Context-Aware Auto-Encoded Graph Neural Model for Dynamic Question Generation using NLP

SURESH DARA, Department of Computer Science and Engineering, PACE Institute of Technology & Sciences, India
CH. SRINIVASULU, Department of Computer Science and Engineering, Institute of Aeronautical Engineering (IARE)., India
CH MADHU BABU, Department of CSE, B V Raju Institute of Technology., India
ANANDA RAVURI, Senior Software Engineer Intel Corporation Hillsboro, 97124 USA
TIRUMALA PARUCHURI, Department of Management Information System, Jazan University, Kingdom of Saudi Arabia
ABHISHEK SINGH KILAK, Department of CSE, Engineering College Bikaner, India
ANKIT VIDYARTHI*, Jaypee Institute of Information Technology Noida, India

Question generation is an important task in natural language processing that involves generating questions from a given text. This paper proposes a novel approach for dynamic question generation using a context-aware auto-encoded graph neural model. Our approach involves constructing a graph representation of the input text, where each node in the graph corresponds to a word or phrase in the text, and the edges represent the relationships between them. We then use an auto-encoder model to learn a compressed representation of the graph that captures the most important information in the input text. Finally, we use the compressed graph representation to generate questions by dynamically selecting nodes and edges based on their relevance to the context of the input text. We evaluate our approach on four benchmark datasets (SQuAD, Natural Questions, TriviaQA, and QuAC) and demonstrate that it outperforms existing state-of-the-art methods for dynamic question generation. In the experimentation, to evaluate the result four performance metrics are used i.e. BLEU, ROUGE, F1-Score, and Accuracy. The result of the proposed approach yields an accuracy of 92% on the SQuAD dataset, 89% with QuAC, and 84% with TriviaQA. while on the natural questions dataset, the model gives 79% accuracy. Our results suggest that the use of graph neural networks and auto-encoder models can significantly improve the accuracy and effectiveness of question generation in NLP. Further research in this area can lead to even more sophisticated models that can generate questions that are even more contextually relevant and natural-sounding.

CCS Concepts: • **Computing methodologies** → **Natural language processing**; *Machine learning*; • **Information systems** → Information retrieval.

Authors' addresses: Suresh Dara, darasuresh@live.in, Department of Computer Science and Engineering, PACE Institute of Technology & Sciences, Ongole, A.P, India; CH. Srinivasulu, schennupalli@gmail.com, Department of Computer Science and Engineering, Institute of Aeronautical Engineering (IARE)., Hyderabad, India; CH Madhu Babu, madhubabu6585@gmail.com, Department of CSE, B V Raju Institute of Technology., Narsapur, 502313 Telangana, India; Ananda Ravuri, ananda.ravuri@gmail.com,ananda.ravuri@intel.com, Senior Software Engineer Intel Corporation Hillsboro, Oregon, 97124 USA; Tirumala Paruchuri, tirumalparuchuri@gmail.com, Department of Management Information System, Jazan University, Jazan, Kingdom of Saudi Arabia; Abhishek Singh Kilak, abhishekkilak@gmail.com, Department of CSE, Engineering College Bikaner, Rajasthan, India; Ankit Vidyarthi*, dr.ankit.vidyarthi@gmail.com, Jaypee Institute of Information Technology Noida, Department of CSE&IT, India.

2375-4699/2023/10-ARTxxx $15.00
https://doi.org/10.1145/3626317

Additional Key Words and Phrases: Natural Language Processing, Graph Neural Network, Question-answer generation, Auto Encoder, Information Processing

1 INTRODUCTION

NLP stands for Natural Language Processing, which is a subfield of artificial intelligence that deals with the interaction between computers and human language. NLP involves developing algorithms and models that can process, understand, and generate natural language text and speech. Some of the applications of NLP include language translation, speech recognition, sentiment analysis, text classification, information retrieval, and question-answering. NLP is a rapidly growing field with many exciting research directions, including deep learning models, language generation models, and multimodal processing.

Question-Answer generation (QAG) is an important task in natural language processing that involves generating questions from a given text [4]. This task is challenging because it requires the model to not only understand the meaning of the text but also to identify the most salient information and generate questions that are contextually relevant and natural-sounding. QAG has numerous applications in education, dialogue systems, and information retrieval. The goal of question generation is to produce natural-sounding questions that are relevant to the given text and that capture the most important information in it.

Existing approaches to question generation typically use template-based methods or rule-based methods that require pre-defined templates or rules for generating questions [23]. While these methods can be effective for generating simple questions, they often struggle to generate more complex questions that require a deeper understanding of the context and meaning of the input text.

Recently, deep learning models have been successfully applied to a variety of application areas using spoken languages by the communities [18]. The most commonly used deep models which are found in these applications are recurrent neural networks, convolutional neural networks, attention-based models, and Graph Neural Networks.

A Graph Neural Network (GNN) is a type of neural network that is designed to operate on graph-structured data. In a graph, nodes represent entities or objects, and edges represent relationships or connections between them. GNNs leverage this structure by learning to propagate information between nodes through the edges. GNNs have been successfully applied in various domains, including social network analysis, recommendation systems, computer vision, and natural language processing [33]. In NLP, GNNs have been used for tasks such as sentence classification, relation extraction, and sentiment analysis.

Besides this, there are other neural models that were used in the past for sentiment analysis on various languages like Hindi, English, Urdu, and other Asian and non-Asian languages. Those deep models were also utilized for QAG at various places. However, all such models often generate questions that are generic and not tailored to the specific context of the input text.

To address this limitation, we propose a novel approach for dynamic question generation using a context-aware auto-encoded graph neural model. Our approach involves constructing a graph representation of the input text, where each node in the graph corresponds to a word or phrase in the text, and the edges represent the relationships between them. We then use an auto-encoder model to learn a compressed representation of the graph that captures the most important information in the input text. Finally, we use the compressed graph representation to generate questions by dynamically selecting nodes and edges based on their relevance to the context of the input text.

Our approach is designed to be context-aware and dynamic, which means that it can generate questions that are tailored to the specific context of the input text and that can be adjusted based on the user's feedback. We

evaluate our approach on several benchmark datasets and demonstrate that it outperforms existing state-of-the-art methods for dynamic question generation. Our results suggest that the use of graph neural networks and auto-encoder models can significantly improve the accuracy and effectiveness of question generation in NLP.

Also, the proposed approach has several advantages over existing methods. First, it does not rely on pre-defined templates or rules, making it more flexible and adaptable to different contexts and domains. Second, it captures the complex relationships between words and phrases in the input text, allowing it to generate more contextually relevant questions. Finally, it can dynamically adjust the question generation process based on the context of the input text, resulting in more natural-sounding questions.

In this paper, we describe our proposed approach in detail and provide experimental results demonstrating its effectiveness. We also discuss potential future directions for research in this area, including the use of more sophisticated graph representations and the incorporation of external knowledge sources to improve question generation performance. The main contributions of the work are summarized as follows:

(1) A novel approach for dynamic question generation using a context-aware auto-encoded graph neural model: The proposed approach constructs a graph representation of the input text, uses an auto-encoder model to learn a compressed representation of the graph, and then generates questions by dynamically selecting nodes and edges based on their relevance to the context of the input text. This approach is more flexible and adaptable to different contexts and domains compared to existing template-based or rule-based methods.
(2) Integration of graph neural networks and auto-encoder models: The proposed approach integrates graph neural networks and auto-encoder models to capture the complex relationships between words and phrases in the input text and generate more contextually relevant questions.
(3) Context-awareness: The proposed approach dynamically adjusts the question generation process based on the context of the input text, resulting in more natural-sounding questions.
(4) Evaluation on benchmark datasets: The proposed approach is evaluated on several benchmark datasets and demonstrates superior performance compared to existing state-of-the-art methods for dynamic question generation.

The rest of the manuscript sections are arranged in the following manner: The various forms of the question-answer pair that can be generated using the language processing and the introduction about GNN with their application areas are described in Section 2. The most relevant and related literature articles description is presented in Section 3. The proposed methodology and its detailed description are presented in Section 4. The experimentation setup and the results are given in Section 5. Finally, in the end, the conclusion with the future work extension possibilities is presented in Section 6.

2 BASIC TERMINOLOGIES USED IN PROPOSED METHODOLOGY

2.1 Question-Answer Pair Types Generated with NLP

Using the Artificial Intelligence approaches in NLP, various types of question-answer pairs can be generated. Some common types are given as:

(1) Close-style question-answer pairs: These types of questions involve filling in the blank in a sentence or paragraph with the correct answer. For example, given the sentence "The capital of India is _____", the generated question-answer pair would be "What is the capital of India?" and "Delhi".
(2) Factoid question-answer pairs: These types of questions are answerable with a specific piece of information or fact. For example, given the sentence "John F. Kennedy was assassinated in 1963", the generated question-answer pair would be "When was John F. Kennedy assassinated?" and "1963".
(3) Definition-style question-answer pairs: These types of questions involve defining a term or concept. For example, given the sentence "Photosynthesis is the process by which plants convert light energy into

chemical energy", the generated question-answer pair would be "What is photosynthesis?" and "The process by which plants convert light energy into chemical energy".

