KG4QG: COMBINING KNOWLEDGE GRAPH WITH LARGE LANGUAGE MODELS FOR MULTI-HOP QUESTION GENERATION

AL HASIB MAHAMUD
Bachelor of Science, Ahsanullah University of Science and Technology, 2019

A thesis submitted
in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

Department of Mathematics and Computer Science
University of Lethbridge
LETHBRIDGE, ALBERTA, CANADA

KG4QG: COMBINING KNOWLEDGE GRAPH WITH LARGE LANGUAGE MODELS FOR MULTI-HOP QUESTION GENERATION

AL HASIB MAHAMUD

Date of Defence: June 18, 2025

Dr. Yllias Chali Thesis Supervisor	Professor	Ph.D.
Dr. Wendy Osborn Thesis Examination Committee Member	Associate Professor	Ph.D.
Dr. John Zhang Thesis Examination Committee Member	Associate Professor	Ph.D.
Dr. Andrew Fiori Chair, Thesis Examination Committee	Associate Professor	Ph.D.

Dedication

This thesis is dedicated to all the mothers of the world who have sacrificed their lives to support their children, just like my mother.

Abstract

Question generation is a task of Natural Language Processing where the goal is to generate fluent, grammatically correct, and error-free questions based on a given input context and optionally an answer. Multi-hop question generation is a more complex task compared to traditional single-hop question generation, as it requires reasoning over multiple information from multiple input contexts in generating multi-hop questions. In our work, we have addressed the challenge of building a multi-hop question generation system by combining the knowledge graphs with Large Language Models (LLMs). We have designed a framework KG4QG (Knowledge Graph for Question Generation), where knowledge graphs are generated from the input contexts. For the knowledge graph embedding, we use a Graph Attention Network, and for input texts embedding, we leverage a Sentence Transformer. Finally, we apply the BART and T5 models as Large Language Models to generate multi-hop questions from our proposed model. Using the HotpotQA dataset to evaluate the performance of our KG4QG framework, our proposed methodology shows enhanced performance over the previous methodologies.

Acknowledgments

First and foremost, I thank Almighty Allah for His countless blessings and for giving me the strength, opportunity, and perseverance to undertake this thesis.

I would like to express my sincere gratitude to my supervisor, Dr. Yllias Chali, for his unwavering support, guidance, and encouragement. He introduced me to the wonderful world of Natural Language Processing, inspired me to explore new research challenges, and tolerated my errors. His door was always open for me whenever I faced any trouble or I had any questions regarding my research and my studies.

I am really grateful to my supervisory committee members, Dr. Wendy Osborn and Dr. John Zhang, for their valuable suggestions, support, and motivation throughout my research journey. I would like to thank Dr. Andrew Fiori for being the Examination chair of my thesis defense.

I am thankful for the financial support provided by the University of Lethbridge's School of Graduate Studies (SGS).

I would like to thank my friends, colleagues, and classmates for their help and support. Thanks to Emam Hossain, Jannatul Maowa, Md. Asif Talukdar, and Ahmed Mohamedeen who have been the source of my motivation, emotional support, and supportive care during my MSc studies.

Last but not least, I am incredibly grateful to my parents and my sister for their love, selfless sacrifices, affection, and support with a heart full of grace. My parents' prayers to Almighty Allah are the luminous flame guiding every step of my life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Question Generation (QG) is the task of automatically generating human-like questions from some input sources like a passage, answer, paragraph, context or a statement (Zhang et al., 2021). As in the field of Artificial Intelligence (AI), QG is considered an important topic, and is gaining focus in both the academic research and industry fields (Zhang et al., 2021). In recent years, QG has been a topic of extensive research. But most of the research is based on the generation of a single question from a single paragraph where the answer to the question depends on a small amount of reasoning (Emerson and Chali, 2023a). From the SQuAD dataset (Rajpurkar et al., 2016), Min et al. (2018) found that approximately 90% of the questions can be generated from a single sentence within the input paragraph (Emerson and Chali, 2023a).

The basic structure of a question generation task can be defined as: input - a sequence of text, and output - a grammatically correct question which is relevant to the input. To generate questions, there are two basic approaches available. One is the rule-based approach and the other is the neural network-based approach. In a rule-based approach, predefined linguistic rules, templates, patterns, and grammatical structures are utilized to generate accurate questions. In the neural network-based approach, the question is predicted on the basis of the neural network architectures. Some neural network architectures which can be used for generating questions are the sequence-to-sequence (Seq2seq) model, the graph neural network (GNN), transformers and large language models (LLMs).

In QG, different types of questions can be generated by following one specific process. Some specific question generation processes are answer-aware question generation, answer-unaware question generation, paragraph-based question generation, single-hop question generation, multi-hop question generation, single reference question generation, and multiple reference question generation. In single-hop question generation, the generated question is dependent only on one passage, and the question is generated using only one sentence of the passage or some relevant information of the passage. It requires simple reasoning to answer the question in the paragraph. Multi-hop questions generation is more complex as it requires multiple reasoning in multiple paragraphs (or even different documents) to answer the question (Lin et al., 2024). Simply, the answer of a multi-hop question is dependent on multiple (two or more) paragraphs. If an individual wants to generate multi-hop question, they must read all input passages, understand the facts from them, understand the relationship between the facts, connect them, and generate a multi-hop meaningful question from the connected information.

The interest of research on multi-hop question generation is increasing as QG can be applied in different areas like the educational system, medical and clinical tools, chatbot components, and also other Natural Language Generation (NLG) tasks like question answering systems. Some real world application of multi-hop question generation system are given below.

- **Search Engine Query Generation**

  As human beings usually struggles to generate actual questions during the time of asking questions in a search engine, a multi-hop question generation model can be applied to express the mind of the person by generating appropriate questions which can lead to deep explanation of a specific topic.

- **Educational System**

  To know the accurate answer, it is required to ask the accurate questions. Multi-hop question generation can be applied in educational systems like enhancing learning ex-

perience for students by generating quiz, test, and assignment questions. To enhance interactive learning systems, multi-hop questions can be effective as they require multiple steps to answer.

- **Natural Language Processing (NLP) Tasks**

  Multi-hop question generation system can be used in different NLP tasks like chatbot design and question answering (QA) systems. When communicating with a chatbot, a multi-hop question generation model can be applied to express complex followup questions. To build up a question answering system, it is required to have a huge amount of labeled question-answer pairs for training when multi-hop reasoning is essential. As manual generation of a dataset is complex and time consuming, the generated questions by the multi-hop question generation system can be added to the dataset and can be deployed during training to build an effective QA system.

- **Medical Applications**

  A multi-hop question generation model can be applied in medical and clinical tools like patient history generation, virtual assistance, and appointment scheduling.

- **Security and Privacy**

  By generating stepwise diagnostic questions, multi-hop question generation system can help identifying threats and vulnerabilities in a network.

In our work, a Knowledge Graph (KG) is generated from input text, while a Graph Attention Network (GAT) is leveraged for graph embedding. To the best of our knowledge, this is the first work where a knowledge graph is combined with an LLM to generate a better result by handling graph-structured data.

The main motivation of this research is to find a strategy to generate complex multi-hop questions that improves upon previous works. Sequence-to-sequence-based transformer models are already capable of generating simple questions where only one single paragraph is required to provide a logical solution to the question. However, the issue arises as to

whether those models are capable of generating complex questions where we need multiple paragraphs to provide a complete solution to the question. The main key factors in the motivation for this thesis are given below.

- **Complex Reasoning**

  A multi-hop question can easily be defined as a question in which multiple sources/-paragraphs are required to answer them. The model should be capable of collecting appropriate information from each paragraph, after which the model should be capable of merging the collected information to generate questions so that complex reasoning of different paragraphs should be ensured by reasoning path.

- **Ensuring Commonsense Reasoning and Mitigating Hallucination**

  One of the major issues and drawbacks of large language models is hallucination. It can reduce the performance of the model. Ensuring commonsense reasoning is an effective way to mitigate hallucination. One of the motivations of this research is to ensure commonsense reasoning of the model so that it does not hallucinate.

- **Supporting Information Retrieval System**

  The generated questions from the model can help in information retrieval, question-answering systems, text summarization, and chatbot design.

1.2 Contributions

The thesis has great impact on the multi-hop question generation system by creating key contributions on the following aspects.

1. **Combining Knowledge Graph with LLMs**

   Usually, the input text is tokenized and converted into embeddings as the input in LLMs. However, in this research, the concatenation of KG embedding with input text embedding is done, which is utilized as input for the LLMs. To the best of our

knowledge, this model offers the unique contribution for first leveraging a KG as input in LLMs.

2. **Integrating SBERT and GAT**

   In this research, a sentence transformer (SBERT[1]) and a GAT are integrated. For input text embedding generation SBERT is used, and for KG embedding generation GAT is used. This approach has enriched the performance of LLMs by handling structured graph data with input text which has made a great contribution in the field of multi-hop question generation.

3. **Triplet Dataset Contribution**

   For this research, we have proposed a triplet dataset that presents the knowledge graph for each input text. This triplet dataset consists of entities and the predicates between two entities. This dataset can be used in the further research of the multi-hop question generation. Also the implementation of this research is made available as an open-source resource which can be considered a contribution of encouraging further research and development in multi-hop question generation.

4. **Enhancement of Multi-hop Question Generation**

   After doing evaluations from the result of the experiments conducted on the HotpotQA [2] dataset, the evaluation has shown a notable increase in the quality of the generated multi-hop questions and the metrics compared to the existing models.

1.3 Thesis Organization

The thesis is organized as follows.

- **Chapter 2** provides the background details and fundamental concepts required to understand the different methodologies and technologies used in the thesis.

[1] sbert.net.

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

- **Chapter 3** discusses the previously completed work and literature review of question generation.
- **Chapter 4** defines the problem definition and proposed methodology of multi-hop question generation. This chapter explains the model architecture, its strength, underlying algorithms, and an explanation of choosing the models.
- **Chapter 5** presents the dataset used in this thesis, evaluation metrics, experimental setup, model implementation, and experimental results.
- **Chapter 6** provides the thesis conclusion and direction for further research on the advancement of multi-hop question generation research.

Chapter 2

Background

This chapter describes the information required to understand the previous research works and the proposed methodology in this thesis. First, we discuss about the knowledge graph. After that we explain different natural language processing concepts like tokenization, co-reference resolution, and word embedding. Then we describe two neural network architectures-sequence-to-sequence learning, and graph attention network. Also, we describe about attention mechanism technique. Furthermore, we discuss about the automatic evaluation metrics BLEU, ROUGE, METEOR, and BERTScore. Finally we explore about large language models and architectures of BART and T5 models.

2.1 Knowledge Graph

A knowledge graph (KG) can be defined as a data management system that combines different data types and uses graphs to visualize the information (Fei et al., 2022; Tang et al., 2019). It can be defined as the collection of triplets (Pietrasik et al., 2024). Each triplet consists of a subject entity to an object entity via a predicate/relationship. So, a triplet can be represented according to equation 2.1.

$$triplet = (s, p, o) \in S \times P \times O \tag{2.1}$$

where,

(s, p, o): A triplet where s is the subject, p is a predicate, and o is the object of the triplet.

S: The set of subjects.

P: The set of predicates/relationships.

O: The set of objects.

The three main elements of a KG are nodes, edges, and triplets. A node represents an entity, concept, or data. An edge connects two nodes in a graph. A predicate/Relationship can be directed or undirected according to the connection of the graph. A KG can be utilized for representing extensive global knowledge and retrieving information (Chaudhary et al., 2025). It can also be used for semantic search (Gulnara and Sabdenov, 2024) and semantic reasoning (Sun et al., 2024).

2.2 Natural Language Processing Concepts

We first describe tokenization, which is a process to convert a text into words. Then we discuss Coreference resolution, and finally we explain embedding, the process of converting words to a dense vector representation.

2.2.1 Tokenization

Tokenization is a fundamental concept of NLP, which converts a piece of text into smaller, more manageable units. For example, converting a sentence into smaller pieces called tokens. Depending on the required form and the method used, the tokens can be individual words, subwords, or characters. Before applying any machine learning model, the text is required to be converted into tokens. This transformation converts the data to a more manageable, understandable, and interpretable, allowing it to be processed properly by models.

