Integrating Knowledge Graph with Large Language Models for Multi-hop Question Generation

Yllias Chali, Al Hasib Mahamud
University of Lethbridge
Lethbridge, Alberta, Canada
{yllias.chali, alhasib.mahamud}@uleth.ca

Abstract

Question generation (QG) is a fundamental task in natural language processing that involves generating fluent and grammatically correct questions from a given input context, optionally conditioned on an answer. Multi-hop question generation (MHQG), a more complex variant, requires reasoning over multiple pieces of information across diverse contexts to formulate coherent questions. In this work, we propose Knowledge Graph for Question Generation (KG4QG), a novel framework that integrates knowledge graphs with large language models to address the challenges of MHQG. Our approach constructs knowledge graphs from input contexts, encodes them using Graph Attention Networks (GAT), and leverages Sentence Transformers for contextual text embeddings. These enriched representations are then fed into large language models—specifically BART and T5—for multi-hop question generation. We evaluate KG4QG on the HotpotQA dataset, demonstrating that our method achieves superior performance compared to existing state-of-the-art approaches, highlighting the effectiveness of combining structured knowledge and pre-trained language models for complex question generation tasks.

Keywords: Multihop Question Generation, Large Language Models, Language Reasoning

1. Introduction

Question Generation (QG) is the task of automatically generating human-like questions from an input source such as a passage, answer, paragraph, context, or statement (Zhang et al., 2021). Within the field of Artificial Intelligence (AI), QG is considered an important research area and has been gaining increasing attention in both academia and industry (Zhang et al., 2021). Although QG has been extensively studied in recent years, most existing research focuses on generating single-hop questions from a single paragraph (Emerson and Chali, 2023a) , where answering the question requires minimal reasoning. For example, Min et al. (2018) found that approximately 90% of questions in the SQuAD dataset (Rajpurkar et al., 2016) could be generated from a single sentence within the input paragraph (Emerson and Chali, 2023a). In contrast, multi-hop question generation (MHQG) involves generating complex questions that require aggregating and reasoning over information from multiple interconnected sources—this process is referred to as multi-hop reasoning. MHQG combines evidence across multiple paragraphs and reasons over them to generate meaningful, answer-related, and factually coherent questions (Su et al., 2020). In single-hop QG, the generated question typically depends on only one passage or a single sentence, requiring only simple reasoning. However, multi-hop QG is inherently more complex, as it requires reasoning across two or more paragraphs—or even across different documents (Lin et al., 2024). To generate a multi-hop question, one must read all relevant passages, extract and understand facts from different sections, identify relationships among these facts, and integrate them into a single coherent question. Interest in MHQG is growing rapidly due to its wide range of applications, including educational systems (Heilman and Smith, 2010; Lindberg et al., 2013; Yao et al., 2018; Zamani et al., 2020; Su et al., 2020), medical and clinical tools (Rao et al., 2022; Du et al., 2022), chatbot systems (Mavi et al., 2024),

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 7867–7882
11–16 May 2026.

and other natural language generation (NLG) tasks such as question answering (Tang et al., 2017; Zhang et al., 2024). In this work, we propose a novel framework, KG4QG (Knowledge Graph for Question Generation), which leverages knowledge graphs generated from input texts to improve the quality of multi-hop question generation. Our major contributions are summarized as follows:

- Combining Knowledge Graphs with Large Language Models (LLMs): While typical LLM architectures use input text directly, our approach concatenates knowledge graph embeddings with input text embeddings and feeds this combined representation into the LLMs, enriching the input information.
- Integration of SBERT and GAT: We integrate Sentence-BERT (SBERT[1]) for input text embeddings and Graph Attention Network (GAT) for knowledge graph embeddings. This hybrid embedding strategy effectively combines structured graph data with unstructured text, significantly enhancing the LLMs' capacity for multi-hop question generation.
- Triplet Dataset Construction: To support our framework, we introduce a novel triplet dataset that includes a knowledge graph for each input text. The dataset contains entities and the predicates (relationships) between entity pairs, providing a structured representation of the input context.
- Performance Enhancement in Multi-hop Question Generation: We evaluate our KG4QG model on the HotpotQA[2] dataset. Experimental results demonstrate a significant improvement in the quality and relevance of the generated multi-hop questions, as well as better performance across evaluation metrics compared to existing state-of-the-art methods.

[1] https://sbert.net/

[2] https://huggingface.co/datasets/hotpotqa/hotpot_qa

Context 1:

The Con Mine (1938-2003) was the first gold mine developed in the Northwest Territories, Canada, just south of Yellowknife. The property was staked by Consolidated Mining and Smelting Company of Canada (Cominco) in September 1935 in response to the discovery of visible gold nearby; the name "Con" is an abbreviation of "Consolidated". The advent of winter prevented any prospecting from being conducted, but work in the summer of 1936 led to the discovery of numerous gold veins. The Con Mine entered production in 1938 and ceased operations in 2003. It has produced over 5000000 ozt of gold from 12,195,585 tons of ore processed. The mine was over 6000 ft deep.

