, and triplets $a_{i,j,k}^{subtitle} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right)$ represent the relationships between nodes $v_{i,j}$ and $v_{j,k}$. The convolution process aggregates features from $N_h\left(v_{i,j}^{subtitle}\right)$ to update $\bar{v}_{i,j}^{subtitle}$, bridging thematic concepts in the title layer with detailed context in the content layer. In the content layer, nodes $v_{i,j}^{content}$ correspond to words or phrases in content sections, with relationships captured in triplets $a_{i,j,k}^{content} = \left(v_{i,j}, r_{i,j,k}, v_{i,k}\right)$. Horizontal convolution aggregates features from $N_h\left(v_{i,j}^{content}\right)$ to update $\bar{v}_{i,j}^{content}$, ensuring that semantic details are effectively represented.

(3) Training and optimisation

With the general process and layer-specific applications of horizontal convolution established, this section focuses on the training and optimisation of the model parameters. By fine-tuning the trainable weights, the model improves feature representation and aggregation within each layer.

The trainable parameters $W^{(l)}$ and $A_{i,j,k}$ are optimized during training using a loss function defined for node classification:

$$\mathcal{L} = -\sum_{i=1}^{n}\sum_{c=1}^{c} y_{i,c}\log\hat{y}_{i,c}$$

where $y_{i,c}$ and $\hat{y}_{i,c}$ denote the ground-truth label of node i and the predicted probability that node i belongs to class c. The parameters are updated using gradient descent methods, such as Adam, with updates defined as:

$$W^{(l)} \leftarrow W^{(l)} - \eta\frac{\partial\mathcal{L}}{\partial W^{(l)}}$$

and

$$A_{i,j,k} \leftarrow A_{i,j,k} - \eta\frac{\partial\mathcal{L}}{\partial A_{i,j,k}}$$

where η denotes the learning rate controlling the step size.

During this stage, horizontal feature convolution is applied to *title*, *subtitle*, and *content layers* of the hierarchical KG, where each node aggregates information from its neighbouring nodes to update its feature representation.

This process enhances the contextual understanding of relationships within each layer. Next, vertical feature convolution will be introduced to further integrate information across different layers of the KG.

4.2.2. Phase B: vertical GCN on multi-layer

This subsection presents the details of vertical feature convolutions, which aim to propagate and aggregate information from nodes across different layers of the hierarchical KG. Unlike horizontal convolution, which gathers features from neighbouring nodes within the same layer, vertical convolution aggregates information from nodes in the layers above and below. This allows nodes to receive cross-layer information, enhancing their understanding of inter-layer relationships.

For *title layer*, vertical feature convolution aggregates information from the *subtitle layer*, which has already incorporated node information from the *content layer*. This hierarchical process allows nodes in the *title layer* to refine their representations by incorporating context from both subordinate layers.

Let $N_v\left(v_{i,j}^{title}\right)$ denote the set of neighbouring nodes from the *subtitle layer* connected to node $v_{i,j}^{title}$, where the subtitle nodes have already aggregated information from the *content layer*. Let $\bar{v}_{i,j}^{title,k}$ denote the updated feature vector of node $v_{i,j}^{title}$ after vertical aggregation, with k indicating the integration of information from both the *subtitle layer* and the *content layer*. The vertical convolution operations are shown in Equation 20:

$$\bar{v}_{i,j}^{title,k} = \sigma\left(W^{(v)}\frac{1}{\left|N_v\left(v_{i,j}^{title}\right)\right|}\sum_{v_{i,k}^{title}\in N_v\left(v_{i,j}^{title}\right)} A_{i,j,k}\cdot\bar{v}_{i,k}^{title}\right) \tag{20}$$

During the training process, the weight matrix $W^{(v)}$ and the individual weight parameters $A_{i,j,k}$ are optimized to minimize the classification loss:

$$\mathcal{L} = -\sum_{i=1}^{n}\sum_{c=1}^{c} y_{i,c}\log\hat{y}_{i,c}.$$

Weights are adjusted via backpropagation as follows:

$$W^{(v)} \leftarrow W^{(v)} - \eta\frac{\partial\mathcal{L}}{\partial W^{(v)}}$$

and

$$A_{i,j,k} \leftarrow A_{i,j,k} - \eta\frac{\partial\mathcal{L}}{\partial A_{i,j,k}},$$

where η represents the learning rate.

For the *subtitle layer* and *content layer*, the convolution aggregates information from adjacent layers. For the *subtitle layer*, information is aggregated from the *title* and *content layer*, while for the content layer, information is aggregated from the *subtitle layer*, which has already incorporated the *title layer's* information. The vertical convolution operations for *subtitle* and *content layers* are shown in Equation 21 and Equation 22 respectively:

$$\vec{v}_{i,j}^{subtitle,k} = \sigma\left(W^{(v)}\frac{1}{\mid N_v\left(v_{i,j}^{subtitle}\right)\mid}\sum_{v_{i,k}^{subtitle}\in N_v\left(v_{i,j}^{subtitle}\right)} A_{i,j,k}\cdot\vec{v}_{i,k}^{subtitle}\right) \tag{21}$$

$$\vec{v}_{i,j}^{content,k} = \sigma\left(W^{(v)}\frac{1}{\mid N_v\left(v_{i,j}^{content}\right)\mid}\sum_{v_{i,k}^{content}\in N_v\left(v_{i,j}^{content}\right)} A_{i,j,k}\cdot\vec{v}_{i,k}^{content}\right). \tag{22}$$

The weight training for both $W^{(v)}$ and $A_{i,j,k}$ for the *subtitle* and *content layers* follows the same optimisation process as the *title layer*.

In practice, two layers of GCN were employed to model both intra-layer and inter-layer interaction within the hierarchical KG. The first layer aggregates initial structural information and updates the node features, while the second layer refines these features to better represent local and global relationships. Through this approach, the QA mechanism effectively leverages the graph's structural properties, enabling more accurate predictions based on learned node features. This is particularly evident in user query retrieval, where each node accesses information from the entire graph not just its immediate neighbours. As a result, the model can more precisely locate and answer specific user queries, which significantly improves the relevance and accuracy of the retrieval process. This also establishes a strong foundation for the next phase, where the internal and external RAG models are developed to further enhance the system's capabilities. The detailed construction and training processes of the KG are outlined in Table 2.

4.3. Stage 3: QA system using constructed KG with RAG

This stage focuses on internal and external RAG models. The internal RAG model processes user queries by matching them with the KG, while the external RAG model queries external databases, including aligning entities between the KG and external sources such as Wiki. Specifically, there are two paths for performing queries.

When a user submits a query, it is processed by a LLM for semantic analysis and vectorisation. The system then matches the query vector with nodes in the hierarchical KG. If relevant nodes are found, the LLM generates a response based on the information from the graph. When the query cannot be fully answered using the KG, the system extracts keywords to perform an external search in Wiki's KG, utilising the FuzzyWuzzy algorithm for precise entity matching. This dual-path approach ensures flexibility and accuracy in generating responses. Next, the details of each RAG model will be discussed.

4.3.1. Internal RAG model

In this model, user queries are transformed into vectors and matched with nodes in the hierarchical KG. The system conducts a multi-layer search through the *title*, *subtitle*, and *content layers* to identify relevant nodes. Multi-hop reasoning is applied when necessary to explore deeper connections. The LLM generates the final response based on the retrieved information from the KG.

(1) Query representation

Let the user query Q be transformed into a vector representation q, which is in a d-dimensional embedding space $\in \mathbb{R}$, where d denotes the dimensionality of the embedding space. The embedding q is generated using a pre-trained language model Embed(Q):

$$\vec{\boldsymbol{q}} = Embed(Q) \in \mathbb{R}$$

This vector representation $\vec{\boldsymbol{q}}$ captures the semantic features of the user's query, which allows it to be compared against the nodes in the KG.