(4) Explanation-style question-answer pairs: These types of questions involve explaining or describing a concept or phenomenon. For example, given the sentence "Water freezes at 0 degrees Celsius", the generated question-answer pair would be "Why does water freeze at 0 degrees Celsius?" and "Because the molecular motion of water slows down and eventually becomes a solid at that temperature".

(5) Multiple Choice Questions (MCQs): These are questions where the respondent is given a set of options to choose from, and must select the correct one. For example, "Which of the following is NOT a stop word in sentiment analysis? (a) the, (b) at, (c) but, (d) cat"

(6) True/False Questions: These are questions where the respondent must indicate whether a given statement is true or false. For example, "NLP is a subfield of computer science. True/False".

(7) Matching Questions: These are questions where the respondent must match one set of items with another set of items. For example, "Match the following natural language processing tasks with their corresponding descriptions: (1) Named Entity Recognition (NER), (2) Sentiment Analysis (SA), (3) Text Summarization (TS). A. Identifying the positive or negative sentiment of a text. B. Extracting named entities such as people, places, or organizations from a text. C. Creating a shorter version of a longer text."

2.2 Graph Neural Network and its Applications

Graph Neural Networks (GNNs) are a class of deep learning models specifically designed to work with graph-structured data. These networks have gained significant attention in recent years due to their effectiveness in various applications where data can be represented as graphs. Some of the recent applications of GNN are: *Social Network Analysis* including community detection, link prediction, and identifying influential nodes, *Recommendation Systems* by capturing user preferences and item relationships effectively, *Biomedical Research* for tasks like protein-protein interaction prediction, drug discovery, and disease classification, *Fraud Detection* in analyzing transaction data and detect fraudulent activities where the transactions can be modeled as a graph, and anomalies are identified using GNN-based techniques. Other application of GNN also includes *Computer Vision* for tasks like object tracking, image segmentation, and scene understanding, *Cybersecurity* to detect network intrusions and anomalies, and in *Physics and Chemistry* where the GNNs are used in materials science to predict material properties, chemical reactions, and molecular structures by representing molecules as graphs of atoms and bonds. Graph Neural Networks have seen wide adoption across diverse domains, with their ability to model complex relationships in data represented as graphs.

3 RELATED WORK

Question Answer Generation (QAG) is an active area of research in Natural Language Processing (NLP), and there have been many works done in this field.

One of the preliminary Studies of the year 2018 proposed a neural network model for generating questions from a given passage of text [34]. The model used an encoder-decoder architecture and was trained on a large corpus of news articles. The model was proved worthy when they compared their work on benchmark parameters. In another work, the authors of [25] introduced a large corpus of factoid questions and answers, and proposed a recurrent neural network model for generating factoid questions from a given answer. The model used a multi-task learning approach and was trained on the large corpus. The authors evaluated the corpus using automatic evaluation metrics and sentence similarity metrics and found it relevant.

Another work for generating news-related question answers was proposed in [27] where the authors proposed a model that generates questions with a specific answer in mind. The model was designed to be position-aware,

meaning it considers the position of the answer in the input text. The model was trained on a large corpus of news articles and achieved state-of-the-art results on several benchmark datasets.

The work on the question generation for an open domain discussion was proposed in [15]. The authors developed a context-enhanced neural question generation model that uses the conversational context to predict question content and pattern and then performs question decoding.

Another work that used the deep model, reinforcement learning, was proposed in the literature to generate the questions in paragraph-level [31]. The authors proposed the Extended Answer-aware Network decoded with Uncertainty-aware Beam Search after being trained with Word-based Coverage Mechanism. Their proposed model targets the answer by its surrounding sentence with an encoder and incorporates the extended answer to paragraph representation using gated paragraph-to-answer attention. On a similar aspect of reinforcement learning some other authors also proposed their methodologies to generate the variant of questions [7, 29].

In the literature, there were works that focus on the domain of healthcare and medical science for which the question-answer pairs were generated [17, 24, 35]. These works cover a single aspect to multi-aspect parameters applied for various domains like medical summarization [17], and question entailment in the medical domain [24, 35]. On the other hand, the authors of [19] present a detailed survey on the textual entailment-based question answering. similarly, using transfer learning, a pre-trained model approach, based question answering system was proposed for the biomedical domain in [30].

For the factoid-based QAG, there were works starting from the dataset benchmark to application areas. The work presented in [10] gave the contribution for the non-factoid question answering benchmark. On the other hand, the works reported in [9, 20, 32] use the memory network and knowledge graphs to generate the factoid question answers for various domains.

Using deep learning, the literature presents significant contributions to generate the question and answers for the given image as a source [5]. Moreover, using the text paragraphs, some of the works on deep learning models are presented in [1, 2, 8, 16, 21, 26] using the various application areas for which QAG is performed. The authors mostly used the slightly improved existing pre-trained models for the question generation.

This is to be noted in the above literature that the used language is English. However, there has been a significant amount of research on Question Answer Generation (QAG) for languages other than English. One such work is presented in [3] that had given the database of non-English for a close-style question-answer generation. Similarly, some more works were present in the literature using the close style for question generation [6, 11, 22].

Other than English, the literature has found many Asian and low-resource languages for the generation of question answers. These languages were identified as the low-resources as found their use in some specific regions having limited populations. The work on Urdu language corpus generation is described in [12]. The paper proposed a QAG system for the Urdu language using transfer learning. The authors use the pre-trained Urdu monolingual models and fine-tune them on a corpus of Urdu text to generate questions and answers.

Using the multilingual aspect, especially focusing on Hindi, and English text to Bangla language-based question-answer generation was proposed in [14, 28]. These papers propose a QAG system in the Bangla language using multilingual languages as a source with the BERT model. The authors fine-tune the model on a corpus of Bengali and other low-resource language text to generate questions and answers.

While the work in [13] used the QAG system for the Tamil language using a sequence-to-sequence model. The authors evaluate the system on a corpus of Tamil text and compare its performance with other QAG systems.

4 PROPOSED METHODOLOGY

In this work, we proposed a newly designed architecture for the Graph Neural model that helps in the generation of the dynamic question-answer pair using the context. The context is found relevant to the problem statement

as sometimes without context the question-answer pair becomes irrelevant irrespective of its linking with the domain. The example of the context-aware and its importance is shown in Table 1.

Table 1. Sample paragraph showing the importance of Context

Paragraph: Taj Mahal is the mausoleum of around 42 acres local in Agra, India. It was built by the Shah Jahan in loving memory of his wife Mumtaz Mahal.		
Within Context	Nearby context	Out of context
What is the Taj Mahal?	For what is the Agra famous for?	What is the architectural style of the Taj Mahal?
Where is the Taj Mahal located?	Is Taj Mahal a historical fortification?	When was the Taj Mahal built?
What type of monument is the Taj Mahal?	What was the purpose of building the Taj Mahal?	Is the Taj Mahal a UNESCO World Heritage site?
How large is the area covered by the Taj Mahal?	Who is the Taj Mahal dedicated to?	What is the cost to built Taj Mahal?

Table 1 presents a sentence about Taj Mahal, one of the seven wonders of the world and situated in India. The table also presents the three columns highlighting the questions using the context of the paragraph. However, the paragraph is small even then twelve questions were generated that were categorized into three sub-classes using the context information. Aside from these, many other questions could also be designed using the given text. Meanwhile, the data present in columns one and two in the table are considered correct and relevant as related to the context. While the last column might seem context related to the text, but it actually does not fall in the context of the given paragraph.

Also, as the passage length increases the count of the number of generated questions will also increase. In addition, using the single information piece there can be multiple questions that can be generated from a single paragraph. This demonstration is shown in Figure 1.

With this motivation here we proposed the context-aware Auto-encoded Graph neural network for the generation of the question-answer pair dynamically.

4.1 Graph Neural Network

A graph neural network (GNN) is a type of neural network designed to operate on graph-structured data. In other words, it is a type of machine learning algorithm that is specifically designed to analyze and make predictions about data that is represented in the form of a graph. Graphs are commonly used to represent complex systems or networks, such as social networks, biological networks, or transportation networks. In these types of systems, nodes represent entities, and edges represent the relationships or interactions between these entities. A graph neural network takes this structure into account and uses it to make predictions about the nodes and edges in the graph. At each layer of the network, the hidden representation of each node is updated based on its neighbors in the graph. This allows the network to capture complex relationships and dependencies between nodes and edges in the graph.