Context 2:

The Giant Mine was a gold mine located on the Ingraham Trail, 5 km north of Yellowknife, Northwest Territories. Giant Mine is within the Kam Group, which is part of the Yellowknife greenstone belt. Gold was discovered on the property and mineral claims staked in 1935 by Johnny Baker, but the true extent of the gold deposits were not known until 1944 when a massive gold-bearing shear zone was uncovered beneath the drift-filled Baker Creek Valley.

Answer:

The Con Mine

Reference Question:

Which mine started production sooner, the Con Mine, or the Giant Mine?

Figure 1: Sample of a Comparison-Type Question. Context 1 provides the information that the Con Mine began production in 1938. Context 2 states that the true extent of the gold deposits at the Giant Mine was not known until 1944. To answer the reference question, "Which mine started production sooner, the Con Mine or the Giant Mine?", it is necessary to compare the information from both contexts. This data sample is taken from the HotpotQA dataset (Yang et al., 2018).

2. Related Works

2.1. Question Generation

Sentence-to-question transformation for factual question generation was first introduced by Mitkov (2006) (Blšták and Rozinajová, 2021). Early research in question generation relied on rule-based approaches, where handcrafted rules were used to convert sentences into questions (Heilman and Smith, 2010). Rus et al. (2010) made a significant contribution to question generation systems by generating questions from both sentences and paragraphs. Subsequently, neural-based approaches have had a major impact on QG, particularly with the introduction of sequence-to-sequence (seq2seq) models based on neural network architectures (Sutskever et al., 2014). Du et al. (2017) were the first to apply an attention-based model for generating reading comprehension questions, marking a key ad-

vancement in neural question generation techniques.

2.2. Multi-hop Question Generation

Although multi-hop question generation (MHQG) is a relatively new area of research, notable advancements have been made in models such as MulQG (Su et al., 2020), CQG (Fei et al., 2022), GNET-QG (Jamshidi and Chali, 2025), and TASE-CoT (Lin et al., 2024). MulQG employs a sequence-to-sequence model that integrates bi-directional LSTM-RNN encoders with a Graph Convolutional Network. CQG presents an effective controlled framework based on a Graph Attention Network (GAT) combined with a pre-trained BERT model. GNET-QG further advances this approach by integrating GAT with sequence-to-sequence models such as BART and T5. Emerson and Chali (2023b) introduced a transformer-based method that generates questions without relying on sentence-level supporting fact information. Lin et al. (2024) proposed a type-aware semantics extraction-based chain-of-thought (CoT) prompting technique for few-shot multi-hop question generation. Despite these advancements, existing models still exhibit certain limitations. Addressing these gaps requires the development of more robust and comprehensive methodologies for multi-hop question generation systems.

3. Proposed Model

3.1. Motivation

The primary motivation of this research is to generate complex multi-hop questions. While sequence-to-sequence-based transformer models have demonstrated strong capabilities in generating simple questions—typically answerable using a single paragraph—it remains uncertain whether these models can effectively generate complex questions that require integrating information across multiple paragraphs. A multi-hop question is defined as one that requires multiple reasoning steps—or "hops"—to arrive at an answer. In such cases, the model must be capable of extracting relevant information from each paragraph, synthesizing that information, and generating questions that reflect complex reasoning across diverse sources. The ability to construct a coherent reasoning path is essential for accurate multi-hop question generation. One of the major challenges with large language models is the issue of hallucination, where models generate content that is not grounded in the input data. This can significantly impact model performance. Incorporating commonsense reasoning into the model is a promising strategy to mitigate hallucination. Therefore, another key motivation of this research is to enhance the model's ability to reason with commonsense knowledge, thereby improving factual consistency and reliability.

3.2. Problem Definition

We define the multi-hop question generation task using a conditional probability framework. Suppose for each answer, the input context C is defined as: $C = (C_1, C_2, ..., C_N)$ where N is the number of paragraphs in the context. The goal is to generate a question Q with the highest probability, where: $Q = (Q_1, Q_2, ..., Q_M)$, and M represents the number of tokens in the question. In this research, we focus on a simplified setting where the input context consists of two paragraphs, i.e., $C = (C_1, C_2)$, since answering the question requires reasoning over both. Based on this, the probability of generating the question Q given context C can be formulated as:

$$P(Q|C) = \Pi_{i=1}^{M} P(Q_i|[Q_1, Q_2, ..., Q_{i-1}], [C_1, C_2])$$

This formulation reflects the auto-regressive nature of the generation process, where each token Q_i is conditioned on the previously generated tokens and the input context.