Table 2. Construction and training of knowledge graph.

Algorithm: construction and training of knowledge graph Input: T (web documents) Output: knowledge graph G= (V, E)	
1	Stage 1 Preprocessing and KG Construction Stage Phase A Preprocessing
2	$T^{title} = \{T_1^{title}, \ldots, T_a^{title}\}$, $T^{subtitle} = \{T_1^{subtitle}, \ldots, T_b^{subtitle}\}$, and $T^{content} = \{T_1^{content}, \ldots, T_c^{content}\}$,
3	Phase B Title Layer Construction
4	**For** each $v_{i,j}$,$v_{i,k} \in T_i^{title}$ Applying LLM to construct the set of all triplets: $A_i^{title} = \left\{ a_{i,j,k}^{title} \middle\| v_{i,j} \in T_i^{title}, v_{i,k} \in T_i^{title} \right\}$ $V_i^{title} = \bigcup_{(v_{i,j}, v_{i,k}) \in a_{i,j,k}} \{v_{i,j}, v_{i,k}\}$ $E_i^{title} = \bigcup_{r_{i,j,k} \in a_{i,j,k}} r_{i,j,k}$ $G_i^{title} = (V_i^{title}, E_i^{title})$ $G^{title} = (V^{title}, E^{title}) = \bigcup_{i=1}^{\|T\|} G_i^{title}$
5	Phase C Subtitle Layer Construction
6	**For** each $v_{i,j}$,$v_{i,k} \in T_i^{subtitle}$ Applying LLM to construct the set of all triplets: $A_i^{subtitle} = \left\{ a_{i,j,k}^{subtitle} \middle\| v_{i,j} \in T_i^{subtitle}, v_{i,k} \in T_i^{subtitle} \right\}$ $V_i^{substitute} = \bigcup_{(v_{i,j}, v_{i,k}) \in a_{i,j,k}} \{v_{i,j}, v_{i,k}\}$ $E_i^{substitute} = \bigcup_{r_{i,j,k} \in a_{i,j,k}} r_{i,j,k}$ $G_i^{subtitle} = (V_i^{subtitle}, E_i^{subtitle})$ $G^{subtitle} = (V^{subtitle}, E^{subtitle}) = \bigcup_{i=1}^{\|T\|} G_i^{subtitle}$
7	Phase D Content Layer Construction **For** each $v_{i,j}$,$v_{i,k} \in T_i^{content}$ Applying LLM to construct the set of all triplets: $A_i^{content} = \left\{ a_{i,j,k}^{content} \middle\| v_{i,j} \in T_i^{content}, v_{i,k} \in T_i^{content} \right\}$ $V_i^{content} = \bigcup_{(v_{i,j}, v_{i,k}) \in a_{i,j,k}} \{v_{i,j}, v_{i,k}\}$ $E_i^{content} = \bigcup_{r_{i,j,k} \in a_{i,j,k}} r_{i,j,k}$ $G_i^{content} = (V_i^{content}, E_i^{content})$ $G^{content} = (V^{content}, E^{content}) = \bigcup_{i=1}^{\|T\|} G_i^{content}$
8	Phase E Inter-Layer Construction
9	$G^{multi-layer} = G^{title} \cup G^{subtitle} \cup G^{content}$ $V^{multi-layer} = V^{title} \cup V^{subtitle} \cup V^{content}$ $E^{multi-layer} = E^{title} \cup E^{subtitle} \cup E^{content}$ V=$V^{title} \cup V^{subtitle} \cup V^{content} \cup V^{multi-layer}$ $E = E^{title} \cup E^{subtitle} \cup E^{content} \cup E^{multi-layer}$ G= (V, E)

(*Continued*)

(2) Multi-Layer KG retrieval process

(a) Title layer retrieval

Recall that the KG $G^{title} = (V^{title}, E^{title})$ consists of node set V^{title} and relation set E^{title}. Let $\vec{v}_i^{title}$ denote the vector of node $v_i^{title} \in V^{title}$. Let $Sim(\cdot)$ denote the similarity function applied to evaluate the similarity between the query vector $\vec{q}$ and node vector $\vec{v}_i^{title}$, as shown in Equation (23). That is,

$$sim\left(\vec{q}\vec{v}_i^{title}\right) = \frac{\vec{q} \cdot \vec{v}_i^{title}}{\| \vec{q} \| \| \vec{v}_i^{title} \|} \tag{23}$$

A threshold θ^{title} is applied to determine whether the similarity score is sufficiently high. In this study, the threshold values were empirically determined through experiments by testing different candidates and selecting those that yielded the best performance on our datasets. These values were then fixed for subsequent evaluations. In practice, when applying the method to different domains, the same experimental

Table 2. (Continued).

10	Stage 2 GCN Training for KG Phase A Horizontal GCN applied on Multi-Layer
11	**For** each title node $v_{i,j}^{title}$ in V^{title}: $N_h\left(v_{i,j}^{title}\right) = \underset{\left(v_{i,j}^{title}, v_x\right) \in G^{title}}{\cup} \{v_x\}$ $\vec{v}_{i,j}^{\,title} = \frac{1}{\left\|N_h\left(v_{i,j}^{title}\right)\right\|} \sum_{v_{i,k}^{title} \in N_h\left(v_{i,j}^{title}\right)} A_{i,j,k} \cdot \vec{v}_{i,k}^{\,title}$ $\vec{v}_{i,j}^{\,title,l+1} = \sigma\left(W^{(l)} \cdot \vec{v}_{i,j}^{\,title}\right)$ **For** each subtitle node $v_{i,j}^{subtitle}$: $N_h\left(v_{i,j}^{subtitle}\right) = \underset{\left(v_{i,j}^{subtitle}, v_x\right) \in G^{subtitle}}{\cup} \{v_x\}$ $\vec{v}_{i,j}^{\,subtitle} = \frac{1}{\left\|N_h\left(v_{i,j}^{subtitle}\right)\right\|} \sum_{v_{i,k}^{subtitle} \in N_h\left(v_{i,j}^{subtitle}\right)} A_{i,j,k} \cdot \vec{v}_{i,k}^{\,subtitle}$ $\vec{v}_{i,j}^{\,subtitle,l+1} = \sigma\left(W^{(l)} \cdot \vec{v}_{i,j}^{\,subtitle}\right)$ **For** each content node $v_{i,j}^{content}$: $N_h\left(v_{i,j}^{content}\right) = \underset{\left(v_{i,j}^{content}, v_x\right) \in G^{content}}{\cup} \{v_x\}$ $\vec{v}_{i,j}^{\,content} = \frac{1}{\left\|N_h\left(v_{i,j}^{content}\right)\right\|} \sum_{v_{i,k}^{content} \in N_h\left(v_{i,j}^{content}\right)} A_{i,j,k} \cdot \vec{v}_{i,k}^{\,content}$ $\vec{v}_{i,j}^{\,content,l+1} = \sigma\left(W^{(l)} \cdot \vec{v}_{i,j}^{\,content}\right)$
12	Phase B Vertical GCN applied on Multi-Layer
13	**For** each title node $v_{i,j}^{title}$ in G^{title}: $N_v\left(v_{i,j}^{title}\right) = \underset{\left(v_{i,j}^{title}, v_x\right) \in E, v_x \in G^{subtitle}}{\cup} \{v_x\}$ $\vec{v}_{i,j}^{\,title} = \frac{1}{\left\|N_v\left(v_{i,j}^{title}\right)\right\|} \sum_{v_{i,k}^{title} \in N_v\left(v_{i,j}^{title}\right)} A_{i,j,k} \cdot \vec{v}_{i,k}^{\,title}$ $\vec{v}_{i,j}^{\,title,k} = \sigma\left(W^{(v)} \cdot \vec{v}_{i,j}^{\,title}\right)$ **For** each subtitle node $v_{i,j}^{subtitle}$ in $G^{subtitle}$ $N_v\left(v_{i,j}^{subtitle}\right) = \underset{\left(v_{i,j}^{subtitle}, v_x\right) \in E \text{and}, v_x \in G^{Content}}{} (v_x)$ $\vec{v}_{i,j}^{\,subtitle} = \frac{1}{\left\|N_v\left(v_{i,j}^{subtitle}\right)\right\|} \sum_{v_{j,k}^{subtitle} \in N_v\left(v_{i,j}^{subtitle}\right)} A_{i,j,k} \vec{v}_{i,k}^{\,subtitle}$ $\vec{v}_{i,j}^{\,subtitle,k} = \sigma\left(W^{(v)} \cdot \vec{v}_{i,j}^{\,subtitle}\right)$ **For** each content node $v_{i,j}^{content}$ in $G^{content}$ $N_v\left(v_{i,j}^{content}\right) = \underset{\left(v_{i,j}^{content}, v_x\right) \in E \text{and}, v_x \in G^{subtitle}}{} (v_x)$ $\vec{v}_{i,j}^{\,content} = \frac{1}{\left\|N_v\left(v_{i,j}^{content}\right)\right\|} \sum_{v_{j,k}^{content} \in N_v\left(v_{i,j}^{content}\right)} A_{i,j,k} \cdot \vec{v}_{i,k}^{\,content}$ $\vec{v}_{i,j}^{\,content,k} = \sigma\left(W^{(v)} \cdot \vec{v}_{i,j}^{\,content}\right)$