The simplest way of tokenize text is splitting the whole text on its whitespaces only. For example, the sentence ”Knowledge is power” would be tokenized as [”Knowledge”, ”is”, ”power”]. Another approach of tokenization is subword tokenization, which splits the text into the smallest meaningful unit. For example, the sentence ”I am not unhappy” would be tokenized as [”I”, ”am”, ”not”, ”un”, ”happy”] using subword tokenization as the word un-

happy is not in the model's vocabulary. There are some open source libraries like NLTK[3], spaCy[4], Gensim[5] to perform the tokenization in different NLP tasks like summarization, question answering, machine translation etc.

2.2.2 Coreference Resolution

Coreference resolution is an NLP task that aims to identify all linguistic expressions in a text that correspond to the same real-world entity (Preiss, 2005). To resolve ambiguity, maintain context across sentences, and ensure scalability for long texts, coreference resolution plays a vital role. Coreference resolution can be utilized in text-to-graph generation, text summarization, and question answering tasks. For example, the sentences "John is a nice person. He always likes to help others", after performing coreference resolution would result in "John is a nice person. John always likes to help others."

2.2.3 Word Embedding

Before feeding the sequence of text into a neural network model, it is required to convert the text to a numerical form. Word embedding is a technique of NLP where words are represented as a dense, fixed-length vector. The Vector Space Model (VSM) introduced by Salton et al. (1975) is considered one of the most influential models for encoding words and documents as vectors (Almeida and Xexéo, 2023). As word embedding is leveraged in many NLP tasks, it has become one of the emerging topics of research (Turney and Pantel, 2010). It is the process of representing words into fixed-length vector representations. Word embedding can be categorized into two types: prediction-based models and count-based models (Baroni et al., 2014; Almeida and Xexéo, 2023; Pennington et al., 2014).

- **Prediction-based Models**: Prediction-based models are mainly based on neural language models (NNLMs), which started with the first large neural language model proposed by Bengio et al. (2003).

[3]https://www.nltk.org/.
[4]https://spacy.io/.
[5]https://pypi.org/project/gensim/.

- **Count-based Models**: Count-based models produce word embedding by utilizing word-context co-occurrence counts globally in a corpus (Turney and Pantel, 2010). This approach does not provide any specific training for algorithms.

Three of the most popular word embedding approaches are Word2Vec (Mikolov et al., 2013a), Glove (Pennington et al., 2014), and FastText (Joulin et al., 2016).

Word2Vec

Word2Vec is a widely-used word embedding approach proposed by Mikolov et al. (2013a) at Google, which represents a high-dimensional vector of real numbers that represents the complex relationships between different entities. To produce a word embedding, the Word2Vec model can use either a continuous bag of words (CBOW) approach or a continuously sliding skip-gram approach (Mikolov et al., 2013a,b).

Glove

Glove (Global Vectors for Word Representation) is a word embedding approach introduced by Pennington et al. (2014) at Stanford. This is an unsupervised learning approach which maps words into a meaningful vector space. This approach leverages global word-word co-occurrence statistics. This method combines two approaches - global matrix factorization and local context window methods.

Sentence Transformer

Sentence transformer, known as Sentence-BERT (SBERT) introduced by Reimers and Gurevych (2019) is a modification of the pre-trained BERT (Devlin et al., 2019)model. This approach generates sentence embeddings using siamese and triplet network structure. It generates semantically meaningful sentence embeddings that can be compared using cosine similarity, so semantically similar sentences are close in vector space. Siameses network architecture consists of two identical neural networks that share and parameteres, it ensures the similar sentences are closed together in the embedding space. SBERT is very useful for

clustering, semantic similarity, sentiment analysis, and information retrieval tasks, which are computationally efficient.

2.3 Sequence-to-Sequence Learning

Sequence-to-Sequence, or Seq2Seq, is a type of neural network architecture that transforms an input sequence to an output sequence. It is widely used in NLP tasks like machine translation, text summarization, question answering, and conversational models. This model is introduced by Sutskever et al. (2014) for the machine translation task between English and French, where two LSTMs (Long Short-Term Memory) (Hochreiter and Schmidhuber, 1997) are combined into one unified model and the model is trained in an end-to-end manner. This model can take a variable-length input text and generate a different length output text, where Cho et al. (2014) shows how the machine can handle variable-length sequence transformations.

Sequence-to-Sequence models are based on two major components - encoder and decoder. The encoder takes the input sequence and converts it into an encoded representation. An encoder usually uses an RNN (Recurrent Neural Network) (Rumelhart et al., 1986) like an LSTM (Hochreiter and Schmidhuber, 1997), or a GRU (Gated Recurrent Unit) (Cho et al., 2014) and convert the input sequence to a context vector. The context vector is the input for the decoder, which generates the output sequence.

2.4 Attention Mechanism

One of the major limitations of Sequence-to-Sequence models is long-range dependencies in data. A fixed-length vector that represents the encoded sequence may not be sufficient to store all the information. So the model performance may decrease if the input sequence is too long. Bahdanau et al. (2016) proposed an attention mechanism which enables solving this problem in the neural machine translation task. This approach allows RNN-based sequence-to-sequence model during decoding to look back in the encoding se-

quence and determine at the current stage which words are more important for generating the output sequence. In producing the output sequence, the attention mechanism provides different weights to the input sequence according to the relative importance of the current stage. The evaluation of the attention mechanism results in the development of the Transformer architecture by Vaswani et al. (2023). A self-attention mechanism is used in the transformer. Self-attention mechanism for each element calculates a weighted average of all input elements, allowing each element to attend to all other elements in the sequence (Vaswani et al., 2023).

Components of Attention Mechanism

The components of the attention mechanism are: Query, Key, and Value. An alignment score is calculated by attention between queries and keys. Then the softmax function is applied to get the attention weight, which is used to calculate a weighted sum of the values.

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.2}$$

where, the set of Queries, packed together into a matrix Q, the keys and values are packed together into matrices K and V, and d_k is the dimensionality of the key vector which scales the dot product in the softmax function.

Multi-head Attention

In a transformer architecture, multi-head attention uses several attention heads in parallel. The outputs are then concatenated and linearly transformed to make the final output. It allows the model to capture more information from the input.

$$MultiHead(Q,K,V) = Concat(head_1,...,head_h)W^O \tag{2.3}$$

$$where\ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{2.4}$$

In the above equations,

W_0 is a parameter matrix, this matrix linearly transform the concatenated outputs of all the heads.

W_i^Q is the parameter matrix for the i-th attention head for Queries.

W_i^K is the parameter matrix for the i-th attention head for Keys.

W_i^V is the parameter matrix for the i-th attention head for Values.

2.5 Graph Attention Network

A Graph Attention Network (GAT) introduced by Veličković et al. (2018) is a neural network architecture. It is a variant of a Graph Neural Networks (GNNs) that works with graph structure data. Compared to a GNN, a GAT is more efficient, effective, and flexible as it combines an attention mechanism into a GNN. GAT assigns an attention coefficient to each neighboring node, according to the neighbor's features during feature aggregation.

GAT Architecture

Input Features

Each node in the graph is represented as a feature vector, which is the numerical representation of each node. The feature vectors are one-dimensional, where each number in the vector represents a feature of the node.

Linear Transformation

The first linear transformation is done on the input features using a weight matrix that encodes how an input vector is mapped to an output vector through multiplication.

Attention Layers

After the linear transformation, the GAT computes the attention coefficient concerning its neighboring nodes, indicating the importance of node features to a node. The coefficient computed by the attention mechanism, where nodes i and j are connected by an edge, can be represented by the equation 2.5.

$$\alpha_{ij} = \frac{exp(LeakyReLU(a^T[Wh_i||Wh_j]))}{\sum_{k\in\mathcal{N}(i)} exp(LeakyReLU(a^T[Wh_i||Wh_k))]} \tag{2.5}$$

where,

a: Weight vector to compute attention score

h_i:Node feature vector of node i

h_j: Node feature vector of node j

W: Weight matrix applied to the h_i and h_j

$\mathcal{N}(i)$: Sets of neighboring nodes for node i

$||$: Concatenation operation

k: Iterator that runs over the set of neighbors of node i

$.^T$: Transposition operation

LeakyReLU (Maas et al., 2013) is an activation function which allows a small, non-zero gradient when the input is negative and solves the problem of dying ReLU. Dying ReLU is the problem in neural network where neurons using the ReLU activation function always output zero and become permanently inactive, regardless of the input.

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases} \tag{2.6}$$

where α is a positive small constant.

Normalization

The attention coefficients are normalized using the softmax function to make the coefficients easily comparable across different nodes.

$$\alpha_{ij} = softmax_j(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k=1}^{N} exp(e_{ik})} \tag{2.7}$$

where e_{ij} is the raw attention score and N is the number of tokens in the sequence.

Feature Aggregation

Using the normalized attention coefficients as weights, the final step is to update the node features of each node as a weighted sum of its neighboring nodes. Feature aggregation is done using equation 2.8.

$$h_i^{'} = \sigma(\sum_{j \epsilon N_i} \alpha_{ij} W h_j) \tag{2.8}$$

σ is a nonlinear activation function. Usually, ReLU (Agarap, 2019) is utilized as a nonlinear activation function.

2.6 Automatic Evaluation Metrics

To understand the model performance and capabilities, evaluation metrics are used in natural language processing tasks like classification, text generation, summarization, translation, etc. The main target of using evaluation metrics is to compare the model output with the reference where the reference can be obtained from the dataset. Utilizing evaluation metrics are better than human evaluations because evaluation metrics are fast, cost-efficient, consistent, reproducible, and standardized by research community compared to human evaluation (Sun, 2010; Belz and Reiter, 2006; Clark et al., 2012). For classification task, classification metrics are Accuracy, Precision, Recall, F-score, for text generation task like summarization, question answering, or machine translation, the language generation metrics are BLEU, ROUGE, METEOR, BERTScore etc.

2.6.1 Accuracy, Precision, Recall and F-score

Accuracy

Accuracy calculates the overall correctness of a model, whether positive or negative.

$$\text{Accuracy} = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positives + False\ Negatives} \tag{2.9}$$

Precision

Precisions evaluates the correctness of positive predictions measured by a model.

$$\text{Precision} = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{2.10}$$

Recall

Recall, also known as sensitivity measures the model's ability to find all relevant instances.

$$\text{Recall} = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{2.11}$$

F-score

F-score is the harmonic mean of precision and recall. It provides a single metric that balances both

$$\text{F1-score} = 2 * \frac{Precision\ *\ Recall}{Precision\ +\ Recall} \tag{2.12}$$

2.6.2 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. Generally, it is up to four words long. BLEU-1, BLEU-2, BLEU-3, and BLEU-4 are the variants of BLEU that differ in the length of n-grams. BLEU-1 means precision of a single word, BLEU-2 means precision of a 2 word sequence, BLEU-3 means precision of a 3 word sequence, and BLEU-4 means precision of a 4 word sequence.

The BLEU score for corpus is calculated as:

$$BLEU = BP.exp(\sum_{i=1}^{N}(w_n log p_n) \tag{2.13}$$

where,

w_n: Weight for each n-gram (1 for BLEU-1, 0.5 for BLEU-2, 0.33 for BLEU-3, and 0.25 for BLEU-4)

p_n:Modified precision for n-grams of size n

BP: Brevity penalty

The major limitations of BLEU are that it does not capture fluency, semantic adequacy, and grammatical errors. Additionally, it only counts the actual matches, but it does not consider sentences with the same meaning.

2.6.3 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.

$$Precision = \frac{Length\ of\ LCS}{Length\ of\ Generated\ Text} \tag{2.14}$$

$$Recall = \frac{Length\ of\ LCS}{Length\ of\ Reference\ Text} \tag{2.15}$$

$$F - score = 2 * \frac{Precision\ *\ Recall}{Precision\ +\ Recall} \tag{2.16}$$

ROUGE has some variants like ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-N, ROUGE-W, etc. ROUGE-N finds the overlap of n-grams between the generated text and the reference text, for example, ROUGE-1 (unigrams) and ROUGE-2 (bigrams) find the overlap between single words and word pairs, respectively. ROUGE-L, which is mostly used as the evaluation metric in the question generation task, is an LCS-based strategy.