3.3. Knowledge Graph

To create the knowledge graph, the input text is first converted into annotated text by per-

forming co-reference resolution. The knowledge graph is then generated using Stanford CoreNLP[3].

3.3.1. Coreference Resolution

In the input text, we perform coreference resolution to identify and link mentions of the same entity. Our primary objective is to construct a knowledge graph from each input text. Coreference resolution helps reduce node redundancy by linking repeated references to the same entity, ensuring that each distinct entity is represented by a single node in the graph. This results in a more accurate and coherent knowledge graph. For this purpose, we use the en_coreference_web_trf model from spaCy[4], which identifies and resolves coreferences by replacing pronouns or ambiguous mentions with their corresponding noun references.

3.3.2. Knowledge Graph

The main objective is to extract entities from the input texts, establish relationships between them, and ultimately generate a graph structure for each input. To create the knowledge graph, we first start the CoreNLP server. We utilize the Open Information Extraction (OpenIE)[5] annotator, which is capable of extracting triplets from the input text. A triplet consists of a subject, a relation, and an object that represents the relationship. Finally, the resulting graph is visualized using the Python NetworkX[6] package.

3.4. Graph Representation

After generating the knowledge graph using Stanford CoreNLP, we created a graph representation for each knowledge graph by constructing PyG[7] compatible tensors. The triplets are converted into the following attributes:

- Node Features: Node features are represented as a dictionary mapping where Each node (subject or object) in the graph is converted to an index number to transform human-readable node labels into machine-readable indices. The embedding for each node is generated using the Sentence Transformer model all-MiniLM-L6-v2[8]. This produces a PyTorch tensor of shape [number of nodes, 384], where 384 corresponds to the output dimensionality of the all-MiniLM-L6-v2 model.
- Edge Index: To represent connections between source and destination nodes, the edge index encodes the graph's connectivity. This is a PyTorch tensor[9] of shape [2, number of edges].
- Edge Attribute: Edge attributes represent embeddings for the relationships between subject and object nodes. These embeddings are also generated using the Sentence Transformer model all-MiniLM-L6-v2. The resulting PyTorch tensor has shape of [number of edges, 384], where 384 is the embedding dimension.

After generating these attributes from the triplets, each question is encoded using the all-MiniLM-L6-v2 model to produce vector representations of the questions. Finally, the graph object representation is created by constructing a Data object from the PyTorch Geometric (PyG)[10] library. This graph data object packages the node features, edge index, edge attributes, and the target variable, where the target variable in this research is the question.

3.5. Graph Embedding

The graph embedding is generated using a Graph Attention Network (GAT) model. First,

[3] https://stanfordnlp.github.io/CoreNLP/
[4] https://spacy.io/
[5] https://stanfordnlp.github.io/CoreNLP/openie.html
[6] https://networkx.org/
[7] https://pyg.org/
[8] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
[9] https://pytorch.org/docs/stable/tensors.html
[10] https://pytorch-geometric.readthedocs.io/en/latest/generated/torch_geometric.data.Data.html

the graph data object is loaded, after which the model is trained and evaluated on a test set derived from the data object. Figure 2 illustrates the five-layer GAT architecture used for generating graph embeddings from the knowledge graphs. The dataset is split into training, validation, and testing sets in a 70:15:15 ratio. During training, Mean Squared Error (MSE) is used as the loss function. Additionally, early stopping and model checkpointing techniques are applied to prevent overfitting and retain the best model. The model is trained for 200 epochs. The configuration details of the GAT model used for optimizing the graph object are presented in Table 1.

Parameters	Values
Optimizer	Adam Optimizer
Learning Rate	0.001
Weight Decay	$1 * 10^{-4}$
Epoch	200
Batch Size	32
Train, Validation, and Test Split	70:15:15

Table 1: GAT Model Parameter Specification

Figure 2 illustrates the architecture of the Graph Attention Network (GAT) model used for graph embedding. A brief explanation of each component is provided below:

- Input: The input to the GAT model is a graph object that contains node features, edge indices, and edge attributes.
- Graph Convolution Layers: The model consists of five graph convolution layers. In the first layer, each node has an input feature vector of size 384, corresponding to the output dimensionality of the Sentence Transformer. The output feature size per attention head is 256.
- Attention Mechanism: A multi-head attention mechanism is used, with 8 independent attention heads. This enables the model to learn diverse attention patterns across different subspaces of the feature vectors.
- Dropout: Dropout is applied after each graph convolution layer and before the normalization layer, with a dropout probability of 20% during training. This helps prevent overfitting.
- Normalization Layer: A normalization layer is applied before the dropout layer to stabilize training by normalizing the outputs of the convolution layers.
- ELU Activation: The Exponential Linear Unit (ELU) activation function is used after each convolution layer to introduce nonlinearity into the model.
- Global Mean Pooling: A global mean pooling layer is applied at the end of the network. After the final convolution layer generates the node embeddings, this pooling operation computes the average of the node embeddings for each graph in the batch.
- Output: The final output is a single vector representation for each graph. The size of this output tensor is [batch size × 384], where 384 corresponds to the final embedding dimension.