procedure can be used to determine suitable threshold values based on data rather than manual rules. Although this approach does not provide the full flexibility of a learned strategy, the performance-based adjustment of thresholds still ensures robustness and reliability, and it can be further refined in the future through adaptive learning methods.

If the similarity score satisfies: $Sim\ (\vec{\boldsymbol{q}}, \vec{v}_i^{\,title}) \geq \theta^{title}$ for all $v_i^{title} \in V^{title}$ Let $V^{title,q}$ be the set of nodes $v_i^{title} \in V^{title}$ that satisfy Equation (23). That is,

$$V^{title,q} = \left\{ v_i^{title} | Sim\left(, \vec{v}_i^{\,title}\right) \geq \theta^{title}, v_i^{title} \in V^{title} \right\}.$$

Herein, it is noticed that $V^{title,q}$ is a set of candidates, where each node in the set is highly relevant to the question Q. Next, for each node, say $v_i^{title,q}$, in the set $V^{title,q}$, the proposed QA system will search for multiple nodes in the *subtitle layer* that are highly related to node $v_i^{title,q}$. The subtitle content stored in these nodes will also become critical content for our continued RAG relevant dataset.

(b) Subtitle layer retrieval

Let V^{inter_12} be the set of nodes in the subtitle layer that are connected any node $v_i^{title,q} \in V^{title,q}$. That is,

$$V^{inter\_12} = \{v_j^{subtitle} | \left(v_i^{title,q}, v_j^{subtitle}\right) \in E, v_j^{subtitle} \in V^{subtitle}, v_i^{title,q} \in V^{title,q}\}.$$

Herein, for each node, $v_j^{subtitle}$, this represents the nodes in the *subtitle layer* that are directly linked to the selected nodes $v_i^{title,q}$ from the *title layer*. Each subtitle node $v_j^{subtitle}$ is connected via an edge $\left(v_i^{title,q}, v_j^{subtitle}\right) \in E$ to the corresponding node $v_i^{title,q}$ in the *title layer*, which has been identified as relevant to the query *Q*.

Let $\vec{v}_j^{subtitle}$ denote the vector form of node $v_j^{subtitle} \in V^{subtitle}$. For each $v_j^{subtitle} \in V^{inter\_12}$, the same similarity computation is performed, as shown in Eq. (24):

$$sim\vec{v}_j^{subtitle} = \frac{\vec{q} \cdot \vec{v}_j^{subtitle}}{\| \vec{q} \| \| \vec{v}_j^{subtitle} \|} \tag{24}$$

A threshold $\theta^{subtitle}$ is applied to determine whether the similarity score between the query vector $\vec{\boldsymbol{q}}$ and the subtitle node vector $\vec{v}_j^{subtitle}$ is sufficiently high. Let $V^{subtitle,Q}$ be the set of subtitle nodes that satisfy the above condition. That is,

$$V^{subtitle,q} = \left\{v_j^{subtitle,q} | Sim\left(, \vec{v}_j^{subtitle}\right) \geq \theta^{subtitle}, v_j^{subtitle} \in V^{inter\_12}\right\}.$$

The set $V^{subtitle,q}$ consists of candidate nodes that are highly relevant to the question q ∈ Q. For each node $v_j^{subtitle,q} \in V^{inter\_12}$, the system searches for related nodes in the content layer. The information in these content nodes provides finer details essential for the continued RAG dataset.

(c) Content layer retrieval

Let V^{inter_23} be the set of nodes in the content layer that are connected to any node $v_j^{subtitle,q} \in V^{subtitle,q}$. That is,

$$V^{inter\_23} = \{v_k^{content} | \left(v_i^{title,q}, v_j^{subtitle,q}, v_k^{content}\right) \in E, v_k^{content} \in V^{content}\}.$$

Here, V^{inter_23} represents the set of content layer nodes connected to relevant subtitle nodes from $V^{subtitle,q}$. Each node $v_k^{content}$ provides detailed information directly linked to the subtitle nodes. For each $v_k^{content} \in V^{inter\_23}$, similarity computation is performed between the query vector $\vec{\boldsymbol{q}}$ and the content node vector $\vec{v}_k^{content}$, as shown in Equation (25):

$$sim\left(\vec{q}, \vec{v}_k^{content}\right) = \frac{\vec{q} \cdot \vec{v}_k^{content}}{\| \vec{q} \| \| \vec{v}_k^{content} \|} \tag{25}$$

Similarly, a threshold $\theta^{content}$ is applied to determine whether the similarity score is sufficiently high. Let $V^{content,q}$ be the set of content nodes that satisfy the above condition:

$$V^{content,q} = \left\{v_k^{content,q} | Sim\left(, \vec{v}_k^{content}\right) \geq \theta^{content}, v_k^{content} \in V^{inter\_23}\right\},$$

the nodes in $V^{content,q}$ provide fine-grained information highly relevant to the query Q.

(d) Multi-Hop reasoning

When high-similarity nodes are identified, the system may also perform multi-hop reasoning by exploring neighbouring nodes at each layer. Let

$$V^q = V^{title,q} \cup^{V^{subtitle,q}} \cup^{V^{content,q}}.$$

That is, the set V^{q_plus} collects all nodes highly related to question q. For each node $v^q \in V^q$, the neighbouring nodes of v^q will be further identified if it is related to q. Let θ^q be the threshold of similarity

between nodes node v^q and q. Let $N(v^q)$ denote the set of neighbours of node v^q. The nodes that related to question q will also be collected in set V^q. That is,

$$V^{q\_plus} = V^q \cup \left\{ v_i | \mathrm{Sim}\left(,\vec{v}_i\right) \geq \theta^q, \mathrm{for} \forall v_i \in N(v^q), \mathrm{for} \forall v^q \in V^q \right\}.$$

(e) Answer generation

To generate accurate and contextually relevant answers, it is important to identify not only explicit relationships between nodes but also implicit connections that may not be directly linked between nodes. This step involves internal knowledge integration through relational checking and triple formation, which ensures that the system captures all potential relationships related to query. By systematically examining and forming triples based on both existing edges and inferred relationships within the hierarchical KG, the system builds a comprehensive set of answer-relevant triples for precise answer generation.