In the context of NLP, especially Question-Answer Generation, Graph neural networks (GNNs) can be used to generate pairs by representing the question-answer pairs as nodes in a graph, and the relationships between them as edges. One approach to using GNNs is to first construct a graph where each question-answer pair is represented as a node. The edges between the nodes can represent various types of relationships, such as the similarity between questions or answers, or the relevance between questions and answers. The GNN is then

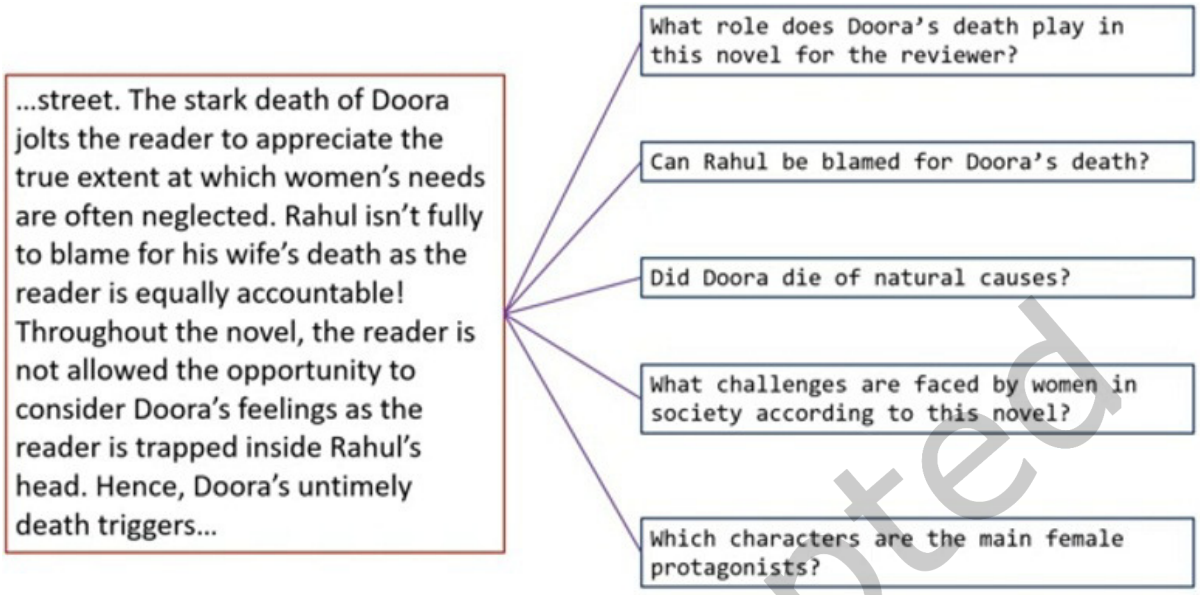

Fig. 1. Sample of generating multiple questions from a paragraph using single context and information

trained on this graph to predict the most likely answer for a given question. During training, the GNN propagates information through the graph to learn the relationships between the nodes, which can help to identify the most relevant answer to a given question.

Another approach is to use a GNN to generate question-answer pairs directly. In this approach, the GNN is trained to generate a sequence of words that corresponds to a question-answer pair. The graph is constructed by encoding the question and answer as nodes and using edges to represent the relationship between them. The GNN then generates the question-answer pair by recursively sampling words based on the information in the graph.

4.2 Auto-Encoded Graph Neural Network

An Auto-encoded Graph Neural Network (AGNN) for a question-answer generation would consist of several layers of nodes and edges, with each layer representing a different level of abstraction in the model. The input to the model would be a graph representing the input text, with nodes corresponding to words and edges corresponding to relationships between them. The model would use a combination of graph convolutional layers and recurrent neural networks to encode the input graph into a vector representation, which would then be fed into a decoder to generate the output question-answer pairs. The decoder would also be implemented using a graph neural network, with the output graph representing the question-answer pair. Finally, the output graph would be decoded into natural language using a sequence-to-sequence model.

In this approach, a graph neural network is used to encode the input text into a graph structure, and an autoencoder is used to generate the corresponding question-answer pairs. It consists of two main components: the graph encoder and the autoencoder. The graph encoder takes the input text and converts it into a graph structure, where each word or phrase is represented as a node in the graph. The edges between the nodes represent the semantic relationships between them. The autoencoder takes the encoded graph structure and generates the

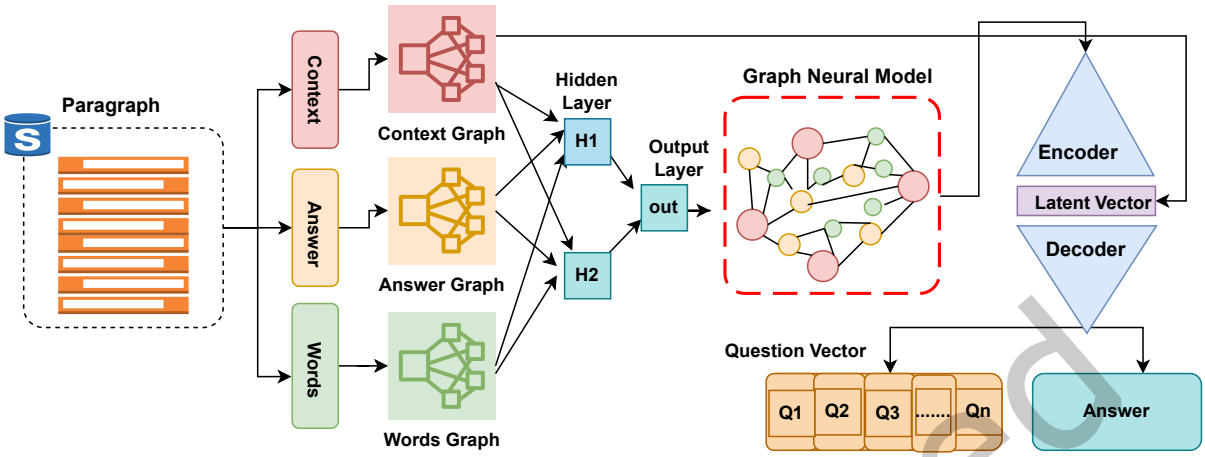

Fig. 2. Flow diagram of the proposed methodology

corresponding question-answer pairs. The autoencoder is trained on a dataset of question-answer pairs, where the input is the question and the output is the corresponding answer. During training, the autoencoder learns to generate the correct answer for a given question by encoding the graph structure of the input text.

Once the autoencoder has been trained, it can be used to generate question-answer pairs for new input texts. The input text is first encoded into a graph structure using the graph encoder, and then the autoencoder generates the corresponding question-answer pairs based on the encoded graph structure.

4.3 Proposed Context-Aware Auto-encoded Graph Neural Network

The proposed Context-aware Auto-encoded Graph Neural Network (CAGNN) is an advanced version of the Auto-encoded Graph Neural Network (AGNN) architecture, which is designed to generate question-answer pairs from a given context by incorporating contextual information. In the CAGNN architecture, the input text is first split into smaller segments or paragraphs, each of which is treated as a separate context. Each context is then encoded into a graph structure using the graph encoder, where each word or phrase is represented as a node and the edges represent the semantic relationships between them.

Next, the contextual information is incorporated into the graph structure using a context-aware attention mechanism. This mechanism assigns different weights to different nodes in the graph, based on their relevance to the current context. The encoded graph structure with contextual information is then fed into the autoencoder, which generates the corresponding question-answer pairs. During training, the autoencoder learns to generate the correct answer for a given question, based on the encoded graph structure with contextual information.

Once the CAGNN has been trained, it can be used to generate question-answer pairs for new input texts. The input text is first split into separate contexts, and each context is encoded into a graph structure with contextual information using the graph encoder and attention mechanism, respectively. The autoencoder then generates the corresponding question-answer pairs based on the encoded graph structure with contextual information. The basic flow diagram of the methodology is shown in Figure 2.

It is to be observed that the input to the proposed model is either a single sentence or a multi-line paragraph distributed among several rows. The input paragraph is considered to be the collection of the triplet i.e. <Bag-of-words, Answer, Context> that is to be fetched individually to form the instance graphs. Later, the instance graphs will be fed into the neural model to generate the *Graph Neural Model* (GNM) having collective instances of the

triplet. Further, the autoencoder is trained on the dataset using the GNM. Finally, the reverse structure, called Decoder will generate the question-answer pair using the context information.

The steps involved in the generation of the Question-answer pairs using the proposed methodology are given as:

(1) Define an input layer with a paragraph P represented as a sequence of words with a maximum length of N words.
(2) Define an autoencoder neural network architecture with two main components: encoder and decoder.
(3) Define a model to encode the context, answer, and Bag-of-Words of paragraph P.
(4) Feed the model output into the GNN layer to obtain a graph representation of the context.
(5) Define a graph neural network (GNN) layer to encode the context, answer, and words of paragraph P.
(6) Train the autoencoder using generated graph neural model as input and its corresponding output.
(7) Define a question decoder neural network that generates questions from the graph representation.
(8) Train the question decoder using the graph representation as input and the corresponding question as output.
(9) During testing, feed the paragraph P into the encoder to obtain the encoded representation.
(10) Feed the encoded representation into the GNN layer to obtain the graph representation of the context.
(11) Finally, Feed the graph representation into the question decoder to generate dynamic questions from the paragraph.