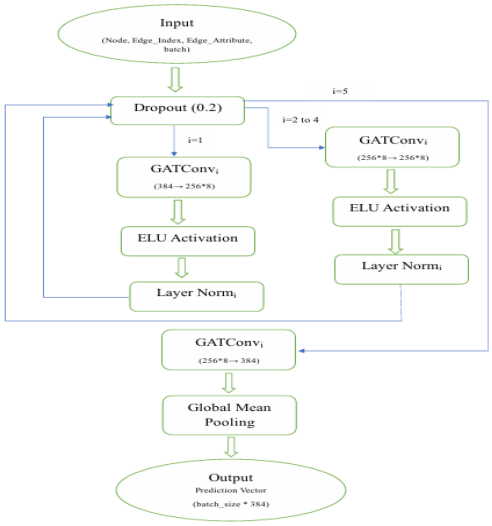

Figure 2: GAT Model Architecture for Graph Embedding

3.6. Text Embedding

The input text is converted into a numerical representation using a Sentence Transformer. Specifically, we utilize the all-MiniLM-L6-v2 model to generate the semantic vector (embedding) of the input text, resulting in an encoded representation suitable for downstream processing.

3.7. Combining Text Embedding and Graph Embedding

The graph embedding is concatenated with the input text embedding using the torch.stack[11] function. This function, from the PyTorch[12] library, combines the input text embedding and the graph embedding along a new dimension. In this research, both the input text embedding and the graph embedding have the same shape. We use torch.stack instead of torch.cat[13] because torch.stack is better suited for batching operations in deep learning, as it introduces an additional dimension, enabling more structured handling of multiple tensors in a batch.

3.8. Encoder

The encoder component of large language models (LLMs) is used to enhance the representation of input data by leveraging the transformer architecture, which includes feed-forward neural networks and a multi-head self-attention mechanism. In this research, we employ both BART-base and T5-base models as separate encoder backbones to evaluate which performs better. To ensure compatibility with the model's input requirements, we tokenize the input text and generate an attention mask, which guides the model in distinguishing between actual content and padding tokens. After generating the attention mask and preparing the tokenized inputs, the combined embedding layer—comprising the input text embedding and graph embedding—is passed into the encoder of the selected LLM (BART-base or T5-base).

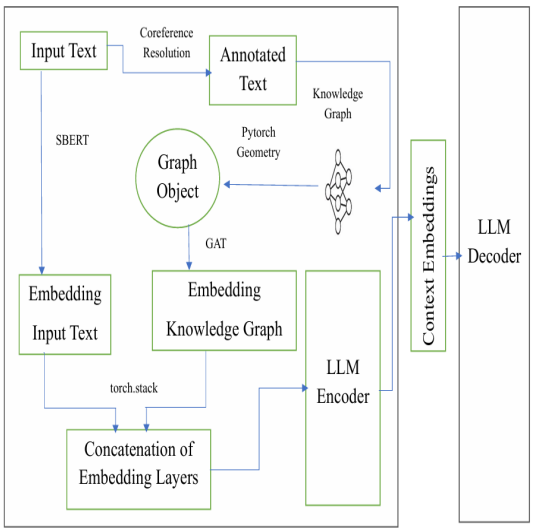

Figure 3: KG4QG Architecture

3.9. Decoder

The decoder of the large language model (LLM), which also follows the transformer architecture, is used for autoregressive generation—that is, it generates questions token by token based on previously predicted tokens. It outputs the probability distribution over the vocabulary to predict the next token in the sequence. The decoder employs masked self-attention, which ensures that the model only attends to previously generated tokens and the current token, thereby preserving the autoregressive property. Figure 3 illustrates the complete architecture of KG4QG, which includes both the encoder and decoder components. The encoder processes the input and generates context embeddings, which are then passed to the decoder to guide the generation of the final question.

[11] https://pytorch.org/docs/stable/generated/torch.stack.html

[12] https://pytorch.org/

[13] https://pytorch.org/docs/stable/generated/torch.cat.html

4. Experiments

4.1. Dataset

In this work, we utilize the HotpotQA dataset (Yang et al., 2018) for training and evaluating our model. The dataset consists of approximately 13,000 samples collected from Wikipedia articles. It was originally designed to train question-answering (QA) systems to perform complex reasoning and provide explanations for answers. Each sample in the HotpotQA dataset includes 10 paragraphs, of which only two are supporting paragraphs that contain the actual evidence needed to answer the question. The remaining eight are distractor paragraphs, which are irrelevant and are therefore discarded in our preprocessing. Additionally, we exclude supporting sentence annotations and remove all samples with yes/no answers. As noted by Su et al. (2020), such binary-answer questions tend to require less reasoning and do not align with the goal of evaluating multi-hop reasoning capabilities. By filtering out these samples, we ensure that our dataset focuses on complex multi-hop question generation. After preprocessing, we partition the dataset into three subsets: training, validation, and testing. The training set comprises 70% of the data, while the validation and testing sets each consist of 15%. Prior to splitting, we randomly shuffle the dataset to ensure that all three subsets are balanced with respect to the types of questions they contain.