To further enhance the credibility of the output and mitigate hallucinated responses, the language model is guided by a prompt that explicitly restricts its generation to the extracted triples in A^q. This prompt-restricted generation ensures that the LLM produces responses grounded only in verified hierarchical KG content, reducing the likelihood of unsupported or fabricated information (Song et al. 2024; W. Zhang and Zhang 2025).

Let A^q denote the set of final triplets which will be used to generate the answer. Let $V_{independent}$ denote the independent set which includes those nodes that are not part of any relationships or triples yet. Initially, the triple set is empty and the independent set is equal to V^q. That is,

A^q=∅ and $V_{independent} = V^q$.

For each pair of nodes $v_i, v_j \in V^q$, if $r_{i,j} = \left(v_i, v_j\right) \in E$, the triple $\left(v_i, r_{i,j}, v_j\right)$ is added to A^q, which is the set of answer-relevant triples. The independent node set is then updated by removing v_i and v_j as follows:

$$V_{independent} \leftarrow V_{independent} \setminus \left\{v_i, v_j\right\}.$$

Otherwise, if the similarity $Sim\left(v_i, v_j\right) \geq \theta$, the relationship is defined as:

$$r_{i,j} = Relation\left(v_i, v_j\right),$$

and the triple $\left(v_i, r_{i,j}, v_j\right)$ is added to A^q, and the independent node set is updated similarly:

$$V_{independent} \leftarrow V_{independent} \setminus \left\{v_i, v_j\right\}.$$

For nodes in $V_{independent}$, which are those that remain unconnected after the above steps, each isolated node $v_i \in V_{indepent}$ is examined for its neighbours $v_j \in N(v_i)$. If $Sim\left(v_i, v_j\right) \geq \theta$, the relationship is defined as:

$$r_{i,j} = Relation\left(v_i, v_j\right),$$

and the triple $\left(v_i, r_{i,j}, v_j\right)$ is added to A^q. At the end of this process, the set A^q contains all triplets representing the relevant relationships extracted from V^q.

An LLM M uses both the query vector $\vec{\boldsymbol{q}}$ and the set A^q, to produce the final response. Additionally, a prompt is provided to guide the model M in interpreting the query context and aligning it with the structured information in A^q. The prompt helps the model use the query and extracted relationships to generate accurate responses.

$A = M\ (p, Q, A^q)$

This answer A, enriched with relevant knowledge from the hierarchical KG, is provided to the user as the system's final output.

As part of this process, the system records the reasoning path from the query to the final answer, including node transitions, relationships, and similarity scores across the title, subtitle, and content layers. These records are currently used for internal validation and analysis of the retrieval process.

4.3.2. External RAG model

If no relevant nodes are found in the internal KG for a given query, the system dynamically shifts to an external retrieval path using Wiki's KG. Let $Q_k = \left(w_{k,1}, w_{k,2}, \ldots w_{k,j}\right)$ represent the set of essential keywords

extracted from the query, where each $w_{k,i}$ denotes a key term within the question. Each entity in the Wiki KG can be represented as $E_j = (L_j, A_j)$, where L_j is the entity label serving as the main name or title for the entity. Let $A_j = (a_{j,1}, a_{j,2}, \ldots a_{j,n})$ denote the set of attributes associated with the entity, where each $a_{j,i}$ provides additional information or descriptive text relevant to E_j

To match the query keywords Q_k with entities in Wiki's KG, let $Sim(\cdot)$ denote the similarity function used to compare each keyword $w_{k,i}$ from Q_k and with both the label L_j and the attributes A_j of each Wiki entity E_j. The FuzzyWuzzy algorithm is applied to measure the similarity between $w_{k,i}$ and each entity's label and attributes. To further enhance retrieval precision, a domain-specific lexicon (covering abbreviations, synonyms, and specialised terminology in the target corpus) was incorporated. In addition, the fuzzy matching rules were refined through human validation to ensure that retrieved entities are semantically aligned and to reduce the likelihood of off-topic or hallucinated results.

To achieve this, each entity E_j is evaluated to determine if it should be included in the set of retrieval results. Let $D_{external}$ denote the set of retrieval results. Specifically, the label L_j of entity E_j in the Wiki KG is verified to check whether the following criterion is satisfied: if $Sim(w_{k,i}, L_j) \geq \theta^{FuzzzyWuzzy}$:

If the criterion is met, the entity E_j is added to $D_{external}$. Let $D_{external} = \{e_1, e_1 \ldots e_n\}$ denote the set of Wiki entities identified as relevant to the query. A prompt combining the user's query and the extracted keywords Q_k, along with the retrieved documents $D_{external}$ is submitted to multiple LLMs for answer generation. Their outputs are cross-verified to ensure factual alignment and consistency. The final answer is represented as:
$A = M(p, Q, D_{external})$,
where M denotes the generation process derived from multiple LLMs. Given the potential incompleteness or inaccuracy of external sources such as Wiki, multiple LLMs cross-verify the outputs to mitigate hallucination and improve factual reliability. The generated answer A is then returned to the user as the system's final output. This answer captures the meaning of the query and includes additional information from the Wiki knowledge source.

By integrating an internal RAG model that retrieves curated structured knowledge with a dynamically updated external source, this dual-source design balances stability and adaptability. While the internal KG provides reliable, domain-specific information, the external Wiki-based retriever enables the system to remain responsive to newly emerging knowledge. Through this integration, the QA system can generate accurate and up-to-date answers, even in fast-changing or open-domain scenarios. The QA System that utilises constructed KG with RAG is outlined in Table 3.

5. Performance evaluation

This section evaluates the proposed KG-based QA system by presenting the experimental setup, dataset construction, and benchmarking. The analysis confirms the system's superior performance across metrics such as accuracy, loss trends, and the impact of Wiki integration, which proves its ability to handle diverse and complex queries compared to existing methods.

5.1. Dataset construction and benchmarking

To train and evaluate the proposed KG-based system, a comprehensive dataset was developed, comprising two main parts. The first part includes 43,286 QA pairs across categories such as Frequently Asked Questions (FAQ), Course-related Questions, and Campus Life. These pairs, sourced from real academic and campus environments and refined by experts, offer a diverse range of topics and scenarios. The second part consists of 13,985 QA pairs with answers generated through the integration of LLMs, knowledge graphs, and Graph Convolutional Networks. These pairs were crafted to evaluate the system's performance across varying levels of complexity.

Additionally, two widely recognised datasets were used for benchmarking. The Complex WebQuestions 1.1 (CWQ) dataset, known for its challenging queries, was employed to assess the system's ability to handle intricate knowledge queries. The WebQuestions dataset, designed for web-based knowledge graph evaluations, provided insights into the system's effectiveness in real-world, internet-based applications.

Table 3. QA system using constructed KG with RAG.