Though ROUGE is a fast and automated approach as an evaluation metric, its main limitation is that it does not consider fluency, coherence, or semantic meaning. ROUGE generates the metric value based on the exact word matching, and it does not consider any paraphrases or synonyms.

2.6.4 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 not only word matching is calculated, but also takes into account the word meaning, stemming, and synonyms.

First, METEOR calculates the alignment between a unigram in the candidate translation and a unigram in the reference translation. Then it calculates the precision and the recall. METEOR combines the precision and recall using the harmonic mean in the equation 2.19,

where recall is provided nine times more weights than precision.

$$Precision = \frac{M}{W_C} \tag{2.17}$$

where,

M: The number of unigrams matches between the candidate translation and the reference translation

W_C: The number of unigrams in the candidate translation

$$Recall = \frac{M}{W_R} \tag{2.18}$$

where,

M: The number of unigrams matches between the candidate translation and the reference translation

W_R: The number of unigrams in the reference translation

$$F_{mean} = \frac{10 * Precision * Recall}{Recall + 9 * Precision} \tag{2.19}$$

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.

2.6.5 BERTScore

BERTScore introduced by Zhang et al. (2020), is an advanced evaluation metric assessing the semantic similarity between a generated text and a reference text. It leverages the contextual embedding from BERT or similar Transformer-based models. Traditional evaluation metrics are based on surface-level word matching or n-gram overlap. On the other hand, BERTScore depends on the deeper semantic meaning and context. The development

of BERTScore has provided more context-aware evaluation metrics in NLP.

BERTScore measures the similarity between the texts. According to the computation of BERT-Score, the generated text and the reference text are first tokenized and passed through a BERT model to generate contextual embedding of each token. The cosine similarity is calculated for each token in the candidate tokens between this candidate token embedding and the embedding of all tokens of the reference text. Finally, precision, recall, and F1-score are calculated. Precision calculates the average of the maximum cosine similarity scores for each candidate token to the reference tokens. Recall calculates the average of the maximum cosine similarity scores for each reference token to the candidate tokens. F1-score is the harmonic mean of the precision and recall. Precision measures how much the candidate text or the generated text is relevant to the reference text, recall measures how much the reference text is relevant to the candidate text or the generated text.

$$F_{BERT} = 2 * \frac{Precision_{BERT}.Recall_{BERT}}{Precision_{BERT} + Recall_{BERT}} \tag{2.20}$$

BERTScore is effective for model selection; it achieves better correlation than common metrics and resolves some of the limitations of traditional metrics (Zhang et al., 2020).

2.7 Large Language Models

Large Language Models (LLMs) are transformer-based models proposed by Vaswani et al. (2023). LLMs contain hundreds of millions to trillions of parameters, which are pre-trained on extensive text (Shanahan, 2023; Zhang et al., 2020; Minaee et al., 2025). Some examples of LLMs are GPT-3 (Brown et al., 2020), BART(Lewis et al., 2020), T5(Raffel et al., 2023), PaLM (Chowdhery et al., 2022), LLaMA (Touvron et al., 2023). LLMs can understand, generate, and interact with human languages.

LLMs are trained in two main phases:

- Pre-training: Pre-training LLMs is the task of training a model on a huge, diverse

dataset of text without any specific task labels.

- Fine-tuning: Fine-tuning LLMs is the process of training a pre-trained model on a specific dataset for a particular task to optimize its performance.

2.7.1 Applications of LLMs

- **Text Generation**: LLMs can be utilized producing appropriate contextual texts on different fields like text summarization, creative writing, mail generation etc.

- **Question Answering**: Designing question answering system and chatbot using LLMs can be highly efficient.

- **Machine Translation**: Multilingual communication like English to French translation or vice versa LLMs can be used for machine translation.

- **Code Generation and Debugging**: We can utilize LLMs for automatic code generation or automatic bug fixing task.

- **Information Retrieval**: LLMs can be leveraged for automatic query generation, document retrieval, and search engine optimization.

2.7.2 BART

BART (Bidirectional and Auto-Regressive Transformers) proposed by Lewis et al. (2020) is a transformer based sequence-to-sequence model. By integrating BERT (Bidirectional Encoder Representations from Transformers) by Devlin et al. (2019) bidirectional encoder with GPT (Generative Pre-training Transformer) by Radford and Narasimhan (2018) left-to-right decoder, this model can be used in language generation, understanding and interaction tasks.

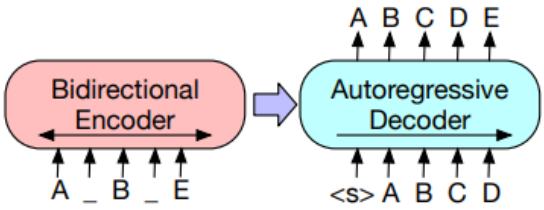


Figure 2.1: BART Architecture (Vaswani et al., 2023).

Architecture

BART model is built based on the standard sequence-to-sequence transformer architecture form Vaswani et al. (2023). BART model is pre-trained by corrupting documents and the cross-entropy loss function is utilized to reconstruct the loss between the decoder's output and the original document. It has main two parts: encoder and decoder.

- **Encoder**

  The encoder part of BART is similar to BERT where a bidirectional attention mechanism is employed which allows the encoder to process the input text by both direction. The differences between BERT and BART are BERT uses an additional feed-forward network before word prediction and in BART decoder, each layer performs cross-attention over the final hidden layer of the encoder.

- **Decoder**

  Like GPT, the BART decoder performs in autoregressive manner. The decoder generates output tokens autoregressively by generating one token at a time in a left-to-right manner.

Fine-tuning can make the BART model work more effectively. BART has roughly 10% more parameters than a model of the same size as the BERT model. The architecture of the BART model proposed by Vaswani et al. (2023) is shown in Figure 2.1. The left side of the image is the BART encoder part, which encodes the input text. The right side of the image is autoregessive decoder which decodes the text. A special token (<s>) is used at the beginning of the decoder, and it generates the output.

Variants of BART Model

According to the model size of BART architecture, BART model has some variants like BART-small, BART-base, BART-large, etc. In Table 2.1, the details of BART architecture with some variants are shown.

Table 2.1: BART Model Architecture Variants.

Model	**# Encoder Layers**	**# Decoder Layers**	**Hidden Size**	**# Parameters(Approximate)**
BART-small	4 to 6	4 to 6	256 to 512	45 to 60 million
BART-base	6	6	768	140 million
BART-large	12	12	1024	406 million

Applications

BART model can be utilized for text generation, text summarization, machine translation, understanding conversational texts, generating responses etc.

2.7.3 T5

The T5 (Text-to-Text Transfer Transformer) is a highly prominent language model proposed by Raffel et al. (2023) and introduced by Google research. Based on its model architecture, training process and hyperparameters, the T5 model is capable for doing different text-to-text problems like text summarization, question answering, and machine translation.

Architecture

The T5 model is based on a standard transformer architecture. This model takes a sequence of text as input and generates a sequence of text as output. This model is also based on an encoder and decoder.

- **Encoder**

  Utilizing an encoder, the input text is converted to a contextualized representation.

- **Decoder**

  A decoder takes the contextualized representation and generates the output tokens one at a time.

One of the key advantages of the T5 model is its unified framework. It transforms every NLP problem into a text-to-text problem and can simplify the new task.

Variants of T5 Model

Though all of the variants of the T5 model have the same architecture, the variants exist because of their model size and performance. In Table 2.2, the details of some variants of the T5 model are shown.

Table 2.2: T5 Model Architecture Variants.

Model	**# Encoder Layers**	**# Decoder Layers**	**Hidden Size**	**# Parameters(Approximate)**
T5-small	6	6	512	70 million
T5-base	12	12	768	240 million
T5-large	24	24	1024	770 million
T5-3B	24	24	1024	2.8 billion
T5-11B	24	24	1024	11 billion

Applications

The T5 model can be applied to different text-to-text generation tasks like question answering, text summarization, text generation, machine translation, and others.

2.8 Summary

This chapter provides the information required to understand the previous works on question generation and the proposed methodology of this thesis. We have discussed the knowledge graph, the basic concepts of natural language processing. We have then provided an overview of sequence-to-sequence learning, the attention mechanism, graph attention network architecture. Then we have described about the automatic evaluation metrics that

are important to understand the performance of our model. Finally, we have discussed large language models, the BART, and T5 model architectures and applications.

Chapter 3

Related Work

As multi-hop question generation has gained specific attention from researchers because of the advancement of Artificial Intelligence (AI), researchers have made unique contributions to question generation and specifically to multi-hop question generation. This chapter discusses the current state-of-the-art in the area of question generation, techniques, and methodologies used, and the scope which a researcher can contribute in the QG field.
At first, different question generation approaches are discussed such as template-based approach, syntax-based approach, semantic-based approach, and sequence-to-sequence models. The emphasis is given to the multi-hop question generation research. We have explored current research on the multi-hop question generation process, the dataset used in the research, and the evaluation approaches used. We have tried to find the architecture, and effectiveness of it, and to point out limitations, if any, in the architecture. This review provides us a path to understanding state-of-the-art models of multi-hop question generation, finding the research scope, and deciding our thesis methodology.

3.1 Question Generation Task

Sentence-to-question transformation for factual question generation was first introduced by Mitkov (2006) (Blšták and Rozinajová, 2021). Rus et al. (2010) had a great impact on question generation systems where a question was generated from a sentence and paragraphs. Zhang et al. (2021) discussed the state-of-the-art, present methodologies, and future scopes of question generation from a given passage or answer. From the analysis of the

paper, the template-based approach, syntax-based approach, and semantic-based approach are the main three approaches of the rule-based approach. On the other hand, traditional Seq2seq models, pre-trained Seq2seq models, generative models, and graph-based models are utilized in the neural network-based approach.

3.1.1 Syntax Based Question Generation

Syntax-based Question Generation is considered one of the fundamental processes that follows the syntactic structure of a sentence to generate meaningful questions. It is a rule-based approach that utilizes syntactic parsing to find the grammatical relationship and transformation rules to generate the questions (Dhole and Manning, 2022; Khullar et al., 2018). The main advantages of the syntax-based approach are that it performs better in generating short questions, and can identify the question type by analyzing syntactic patterns. However, it often struggles to generate complex questions. Generating questions with grammatical errors are the major limitations of this approach (Heilman and Smith, 2010; Yao, 2010).

3.1.2 Semantic Based Question Generation

Semantic-based Question Generation focuses on the deep understanding of the text by extracting meaning, relations, and entities from the text (Flor and Riordan, 2018). This approach utilizes Semantic Role Labeling (SRL) which identifies the semantic relationships in a sentence (Màrquez et al., 2008). This approach can generate wider variety of accurate questions by using deeper meaning of the sentence (Yao et al., 2012; Ma et al., 2020).

3.1.3 Template Based Question Generation

Template-based Question Generation uses predefined templates to generate questions where templates can be defined as predefined structures with placeholders (Biermann et al., 2018; Liu et al., 2018; Chen and Aist, 2009). Though this approach is very straightforward, it can generate questions ensuring grammatical correctness, designing high-quality

templates are major drawbacks of this approach because it requires human efforts and linguistic diversity (Yao, 2010).

3.1.4 Sequence-to-Sequence Approach

Sequence-to-sequence (seq2seq) models (Sutskever et al., 2014) are data-driven approaches using neural network architecture to generate questions. An encoder process the input text and a decoder generates the question.

Du et al. (2017) applied an RNN encoder-decoder architecture as a fully data-driven neural network approach for question generation. A bi-directional LSTM is used to encode the sentence. The attention mechanism is used to make the model focused on certain elements of the input. The SQuAD (Rajpurkar et al., 2016) dataset is utilized in this work which contains 100k questions from 536 articles. The seq2seq model with attention mechanism where pre-trained word embedding is implemented provides better results than without pre-trained word embedding seq2seq model with attention mechanism. The main contribution of this paper is to propose an attention mechanism first for generating questions.

Sun et al. (2018) proposed an attention-based pointer generator model where a bi-directional LSTM works as an encoder and unidirectional LSTM works as a decoder. Local attention is used to narrow the focus in the input sequence and word position embedding is incorporated. MS Marco (Nguyen et al., 2016) and SQuAD (Rajpurkar et al., 2016) datasets are used in this research. The main contribution of this research is that the model does not copy any irrelevant words from the input.