4.2. Fine-tuning

To adapt the pre-trained models for the task of multi-hop question generation, we perform fine-tuning using both input texts and their corresponding knowledge graphs. For each input text, a knowledge graph is generated, and the text embedding is concatenated with the graph embedding to form the input for fine-tuning. This process is carried out using the HotpotQA dataset. Both BART-base and T5-base models are fine-tuned for 50 epochs. Early stopping is applied as a regularization technique to prevent overfitting. Specifically, the patience parameter is set to 3, meaning that if no improvement is observed in the validation loss for three consecutive epochs, training is halted. The best performance for the T5-base model is achieved after 50 epochs, while the best outcome for the BART-base model is obtained after 20 epochs. We use the Adam optimizer (Kingma and Ba, 2017) with a learning rate of 10^{-4}. The Cross-Entropy Loss function (Mao et al., 2023) is employed to calculate the training loss. The full training configuration is summarized in Table 2.

Model Name	Parametrs	Value
BART-base	Optimizer	AdamW
	Learning rate	1e-4
	Loss function	Cross-entropy
	Weight decay	0.01
	Epoch	20
	Early stopping	3
	Batch size	8
	Data split	0.70 : 0.15 :0.15
T5 -base	Optimizer	AdamW
	Learning rate	1e-4
	Loss function	Cross-entropy
	Weight decay	0.01
	Epoch	50
	Early stopping	3
	Batch size	8
	Data split	0.70 : 0.15 :0.15

Table 2: Parameter Specification to fine-tune KG4QG Model

4.3. Execution Details

All implementation is done in the Python programming language using PyTorch (Paszke et al., 2019). The BART-base[14] model is used in this work. Developed by Facebook AI, BART-base is a pre-trained model that combines a bidirectional encoder (similar to BERT) with an autoregressive decoder (similar to GPT). This model is not pre-trained for any specific downstream task and can be applied to various text generation tasks such as summarization, machine translation, and question answering. The model has approximately 139 million parameters, and its maximum input sequence

[14] https://huggingface.co/facebook/bart-base

length is 1024 tokens. We have also used the T5-base[15] model in this work. T5-base is the base-sized version of the T5 model developed by Google (Raffel et al., 2023). It is pre-trained on a span corruption task (Ye et al., 2024) using the C4 dataset (Dodge et al., 2021). This model contains approximately 220 million parameters and supports a maximum sequence length of 512 tokens.

4.4. Experimental Setup

The main computational resources for this research are servers that provide A100 GPUs, and V100 GPUs. We utilized the Distributed Data Parallel (DDP)[16] feature from PyTorch to leverage 4 GPUs simultaneously. Details, code, and implementations are available in our GitHub[17] repository.

5. Automatic Evaluations

To evaluate the performance of KG4QG, we utilized standard automated evaluation metrics. The generated questions from our model were compared against the reference questions in the test dataset using BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), and METEOR (Lavie and Agarwal, 2007). These metrics were selected due to their widespread adoption in the question generation literature. Evaluating our model with these metrics allows for a quantitative comparison of the effectiveness of KG4QG against prior work in multi-hop question generation. To demonstrate the efficacy of KG4QG with both BART and T5 backbones, we compared it with several existing models, including: MulQG by Su et al. (2020), CQG by Fei et al. (2022), Transformer Based approach by Emerson and Chali (2023a), GNET-QG by Jamshidi and Chali (2025), TASE-CoT by Lin et al. (2024), DCQG by Cheng et al. (2021), KGEL by Cao et al. (2023) and MultiFactor by Xia et al. (2023).

[15] https://huggingface.co/google-t5/t5-base

[16] https://docs.alliancecan.ca/wiki/PyTorch

[17] https://github.com/AlHasibMahamud/KG4QG_MultihopQG

Among these models, MulQG, GNET-QG, DCQG, and KGEL are graph-based approaches, making them particularly relevant for comparison. Notably, KGEL also incorporates knowledge graphs, similar to our model. We also included CQG, TASE-CoT, and the Transformer-based approach due to their strong reported performance on standard evaluation metrics. As shown in Table 3, our proposed model KG4QG outperforms all existing models across the evaluation metrics, with two exceptions: a slight shortfall on BLEU-1, where the difference with MultiFactor is 0.79, and on METEOR, where the difference with GNET-QG is 3.79.