The algorithm to generate question-answer pair using the context is presented in Algorithm 1.

4.4 Detailed Model Architecture for QAG

The detailed model architecture for the QAG generation is described by using the two main modules i.e. *Encoding* and *Decoding*. In the encoding phase of the model, the input data in form of the paragraph is transformed into the latent semantic cascaded vector embedded with the triplet <Context, Answer, Bag-of-words> while doing model training. The trained model vector is later fed into the decoding phase to generate the vector of questions, related to the context information with the answer. In the question vector, multiple questions will be formed which are having the same answer and context information. The detailed architecture with internal workflow is shown in Figure 3.

4.4.1 Encoding for Individual components. The input paragraph P is splitted into three component vectors i.e. <Bag-of-words, Answer, context>. The paragraph is scanned and filtered out the stopwords to form the Bag-of-words and known as word tokens w= {w1, w2,..., wn}. The answer is splitted to word-level tokens as a = {a1, a2, ..., am}, and context as c= {c1, c2,..., cn} respectively. Later on, to tag the given words to the context, answer tagging embeddings are to be used. To represent the contextual information the three component vectors are fed into the LSTM deep network and are denoted by $W_0 \in R^{hxn}$, $a_0 \in R^{hxn}$, and $c_0 \in R^{hxn}$ respectively, in which h is the hidden state dimensions in LSTM. The working architecture of the LSTM is shown in Figure 4.

The LSTM vector output is processed with the attention-based layer to generate the respective graphs for Bag-of-words, answer, and context. To do this, first, we compute the alignment matrix, using Equation 1.

$$M = c0^T.a0 \tag{1}$$

The attention layer in the Figure 3 has its importance to filter the response. Attention mechanisms play a crucial role in enhancing the capabilities of Graph Neural Networks (GNNs) for question-answer generation (QAG). Attention mechanisms enable GNNs to focus on relevant information within the graph when updating node embeddings. In QAG, this means that the model can give more attention to nodes (words or entities) that are contextually important for understanding the question and generating accurate answers. Also, attention mechanisms help GNNs effectively handle long contexts by allowing the model to attend to the most informative

Algorithm 1: An Algorithm to generate Question-Answer pair from a given paragraph

Data: A paragraph P
Result: Dynamic questions-answer pairs generated from paragraph P
Preprocessing:
a. Tokenize paragraph P into a sequence of words with a maximum length of N.
b. Construct a vocabulary V from the words in P.
c. Map each word in P to a unique integer in V.
Autoencoder Training:
a. Define the autoencoder neural network architecture with an encoder and decoder.
b. Train the autoencoder using P as input and output.
c. Save the encoder weights for later use.
Graph Neural Network Encoding:
a. Define a GNN layer to encode the context of the paragraph.
b. Construct a graph representation of the context using the encoded representation obtained from the autoencoder.
c. Apply the GNN layer to the graph representation to obtain a new representation that captures the contextual information of the paragraph.
Question Decoder Training:
a. Define the question decoder neural network that generates questions from the graph representation.
b. Train the question decoder using the graph representation as input and the corresponding question as output.
Question Generation:
a. During testing, feed P into the autoencoder to obtain the encoded representation.
b. Feed the encoded representation into the GNN layer to obtain the graph representation of the context.
c. Feed the graph representation into the question decoder to generate dynamic questions from P.
Predicted Output:
Return the dynamic questions generated from P.

nodes and edges while ignoring irrelevant or less relevant parts of the graph. This is particularly important for memory efficiency and reducing computational complexity. Thus, GNNs with attention can capture complex relationships and dependencies within the graph. This is especially valuable when generating answers that require understanding the connections between various pieces of information present in the text, such as cause-and-effect relationships, comparisons, or logical reasoning.

Next, the matrix M is being normalized in both orientations i.e. row (M_R) and column-wise (M_C) as shown with Equation 2. Here, the (M_R) gives the answer token relevance to the question, and similarly (M_C) gives each context token relevance to the answer.

$$\begin{aligned} M_R &= softmax(M) \\ M_C &= softmax(M^T) \end{aligned} \tag{2}$$

The generated new answer form using the context information is given by Equation 3.

$$A^{'} = c0.M_R \tag{3}$$

The modified context information related to the answer tag is calculated by concatenation of the previous and the new answer forms processed with attention matrix M_C and is given by Equation 4.

$$c1^{'} = [A0 : A^{'}].M_C \tag{4}$$

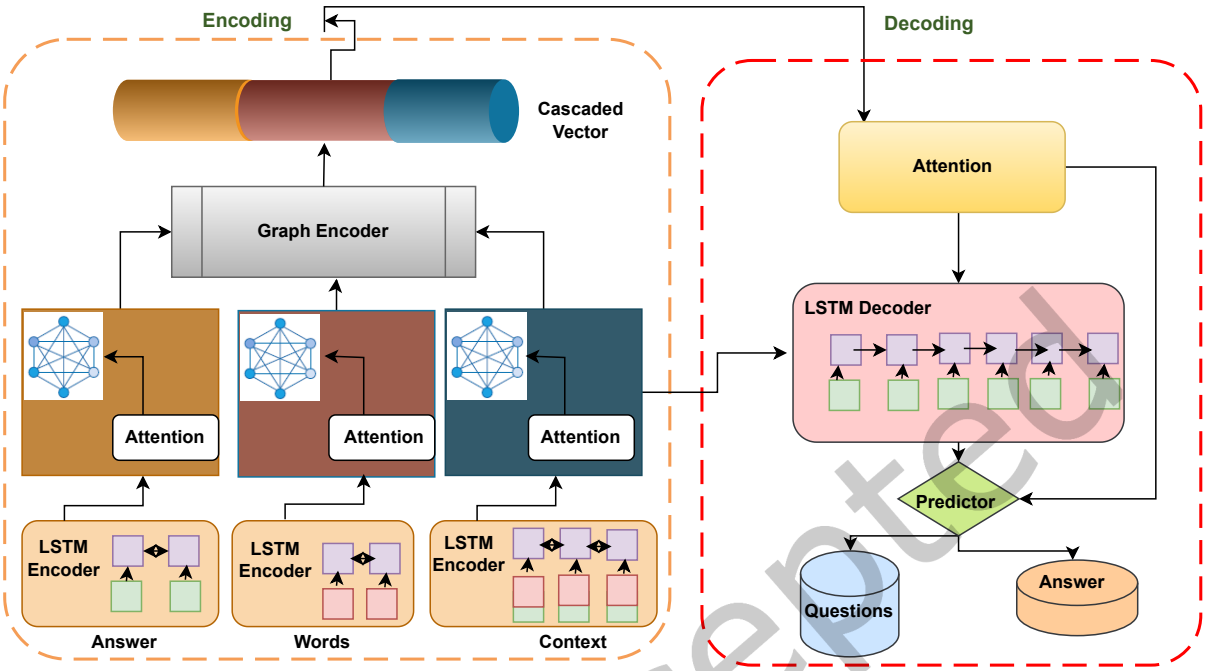

Fig. 3. Internal workflow of the model for Question-Answer pair generation

This will result in the formation of the encoding of the context to form the context graph. Similarly, the other two components are processed individually to generate the individual graphs of the answer tokens and the Bag-of-words.

4.4.2 Graph Encoding. In the next phase, the individual graphs are encoded to give a useful and meaningful question-answer pair within the context. Here the neural architecture is of three neurons in the input layer (one of each graph), two hidden neurons, and one output neuron is used for generating the Graph neural network. The output of this neural network is the graph model having interconnection between the Questions, Answers, and Context.

The graph is composed of nodes and edges where the nodes are the representation of the questions, their answers, and the context information. The edges are the interconnection between these nodes suggesting the relevance of the context in which questions and answers are linked to each other. More specifically, the connected edges in between the nodes of any pair exist in a graph if both (say Question terms and Answer) are present in the same sentence, or appear in the same paragraphs. Also, there will be no interconnection or edges in a graph if the nodes are having the same entity i.e. in between Question-Question, Answer-Answer, and Context-context. It is because this will not create any useful information to generate the question-answer pair.