Algorithm: QA System using constructed KG with RAG
Input: the query Q
Output: the answer $\widehat{A}$

1	Internal RAG Model
2	(1) Query Representation
3	$\vec{q} = \text{Embed}(Q) \in \mathbb{R}^d$
4	(1) Multi-Layer KG Retrieval Process
5	(a) Title Layer Retrieval
6	Find title-layer nodes related to $\vec{q}$: $V^{title,q} = \left\{ v_i^{title} \middle| Sim\left(, \vec{v}_i^{title}\right) \geq \theta^{title} \right\}$
7	(a) Subtitle Layer Retrieval
8	Identify intermediate subtitle candidates connected to $V^{title,q}$: $V^{inter\_12} = \{v_j^{subtitle} \mid \left(v_i^{title,q}, v_j^{subtitle}\right) \in E, v_j^{subtitle} \in V^{subtitle}, v_i^{title,q} \in V^{title,q}\}$ From V^{inter_12}, select subtitle-layer nodes similar to $\vec{q}$: $V^{subtitle,q} = \left\{ v_j^{subtitle,q} \middle| Sim\left(, \vec{v}_j^{subtitle}\right) \geq \theta^{subtitle} \right\}$
9	(a) Content Layer Retrieval
10	Identify intermediate content candidates connected to $V^{subtitle,q}$: $V^{inter\_23} = \{v_k^{content} \mid \left(v_i^{title,q}, v_j^{subtitle,q}, v_k^{content}\right) \in E, v_k^{content} \in V^{content}\}$ From V^{inter_23}, select content-layer nodes similar to $\vec{q}$: $V^{content,q} = \left\{ v_k^{content,q} \middle| Sim\left(, \vec{v}_k^{content}\right) \geq \theta^{content} \right\}$
11	(a) Multi-Hop Reasoning
12	Combine all selected nodes and include other nodes with similarity $\geq \theta^q$: $V^{q\_plus} = V^q \cup^{\{} v_i \mid \text{Sim}\left(, \vec{v}_i\right) \geq \theta^q\}$
13	(a) Answer Generation

14	$A^q = \emptyset$
	$V_{independent} = V^q$
	For each v_i, v_j in V^q
	Add existing edges as triples
	if $v_i, v_j \in E$:
	$r_{i,j} = Relation\left(v_i, v_j\right)$
	$A^q = A^q \cup \left\{\left(v_i, r_{i,j}, v_j\right)\right\}$
	$V_{independent} = V_{independent} \setminus \left\{v_i, v_j\right\}$
	For each v_i, v_j in V^q
	Infer new relations by similarity
	if $v_i, v_j \notin E$ and $Sim\left(v_i, v_j\right) \geq \theta$:
	$r_{i,j} = Relation\left(v_i, v_j\right)$
	$A^q = A^q \cup \left\{\left(v_i, r_{i,j}, v_j\right)\right\}$
	$V_{independent} = V_{independent} \setminus \left\{v_i, v_j\right\}$
	For each v_j, in $V_{indipendent}$
	Handle isolated nodes
	for each V_j in $N(v_i)$:
	if $Sim\left(v_i, v_j\right) \geq \theta$:
	$r_{i,j} = Relation\left(v_i, v_j\right)$
	$A^q = A^q \cup \left\{\left(v_i, r_{i,j}, v_j\right)\right\}$
	$\widehat{A} = M\ (p, Q, A^q)$

Input: a user query Q
Output: the final answer A

15	External RAG Model
16	$Q_k = \left(w_{k,1}, w_{k,2}, \ldots w_{k,j}\right)$ $D_{external} = \emptyset$ **For** each keyword $w_{k,i}$ in Q_k: **For** each entity E_j with label L_j in the Wiki KG: if $Sim(w_{k,i}, L_j) \geq \theta^{FuzzzyWuzzy}$: $D_{external} = D_{external} \cup^{E_j}$ $A = M\ (p, Q, D_{external})$

5.2. Environment settings and simulation results

Table 4 provides the environment settings that form the foundation for the proposed mechanism, detailing the computational environment and software stack. The environment includes an Intel i7-12700 CPU and an NVIDIA A100 GPU with 40GB memory, supported by Python 3.8.18. Key libraries and frameworks such as

Torch 2.1.2 for deep learning, Transformers 4.15.0 for natural language processing, and Numpy 1.24.4 for numerical computations ensure a robust implementation. Additionally, NetworkX 2.7.1 and DGL 0.8 handle graph processing.

Building on this computational foundation, Table 5 highlights the accuracy improvements achieved by integrating Wiki information into three feature aggregation methods, including HFA (Horizontal Feature Aggregation), VFA (Vertical Feature Aggregation), and CHVFA (Combined Horizontal and Vertical Feature Aggregation). The results show consistent performance enhancements across all methods, with CHVFA achieving the highest accuracy gain. These findings demonstrate how Wiki data provides additional contextual knowledge, which enriches the feature representations and enables the model to capture more nuanced patterns and relationships. This evaluation of how Wiki data enhances feature aggregation methods naturally leads to examining its impact on other aspects, like language modelling. Figure 6 explores this further by analysing BLEU scores across different n-gram levels, providing more detailed insights into how Wiki data improves semantic and contextual understanding.

Figure 6 compares the performance of the proposed Hi-KG (hierarchical KG) and Hi-KG+Wiki using BLEU scores across different n-gram levels (1-gram to 5-gram). The results clearly show that Hi-KG+Wiki consistently outperforms Hi-KG at all levels, with the performance gap widening as the n-gram level increases. This indicates that the integration of Wiki data significantly enhances the model's semantic and contextual understanding. By providing richer information, Wiki data enables the model to better capture long-range dependencies and subtle language patterns, particularly at higher n-gram levels.

Table 6 compares the performance of proposed QA mechanisms against existing studies, including KV-Mem, GraftNet, PullNet, EmbedKGQA, NSM, TransferNet, SR+NSM, SR+NSM+E2E, UniKGQA across two datasets: WebQSP and CWQ. The evaluation metrics include Hit@1, which measures the accuracy of the top-ranked answers, and F1-Score, which evaluates the balance between precision and recall. Earlier methods, such as KV-Mem, show relatively low performance, while models like TransferNet and UniKGQA demonstrate significant improvements. The proposed KG-based QA system further outperforms all other approaches, achieving a Hit@1 of 80.4% and an F1-Score of 72.3% on WebQSP, as well as a Hit@1 of 55.1% and an F1-Score of 54.3% on CWQ. This occurs because the proposed mechanisms effectively combines KG embeddings, hierarchical reasoning, and retrieval-augmented mechanisms. This model integration strengthens the system's ability to interpret and resolve complex queries with greater precision and reliability.

Figure 7 illustrates how the loss changes for two models (GCN and LLM) during training and evaluation. The horizontal axis represents the number of training epochs (up to 200), and the vertical axis indicates the loss values. In the first 50 epochs, both models experience a rapid reduction in their training and evaluation loss. Afterwards, the losses of GCN gradually stabilise, with small changes between epochs.

The proposed GCN model consistently performs better than the LLM model. Its training and evaluation losses stay low and stable throughout, demonstrating strong generalisation and training stability. On the other hand, while the LLM model also reduces its losses, its evaluation loss remains higher than its training loss in later stages, indicating a potential overfitting to the training data. The better performance of the GCN

Table 4. Environment settings.

Environment	Specification
CPU	i7-12700
GPU	NVIDIA A100 GPU (40 G)
Python	3.8.18
Torch	2.1.2
Numpy	1.24.4
NetworkX	2.7.1
DGL	0.8

Table 5. Effect of Wiki integration on aggregation methods.

Method	Accuracy	Accuracy (+Wiki)
HFA	0.69	0.75
VFA	0.72	0.78
CHVFA	0.78	0.84

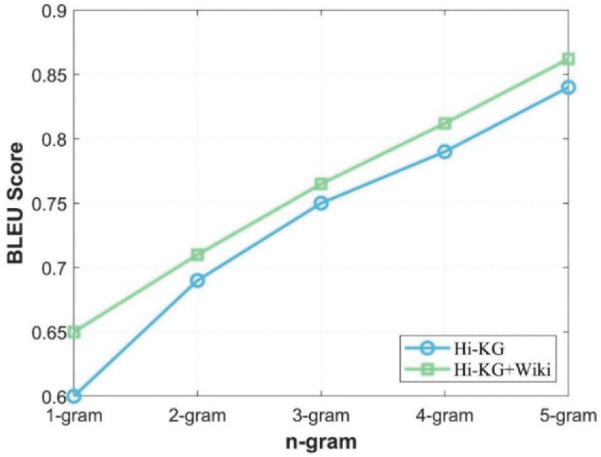


Figure 6. BLEU scores for different *n*-gram.

model is likely due to its use of three-layer KG and the training by applying the GCN to capture more complex patterns in the data.