6. Human Evaluation

In addition to automatic evaluation, we conducted a human evaluation to perform a more comprehensive assessment, as n-gram-based metrics alone may not fully capture the quality of generated questions (Fei et al., 2022). From the test set, we randomly selected 500 samples and asked five native English-speaking annotators to rate each generated question on a scale from 1 (very poor) to 5 (very good). Ratings were based on the following four criteria: Completeness: Whether the questions are fully formed, clear, and logically structured; Fluency: Whether the questions are grammatically correct and natural to read; Answerability: Whether the questions can be answered based on the provided input context; and Multi-hop Relevance: Whether the questions require reasoning across multiple pieces of information from different sources. For each sample, we averaged the scores from all five annotators. The spearman inter-anotator agreement was almost .6. The results, summarized in Table 4, compare KG4QG against the baseline systems BART and GNET-QG. Our model KG4QG achieved a 79% success rate in generating valid multi-hop questions, surpassing both baselines. Furthermore, KG4QG outperformed BART and GNET-QG across all human evaluation metrics, indicating higher-quality question generation. However, the human-generated questions still scored higher overall

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
MulQG	40.15	26.71	19.73	15.20	35.30	20.51
CQG	49.71	37.04	29.93	25.09	41.83	27.45
Transformer Based	42.13	30.44	23.84	19.42	39.26	22.78
GNET-QG	49.72	38.95	32.88	27.93	40.25	**49.87**
TASE-CoT	45.89	34.06	27.11	22.37	39.68	23.39
DCQG	-	-	21.07	15.26	-	19.99
KGEL	41.93	28.04	20.83	16.13	19.70	35.28
MultiFactor	**54.17**	41.50	33.74	28.22	28.60	44.17
KG4QG (T5 Backbone)	47.08	39.11	36.32	34.27	41.48	40.41
KG4QG (BART Backbone)	53.38	**44.81**	**41.24**	**38.56**	**47.11**	46.08

Table 3: Performance Comparison

Models	Completeness	Fluency	Answerability	Multi-hop Relevance
BART	3.96	3.86	3.97	54.0%
GNET-QG	4.14	3.94	4.18	76.0%
KG4QG	4.32	4.21	4.40	79.0%
Human	4.38	4.32	4.52	84.0%

Table 4: Human Evaluation Scores

than those produced by KG4QG, highlighting the remaining gap and the potential for future improvements in multi-hop question generation.

7. Conclusion

In this work, we proposed a novel approach for automatic multi-hop question generation to address the challenges inherent in generating questions that require reasoning across multiple pieces of information. We utilized the HotpotQA dataset, a widely recognized and extensively used benchmark in the research community for both multi-hop question generation and multi-hop question answering tasks. Our approach is based on the generation of knowledge graphs from the input text. We use a Graph Attention Network (GAT) for generating graph embeddings, which are then concatenated with input text embeddings. Both BART and T5 are employed as large language models, with BART demonstrating stronger performance overall. The integration of GAT, SBERT, and BART contributes significantly to the field of multi-hop question generation. Evaluation metrics indicate that our model KG4QG outperforms existing methods across various benchmarks. In the appendix, we present qualitative comparisons between the generated questions and reference questions, further illustrating the effectiveness of our approach. As multi-hop question generation is still a relatively new research area, there remains substantial room for improvement. Future research could explore low-resource multi-hop question generation, enhance cross-lingual and multilingual capabilities, and incorporate advanced prompting techniques such as Vanilla Prompting, Tree of Thoughts (ToT) prompting, and Graph of Thoughts (GoT) prompting, which may further improve the performance of large language models in this domain.

8. Acknowledgments

We would like to thank the anonymous reviewers for their useful comments. The research reported in this paper was conducted at the University of Lethbridge and supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada and the University of Lethbridge.

9. Bibliographical References

Miroslav Blšták and Viera Rozinajová. 2021. Automatic question generation based on sentence structure analysis using machine learning approach. *Natural Language Engineering*, 28(4):487–517.

Zhen Cao, Pengfei Li, Yong Zhong, and Shaobo Li. 2023. Multi-hop question generation with knowledge graph-enhanced language model. *Applied Sciences*, 13.

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. 2021. Guiding the growth: Difficulty-controllable question generation through step-by-step rewriting. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5968–5978, Online. Association for Computational Linguistics.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus.

Huifang Du, Zhongwen Le, Haofen Wang, Yunwen Chen, and Jing Yu. 2022. Cokg-qa: Multi-hop question answering over covid-19 knowledge graphs. *Data Intelligence*, 4(3):471–492.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

John Emerson and Yllias Chali. 2023a. Efficient multi-hop question generation. *Procedia Computer Science*, 222:217–222.