Rabin et al. (2023) proposed a question generation model for generating one or multiple questions using the consistency parser. SNLI (Bowman et al., 2015) dataset used in this research work contains natural langauge inference (NLI) pairs that have two sentences denoting a premise and a hypothesis. The authors have filtered those questions with common answers and finally selected questions that only use information known to the user. The major contribution of this research is to propose an approach for gap-focus question gener-

ation that involves information gaps between the interlocutors during communication and utilize NLI dataset.

Oh et al. (2023) proposed a multiple reference question generation system utilizing LLMs (Chatgpt and GPT-3) to generate reference augmentation questions by paraphrasing each question. Each question is paraphrased 20 times. The Quiz design dataset (Laban et al., 2022) is used in this research which contains 3,164 human-annotated samples.

Semantic Role Labeler (SRL) can be combined with the seq2seq approach where SRL generates the semantic representation of input, which is used to train the seq2seq models. Naeiji et al. (2023) proposed an approach where using SRL BERT, each answer is converted to its semantic representation. Each question is also converted to its semantic representation. If a semantic role label occurs both in the answer and question, then the question is replaced by the label. Otherwise, cosine similarities are assigned to generate the question by comparing each label word with a possible n-gram from the question. The highest similarities are replaced by the question. T5 and BART are used as Seq2seq models. Three datasets are used in this work: SQuAD (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2017), and CarManuals (Alfassy et al., 2022).

3.1.5 Multi-hop Question Generation

Though multi-hop question generation is a new area of research and some advances have been made in this field, multi-hop QG has gained great attention from researchers now. By generating multi-hop questions, reasoning-based questions can be ensured. To solve the unique challenges of multi-hop QG, researchers have proposed some methodologies. Some of the approaches are based on graph neural networks, large language models, prompting, attention mechanisms, advancement of encoder-decoder architecture, fusion networks, and others.

Su et al. (2020) proposed MulQG that is a sequence-to-sequence based model to generate multi-hop questions without any sentence level information. In the MulQG model, two

Table 3.1: Comparison of Seq2seq models in QG task.

Paper Title	**Methodology**	**Dataset**
Learning to Ask: Neural Question Generation for Reading Comprehension (Du et al., 2017)	Seq2Seq with Bi-directional LSTM	SQuAD
Question Generation Using Sequence-to-Sequence Model with Semantic Role Labels (Naeiji et al., 2023)	T5 and BART	SQuAD, NewsQA and Car Manuals
Answer-focused and Position-Aware Neural Question Generation (Sun et al., 2018)	Attention Based Pointer Generator	SQuAD and MARCO
Open-World Factual Consistent Question Generation (Maheshwari et al., 2023)	Pegasus large and BART-large	ELI5
Question Generation for Question Answering (Duan et al., 2017)	CNN and RNN	Community QA
Event Extraction as Question Generation and Question Answering (Lu et al., 2023)	BART and T5	ACE 2005
RQUGE: Reference-Free Metric for Evaluating Question Generation by Answering the Question (Mohammadshahi et al., 2023)	GPT-2 and T5	SQuAD, NQ, and MS-MARCO
Evaluation of Question Generation Needs More References (Oh et al., 2023)	Chatgpt and GPT 3	Quiz Design Dataset

bidirectional LSTM-RNNs encoders are used to obtain initial contextual representations. A third bidirectional LSTM is used to create answer-aware context encoding. They have constructed the entity graph with the name entities in contexts as node where the edges between entities are created if they are in the same sentence, or in the same paragraphs. A Graph Convolution Network (GCN) is leveraged to perform multi-hop context encoding and use biattention mechanism (Seo et al., 2018) to update multi-hop answer encoding. Lastly, a reasoning gate is applied on the answer-aware context encoding. The decoder part is consisted of a uni-directional LSTM and a maxout pointer generator. For this research, HotpotQA dataset is utilized.

To ensure the complexity of the generated questions, Fei et al. (2022) proposed CQG which is a effective controlled framework. Entity graph is constructed following Qiu et al. (2019) from the input documents and Graph Attention Network (GAT) (Veličković et al., 2018) is employed. The answer is concatenated with the text and the sequence is passed to pre-trained BERT model. A transformer based decoder is constructed as controlled generation framework. The HotpotQA dataset is used for experiments of this research.

Emerson and Chali (2023b) proposed a multi-hop question generation approach that applies a transformer model without utilizing any sentence level supporting fact information. Concepts that are effective in single-hop question generation are incorporated in this research including a copy mechanism and placeholder tokens.

Jamshidi and Chali (2025) proposed GNET-QG, which integrates a GAT with sequence-to-sequence models. Constructing entity graph by nodes representing entities and edges representing relationships between entities, a multi-head GAT is applied before linear transformation to create enriched input text. BART and T5 are used as pre-trained transformer models. After fine-tuning the model with the HotpotQA dataset, better performance is obtained by BART when compared to the T5 model.

Cao et al. (2023) proposed knowledge graph enhanced language model (KGEL). The whole methodology can be split into three steps. In the first step, a GPT-2 model is utilized for en-

coding the context and answer. An answer-aware GAT with bi-directional attention mechanism provides the path to interact between the answer and the knowledge graph. Multi-hop questions are generated from the enhanced context representation using multi-head self attention module (Vaswani et al., 2023). The performance of the KGEL model is evaluated on the HotpotQA dataset.

Chapter 4

Multi-hop Question Generation

In this chapter, the problem definition is described first. Later, this research's proposed methodology is described in detail of its various components. The whole architecture of KG4QG is shown in Figure 4.1.

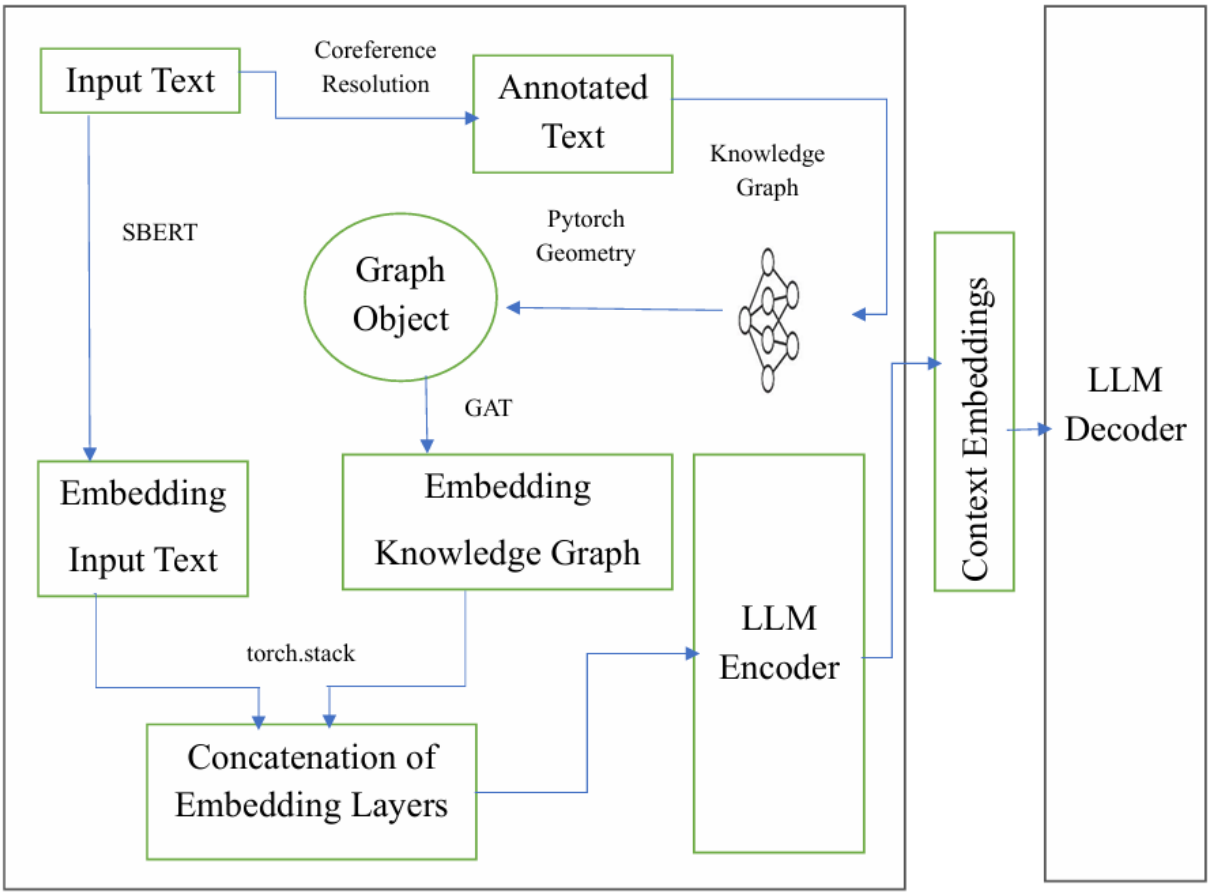


Figure 4.1: Full Architecture of KG4QG.

4.1 Problem Definition

In this section, we define the multi-hop question generation task based on conditional probability. Suppose for each answer, an input context C is defined as $C = (C_1, C_2, ..., C_N)$ where C_i is one paragraph and N is the number of paragraphs in the context. We want to generate question $Q = (Q_1, Q_2, ..., Q_M)$, where M represents the number of tokens in the question. The probability of the generated question can be defined by equation 4.1.

$$P(Q|C) = \Pi_{i=1}^{M} P(Q_i|[Q_1, Q_2, ..., Q_{i-1}], [C_1, C_2, C_3, ...C_N]) \tag{4.1}$$

As for this research, the input context C is generated by considering two paragraphs as the answer of the question can be gained by the reasoning path of the two paragraphs, then equation 4.1 becomes,

$$P(Q|C) = \Pi_{i=1}^{M} P(Q_i|[Q_1, Q_2, ..., Q_{i-1}], [C_1, C_2]) \tag{4.2}$$

4.2 Knowledge Graph Creation

In this section, we discuss the process of creating a Knowledge Graph (KG). To create the knowledge graph, the input text is converted to annotated text by performing coreference resolution on the input text. Then the knowledge graph is generated by using Stanford CoreNLP [6]. The whole procedure is shown in Figure 4.2.

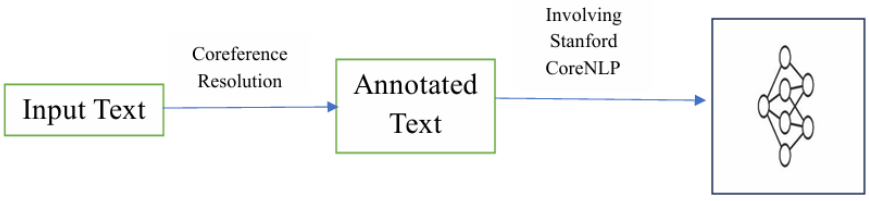


Figure 4.2: Knowledge Graph Creation.

[6] https://stanfordnlp.github.io/CoreNLP/.

4.2.1 Coreference Resolution

In the input text, we have done a co-reference resolution to identify and link the same entities. Our main aim is to create a knowledge graph from each input text. Coreference resolution helps remove the redundancy in the nodes and links the same entities, ensuring each entity corresponds to a single node in the graph. This results in a more accurate graph. A coreference resolution model from spaCy [7] called en_coreference_web_trf is used in this research which identifies and refers to the same entity (e.g., replacing the pronoun with its actual noun). Listing 4.1 shows the input data sample from the HotpotQA dataset and Listing 4.2 shows the output of performing coreference resolution in the input data sample.