Figure 8 evaluates the performance of both models across three categories (FAQ, Course-related, and Campus Life) at various data ratios (0.2 to 1). As the data ratio increases, both models perform better, with larger datasets contributing to improve outcomes. The GCN model achieves superior results across all metrics and categories, especially when data is limited. This highlighted its capacity to learn effectively from restricted datasets and to capture complex patterns across diverse categories. While the LLM also improves with more data, its performance levels off, showing challenges in handling complex relationships as well as GCN.

Figure 9 focuses on how embedding lengths (128 to 768) affect the accuracy of the same three tasks: FAQ, Course-related, and Campus Life. The results show a consistent trend: accuracy improves as the embedding length increases. This is because longer embedding vectors can better capture complex contextual and semantic information, which is essential for understanding and responding to queries effectively. The figure also reveals that shorter embeddings may work for simpler tasks, but tasks needing deeper understanding improve much more with longer embeddings. For example, FAQ tasks improve steadily, while Course-related and Campus Life tasks see bigger gains as embedding lengths increase. This is because these tasks depend more on rich context and detailed relationships. The results suggest that choosing the right embedding length depends on how complex the task is.

Figure 10 presents the Friedman test results from the ablation study, which compares the $\mathcal{F}1^{\mathcal{M}}$ performance of different mechanisms. The box plots reveal notable differences in the $\mathcal{F}1^{\mathcal{M}}$ distributions across the mechanisms. The proposed GCN exhibits a higher median $\mathcal{F}1^{\mathcal{M}}$ and a more consistent distribution, whereas the LLM mechanism shows relatively lower $\mathcal{F}1^{\mathcal{M}}$. The Friedman test results (chi-square value: 5.00, p-value: 0.02535) confirm that these differences are statistically

Table 6. QA mechanisms performance on WebQSP and CWQ datasets.

	WebQSP		CWQ	
Mechanism	Hit@1	$\mathcal{F}1^{\mathcal{M}}$	Hit@1	$\mathcal{F}1^{\mathcal{M}}$
KV-Mem (Miller et al. 2016)	46.7	34.5	18.4	15.7
GraftNet (Sun et al. 2018)	66.4	60.4	36.8	32.7
PullNet (Sun, Bedrax-Weiss, and Cohen 2019)	68.1	–	45.9	–
EmbedKGQA (Saxena, Tripathi, and Talukdar 2020)	66.6	–	–	–
NSM (He et al. 2021)	68.7	62.8	47.6	42.4
TransferNet (Shi et al. 2021)	71.4	–	48.6	–
SR+NSM (Zhang et al. 2022)	68.9	64.1	50.2	47.1
SR+NSM+E2E (Zhang et al. 2022)	69.5	64.1	49.3	46.3
UniKGQA (Jiang et al. 2022)	75.1	70.2	50.7	48.0
The proposed QA	80.4	72.3	55.1	54.3

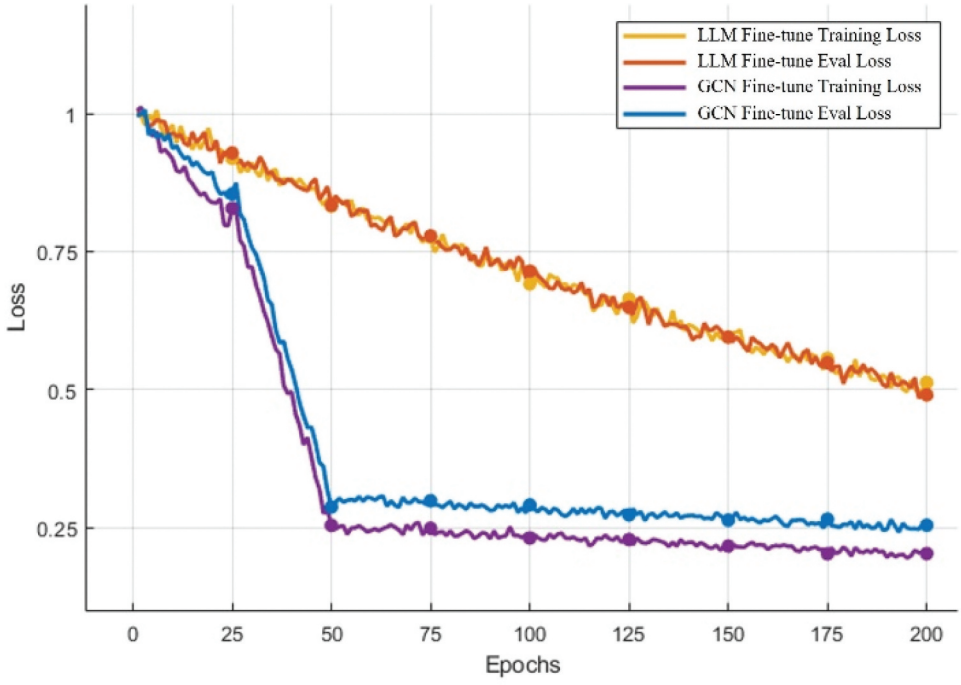
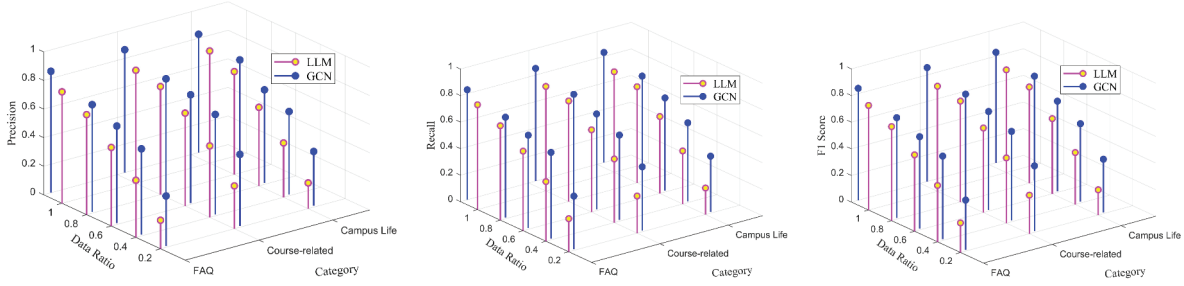


Figure 7. Comparison of training and evaluation loss trends for GCN and LLM models during fine-tuning.



(a) Precision (b) Recall (c) F1-Score

Figure 8. Performance comparison of GCN and LLM models at different data ratios.

significant. These findings support the superior effectiveness of the proposed GCN in enhancing $\mathcal{F}1^{\mathcal{M}}$.

6. Conclusion with further discussion

This study proposed a novel QA system that uses a hierarchical KG integrated with RAG to enhance retrieval and reasoning process. The hierarchical KG organises information across title, subtitle, and content layers, allowing for effective semantic representation and retrieval. Two key challenges were addressed in this system. First, the approach improves retrieval relevance and contextual accuracy in QA systems by employing GCNs for horizontal and vertical feature aggregation across layers. This method leverages both local and global relationships within the KG. Second, the RAG mechanism uses external sources, including Wiki, to address gaps in internal knowledge and deliver comprehensive and accurate responses. Experimental results show that the proposed system outperformed existing approaches, with significant gains in precision, recall, and F1-score.