Using the modified context information, as given in Equation 4, the context encoding is performed using Equation 5.

$$C1 = LSTM(c0, c1^{'}) \tag{5}$$

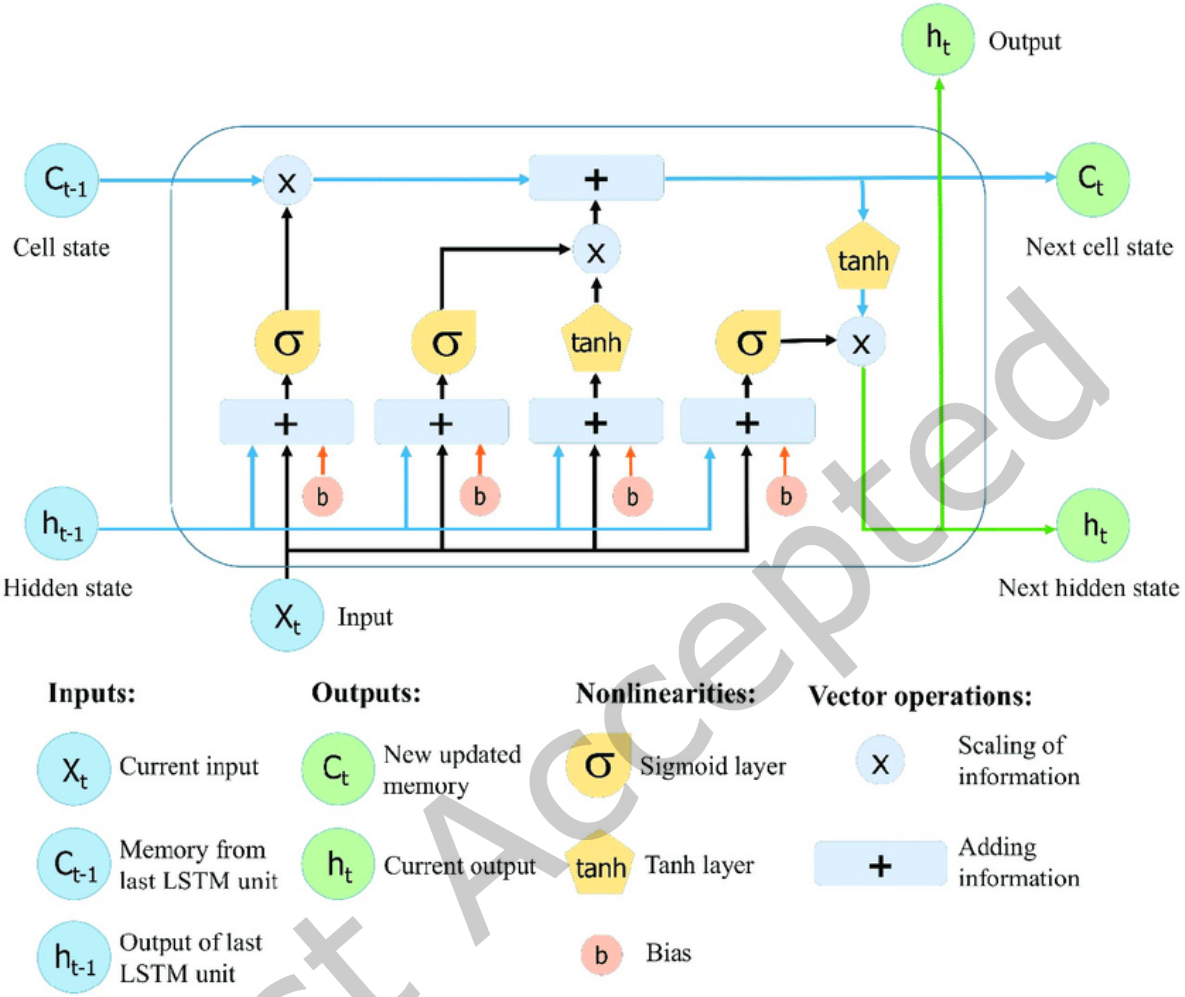

Fig. 4. Architecture of the LSTM for vector encoding

Using context encoding *C1*, the entities encoding matrices M_i is calculated which is binary in nature having two values {0 and 1}. The value under $M_{i,j}$=1, if the tuple is having an impact with context, otherwise it should be 0.

Next, to generate the latent cascaded vector the graph is processed using the encoded matrices M_i. For this, initially, the irrelevant entities which are out of context information are ruled out and the entity nodes that are related to Question-answering with context are allowed to disseminate information. The latent vector of the question-answer, represented by $L = \{l1, l2, ..., lm\}$, where li is the relatedness of the question-answer to the context, is given by Equation 6.

$$L = Sigmoid(a0^T.V.c0) \tag{6}$$

Where V is the linear projection matrix, a0 is the initial answer encoding, and c0 is the initial context encoding.

The final latent encoding vector is calculated by cascading the context vector with the latent vector L.

4.5 Graph Neural Network Model Parameters

In question-answer generation (QAG) within Natural Language Processing (NLP), Graph Neural Networks (GNNs) can be used to model relationships and dependencies among words, entities, or concepts in a text. The parameters of a GNN-based QAG model typically include the following:

- Graph Representation:

  Node Features: These are the embeddings or representations of individual words, entities, or subword tokens in the input text. These embeddings are often initialized using pre-trained word embeddings (e.g., Word2Vec, GloVe, or BERT embeddings). While in this work, the node features are learned during training.

  Edge Features: In many places edges between nodes (e.g., words or entities) have associated features. These features can capture linguistic relationships like dependency parse labels or semantic similarities.

  Adjacency Matrix: The graph structure is represented using an adjacency matrix, where each entry indicates the presence or strength of a connection (edge) between nodes (words or entities). This matrix is often sparse and binary, indicating whether two nodes are connected or not.

- Graph Convolutional Layers: The convolutional filters in the graph convolution layer are the learnable parameters that perform convolution operations on node features. The filters aggregate information from neighboring nodes and update node embeddings. While the pooling operations in GNN architecture are applied to reduce the dimensionality of node embeddings or aggregate information hierarchically.

- Message Passing Mechanism: These functions define how information is exchanged (or messaged) between neighboring nodes during the forward pass. Here we have used the LSTM-attention-based aggregation for message passing.

- Update Functions: It takes the aggregated messages and updates the node embeddings based on this information. To do the update process we have used the long short-term memory (LSTM) cells in model designing.

- Attention Mechanisms: With the use of the attention mechanism the model has learnable attention weights that control the importance of different nodes during message aggregation. Attention can be used to capture context and relationships more effectively.

- Loss Function: The loss function measures the difference between the predicted answer spans and the ground truth spans in the training data. We have used the cross-entropy loss on the specific formulation of the QAG task.

- Regularization Techniques: Dropout layers are added to prevent overfitting by randomly dropping out a fraction of the neurons during training. The dropout is set here with the value 0.3, and L2 regularization terms are added to the loss function to penalize large weights and encourage weight sparsity.

- Optimization Algorithm: An optimization algorithm SGD is used to update the model parameters during training by minimizing the loss function.

Table 2. Comparative analysis of the used datasets

Dataset	Number of QA Pairs	Question Type	Answer Type	Context Source	Domain
SQuAD	100,000+	Factoid	Short text span	Wikipedia articles	General knowledge
Natural Questions	300,000+	Factoid, List, Yes/No	Short text span, List	Google search snippets	General knowledge
TriviaQA	650,000+	Factoid, List	Short text span, List	Web pages, books	General knowledge
QuAC	14,000+	Contextual	Free-form text	Conversations	Conversational

- Learning Rate: The learning rate determines the step size during parameter updates and is a critical hyperparameter to tune. Here we have taken the learning rate as 0.001. While the batch Size is taken as 32 in experimentation.

5 EXPERIMENTATION AND RESULTS

5.1 Dataset Description

To evaluate the performance of the proposed model, four different datasets were used for model training and testing. The description of these datasets is given as,

(1) SQuAD (Stanford Question Answering Dataset): SQuAD[1] is a reading comprehension dataset, consisting of questions posed by crowd workers on a set of Wikipedia articles, where the answer to every question is a segment of text from the corresponding reading passage. The dataset contains over 100,000 question-answer pairs and is used for training and evaluating question-answering systems.
(2) Natural Questions Dataset: Natural Questions[2] is a dataset of question-answer pairs, consisting of questions derived from query logs and their corresponding answers extracted from Wikipedia articles. The dataset contains around 3,000 question-answer pairs and is used for evaluating information retrieval systems.
(3) TriviaQA: TriviaQA[3] is a large-scale dataset of question-answer pairs, created by the University of Washington and the Allen Institute for Artificial Intelligence. The dataset contains over 650,000 question-answer pairs, covering a wide range of topics and domains, including sports, history, and science.
(4) QuAC (Question Answering in Context): QuAC[4] is a dataset for question-answering in context, created by the Allen Institute for Artificial Intelligence. The dataset contains over 14,000 question-answer pairs, where the questions are designed to be answerable only with context from a preceding paragraph.

The table comparing the key differences between the SQuAD, Natural Questions, TriviaQA, and QuAC datasets is shown in Table 2.

In all these datasets mostly the Question-answer pairs were generated by both humans and machines. Also, in the analysis of the dataset, it is found that the questions were generated using the basic question keywords like *which, what, who, when, where, and how many*. Using such keywords, for a single context multiple questions can be generated. However, to measure the performance of the generated questions different measures can be used like relevancy, type of question, grammatical right, and ambiguity.