Listing 4.1: Input Data Sample for Coreference Resolution.

```
{
    {
    'p1': {
    'title': 'Shirley Temple',
    'context': "Shirley Temple Black (April 23, 1928
       February 10, 2014) 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, she was
       named United States ambassador to Ghana and to
       Czechoslovakia and also served as Chief of Protocol
       of the United States."
    },
    'p2': {
    'title': 'Kiss and Tell (1945 film)',
    'context': "Kiss and Tell is a 1945 American comedy
```

[7] https://spacy.io/.

```
        film starring then 17-year-old Shirley Temple as
        Corliss Archer. In the film, two teenage girls cause
        their respective parents much concern when they
        start to become interested in boys. The parents'
        bickering about which girl is the worse influence
        causes more problems than it solves."
    }
    }
}
```

Listing 4.2: Output of Coreference Resolution.

```
{
    {
    'p1': {
    'title': 'Shirley Temple',
    'context': "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."
    },
    'p2': {
    'title': 'Kiss and Tell (1945 film)',
    'context': "Kiss and Tell (1945 film) is a 1945
        American comedy film starring then 17-year-old
```

```
        Shirley Temple as Corliss Archer. In Kiss and Tell
        (1945 film) , two teenage girls cause two teenage
        girls respective parents much concern when two
        teenage girls start to become interested in boys.
        their respective parents bickering about which girl
        is the worse influence causes more problems than The
        parents' bickering solves."
    }
    }
}
```

4.2.2 Creating Knowledge Graph Involving Stanford CoreNLP

The main aim was to involve Stanford CoreNLP to extract the entities from the input texts, create relationships between them, and finally generate a graph structure for each input text. To create the knowledge graph, we first have to start the CoreNLP server. The Open Information Extraction (OpenIE) [8] annotator is used as it can extract triplets from the input text. A triplet is a representation of a subject, a relation, and the object of the relation. Finally, the graph is visualized using the NetworkX [9]package in Python. Listing 4.3 is the sample input for knowledge graph generation where Table 4.1 are the extarcted triplets from this sample input involving Stanford CoreNLP server. Table 4.2 shows the node mapping and edge mapping from the extracted triplets. Figure 4.3 is the final knowledge graph representation of this input sample.

[8]https://stanfordnlp.github.io/CoreNLP/openie.html.
[9]https://networkx.org/.