John Emerson and Yllias Chali. 2023b. Multi-hop question generation without supporting fact information. In *The International FLAIRS Conference Proceedings*, volume 36.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuanjing Huang. 2022. CQG: A simple and effective controlled generation framework for multi-hop question generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906, Dublin, Ireland. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Samin Jamshidi and Yllias Chali. 2025. GNET-QG: Graph network for multi-hop question generation. In *Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK)*, pages 20–26, Abu Dhabi, UAE. International Committee on Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612, Barcelona, Spain.

Zefeng Lin, Weidong Chen, Yan Song, and Yongdong Zhang. 2024. Prompting few-shot multi-hop question generation via comprehending type-aware semantics. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3730–3740, Mexico City, Mexico. Association for Computational Linguistics.

David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning online. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 105–114, Sofia, Bulgaria. Association for Computational Linguistics.

Julie Beth Lovins. 1968. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11(1-2):22–31.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. 2023. Cross-entropy loss functions: Theoretical analysis and applications.

Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. 2024. Multi-hop question answering.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Dattaraj J. Rao, Shraddha S. Mane, and Mukta A. Paliwal. 2022. Biomedical multi-hop question answering using knowledge graph embeddings and language models.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Christian Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics.

Dan Su, Yan Xu, Wenliang Dai, Ziwei Ji, Tiezheng Yu, and Pascale Fung. 2020. Multi-hop question generation with graph convolutional network. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4636–4647, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks.

Zehua Xia, Qi Gou, Bowen Yu, Haiyang Yu, Fei Huang, Yongbin Li, and Nguyen Cam-Tu. 2023. Improving question generation with multi-level content planning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 800–814, Singapore. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Kaichun Yao, Libo Zhang, Tiejian Luo, Lili Tao, and YanJun Wu. 2018. Teaching machines to ask questions. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 4546–4552. AAAI Press.

Ke Ye, Heinrich Jiang, Afshin Rostamizadeh, Ayan Chakrabarti, Giulia DeSalvo, Jean-François Kagy, Lazaros Karydas, Gui Citovsky, and Sanjiv Kumar. 2024. Spactor-t5: Pre-training t5 models with span corruption and replaced token detection.

Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *Proceedings of The Web Conference 2020*, WWW '20, page 418–428, New York, NY, USA. Association for Computing Machinery.

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Liu Yong, and Shen Huang. 2024. End-to-end beam retrieval for multi-hop question answering. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1718–1731, Mexico City, Mexico. Association for Computational Linguistics.

Ruqing Zhang, Jiafeng Guo, Lu Chen, Yixing Fan, and Xueqi Cheng. 2021. A review on question generation from natural language text. *ACM Trans. Inf. Syst.*, 40(1).

A. Appendix

A.1. Model Prediction Examples

Example 1
Predicted Question: The winners of the 2017 Canadian Olympic Curling Trials will compete in a sports event that will be held in what city?
Reference Question: Where did the winners of the 2017 Canadian Olympic Curling Trials go on to represent Canada?
Comparison: Both questions are looking for the same information. The predicted question is comparatively straighter asking "will be held in what city?".

Example 2
Predicted Question:"Rings Around the World" is a song by a band that was formed in what year?
Reference Question: "Rings Around the World" is a song by what band formed in 1993?
Comparison: The predicted question and the reference question are asking different questions. The predicted question and the reference question both are related to the given input contexts.

Example 3
Predicted Question: To which group does the singer, who stars with Lee Sang-woo in the TV series All About My Mom belong?
Reference Question: To which band does the actress belong, who starred with Lee Sang-woo in the film All About My Mum?
Comparison: Though both of the questions are very similar in structure, they are not the same. The predicted question is based on the input contexts where the reference question is not based on the given context.

Example 4
Predicted Question: Helter Skelter chronicled the case of the murderer who was part of whose "family"?
Reference Question: Helter Skelter presented the firsthand account of the murder who was a part of whose "family"?
Comparison: The both questions are effectively asking for the same information. The predicted question is using "chroncicled" instead of "presented the firsthand account", but the meaning of both scenarios are same.

Example 5
Predicted Question: The composer who wrote the music for the musical "The Fantastics" was born in what year?
Reference Question: The composer who wrote the music for The Secret Garden was born in which year?
Comparison: The predicted question and the reference question are asking different questions. The reference question is based on the input contexts, but the predicted question is not based on the input contexts.

Example 6
Predicted Question: Howdilly Doodilly is an album that features music themes based on a character from what television show?
Reference Question: Howdilly Doodilly is an album that features music themes based on a character from what TV show?
Comparison: Both questions are identically same.