[1]https://www.kaggle.com/datasets/stanfordu/stanford-question-answering-dataset
[2]https://ai.google.com/research/NaturalQuestions
[3]http://nlp.cs.washington.edu/triviaqa/
[4]https://quac.ai/

5.2 Performance Evaluation Metrics

One of the traditional ways to evaluate the correctness of the generated questions is human evaluation. It involves collecting feedback from human experts on the quality of the generated questions. This is a subjective evaluation method and is commonly used to complement objective evaluation metrics. But, in the new evaluation system, also called the modern approach, the evaluation of question-answer pairs is being done through advanced measures. In this work, there are several performance metrics used to evaluate the quality of the question-answer generation system. These are given as:

(1) BLEU (Bilingual Evaluation Understudy): BLEU is a metric that is commonly used to evaluate machine translation systems. It measures the similarity between the generated questions and the reference questions provided by human experts. The metric ranges from 0 to 1, where 1 indicates perfect similarity between the generated questions and the reference questions. The mathematical form of which is given in Equation 7.

$$BLEU = BP * exp(\frac{1}{n} * \sum(log(p_i))) \tag{7}$$

where BP is the brevity penalty, n is the number of generated questions, and p_i is the precision of the i^{th} n-gram.

(2) ROUGE (Recall-Oriented Understudy for Gisting Evaluation): ROUGE is a family of metrics that are commonly used to evaluate text summarization systems. It measures the overlap between the generated questions and the reference questions at the word and phrase level. It is given in mathematical form using Equation 8.

$$ROUGE = \frac{(Bleu - N\ precision * Bleu - L\ precision * Bleu - S\ precision)}{3} \tag{8}$$

where Bleu-N precision, Bleu-L precision, and Bleu-S precision are the precisions of the n-gram overlap, the longest common subsequence, and the skip-bigram overlap, respectively.

(3) F1 score: F1 score is a widely used evaluation metric in NLP. It measures the balance between precision and recall. It is the harmonic mean of precision and recall and ranges from 0 to 1, where 1 indicates perfect performance. It is calculated using Equation 9.

$$F1 - score = \frac{2 * (precision * recall)}{(precision + recall)} \tag{9}$$

where precision is the number of true positives divided by the sum of true positives and false positives, and recall is the number of true positives divided by the sum of true positives and false negatives.

(4) Accuracy: Accuracy measures the percentage of generated questions that match the reference questions. It is a simple and intuitive metric that is commonly used in NLP evaluation. It is given as described in Equation 10.

$$Accuracy = \frac{number\ of\ correctly\ generated\ questions}{total\ number\ of\ questions} \tag{10}$$

where the correctly generated questions are those that match the reference questions.

5.3 Baseline Models

Baseline methods for question-answer generation typically involve simple heuristics or rule-based algorithms that rely on pattern matching and keyword extraction to generate answers. These methods may not be as accurate or flexible as more advanced machine learning approaches, but they can still be useful in certain applications, particularly those involving structured data or well-defined domains. These baseline methods are categorized into four groups i.e.

- Rule-based methods: These methods rely on hand-crafted rules to generate questions and answers. For instance, one rule could be to extract the subject, verb, and object from a given sentence to generate a question.
- Information retrieval-based methods: These methods extract information from a large corpus of text using search engines and other tools, and use that information to generate questions and answers.
- Template-based methods: These methods use pre-defined templates to generate questions and answers. For example, a template could be used to generate questions about a specific topic or event.
- Sequence-to-sequence models: These are neural network models that take a sequence of words as input and generate a sequence of words as output. They can be used for question-answer generation by training them on a large dataset of questions and answers.

While these methods can provide a useful starting point for question-answer generation, they often have limitations in terms of accuracy and the ability to handle complex questions and contexts. Therefore, more advanced techniques, such as deep learning models like transformer-based models, are now being used for question-answer generation with higher accuracy and performance.

In this work, as a baseline, we selected two base algorithms for word embedding approaches i.e. *GLoVE*and *BERT* with the three encoding-decoding models i.e. *MLP, CNN*, and *BiLSTM*. As a baseline, the six different models are used i.e. *GLoVE+MLP, GLoVE+CNN, GLoVE+BiLSTM, BERT+MLP, BERT+CNN, and BERT+BiLSTM*.

5.3.1 Word Embedding Models. GloVe (Global Vectors for Word Representation) is a popular method for generating word embeddings, which are dense vector representations of words that capture their meaning and context. GloVe is often used as a pre-processing step in question-answering systems, where it is used to convert words in the text into numerical representations that can be processed by machine learning models.

BERT (Bidirectional Encoder Representations from Transformers) can be used for word embedding in question answer generation by fine-tuning the pre-trained model on a specific task, such as question answering. In this approach, the pre-trained BERT model is used as a feature extractor for the input text, which is then fed into a classifier or decoder to generate the answer. The BERT model is fine-tuned on a training dataset of question-answer pairs, using techniques such as masked language modeling and next-sentence prediction, to adapt it to the specific task of question-answering.

During inference, the input question is first encoded using BERT to generate a contextualized word embedding for each token in the question. These embeddings are then used as input to a decoder or classifier to generate the answer.

5.3.2 Baseline Encoding-Decoding. MLP (multilayer perceptron) can be used as an encoder-decoder in question answer generation, where the encoder maps the input question into a fixed-length vector representation, and the decoder generates the answer based on this representation. In this approach, the input question is first tokenized into a sequence of words, and each word is represented as a vector using pre-trained word embedding. The word vectors are then fed into the encoder, which consists of one or more hidden layers of neurons that transform the input sequence into a fixed-length vector representation, often referred to as the context vector.

The context vector is then used as the initial hidden state of the decoder, which is typically another MLP that generates the answer word by word. At each time step, the decoder takes the current word and the previous hidden state as input and produces a probability distribution over the possible next words in the answer. The decoder selects the word with the highest probability and uses it as input for the next time step until an end-of-sequence token is generated or a maximum answer length is reached.

CNN (convolutional neural network) can be used as an encoder-decoder in a question-answer generation, where the encoder maps the input question into a fixed-length vector representation, and the decoder generates the answer based on this representation. In this approach, the input question is first tokenized into a sequence of

words, and each word is represented as a vector using pre-trained word embeddings. The word vectors are then fed into a 1D convolutional neural network, which consists of one or more convolutional layers with max pooling, followed by one or more fully connected layers that output the context vector.

The context vector is then used as the initial hidden state of the decoder, which is typically another CNN that generates the answer word by word. At each time step, the decoder takes the current word and the previous hidden state as input and produces a probability distribution over the possible next words in the answer. The decoder selects the word with the highest probability and uses it as input for the next time step until an end-of-sequence token is generated or a maximum answer length is reached.

BiLSTM (Bidirectional Long Short-Term Memory) can also be used as an encoder-decoder in a question-answer generation. In this approach, the input question is tokenized and each word is represented as a vector using pre-trained word embeddings. The word embeddings are then fed into a bidirectional LSTM, where the forward and backward LSTM layers process the input sequence in opposite directions and concatenate their outputs. This creates a context vector that captures both the past and future context of each word in the sequence.

The context vector is then used as the initial hidden state of the decoder, which is typically another LSTM that generates the answer word by word. The decoder takes the context vector and the previously generated word as input at each time step and produces a probability distribution over the possible next words in the answer. The decoder selects the word with the highest probability and uses it as input for the next time step until an end-of-sequence token is generated or a maximum answer length is reached.

5.3.3 Cascaded Baseline Models. In experimentation, the word embedding model is cascaded with the encoder-decoder model, one at a time, to generate the cascaded baseline models. In the end, the following six baseline models were generated which are given below:

1 *GLOVE+MLP*: It consists of an Input layer, followed by a GLOVE Embedding layer. Later the output is fed into the flattened layer and at last to the Dense layer, with a sigmoid activation function, which acts as our output layer.
2 *BERT+MLP*: It consists of an Input Layer, followed by the BERT layer. Later the output is fed into the flattened layer and at last to the Dense layer, with a sigmoid activation function, which acts as our output layer.
3 *GLOVE+CNN*: It consists of an Input layer, followed by a GLOVE Embedding layer. The embeddings are passed to three Conv1D layers with kernel sizes 2,3 and 4 respectively. Each Conv1D layer is followed by a Dropout layer of value 0.2, then a MaxPooling 1D layer. The three final flatten layers are concatenated and passed through a Dense Layer with the ReLU activation function.
4 *BERT+CNN*: It consists of an Input Layer, followed by the BERT Layer. The embeddings are passed to three Conv1D layers with kernel sizes 2,3,4 in that order, i.e., the embeddings go to the kernel-size-2 layer, then the output is passed to the second Conv1D layer. It is followed by a GlobalMaxPool 1D Layer, a Dense layer with ReLU activation function, and a Dropout layer with a value of 0.2.
5 *GLOVE+BiLSTM*: It consists of an Input layer, followed by a GLOVE Embedding layer followed by the Bidirectional-LSTM layer, and again a Dropout layer with the same value following the first Dropout Layer.
6 *BERT+BiLSTM*: It consists of an Input Layer, followed by the BERT Layer. A Dropout layer with a value of 0.3 follows a Bidirectional-LSTM layer.

5.4 Result

The main results of the proposed methodology using the four performance metrics on four different datasets are shown in Table 3.