Listing 4.3: 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.
```

Table 4.1: Extracted Triplets from Sample Input Involving Stanford CoreNLP.

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

4.2.3 Graph Representation Creation

After generating the knowledge graph, we created a graph representation of each knowledge graph by constructing PyG [10] compatible tensors. The triplets are converted to the following attributes.

- **Node Features**

  A node features is a dictionary mapping where each node (subject or object) of the graph is converted to an index number to convert human readable node labels to ma-

[10] ttps://pyg.org/.

Table 4.2: Triplets to Nodes and Edges Generation.

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

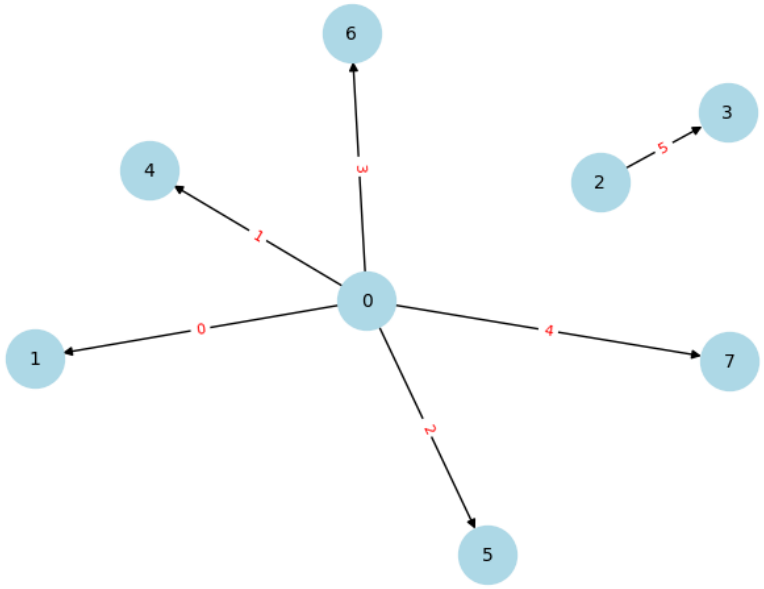


Figure 4.3: Graph Representation with Nodes and Edges.

chine readable index number. The embedding for each node is generated using the Sentence Transformer model all-Minim-L6-v2 [11].This is a Pytorch Tensor of shape [number of nodes in the graph, 384]. The value 384 comes from the output dimensionality of the all-Minim-L6-v2 Sentence Transformer model.

- **Edge Index**

  To represent the connection between source to destination of an edge, the edge index represents the connectivity. This is a Pytorch Tensor [12] of shape [2, number of edges].

- **Edge Attribute**

  The edge attribute represents the embeddings of each subject to object relations. The embedding for each attribute are generated using the Sentence Transformer model all-Minim-L6-v2. This is a Pytorch Tensor of shape [2, number of edges] where the embedding dimension is 384.

After generating the attributes from the triplets, each row of questions is encoded by the

[11] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.
[12] https://pytorch.org/docs/stable/tensors.html.

Table 4.3: Specification of Parameters for GAT Model.

Parameters	Values
Optimizer	Adam Optimizer
Learning Rate	0.001
Loss Function	Mean Squared Error (MSE)
Patience (early stopping)	15
Scheduler	ReduceLROnPlateau
Patience (ReduceLROnPlateau)	5
mode (ReduceLROnPlateau)	'min'
Weight Decay	$1 * 10^{-4}$
Epoch	200
Batch Size	32
Train, Validation, and Test Split	70:15:15

all-MiniLM-L6-v2 model to generate the space vector of the questions. Finally, the graph object representation is generated by creating a Data object from the PyTorch Geometric (PyG) [13] library. The graph data object is created by packaging the node features, edge index, edge attributes, and the target variable, where the target variable is the question in this research.

4.3 Graph Embedding Creation

The graph embedding is generated by the graph attention network (GAT) model. At first, we load the graph data object, then the model is trained, and the evaluation is done on a test set from the data object. The GAT architecture is shown in Figure 4.4. The training, testing and validation split is done as 70:15:15 ratio. During training, the Mean Squared Error (MSE) is used to calculate the loss. Early stopping and model checkpoint are also used. The model is trained for 200 epochs. The configuration of the GAT model to create graph object optimization is shown in Table 4.3.

The brief explanation of the GAT model architecture for graph embedding generation is shown in Figure 4.4.

[13] https://pytorch-geometric.readthedocs.io/en/latest/generated/torch_geometric.data.Data.html.

- **Input**

  The graph object, which holds the information of node features, edge index, and edge attribute, is the input for the GAT model.

- **Graph Convolution Layer**

  A total of five convolution layers are specified as graph convolution layers, where in the first layer, the total number of input features for each node is 384, which is the dimensionality of the feature vector of each node, and the total number of output features for each node per head is 256.

- **Attention Mechanism**

  A multi-head attention mechanism is used, as it allows learning different attention patterns. The number of independent attention head set to 8.

- **Dropout**

  A dropout layer is added after every Convolution layer and before the Layer Normalization layer, where the dropout probability of the neuron is 20% during training.

- **Layer Normalization**

  Before the dropout layer, normalization is added to normalize the output of the Convolution layer.

- **ELU Activation**

  ELU is an activation function used after the Convolution layer, so non-linearity is introduced in the model.

- **Global Mean Pooling**

  At the end, a global mean pooling layer is added, which is a pooling operation. After the final Convolution layer, when the final node embeddings are generated, this layer averages them for each graph in the batch.

- **Output**

  The final output is a single vector representation of each graph, whose size is batch size * 384.

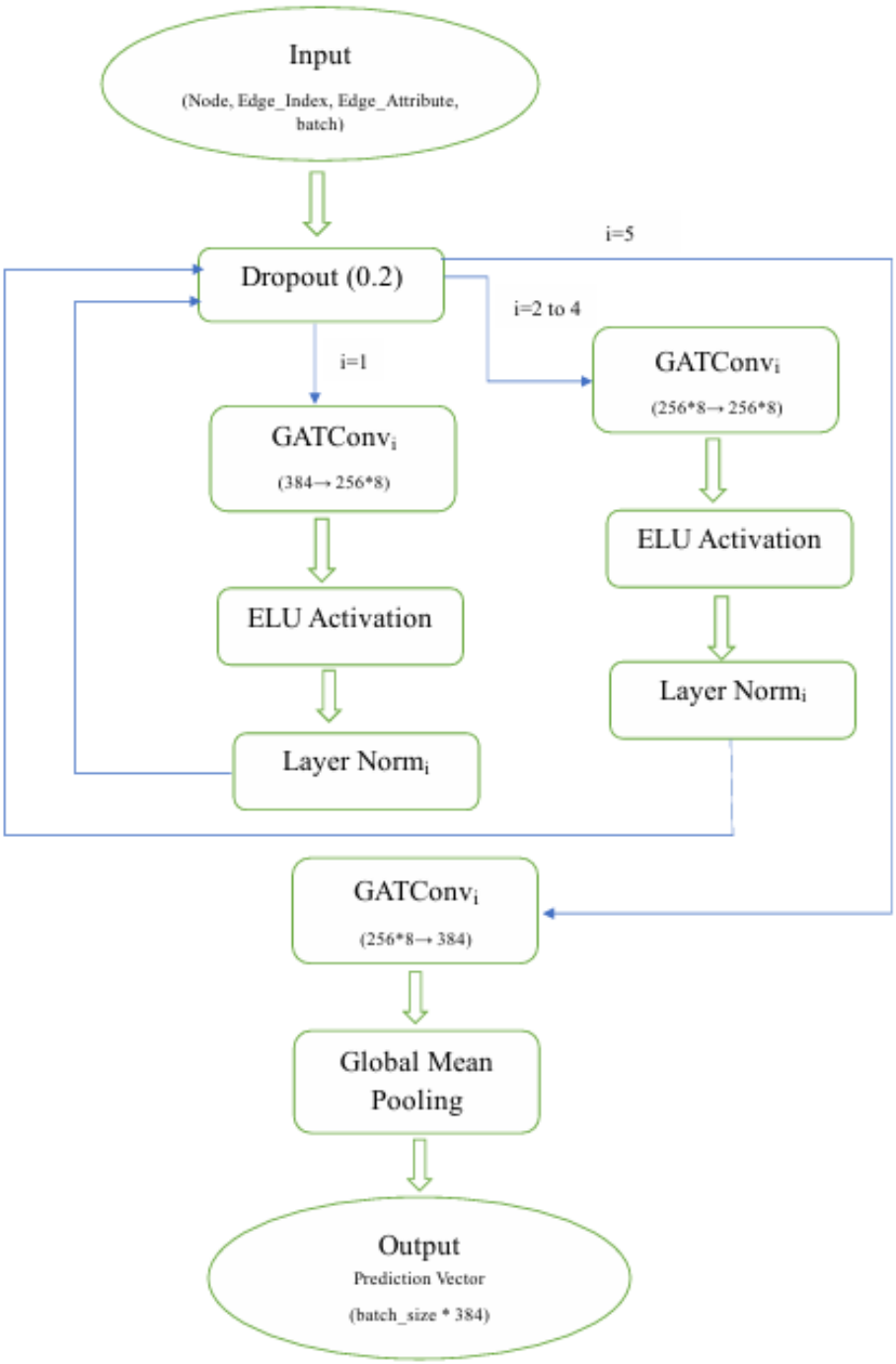


Figure 4.4: Visualization of GAT Model Architecture for Graph Embedding.

4.4 Input Text Embedding Creation

The input text is converted to a numerical representation using the Sentence Transformer. We utilize the all-Minim-L6-v2 model to generate the space vector of the input text, so the encoded input text is generated.

4.5 Combining of Input Text Embedding and Graph Embedding

The graph embedding is concatenated at the end of the input text embedding by torch.stack[14]. The torch.stack function is in the PyTorch[15] library, which is used to concatenate the input text embedding and the graph embedding along with a new dimension. Both input text embedding and graph text embedding have the same shape. In this research, torch.stack is used over torch.cat [16] as the torch.stack function is better for batching operations in deep learning.

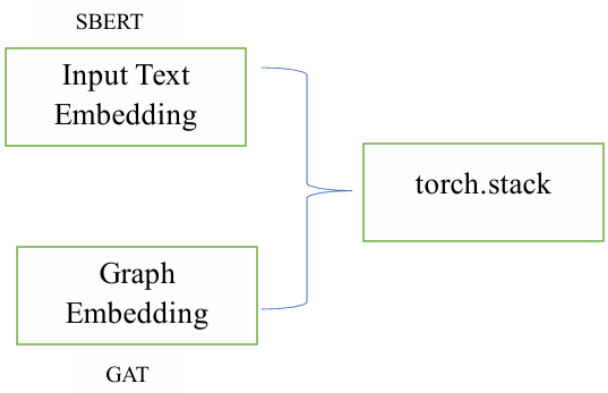


Figure 4.5: Concatenation of Input Text Embedding and Graph Embedding.

4.6 Encoder

The encoder of a large language model is used to improve the representation of input by utilizing a transformer architecture, which includes feed-forward neural networks and a

[14]https://pytorch.org/docs/stable/generated/torch.stack.html.
[15]https://pytorch.org/.
[16]https://pytorch.org/docs/stable/generated/torch.cat.html.

multi-head self-attention mechanism. As a backbone, we have utilized both BART and T5 as encoders separately to determine which model performs better. In both cases, we used the base models. To ensure that the text fits the input length of the model and to generate an attention mask for the input text, we have fed tokenization into the LLM encoder component. We have generated the attention mask from the input text and then the combined embedding layer is ready to be fed into the encoder of the LLMs (BART-base and T5-base). Figure 4.6 describes the encoder components and its input in this research.

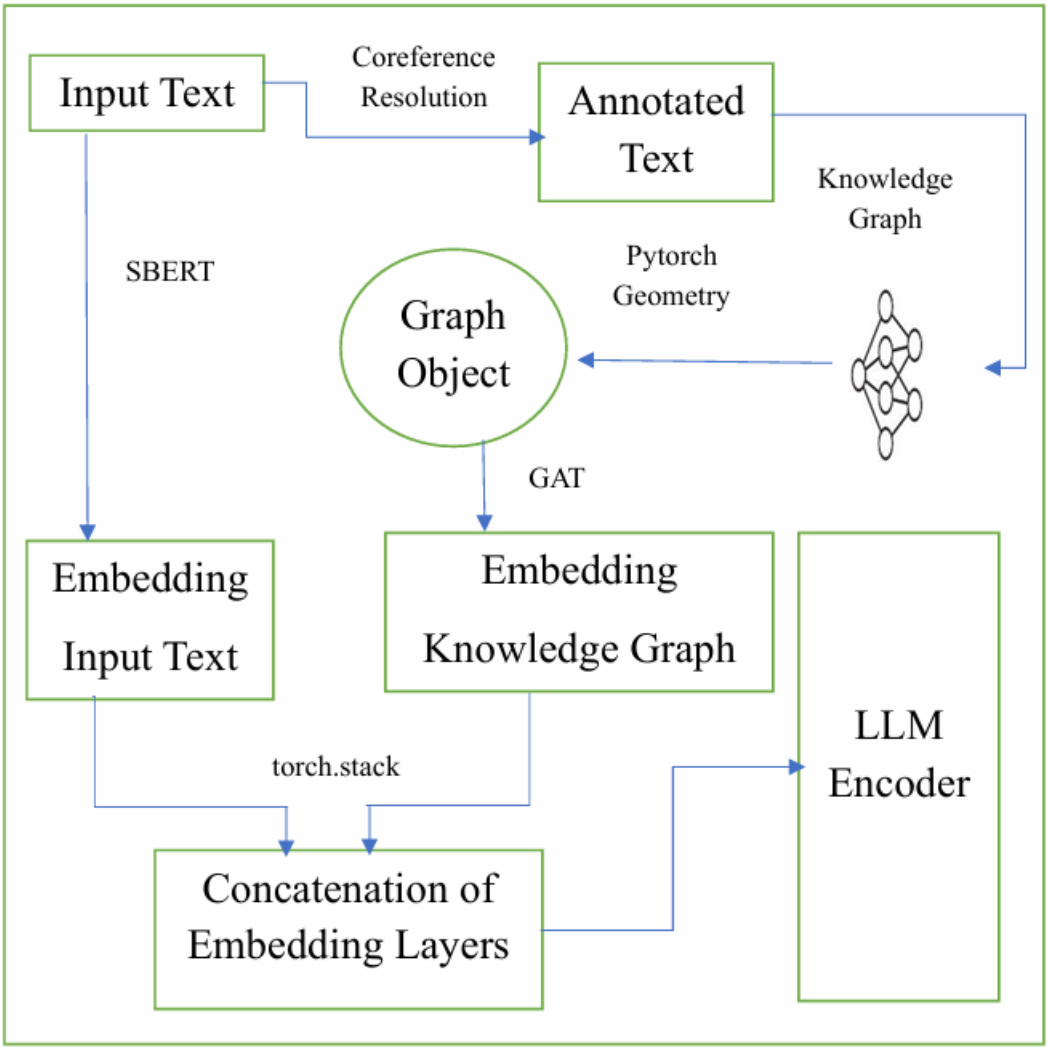


Figure 4.6: Encoder Part of KG4QG.

4.7 Decoder

The LLM decoder which follows the transformer architecture is used for autoregressive generation. That is to say, based on previous predictions, it produces questions token by token. It produces the prediction of likelihoods to generate the next token in the vocabulary. The decoder uses mask self-attention so the model ensures autoregressive masking, so the next prediction of word is based on the words that are previously examined in the sequence plus the current word. Figure 4.1 shows the entire architecture of KG4QG, which contains both the encoder and decoder. The encoder finally produces the context embeddings, which are the input of the decoder.

4.8 Execution Details

In this research, using KG4QG (Knowledge Graph for Question Generation), we have used both BART and T5. All of the implementation is done in Python programming language using PyTorch (Paszke et al., 2019). PyTorch is popular to researchers because of its flexibility, documentation, features, and efficiency. We have used the base models in both BART and T5.

The Facebook/BART-base model is used in this research. The Facebook/BART-base [17] is a pre-trained model which is developed by Facebook AI. This model combines a bidirectional encoder (like BERT) with an autoregressive decoder (like GPT). This model is not pre-trained for any specific task and can be used in different text generation tasks like summarization, machine translation, question answering and others. The total number of parameters of this model is approximately 139 millions and the maximum sequence length is 1024 tokens.

We have also used the T5-base [18] model in this research. The T5-base is a pre-trained model which is a base size version of the T5 model developed by Google (Raffel et al., 2023). This model is pre-trained on the span corruption task (Ye et al., 2024) using the C4

[17] https://huggingface.co/facebook/bart-base.
[18] https://huggingface.co/google-t5/t5-base.

dataset (Dodge et al., 2021). The total number of parameters of this model is approximately 220 million, and the maximum sequence length is 512 tokens.

4.8.1 Fine-tuning

To make the pre-trained models task-specific on multi-hop question generation, fine-tuning is done. The fine-tuning is done based on input texts and knowledge graphs. That means, for each input texts a knowledge graph is generated, and both the text and knowledge graph are concatenated for fine-tuning. The HotpotQA dataset is used for this process. For both BART and T5, the models are trained for 50 epochs, where early stopping is used as a regularization technique to prevent the model from overfitting. Based on validation loss the patience parameter is fixed as 3, which means if no change occurs in validation loss, after three epochs, the training will be stopped. The best outcome of the T5-base model is achieved after 50 epochs, where for the Facebook/BART-base model the best outcome is achieved after 20 epochs. The Adam optimizer (Kingma and Ba, 2017) is used, where the Cross-Entropy function (Mao et al., 2023) is used to calculate the loss. The learning rate is fixed as 10^{-4}. The whole configuration is shown in Table 4.4.

Table 4.4: Specification of parameters to fine-tune the models for KG4QG.

Model Name	**Parametrs**	**Value**
BART-base	Optimizer	AdamW
	Learning rate	1e−4
	Loss function	Cross-entropy
	Weight decay	0.01
	Epoch	20
	Patience (early stopping)	3
	Batch size	8
	Train, validation and test 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
	Patience (early stopping)	3
	Batch size	8
	Train, validation and test split	0.70 : 0.15 :0.15

4.8.2 Experiment Setup

The main computational resources of this research are the Narval and Cedar servers from Compute Canada [19]. The Narval server provides A100 GPUs and Cedar provides V100 GPUs. We have used the Distributed Data Parallet (DDP) [20] feature from PyTorch so that we can utilize 4 GPUs simultaneously. Details, code and implementations are available on our Github [21] repository.

[19] https://docs.alliancecan.ca/.
[20] https://docs.alliancecan.ca/wiki/PyTorch.
[21] https://github.com/AlHasibMahamud/KG4QG_MultihopQG.

Chapter 5

Experiments and Results

In this chapter, we will discuss the results we have obtained from evaluating our proposed methodology of KG4QG. At first, we discuss about the HotpotQA dataset which is used in our research. Then we discuss the performance of our model and the results based on the evaluation metrics.

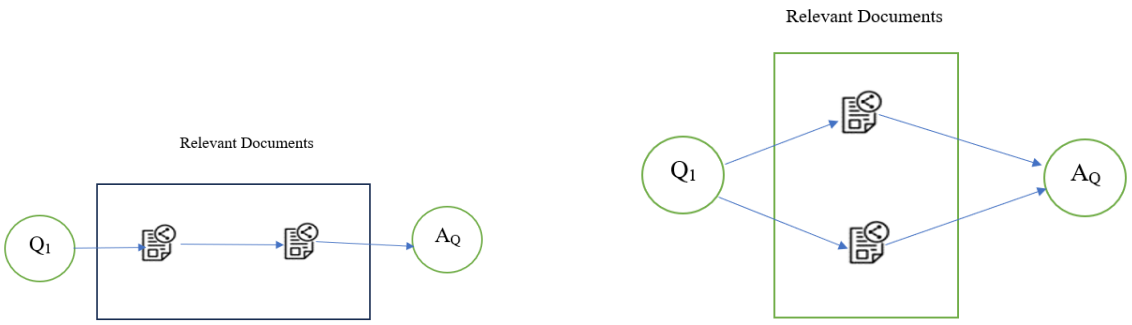
5.1 Dataset

In this research, we have utilized HotpotQA (Yang et al., 2018) dataset for training and evaluating our model. This dataset is consists of 113k samples which are collected from Wikipedia articles. This dataset was generated for the purpose to train question-answering (QA) systems to perform complex reasoning and provide explanations to answers. From the dataset, each sample consists of the following information:

- Unique ID for each sample
- Ten context paragraphs with title
- Reference Question
- Question Type (Bridge or Comparison)
- Question Difficulty
- Supporting Sentences

In the HotpotQA dataset, there are mainly two types of questions based on multi-hop reasoning tasks: bridge and comparison.

- Bridge: A bridge-type question links two remotely located supporting facts by utilizing the step-specific sub-question provided by the controller.
- Comparison: A comparison-type question involves comparing two or more elements, entities, or concepts to highlight similarities and differences between them.



(a) Bridge (b) Comparison

Figure 5.1: Answer Retrieval Path Types for Multi-hop Questions(Samadi and Rafiei, 2023).

Below Figures 5.2 and 5.3 are one of each instance for bridge-type and comparison-type questions sample from HotpotQA dataset. For a bridge-type question, it is required to integrate separate information from different contexts. The question of Figure 5.2 can be classified as bridge-type question as it is required to connect different information from two contexts which is described below.

- **Context 1** provides the information about Bailey Willams that he made his debut against Melbourne in round 82,016 at the Melbourne Cricket Ground.
- **Context 2** states that the Melbourne Cricket Ground (MCG), also known simply as ”The G”.

To get the answer of the reference question ”What is the nickname of the stadium where Bailey Willams played his debut AFL football game?”, it is required to bridge the above two information. The reasoning steps are given below.

- From Context 1, it is required to find the cricket ground name from where Bailey Willams debuted.

- From Context 2, it is essential to find the nickname of the cricket ground where Bailey Willams debuted.

Context 1:

Bailey Williams (born 10 October 1997) is an Australian rules footballer playing for the Western Bulldogs in the Australian Football League (AFL). He was drafted by the Western Bulldogs with their fourth selection and forty-eighth overall in the 2015 national draft. He made his debut in the thirty-two point win against Melbourne in round 8, 2016 at the Melbourne Cricket Ground.

Context 2:

The Melbourne Cricket Ground (MCG), also known simply as "The G", is an Australian sports stadium located in Yarra Park, Melbourne, Victoria. Home to the Melbourne Cricket Club, it is the 10th-largest stadium in the world, the largest in Australia, the largest in the Southern Hemisphere, the largest cricket ground by capacity, and has the tallest light towers of any sporting venue. The MCG is within walking distance of the city centre and is served by the Richmond railway station, Richmond, and the Jolimont railway station, East Melbourne. It is part of the Melbourne Sports and Entertainment Precinct.