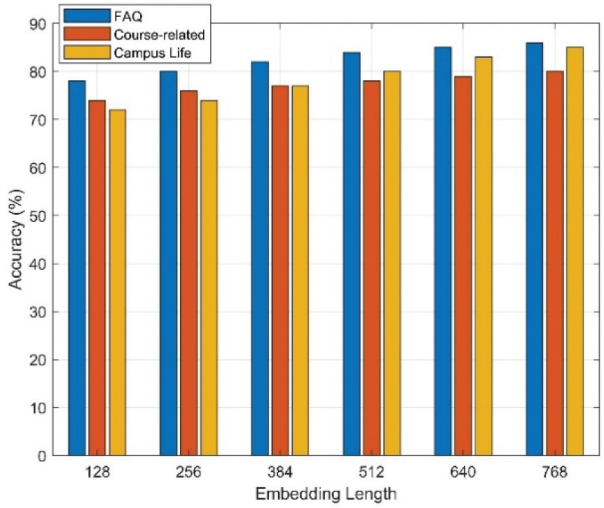


Figure 9. Impact of embedding length on task accuracy across contextual complexity.

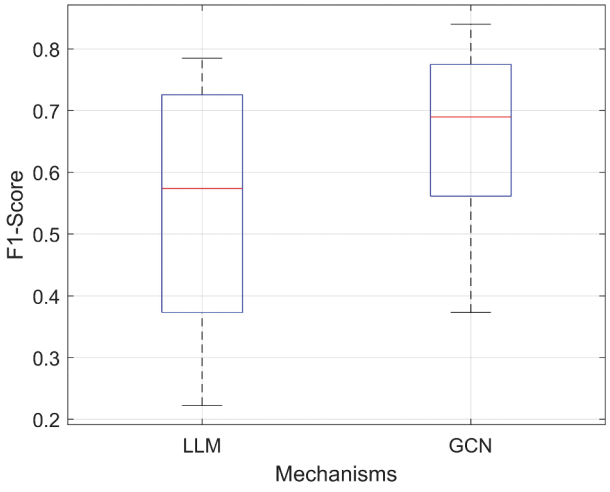


Figure 10. Ablation study of F1-score distributions for LLM and GCN mechanisms.

Future work will explore dynamic KG and adaptive learning mechanisms to further improve performance in real-world applications. Incorporating more advanced entity alignment methods will also enhance the RAG mechanism's ability to merge information from diverse external sources. Moreover, while this study adopts empirically determined fixed thresholds for node relevance, future research will investigate adaptive thresholding strategies to enhance transferability across domains and reduce manual adjustment.

In addition, to improve explainability, the reasoning paths used during multi-hop retrieval, which are currently recorded for internal validation, will be made visible in future versions. This improvement is expected to help users understand how answers are generated, which is especially important in domains such as education and law where transparency is essential.

Beyond explainability, the long-term sustainability of QA systems critically depends on effective knowledge maintenance, which plays a crucial role in ensuring accuracy and relevance over time. In the broader context, knowledge processing encompasses multiple stages, including data collection, validation, storage, updating, modification, and application (Evans, Dalkir, and Bidian 2014). The present system primarily focuses on the application stage, where pre-built hierarchical KGs and RAG mechanisms are applied to deliver accurate and contextually relevant answers. As the datasets used for system evaluation are campus-related and therefore subject to relatively slow content changes compared to fast-evolving domains such as

news or finance, correctness and reliability are prioritised over real-time updates. Nevertheless, future work will also consider integrating dynamic KG update mechanisms, recognising that sustained knowledge maintenance is essential for improving adaptability in more rapidly changing environments.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

The author(s) reported there is no funding associated with the work featured in this article.

Author contributions

Wan-Chi Yang: She led the development of the research framework and took charge of designing the hierarchical knowledge graph-based QA system. Her work included conducting the experiments, handling data preprocessing, constructing KG structure, and preparing the manuscript.

Xuan Li: He contributed to the implementation of the RAG model and analysed experimental results.

Chih-Yung Chang: He guided overall research direction and algorithm development. He supervised the project, ensured the quality of the manuscript, and managed all correspondence with the journal.

Diptendu Sinha Roy: He provided expertise in graph convolutional networks. He helped refine the training methodology and offered a thorough review of the study's technical aspects.

References

ANSI/IEEE. 1983. *IEEE Standard Glossary of Software Engineering Terminology*. IEEE.

Bordes, A., N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. 2013. "Translating Embeddings for Modeling Multi-Relational Data." In *Advances in Neural Information Processing Systems 26*, edited by C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, 2787–2795. Red Hook, NY: Curran Associates, Inc.

De Nicola, A., A. Formica, I. Mele, M. Missikoff, and F. Taglino. 2024. "A Comparative Study of LLMs and NLP Approaches for Supporting Business Process Analysis." *Enterprise Information Systems* 18 (10): 2415578. https://doi.org/10.1080/17517575.2024.2415578.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. 2019. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, edited by J. Burstein, C. Doran, and T. Solorio, 4171–4186. Stroudsburg, PA: Association for Computational Linguistics.

Evans, M., K. Dalkir, and C. Bidian. 2014. "A Holistic View of the Knowledge Life Cycle: The Knowledge Management Cycle (KMC) Model." *Electronic Journal of Knowledge Management* 12 (2): 85–97. http://www.ejkm.com .

Fang, Y., J. Deng, F. Zhang, and H. Wang. 2023. "An Intelligent Question-Answering Model over Educational Knowledge Graph for Sustainable Urban Living." *Sustainability* 15 (2): 1139. https://doi.org/10.3390/su15021139.

Guo, Q., X. Wang, Z. Zhu, P. Liu, and L. Xu. 2023. "A Knowledge Inference Model for Question Answering on an Incomplete Knowledge Graph." *Applied Intelligence* 53 (7): 7634–7646. https://doi.org/10.1007/s10489-022-03927-0.

Guu, K., K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang. 2020. "Realm: Retrieval-Augmented Language Model Pre-Training." In *Proceedings of the 37th International Conference on Machine Learning*, edited by H. Daumé III and A. Singh, Proceedings of Machine Learning Research (PMLR), 119, 3929–3938.

Han, H., J. Wang, and X. Wang. 2024. "Leveraging Knowledge Graph Reasoning in a Multihop Question Answering System for Hot Rolling Line Fault Diagnosis." *IEEE Transactions on Instrumentation and Measurement* 73:1–14. https://doi.org/10.1109/TIM.2023.3341130.

Hao, Y., H. Yu, and J. You. 2025. "Beyond Facts: Evaluating Intent Hallucination in Large Language Models." Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vienna, Austria: 7046–7069. Association for Computational Linguistics. https://doi.org/10.18653/v1/2025.acl-long.349.

He, G., Y. Lan, J. Jiang, W.-X. Zhao, and J.-R. Wen. 2021. "Improving Multi-Hop Knowledge Base Question Answering by Learning Intermediate Supervision Signals." In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, edited by L. Lewin-Eytan, D. Carmel, E. Yom-Tov, E. Agichtein, and E. Gabrilovich, 553–561. New York, NY, USA: Association for Computing Machinery (ACM). https://doi.org/10.1145/3437963.3441753.

Hui, K., A. Yates, K. Berberich, and G. de Melo. 2017. "PACRR: A Position-Aware Neural IR Model for Relevance Matching." In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, edited by M. Palmer, R. Hwa, and S. Riedel, 1049–1058. Copenhagen, Denmark: Association for Computational Linguistics.

Ji, Z., N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, D. Chen, W. Dai, H. S. Chan, A. Madotto, and P. Fung.. 2023. "Survey of Hallucination in Natural Language Generation." *ACM Computing Surveys* 55 (12): Article 248. 1–38. https://doi.org/10.1145/3571730.

Jiang, J., K. Zhou, W.-X. Zhao, and J.-R. Wen. 2022. "UniKGQA: Unified Retrieval and Reasoning for Solving Multi-Hop Question Answering Over Knowledge Graph." *ArXiv Preprint*. arXiv:2212.00959. https://doi.org/10.48550/arXiv.2212.00959.