Table 3. Performance analysis of the proposed approach on multiple datasets using different performance measures

Approach	Datasets	BLEU	ROUGE	F1-Score	Accuracy
Proposed Approach	SQuAD	0.68	0.57	0.87	0.92
	Natural Questions	0.48	0.49	0.75	0.79
	TriviaQA	0.55	0.50	0.79	0.84
	QuAC	0.57	0.52	0.81	0.89

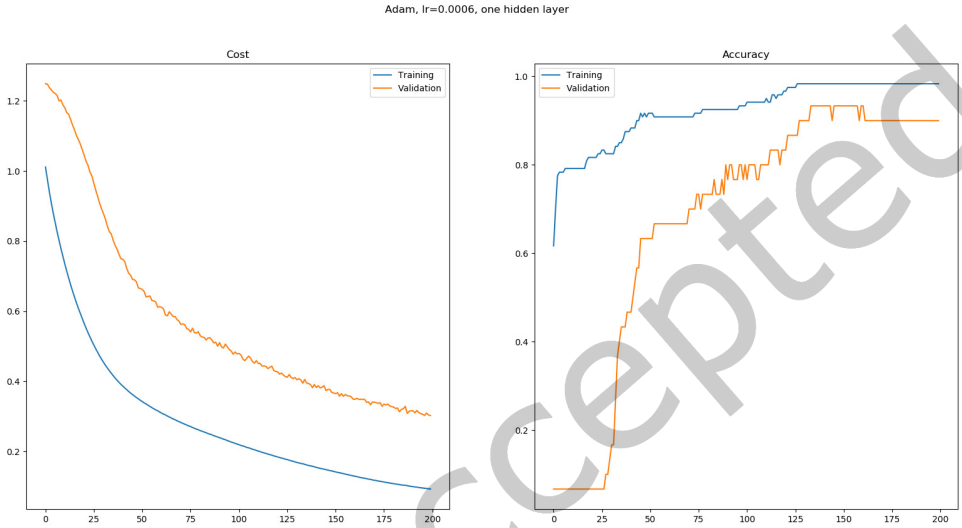

Fig. 5. Performance of the model using per-epoch computation

The performance of the model using the per-epoch computations is presented in Figure 5. The model has been experimented with by using an Adam optimizer with a learning rate of 0.0005 and is run for 200 Epochs. Also, the Softmax is used at the output with loss as catogorical-crossentropy.

The comparative analysis of the proposed method with the baseline methods using the different performance measures are presented in Table 4, Table 5, Table 6, and Table 7. The comparative results are presented using a single dataset in each table. All the experiments are performed using the same hyper-parameters of a model in each experiment.

5.5 Qualitative analysis

After the training, the proposed model is qualitatively analyzed by generating the question-answer pair from the given passage. Qualitative analysis of question-answer pairs generated using a passage involves a manual evaluation of the generated output to assess its relevance, coherence, and correctness. It can be performed by experts in the field or through crowdsourcing. It is an important complement to quantitative metrics such as BLEU, ROUGE, F1, and accuracy, as it can provide deeper insights into the strengths and weaknesses of the model and help guide further development and improvement. In the NLP domain, some of the most commonly known qualitative analysis methods are given as:

Table 4. Comparative analysis of the proposed model with baseline models on the SQuAD dataset

Model	BLEU	ROUGE	F1-Score	Accuracy
GLOVE+MLP	0.39	0.33	0.70	0.80
BERT+MLP	0.42	0.36	0.72	0.82
GLOVE+CNN	0.45	0.37	0.77	0.82
BERT+CNN	0.48	0.44	0.79	0.85
GLOVE+BiLSTM	0.47	0.48	0.80	0.88
BERT+BiLSTM	0.55	0.51	0.84	0.89
T5 Transformer	0.60	0.51	0.83	0.89
Proposed Method	0.68	0.57	0.87	0.92

Table 5. Comparative analysis of the proposed model with baseline models on the Natural Question dataset

Model	BLEU	ROUGE	F1-Score	Accuracy
GLOVE+MLP	0.23	0.28	0.62	0.57
BERT+MLP	0.31	0.34	0.66	0.60
GLOVE+CNN	0.34	0.30	0.61	0.66
BERT+CNN	0.34	0.35	0.67	0.68
GLOVE+BiLSTM	0.39	0.39	0.70	0.71
BERT+BiLSTM	0.42	0.44	0.73	0.74
T5 Transformer	0.41	0.47	0.73	0.75
Proposed Method	0.48	0.49	0.75	0.79

Table 6. Comparative analysis of the proposed model with baseline models on the TriviaQA dataset

Model	BLEU	ROUGE	F1-Score	Accuracy
GLOVE+MLP	0.31	0.20	0.60	0.49
BERT+MLP	0.32	0.24	0.68	0.56
GLOVE+CNN	0.36	0.28	0.67	0.55
BERT+CNN	0.40	0.32	0.69	0.68
GLOVE+BiLSTM	0.43	0.40	0.72	0.72
BERT+BiLSTM	0.49	0.44	0.71	0.78
T5 Transformer	0.54	0.50	0.78	0.84
Proposed Method	0.55	0.50	0.79	0.84

- Human evaluation: This involves having human evaluators read and rate the quality of the generated question-answer pairs. The evaluators can use a scoring system or provide written feedback on the relevance, coherence, and correctness of the output.
- Error analysis: This involves identifying the errors made by the model and analyzing the patterns of these errors. This can help identify areas for improvement in the model architecture or training data.
- Case studies: This involves examining individual examples of generated question-answer pairs and assessing their quality in context. This can help identify areas where the model is performing well and areas that require improvement.

Table 7. Comparative analysis of the proposed model with baseline models on the QuAC dataset

Model	BLEU	ROUGE	F1-Score	Accuracy
GLOVE+MLP	0.22	0.37	0.67	0.70
BERT+MLP	0.28	0.41	0.72	0.71
GLOVE+CNN	0.34	0.42	0.71	0.76
BERT+CNN	0.40	0.46	0.76	0.80
GLOVE+BiLSTM	0.45	0.48	0.79	0.82
BERT+BiLSTM	0.51	0.51	0.81	0.86
T5 Transformer	0.55	0.51	0.80	0.86
Proposed Method	0.57	0.52	0.81	0.89

- Comparison to reference answers: This involves comparing the generated question-answer pairs to the reference answers in the evaluation dataset. This can help assess the accuracy of the model and identify areas where the model is struggling.

The sample of the question-answer pairs generated by using the proposed model is given in Table 8.

6 CONCLUSION

This work proposed a new approach to generate dynamic and diverse questions from a given passage of text using the context-aware auto-encoded graph neural model. The approach used a graph neural network to capture the contextual relationships between words and phrases in the passage, and an auto-encoder to reconstruct the passage to generate questions.

The paper presented the results of the experiments conducted on the four different datasets and demonstrated that the proposed method outperforms the existing state-of-the-art models in terms of BLEU, ROUGE, F1-score, and Accuracy metrics. The qualitative analysis showed that the questions generated by the proposed method were more diverse and relevant to the passage compared to the baseline models.

Overall, the paper contributes to the field of question generation in NLP by introducing a novel approach that leverages the power of graph neural networks and auto-encoders. The proposed method can be used to generate high-quality questions for various applications, such as education, chatbots, and question-answering systems.

In the future, GNNs can be applied to cross-lingual QAG, where questions and answers can be generated in multiple languages. This is valuable for applications like machine translation and cross-cultural information retrieval.

DECLARATIONS

Competing interests: The authors declare that they have no competing interests.

DATA AVAILABILITY STATEMENT

No data is generated or associated with this work. However, the implementation code is accessible to the corresponding author upon reasonable request.

FUNDING INFORMATION

No funding is available for this work.

Table 8. Sample Question-answer pair generated using the proposed approach

Case Study #1
Passage: The world's highest mountain is Mount Everest. It is located in the Himalayas and stands at a height of 8,848 meters above sea level. The mountain was first scaled by Sir Edmund Hillary and Tenzing Norgay in 1953.

Q: What is the world's highest mountain?
A: Mount Everest
Q: Where is Mount Everest located?
A: Mount Everest is located in the Himalayas.
Q: How tall is Mount Everest?
A: Mount Everest stands at a height of 8,848 meters above sea level.
Q: Who was the first person to climb Mount Everest?
A: Sir Edmund Hillary and Tenzing Norgay were the first people to climb Mount Everest in 1953.

Case Study #2
Passage: The Great Barrier Reef is the world's largest coral reef system, located off the coast of Australia. It is composed of over 2,900 individual reefs and 900 islands stretching for over 2,300 kilometers. The Great Barrier Reef is home to thousands of different species of marine life, including over 1,500 species of fish and 600 types of coral. It is a popular tourist destination and supports a multi-billion dollar industry.