Answer:

The G

Reference Question:

What is the nickname of the stadium where Bailey Williams played his debut AFL football game?

Figure 5.2: Bridge Type Question Sample.

As with bridge-type questions, it is essential to integrate information from different contexts. In this example we need to integrate information from above two contexts. From Context 1, we get the information that at the Melbourne Cricket Ground, Bailey Williams had his debut and from Context 2, we can see that Melbourne Cricket Ground is sim-

ply known as the G. By integrating these information, we ensure multi-hop reasoning and can get the answer of this question. The question of Figure 5.3 can be classified as a

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 5.3: Comparison Type Question Sample.

comparison-type question as in comparison type question it is required to gather information from different contexts and compare any property that both of the contexts share. From this sample:

- **Context 1** provides the information that the Con Mine entered production in 1938.
- **Context 2** provides the information that for the Giant Mine the true extent of the gold deposits were not known until 1944.

To get the answer of the reference question "Which mine started production sooner, the Con Mine or the Giant Mine?", it is required to do comparison of the above two information. The reasoning path is given below.

- From Context 1, it is required to find the time when the Con Mine started production.
- From Context 2, it is needed to be understood the Giant Mine started mine production before or after the Con Mine.

As the comparison-type question, it is required to do comparison between information from different contexts. In this sample we need to do comparsion between above two contexts. From Context 1, we get the information that the Con Mine started mine production in 1938. On the other hand, from Context 2 we can see that the Giant Mine started mine production after 1938 as the true extent of the gold deposits were not known until 1944. After comparing these information the multi-hop reasoning is ensured and we can get the answer of this question.

5.1.1 Data Cleaning

For data cleaning, we followed the path according to Emerson and Chali (2023a). In the HotpotQA dataset, each sample consists of 10 paragraphs where only two paragraphs are supportive paragraphs containing the actual evidence which are needed to answer the question, while the other eight paragraphs are distractor paragraphs that are not relevant to answer the question, which we discard from the each sample. We have also discarded supporting sentence information from the dataset so the model can learn to generate questions from the complete input paragraphs. We have removed samples from the dataset which is containing yes or no answer. According to Su et al. (2020), it is important to filter out all yes/no answer based data samples from the HotpotQA dataset when focusing on the multi-hop ability as yes/no answer based questions are easier and requires less reasoning. After removing those samples we have ensured complex multi-hop reasoning in our dataset. After removing those samples, we have total 91,911 samples where the number of bridge type

question samples is 78,909 and the number of comparison type question samples is 13,002. After filtering out the dataset, we have partitioned the dataset into three parts: training dataset, validation dataset, and testing dataset. The training dataset is consists of 70% of the whole dataset where validation dataset and testing dataset each consists of 15% of the whole dataset. Before splitting, we shuffled the dataset randomly so training, testing and validation datasets each contain a balanced distribution on sample question types. The purpose of creating training, testing, and validation datasets are given below.

- Training dataset: To adjust the parameters of the model to minimize the error based on this data.
- Validation dataset: To tune hyperparameters and prevent overfitting.
- Testing dataset: To evaluate the model after training to see how the model performs on unseen data.

Table 5.1: HotpotQA: Train, Validation, and Test Distribution.

	Training	**Validation**	**Test**
No of Samples	64337	13787	13787

5.2 Evaluation Results and Discussion

To evaluate the performance of KG4QG, we have utilized automated evaluation metrics. We have compared the generated questions from the model with the reference questions of the test dataset based on the evaluation metrics. The used evaluation metrics are BLEU (Papineni et al., 2002), ROUGE (Lin, 2004),METEOR (Lavie and Agarwal, 2007), and BERTScore (Zhang et al., 2020). These metrics are chosen based on their widespread utilization in the question generation task. Evaluation of our model based on these metrics provides us the path to compare the effectiveness of KG4QG to previous research works on multi-hop question generation. The performance of our model KG4QG is shown in

Table 5.2 based on the evaluation metrics of BLEU, ROUGE, and METEOR. Based on BERTScore, the model performance is shown in Table 5.3.

Table 5.2: Performance of KG4QG model.

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L	METEOR
T5 backbone	47.08	39.11	36.32	34.27	43.42	33.88	41.48	40.41
BART backbone	**53.38**	**44.81**	**41.24**	**38.56**	**49.58**	**38.05**	**47.11**	**46.08**

Table 5.3: Performance of KG4QG model based on BERT-Score.

	Precison	Recall	F1-Score
T5 backbone	89.70%	89.40%	89.54%
BART backbone	**91.20%**	**90.73%**	**90.95%**

Table 5.4: Performance Comparison between KG4QG and Existing Models.

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

To demonstrate the efficiency of our model KG4QG with BART backbone and T5 backbone, we performed a comparison with existing models on multi-hop question generation 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). MulQG, GNET-QG, DCQG, and KGEL models are chosen for comparison because these models are graph based approaches where KGEL is a

knowlegde graph approach. Also, we have compared KG4QG with CQG, TASE-CoT, and Transformer based approaches as they achieved significant results in evaluation metrics. From Table 5.4, we can see that our model KG4QG outperforms across all metrics comparing the existing models expect on BLEU-1 where the difference with MultiFactor is 0.79 and METEOR where the difference with GNET-QG is 3.79. Our model KG4QG outperforms other methodologies because it combines the strengths of structured and unstructured data representations. Text embedding captures contextual information, whereas knowledge graph embedding encodes explicit relationships. As multi-hop question generation requires reasoning over multiple information, a knowledge graph properly represents the connections between facts. Contextual information from text and explicit relationships from the knowledge graph are generating high-quality, relevant multi-hop questions.

Chapter 6

Conclusion

In this chapter, we will discuss the summary of the research presented in this thesis. Then we will discuss the future potential work scope that is required for further improvement of multi-hop question generation system.

6.1 Summary

In this thesis, we have proposed a new approach for automatic multi-hop question generation system to address the challenges of multi-hop question generation that requires multiple reasoning over information. We utilize the HotpotQA dataset which is mostly popular and highly used in research community for multi-hop question generation and multi-hop question answering tasks. Our approach is based on Knowledge Graph generation from the input text where GAT is leveraged for graph embedding generation which is concatenated with input text embedding. BART and T5 both are used in this thesis as large language models where BART provides significant result over T5. The integration of GAT, SBERT, and BART has shown an important contribution in the field of multi-hop question generation where the evaluation metrics have shown a great impact of results comparing to the existing models. In the appendix, we have shown the quality of generated questions of the model comparing to the reference questions where comparison are also given between predicted questions and reference questions.

6.2 Fututre Work

As multi-hop question generation is relatively a new area of research, there are still some directions to make improvements in future research.

- Low Resource Multi-hop Question Generation:

  There are multiple languages which do not have proper datasets for doing effective research on multi-hop question generation system. Enriching cross lingual and multilingual capacity can help to improve the ability of multi-hop question generation system for low resource languages.

- Prompting Techniques:

  Utilizing different prompting techniques like Vanilla prompting, tree of thoughts prompting, graph of thoughts prompting may improve the performance of LLMs for multi-hop question generation.

- Human in the loop (HTL):

  Leveraging human feedback during the time of training process can be effective to generate accurate multi-hop questions.

- Domain Specific Multi-hop Question Generation:

  During fine-tuning, leveraging specific dataset based on domain can be helpful for specific domains like medicine, agriculture, economics, science etc.

- Evaluation Metrics for Multi-hop Questions:

  Proposing new evaluation metrics which can measure multi-hop relevance, reasoning, and answerability will be very impactful to measure the quality of predicted questions.

- Mitigating Bias in Multi-hop Question Generation:

  Analyzing potential biases (like gender, demography, culture) in the generated questions and developing mitigating techniques is a good idea of further research in this

field.

- Multimodal Multi-hop Question Generation:

  Generating questions based on input text and image or table can be very impactful in different fields like medicine, education and others.

Bibliography

Abien Fred Agarap. Deep Learning using Rectified Linear Units (ReLU), 2019. URL https://arxiv.org/abs/1803.08375.

Amit Alfassy, Assaf Arbelle, Oshri Halimi, Sivan Harary, Roei Herzig, Eli Schwartz, Rameswar Panda, Michele Dolfi, Christoph Auer, Kate Saenko, PeterW. J. Staar, Rogerio Feris, and Leonid Karlinsky. FETA: Towards Specializing Foundation Models for Expert Task Applications, 2022.

Felipe Almeida and Geraldo Xexéo. Word Embeddings: A Survey, 2023. URL https://arxiv.org/abs/1901.09069.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate, 2016. URL https://arxiv.org/abs/1409.0473.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In Kristina Toutanova and Hua Wu, editors, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1023. URL https://aclanthology.org/P14-1023/.

Anya Belz and Ehud Reiter. Comparing Automatic and Human Evaluation of NLG Systems. 01 2006.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.*, 3(null):1137–1155, March 2003. ISSN 1532-4435.

Lukas Biermann, Sebastian Walter, and Philipp Cimiano. A Guided Template-Based Question Answering System over Knowledge Graphs. In *International Conference Knowledge Engineering and Knowledge Management*, 2018. URL https://api.semanticscholar.org/CorpusID:53957812.

Miroslav Blšták and Viera Rozinajová. Automatic question generation based on sentence structure analysis using machine learning approach. *Natural Language Engineering*, 28 (4):487–517, June 2021. ISSN 1469-8110. doi: 10.1017/s1351324921000139. URL http://dx.doi.org/10.1017/S1351324921000139.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In Lluís Màrquez, Chris

Callison-Burch, and Jian Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Zhen Cao, Pengfei Li, Yong Zhong, and Shaobo Li. Multi-Hop Question Generation with Knowledge Graph-Enhanced Language Model. *Applied Sciences*, 13, 05 2023. doi: 10.3390/app13095765.

Akhil Chaudhary, Enayat Rajabi, Somayeh Kafaie, and Evangelos Milios. Fact retrieval from knowledge graphs through semantic and contextual attention. *Expert Systems with Applications*, 282:127612, 2025. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2025.127612. URL https://www.sciencedirect.com/science/article/pii/S0957417425012345.

Wei Chen and Gregory Aist. Generating Questions Automatically from Informational Text. 2009. URL https://api.semanticscholar.org/CorpusID:15246592.

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. Guiding the Growth: Difficulty-Controllable Question Generation through Step-by-Step Rewriting. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *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, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.465. URL https://aclanthology.org/2021.acl-long.465/.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, 2014. URL https://arxiv.org/abs/1406.1078.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari,

Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling Language Modeling with Pathways, 2022. URL https://arxiv.org/abs/2204.02311.

Alexander Clark, Chris Fox, and Shalom Lappin, editors. *The Handbook of Computational Linguistics and Natural Language Processing*. Blackwell Handbooks in Linguistics. Wiley-Blackwell, 2012. ISBN 9781118347188. URL https://books.google.ca/books?id=6BJOwNHD1osC.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Kaustubh D. Dhole and Christopher D. Manning. Syn-QG: Syntactic and Shallow Semantic Rules for Question Generation, 2022. URL https://arxiv.org/abs/2004.08694.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus, 2021. URL https://arxiv.org/abs/2104.08758.

Xinya Du, Junru Shao, and Claire Cardie. Learning to Ask: Neural Question Generation for Reading Comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1123. URL https://aclanthology.org/P17-1123.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. Question Generation for Question Answering. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1090. URL https://aclanthology.org/D17-1090.

John Emerson and Yllias Chali. Efficient Multi-hop Question Generation. *Procedia Computer Science*, 222:217–222, 2023a. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2023.08.159. URL https://www.sciencedirect.com/science/article/pii/S1877050923009249.

John Emerson and Yllias Chali. Multi-hop Question Generation without Supporting Fact Information. In *The International FLAIRS Conference Proceedings*, volume 36, 2023b.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuanjing Huang. CQG: A Simple and Effective Controlled Generation Framework for Multi-hop Question Generation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.475. URL https://aclanthology.org/2022.acl-long.475/.