A.2. Evaluation Metrics

- **BLEU**
  BLEU(Bilingual Evaluation Understudy) first introduced by Papineni et al. (2002) This is a metric for evaluating the performance of machine-translated text with the reference translation by comparing the generated translation to the reference translation. It assigns a score from zero to one based on how close the generated translated text is to the reference text, where zero indicates no similarity and one indicates perfect similarity.
  BLEU evaluates the translation on the corpus level, that is, it calculates how many words in the generated translation match with the reference translation. BLEU calculates the proportion of the n-grams,

which means it evaluates the sequence of n consecutive words that appear both in the generated translation and the reference translation.

- **ROUGE**
  ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a widely popular evaluation metric proposed by Lin (2004), mostly used for text summarization and machine translation tasks. ROUGE works on the basis of Longest Common Subsequence (LCS) (Lin and Och, 2004) between the generated text and the original text. After calculating LCS, it calculates the precision and recall between the generated text and the reference text. The calculated precision and recall helps to find the F-score between the generated text and the reference text. ROUGE metric value can be between zero to one, where one means higher similarity between the generated text and the reference text.

- **METEOR**
  METEOR (Metric for Evaluation of Translation with Explicit Ordering) is an evaluation metric introduced by Lavie and Agarwal (2007) to evaluate machine translation tasks. To solve the limitations of earlier evaluation metrics like BLEU, METEOR provides a more holistic approach where only word matching is not calculated, but also takes into account the word meaning, stemming, and synonyms. METEOR also allows stemming (matching root words) (Lovins, 1968) and synonyms to perform the calculation. By considering stemming and synonyms, METEOR is more suitable for evaluating paraphrased texts.

A.3. Workflow of Knowledge Graph Generation

Sample Input for Knowledge Graph Generation

Shirley Temple was an American actress, singer, dancer, businesswoman, and diplomat who was Hollywood's number one box-office draw as a child actress from 1935 to 1938. As an adult, Shirley Temple was named United States ambassador to Ghana and to Czechoslovakia and also served as Chief of Protocol of the United States.

Subject	**Relation**	**Object**
Shirley Temple	was	American actress
Hollywood	has	number one box office draw as child actress from 1935 to 1938
Shirley Temple	was named United States ambassador As	adult
Shirley Temple	was named United States ambassador to	Ghana
Shirley Temple	was named United States ambassador to	Czechoslovakia
Shirley Temple	served as	Chief of Protocol of United States

Table 5: Extracted Triplets from Sample Input

Node Number	Node Name
0	Shirley Temple
1	American actress
2	Hollywood
3	number one box office draw as child actress from 1935 to 1938
4	adult
5	Ghana
6	Czechoslovakia
7	Chief of Protocol of United States

(a) Node Mapping

Edge Number	Node to Node Connectivity	Relation
0	$Shirley\ Temple \rightarrow American\ actress$	was
1	$Shirley\ Temple \rightarrow adult$	was named United States ambassador As
2	$Shirley\ Temple \rightarrow Ghana$	was named United States ambassador to
3	$Shirley\ Temple \rightarrow Czechoslovakia$	was named United States ambassador to
4	$Shirley\ Temple \rightarrow Chief\ of\ Protocol\ of\ United\ States$	served as
5	$Hollywood \rightarrow number\ one\ box\ office\ draw\ as\ child\ actress\ from\ 1935\ to\ 193$	has

(b) Edge Mapping

Table 6: Triplets to Nodes and Edges Generation

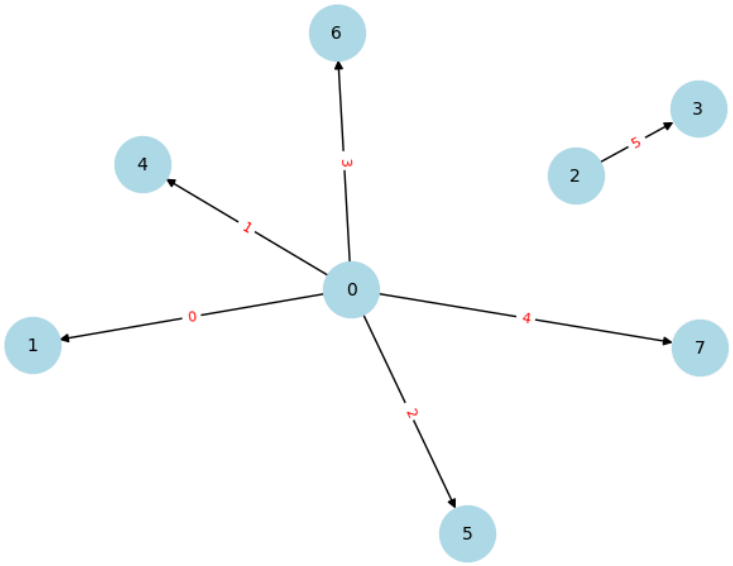

Figure 4: Graph Representation with Nodes and Edges