Q: Where is the Great Barrier Reef located?
A: The Great Barrier Reef is located off the coast of Australia.
Q: How long is the Great Barrier Reef?
A: The Great Barrier Reef stretches for over 2,300 kilometers.
Q: How many individual reefs and islands make up the Great Barrier Reef?
A: The Great Barrier Reef is composed of over 2,900 individual reefs and 900 islands.
Q: How many different species of fish live in the Great Barrier Reef?
A: Over 1,500 species of fish live in the Great Barrier Reef.
Q: How much does the Great Barrier Reef support the tourism industry?
A: The Great Barrier Reef supports a multi-billion dollar industry.

Case Study #3
Passage: Taj Mahal is an ivory-white marble mausoleum on the right bank of the river Yamuna in the Indian city of Agra. It was commissioned in 1632 by the Mughal emperor, Shah Jahan, to house the tomb of his favorite wife, Mumtaz Mahal.

Q: What is the Taj Mahal?
A: The Taj Mahal is an ivory-white marble mausoleum in the Indian city of Agra.
Q: Who commissioned the Taj Mahal?
A: The Taj Mahal was commissioned by the Mughal emperor, Shah Jahan.
Q: When was the Taj Mahal commissioned?
A: The Taj Mahal was commissioned in 1632.
Q: What was the purpose of building the Taj Mahal?
A: The purpose of building the Taj Mahal was to house the
tomb of Shah Jahan's favorite wife, Mumtaz Mahal.

REFERENCES

[1] Abdelrahman Abdallah, Mahmoud Kasem, Mohamed A Hamada, and Shaymaa Sdeek. 2020. Automated question-answer medical model based on deep learning technology. In *Proceedings of the 6th International Conference on Engineering & MIS 2020*. 1–8.

[2] Rajat Agarwal, Vaishnav Negi, Akshat Kalra, and Ankush Mittal. 2022. Deep Learning and Linguistic Feature Based Automatic Multiple Choice Question Generation from Text. In *Distributed Computing and Intelligent Technology: 18th International Conference, ICDCIT 2022, Bhubaneswar, India, January 19–23, 2022, Proceedings*. Springer, 260–264.

[3] Andreas Chandra, Affandy Fahrizain, Simon Willyanto Laufried, et al. 2021. A survey on non-english question answering dataset. *arXiv preprint arXiv:2112.13634* (2021).

[4] Bidyut Das, Mukta Majumder, Santanu Phadikar, and Arif Ahmed Sekh. 2021. Automatic question generation and answer assessment: a survey. *Research and Practice in Technology Enhanced Learning* 16, 1 (2021), 1–15.

[5] Bidyut Das, Arif Ahmed Sekh, Mukta Majumder, and Santanu Phadikar. 2021. Can deep learning solve a preschool image understanding problem? *Neural Computing and Applications* 33, 21 (2021), 14401–14411.

[6] Alexander R Fabbri, Patrick Ng, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. Template-based question generation from retrieved sentences for improved unsupervised question answering. *arXiv preprint arXiv:2004.11892* (2020).

[7] Deepak Gupta, Hardik Chauhan, Akella Ravi Tej, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Reinforced multi-task approach for multi-hop question generation. *arXiv preprint arXiv:2004.02143* (2020).

[8] Tianyong Hao, Xinxin Li, Yulan He, Fu Lee Wang, and Yingying Qu. 2022. Recent progress in leveraging deep learning methods for question answering. *Neural Computing and Applications* (2022), 1–19.

[9] Zhifeng Hao, Junhao Chen, Wen Wen, Biao Wu, and Ruichu Cai. 2022. Motif-based memory networks for complex-factoid question answering. *Neurocomputing* 485 (2022), 12–21.

[10] Helia Hashemi, Mohammad Aliannejadi, Hamed Zamani, and W Bruce Croft. 2020. ANTIQUE: A non-factoid question answering benchmark. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II 42*. Springer, 166–173.

[11] Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2022. Instance-aware prompt learning for language understanding and generation. *arXiv preprint arXiv:2201.07126* (2022).

[12] Samreen Kazi and Shakeel Khoja. 2021. UQuAD1. 0: Development of an Urdu Question Answering Training Data for Machine Reading Comprehension. *arXiv preprint arXiv:2111.01543* (2021).

[13] Aravind Krishnan, Srinivasa Ramanujan Sriram, Balaji Vishnu Raj Ganesan, and S Sridhar. 2023. An Extractive Question Answering System for the Tamil Language. *Advances in Science and Technology* 124 (2023), 312–319.

[14] Gokul Karthik Kumar, Abhishek Singh Gehlot, Sahal Shaji Mullappilly, and Karthik Nandakumar. 2022. Mucot: Multilingual contrastive training for question-answering in low-resource languages. *arXiv preprint arXiv:2204.05814* (2022).

[15] Yanxiang Ling, Fei Cai, Jun Liu, Honghui Chen, and Maarten de Rijke. 2023. Generating Relevant and Informative Questions for Open-Domain Conversations. *ACM Transactions on Information Systems* 41, 1 (2023), 1–30.

[16] Bang Liu, Haojie Wei, Di Niu, Haolan Chen, and Yancheng He. 2020. Asking questions the human way: Scalable question-answer generation from text corpus. In *Proceedings of The Web Conference 2020*. 2032–2043.

[17] Khalil Mrini, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter Chang, Emilia Farcas, and Ndapandula Nakashole. 2021. UCSD-adobe at MEDIQA 2021: Transfer learning and answer sentence selection for medical summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*. 257–262.

[18] Himadri Mukherjee, Subhankar Ghosh, Shibaprasad Sen, Obaidullah Sk Md, KC Santosh, Santanu Phadikar, and Kaushik Roy. 2019. Deep learning for spoken language identification: Can we visualize speech signal patterns? *Neural Computing and Applications* 31 (2019), 8483–8501.

[19] Aarthi Paramasivam and S Jaya Nirmala. 2022. A survey on textual entailment based question answering. *Journal of King Saud University-Computer and Information Sciences* 34, 10 (2022), 9644–9653.

[20] Keqin Peng, Chuantao Yin, Wenge Rong, Chenghua Lin, Deyu Zhou, and Zhang Xiong. 2021. Named entity aware transfer learning for biomedical factoid question answering. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 19, 4 (2021), 2365–2376.

[21] Qinjun Qiu, Miao Tian, Kai Ma, Yong Jian Tan, Liufeng Tao, and Zhong Xie. 2023. A question answering system based on mineral exploration ontology generation: A deep learning methodology. *Ore Geology Reviews* (2023), 105294.

[22] Siyu Ren and Kenny Q Zhu. 2021. Knowledge-driven distractor generation for cloze-style multiple choice questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4339–4347.

[23] Ellen Riloff and Michael Thelen. 2000. A rule-based question answering system for reading comprehension tests. In *ANLP-NAACL 2000 workshop: reading comprehension tests as evaluation for computer-based language understanding systems*.

[24] Mourad Sarrouti, Asma Ben Abacha, and Dina Demner-Fushman. 2021. Multi-task transfer learning with data augmentation for recognizing question entailment in the medical domain. In *2021 IEEE 9th International Conference on Healthcare Informatics (ICHI)*. IEEE, 339–346.

[25] Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. *arXiv preprint arXiv:1603.06807* (2016).
[26] Dhruv Sharma, Sanjay Purushotham, and Chandan K Reddy. 2021. MedFuseNet: An attention-based multimodal deep learning model for visual question answering in the medical domain. *Scientific Reports* 11, 1 (2021), 19826.
[27] Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 conference on empirical methods in natural language processing*. 3930–3939.
[28] Tasmiah Tahsin Mayeesha, Abdullah Md Sarwar, and Rashedur M Rahman. 2021. Deep learning based question answering system in Bengali. *Journal of Information and Telecommunication* 5, 2 (2021), 145–178.
[29] Liuyin Wang, Zihan Xu, Zibo Lin, Haitao Zheng, and Ying Shen. 2020. Answer-driven deep question generation based on reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics*. 5159–5170.
[30] Wonjin Yoon, Jinhyuk Lee, Donghyeon Kim, Minbyul Jeong, and Jaewoo Kang. 2020. Pre-trained language model for biomedical question answering. In *Machine Learning and Knowledge Discovery in Databases: International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part II*. Springer, 727–740.
[31] Hongwei Zeng, Zhuo Zhi, Jun Liu, and Bifan Wei. 2021. Improving paragraph-level question generation with extended answer network and uncertainty-aware beam search. *Information Sciences* 571 (2021), 50–64.
[32] Chen Zhao, Chenyan Xiong, Xin Qian, and Jordan Boyd-Graber. 2020. Complex factoid question answering with a free-text knowledge graph. In *Proceedings of The Web Conference 2020*. 1205–1216.
[33] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI open* 1 (2020), 57–81.
[34] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2018. Neural question generation from text: A preliminary study. In *Natural Language Processing and Chinese Computing: 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8–12, 2017, Proceedings 6*. Springer, 662–671.
[35] Shuohua Zhou and Yanping Zhang. 2021. Datlmedqa: a data augmentation and transfer learning based solution for medical question answering. *Applied Sciences* 11, 23 (2021), 11251.

Received 2023; revised xxx; accepted xxx