Michael Flor and Brian Riordan. A Semantic Role-based Approach to Open-Domain Automatic Question Generation. In Joel Tetreault, Jill Burstein, Ekaterina Kochmar, Claudia Leacock, and Helen Yannakoudakis, editors, *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 254–263, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-0530. URL https://aclanthology.org/W18-0530/.

Bektemyssova Gulnara and Aidos Sabdenov. Building a Semantic Knowledge Graph Search Model for Finding a Causal Answer. *Revue d'Intelligence Artificielle*, 38:243–250, 02 2024. doi: 10.18280/ria.380125.

Michael Heilman and Noah A. Smith. Good Question! Statistical Ranking for Question Generation. In Ron Kaplan, Jill Burstein, Mary Harper, and Gerald Penn, editors, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/N10-1086/.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9 (8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Samin Jamshidi and Yllias Chali. 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, January 2025. International Committee on Computational Linguistics. URL https://aclanthology.org/2025.genaik-1.3/.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification, 2016. URL https://arxiv.org/abs/1607.01759.

Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. Automatic Question Generation using Relative Pronouns and Adverbs. In Vered Shwartz, Jeniya Tabassum, Rob Voigt, Wanxiang Che, Marie-Catherine de Marneffe, and Malvina Nissim, editors, *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-3022. URL https://aclanthology.org/P18-3022/.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Philippe Laban, Chien-Sheng Wu, Lidiya Murakhovs'ka, Wenhao Liu, and Caiming Xiong. Quiz Design Task: Helping Teachers Create Quizzes with Automated Question Generation, 05 2022.

Alon Lavie and Abhaya Agarwal. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In Chris Callison-Burch, Philipp Koehn, Cameron Shaw Fordyce, and Christof Monz, editors, *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/W07-0734/.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703/.

Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

Chin-Yew Lin and Franz Josef Och. 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, July 2004. doi: 10.3115/1218955.1219032. URL https://aclanthology.org/P04-1077/.

Zefeng Lin, Weidong Chen, Yan Song, and Yongdong Zhang. Prompting Few-shot Multi-hop Question Generation via Comprehending Type-aware Semantics. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3730–3740, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.236. URL https://aclanthology.org/2024.findings-naacl.236/.

Tianyu Liu, Bingzhen Wei, Baobao Chang, and Zhifang Sui. *Large-Scale Simple Question Generation by Template-Based Seq2seq Learning*, pages 75–87. 01 2018. ISBN 978-3-319-73617-4. doi: 10.1007/978-3-319-73618-1_7.

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

Di Lu, Shihao Ran, Joel Tetreault, and Alejandro Jaimes. Event Extraction as Question Generation and Answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1666–1688, Toronto,

Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.143. URL https://aclanthology.org/2023.acl-short.143.

Xiyao Ma, Qile Zhu, Yanlin Zhou, Xiaolin Li, and Dapeng Wu. Improving Question Generation with Sentence-level Semantic Matching and Answer Position Inferring, 2020. URL https://arxiv.org/abs/1912.00879.

Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, volume 28, 2013. URL https://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf.

Himanshu Maheshwari, Sumit Shekhar, Apoorv Saxena, and Niyati Chhaya. Open-World Factually Consistent Question Generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2390–2404, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.151. URL https://aclanthology.org/2023.findings-acl.151.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. Cross-Entropy Loss Functions: Theoretical Analysis and Applications, 2023. URL https://arxiv.org/abs/2304.07288.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, 2013a. URL https://arxiv.org/abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality, 2013b. URL https://arxiv.org/abs/1310.4546.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. Efficient and Robust Question Answering from Minimal Context over Documents, 2018. URL https://arxiv.org/abs/1805.08092.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large Language Models: A Survey, 2025. URL https://arxiv.org/abs/2402.06196.

Alireza Mohammadshahi, Thomas Scialom, Majid Yazdani, Pouya Yanki, Angela Fan, James Henderson, and Marzieh Saeidi. RQUGE: Reference-Free Metric for Evaluating Question Generation by Answering the Question. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6845–6867, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.428. URL https://aclanthology.org/2023.findings-acl.428.

Lluís Màrquez, Xavier Carreras, Kenneth Litkowski, and Suzanne Stevenson. Semantic Role Labeling: An Introduction to the Special Issue. *Computational Linguistics*, 34: 145–159, 06 2008. doi: 10.1162/coli.2008.34.2.145.

Alireza Naeiji, Aijun An, Heidar Davoudi, Marjan Delpisheh, and Muath Alzghool. Question Generation Using Sequence-to-Sequence Model with Semantic Role Labels. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2830–2842, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.207. URL https://aclanthology.org/2023.eacl-main.207.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. November 2016. URL https://www.microsoft.com/en-us/research/publication/ms-marco-human-generated-machine-reading-comprehension-dataset/.

Shinhyeok Oh, Hyojun Go, Hyeongdon Moon, Yunsung Lee, Myeongho Jeong, Hyun Seung Lee, and Seungtaek Choi. Evaluation of Question Generation Needs More References. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6358–6367, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.396. URL https://aclanthology.org/2023.findings-acl.396.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

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. PyTorch: An Imperative Style, High-Performance Deep Learning Library, 2019. URL https://arxiv.org/abs/1912.01703.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162/.

Marcin Pietrasik, Marek Reformat, and Anna Wilbik. Hierarchical Blockmodelling for Knowledge Graphs, 2024. URL https://arxiv.org/abs/2408.15649.

Judita Preiss. Ruslan Mitkov, Anaphora Resolution. Longman. 2002. ISBN 0-582-32505-6. Price £49.99 (hardback). xiii + 220 pages. *Natural Language Engineering*, 11:439 – 440, 12 2005. doi: 10.1017/S1351324905214006.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. Dynamically Fused Graph Network for Multi-hop Reasoning. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6140–6150, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1617. URL https://aclanthology.org/P19-1617/.

Roni Rabin, Alexandre Djerbetian, Roee Engelberg, Lidan Hackmon, Gal Elidan, Reut Tsarfaty, and Amir Globerson. Covering Uncommon Ground: Gap-Focused Question Generation for Answer Assessment. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 215–227, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.20. URL https://aclanthology.org/2023.acl-short.20.

Alec Radford and Karthik Narasimhan. Improving Language Understanding by Generative Pre-Training. 2018. URL https://api.semanticscholar.org/CorpusID:49313245.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, 2023. URL https://arxiv.org/abs/1910.10683.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv e-prints*, art. arXiv:1606.05250, 2016.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Jian Su, Kevin Duh, and Xavier Carreras, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264/.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, 2019. URL https://arxiv.org/abs/1908.10084.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986. ISBN 026268053X.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Christian Moldovan. The First Question Generation Shared Task Evaluation Challenge. In John Kelleher, Brian Mac Namee, and Ielka van der Sluis, editors, *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics, July 2010. URL https://aclanthology.org/W10-4234/.

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, November 1975. ISSN 0001-0782. doi: 10.1145/361219.361220. URL https://doi.org/10.1145/361219.361220.

Mohammadreza Samadi and Davood Rafiei. Performance Prediction for Multi-hop Questions. *arXiv preprint arXiv:2308.06431*, 2023.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional Attention Flow for Machine Comprehension, 2018. URL https://arxiv.org/abs/1611.01603.

Murray Shanahan. Talking About Large Language Models, 2023. URL https://arxiv.org/abs/2212.03551.

Dan Su, Yan Xu, Wenliang Dai, Ziwei Ji, Tiezheng Yu, and Pascale Fung. Multi-hop Question Generation with Graph Convolutional Network. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4636–4647, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.416. URL https://aclanthology.org/2020.findings-emnlp.416/.

Kai Sun, Jiapu Wang, Huajie Jiang, Yongli Hu, and Baocai Yin. Query-Enhanced Adaptive Semantic Path Reasoning for Inductive Knowledge Graph Completion, 2024. URL https://arxiv.org/abs/2406.02205.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. Answer-focused and Position-aware Neural Question Generation. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1427. URL https://aclanthology.org/D18-1427.

Yanli Sun. Mining the Correlation between Human and Automatic Evaluation at Sentence Level. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010. European Language Resources Association (ELRA). URL https://aclanthology.org/L10-1051/.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks, 2014. URL https://arxiv.org/abs/1409.3215.

Yachen Tang, Tingting Liu, Guangyi Liu, Jie Li, Renchang Dai, and Chen Yuan. Enhancement of Power Equipment Management Using Knowledge Graph. In *2019 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)*, pages 905–910. IEEE, 2019.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien

Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, 2023. URL https://arxiv.org/abs/2302.13971.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. NewsQA: A Machine Comprehension Dataset. In Phil Blunsom, Antoine Bordes, Kyunghyun Cho, Shay Cohen, Chris Dyer, Edward Grefenstette, Karl Moritz Hermann, Laura Rimell, Jason Weston, and Scott Yih, editors, *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2623. URL https://aclanthology.org/W17-2623.

P. D. Turney and P. Pantel. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188, February 2010. ISSN 1076-9757. doi: 10.1613/jair.2934. URL http://dx.doi.org/10.1613/jair.2934.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2023. URL https://arxiv.org/abs/1706.03762.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, 2018. URL https://arxiv.org/abs/1710.10903.

Zehua Xia, Qi Gou, Bowen Yu, Haiyang Yu, Fei Huang, Yongbin Li, and Nguyen Cam-Tu. Improving Question Generation with Multi-level Content Planning. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 800–814, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.57. URL https://aclanthology.org/2023.findings-emnlp.57/.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Xuchen Yao. Question Generation with Minimal Recursion Semantics. 2010. URL https://api.semanticscholar.org/CorpusID:18796690.

Xuchen Yao, Gosse Bouma, Yi Zhang, Paul Piwek, and Kristy Boyer. Semantics-based Question Generation and Implementation. *Dialogue Discourse*, 3, 03 2012. doi: 10.5087/dad.2012.202.

Ke Ye, Heinrich Jiang, Afshin Rostamizadeh, Ayan Chakrabarti, Giulia DeSalvo, Jean-François Kagy, Lazaros Karydas, Gui Citovsky, and Sanjiv Kumar. SpacTor-T5: Pre-training T5 Models with Span Corruption and Replaced Token Detection, 2024. URL https://arxiv.org/abs/2401.13160.

Ruqing Zhang, Jiafeng Guo, Lu Chen, Yixing Fan, and Xueqi Cheng. A Review on Question Generation from Natural Language Text. *ACM Trans. Inf. Syst.*, 40(1), sep 2021. ISSN 1046-8188. doi: 10.1145/3468889. URL https://doi.org/10.1145/3468889.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating Text Generation with BERT, 2020. URL https://arxiv.org/abs/1904.09675.

Appendix A

Evaluation of Model Prediction

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: What is the parent company of the company that created Mickey Mouse Club?
Reference Question: What American company that oversees various websites and interactive media owned by The Walt Disney Company also created Mickey Mouse Toddler, Preschool, and Kindergarten video Games?
Comparison: The answer of the predicted question and reference question can be different as the predicted question is asking the parent company that created Mickey Mouse Club but the reference company is asking the company that oversees various websites and interactive media, and also created those specific Mickey Mouse educational video games. The predicted question is more straightforward by asking ”the parent company of the company that created Mickey Mouse Club?” but the reference question is lengthy.

Example 3
Predicted Question: Who was born earlier, Jean-Jacques Favier or Reinhold Ewald?
Reference Question: Who was born first, Reinhold Ewald or Jean-Jacques Favier?
Comparison: Both questions are almost identical in content.

Example 4
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 5
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 6
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 scenario are same.

Example 7
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 8
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.

Example 9
Predicted Question: Who was born first, Maurizio Cheli or Hans Schlegel?
Reference Question: who is older Hans Schlegel or Maurizio Cheli?
Comparison: The predicted question and the reference question both are asking essentially the same information. The predicted question ”who was born first” is equivalent to the reference question ”who is older”.

Overall Assessment

Clarity: In some cases, the predicted questions are clearer compared to the reference questions.
Completeness: The reference questions are more detailed.
Relevance: In some cases, the predicted questions are more relevant to the input contexts.
Multi-hop Reasoning: Predicted questions and reference questions both have ensured multi-hop reasoning that required multiple connections of information from different information to generate the answers of the questions