Khushhal, S., A. Majid, S. Abbas, M. A. Nadeem, and S. A. N. 2020. "Question Retrieval Using Combined Queries in Community Question Answering." *Journal of Intelligent Information Systems* 55 (2): 307–327. https://doi.org/10.1007/s10844-020-00612-x.

Lai, C., and S. Qiu. 2025. "MKER: Multi-Modal Knowledge Extraction and Reasoning for Future Event Prediction." *Complex & Intelligent Systems* 11:138. https://doi.org/10.1007/s40747-024-01741-4.

Lang, Q., X. Liu, and W. Jia. 2023. "AFS Graph: Multidimensional Axiomatic Fuzzy Set Knowledge Graph for Open-Domain Question Answering." *IEEE Transactions on Neural Networks and Learning Systems* 34 (12): 10904–10918. https://doi.org/10.1109/TNNLS.2022.3171677.

Lv, S., D. Guo, J. Xu, D. Tang, N. Duan, M. Gong, L. Shou, D. Jiang, G. Cao, and S. Hu. 2020. "Graph-Based Reasoning Over Heterogeneous External Knowledge for Commonsense Question Answering." In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, edited by V. Conitzer and F. Sha, 8449–8456. New York, NY: AAAI Press.

Miller, A., A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston. 2016. "Key-Value Memory Networks for Directly Reading Documents." In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, edited by J. Su, K. Duh, and X. Carreras, 1400–1409. Austin, TX: Association for Computational Linguistics.

Modi, S., Y. Lin, L. Cheng, G. Yang, L. Liu, and W. J. Zhang. 2011. "A Socially Inspired Framework for Human State Inference Using Expert Opinion Integration." *IEEE/ASME Transactions on Mechatronics* 16 (5): 874–878. https://doi.org/10.1109/TMECH.2011.2161094.

Mohamed, S., H. A. A. Nomer, R. Yousri, A. W. Mohamed, A. Soltan, and M.S. Darweesh. 2023. "Energy Management for Wearable Medical Devices Based on Gaining–Sharing Knowledge Algorithm." *Complex & Intelligent Systems* 9:6797–6811. https://doi.org/10.1007/s40747-023-01101-8.

Noraset, T., L. Lowphansirikul, and S. Tuarob. 2021. "WabiQA: A Wikipedia-Based Thai Question-Answering System." *Information Processing & Management* 58 (1): 102431. https://doi.org/10.1016/j.ipm.2020.102431.

Ram, O., Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham. 2023. "In-Context Retrieval-Augmented Language Models." *Transactions of the Association for Computational Linguistics* 11:1316–1331. https://doi.org/10.1162/tacl_a_00605.

Ren, J., F. Xia, X. Chen, J. Liu, M. Hou, A. Shehzad, N. Sultanova, and X. Kong. 2021. "Matching Algorithms: Fundamentals, Applications and Challenges." *IEEE Transactions on Emerging Topics in Computational Intelligence* 5 (3): 332–350. https://doi.org/10.1109/TETCI.2021.3067655.

Rubin, O., and J. Berant. 2023. "Long-Range Language Modeling with Self-Retrieval." *ArXiv Preprint* arXiv:2306.13421. 12:1197–1213. https://doi.org/10.1162/tacl_a_00693.

Saxena, A., A. Tripathi, and P. Talukdar. 2020. "Improving Multi-Hop Question Answering Over Knowledge Graphs Using Knowledge Base Embeddings." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, edited by D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, 4498–4507. Stroudsburg, PA: Association for Computational Linguistics.

Shi, J., S. Cao, L. Hou, J. Li, and H. Zhang. 2021. "Transfernet: An Effective and Transparent Framework for Multi-Hop Question Answering over Relation Graph." In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021), edited by M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, 4149–4158. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. *ArXiv Preprint* arXiv:2104.07302.

Simhi, A., I. Itzhak, F. Barez, G. Stanovsky, and Y. Belinkov. 2025. "Trust Me, I'm Wrong: High-Certainty Hallucinations in LLMs." *ArXiv Preprint* arXiv:2502.12964. https://arxiv.org/abs/2502.12964.

Song, J., X. Wang, J. Zhu, Y. Wu, X. Cheng, R. Zhong, and C. Niu. 2024. "RAG-HAT: A Hallucination-Aware Tuning Pipeline for LLM in Retrieval-Augmented Generation." In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, edited by F. Dernoncourt, D. Preotiuc-Pietro, and A. Shimorina, 1548–1558. Miami, FL: Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.emnlp-industry.113.

Sorokin, D., and I. Gurevych. 2018. "Modeling Semantics with Gated Graph Neural Networks for Knowledge Base Question Answering." In *Proceedings of the 27th International Conference on Computational Linguistics*, edited by E. M. Bender, L. Derczynski, and P. Isabelle, 3306–3317. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Sun, H., T. Bedrax-Weiss, and W. W. Cohen. 2019. "Pullnet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text." In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, edited by K. Inui, J. Jiang, V. Ng, and X. Wan, 2380–2390. Hong Kong, China: Association for Computational Linguistics.

Sun, H., B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen. 2018. "Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text." In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, edited by E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, 4231–4242. Brussels, Belgium: Association for Computational Linguistics.

Sun, Y., Q. Shi, L. Qi, and Y. Zhang. 2022. "jointlk: Joint Reasoning with Language Models and Knowledge Graphs for Commonsense Question Answering." Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2022), edited by M. Carpuat, M.-C. de Marneffe, and I. V. M. Ruiz. Seattle, United States: 5049–5060. Association for Computational Linguistics. 10.18653/v1/2022.naacl-main.372.

Van Woensel, W., and S. Motie. 2024. "NLP4PBM: A Systematic Review on Process Extraction Using Natural Language Processing with Rule-Based, Machine and Deep Learning Methods." *Enterprise Information Systems* 18 (11): 2417404. https://doi.org/10.1080/17517575.2024.2417404.

Yu, H. Y., A. Ogbeyemi, W. J. Lin, J. He, W. Sun, and W. J. Zhang. 2021. "A Semantic Model for Enterprise Application Integration in the Era of Data Explosion and Globalisation." *Enterprise Information Systems* 17 (4). https://doi.org/10.1080/17517575.2021.1989495.

Zafar, A., S. K. Sahoo, H. Bhardawaj, A. Das, and A. Ekbal. 2024. "KI-MAG: A Knowledge-Infused Abstractive Question Answering System in Medical Domain." *Neurocomputing*. https://doi.org/10.1016/j.neucom.2023.127141.

Zhang, J., Z. Pei, W. Xiong, and Z. Luo. 2020. "Answer Extraction with Graph Attention Network for Knowledge Graph Question Answering." Proceedings of the 2020 IEEE 6th International Conference on Computer and Communications (ICCC), Chengdu, China: 1645–1650. IEEE. https://doi.org/10.1109/ICCC51575.2020.

Zhang, J., X. Zhang, J. Yu, J. Tang, C. Tang, C. Li, and H. Chen. 2022. "Subgraph Retrieval Enhanced Model for Multi-Hop Knowledge Base Question Answering." Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), edited by S. Muresan, P. Nakov, and A. Villavicencio. Dublin, Ireland: 5773–5784. Association for Computational Linguistics. 10.18653/v1/2022.acl-long.396.

Zhang, M., T. He, and M. Dong. 2024. "Meta-Path Reasoning of Knowledge Graph for Commonsense Question Answering." *Frontiers of Computer Science* 18 (1): 181303. https://doi.org/10.1007/s11704-022-2336-6.

Zhang, W. 1994. "*An Integrated Environment for CAD/CAM of Mechanical Systems*. Doctoral dissertation, Delft, The Netherlands: Delft University of Technology.

Zhang, W., and J. Zhang. 2025. "Hallucination Mitigation for Retrieval-Augmented Large Language Models: A Review." *Mathematics* 13 (5): 856. https://doi.org/10.3390/math13050856.