## Question-driven text summarization using an extractive-abstractive framework

# Question-driven text summarization using an extractive-abstractive framework

Mahsa Abazari Kia[1], Aygul Garifullina[2], Mathias Kern[2], Jon Chamberlain[1], Shoaib Jameel[3]

[1]School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK

[2]BT, Adastral Park, Ipswich, UK

[3]Department of Electronics and Computer Science, University of Southampton, Southampton, UK

**Correspondence**

Mahsa Abazari Kia, Computing, Mathematics, Engineering and Natural Sciences Faculty, Northeastern University London, London, UK.

Email: mahsa.abazari@nulondon.ac.uk

## Abstract

Question-driven automatic text summarization is a pop-ular technique to produce concise and informative answers to specific questions using a document collec-tion. Both query-based and question-driven summariza-tion may not produce reliable summaries nor contain relevant information if they do not take advantage of extractive and abstractive summarization mechanisms to improve performance. In this article, we propose a novel extractive and abstractive hybrid framework designed for question-driven automatic text summarization. The framework consists of complimentary modules that work together to generate an effective summary: (1) discov-ering appropriate non-redundant sentences as plausi-ble answers using an open-domain multi-hop question answering system based on a convolutional neural net-work, multi-head attention mechanism and reasoning process; and (2) a novel paraphrasing generative adver-sarial network model based on transformers rewrites the extracted sentences in an abstractive setup. Exper-iments show this framework results in more reliable abstractive summary than competing methods. We have performed extensive experiments on public datasets, and the results show our model can outperform many question-driven and query-based baseline methods (an R1, R2, RL increase of 6%–7% for over the next highest baseline).

**KEYWORDS**

abstractive text summarization, hybrid text summarization, multi-hop QA, question-driven text summarization

# 1 | INTRODUCTION

With the advent of the Internet and social media, we have seen an exponential rise in text generated every day. It is difficult to keep pace with the quantity of information conveyed. By reducing extraneous material from documents, text summarizing minimizes the time to comprehend the content, allowing a reader to focus on important aspects rapidly. Shortening a text document while maintaining its overall meaning and information conveyed is the goal of text summarization.[1] Automatic text summarization (ATS) systems are categorized as single-document or multi-document systems.[2] The former generates a single document summary, whereas the latter generates a summary for a group of documents. ATS systems are created by employing either an extractive or abstractive approach. Another class of models jointly models abstractive and extractive paradigms, mainly hybrid approaches. In extractive models, the goal is to extract sentences or pieces of text from the document to optimize the information conveyed by removing redundancy. The task of rephrasing the language by comprehending the semantic information conveyed by the documents is known as abstractive text summarization, which necessitates a thorough understanding of natural language processing.[3] Abstractive text summarization can be regarded as more appealing than extractive summarization, but it is challenging to perform because it requires the capability of generating new sentences.[4] Hybrid approaches combine extractive, and abstractive models, which exploit the complementary advantages between the two.[5]

Compared to extractive summarization, the content produced by abstractive methods often suffers issues such as data redundancy, poor readability, and major semantic diversion from the source document(s). The majority of contemporary abstractive summarization models are built on neural networks with sequence-to-sequence (seq2seq).[6–11] They are composed of encoders for the purpose of comprehending the input sequence and decoders for the purpose of generating the output sequence. However, there are four significant drawbacks to generating reasonable text with seq2seq neural networks: (1) the out-of-vocabulary (OOV) problem; (2) continually producing a particular word or phrase, which introduces redundancy; (3) test-time exposure bias; and (4) non-optimized learning for evaluation metrics used by models in disciplines such as text summarization and machine translation. As a result, they cannot generate appropriate abstractive summaries since they cannot convey the semantics of the document.[12,13] To recreate key content, abstractive summarization requires advanced natural language techniques for reading and understanding the text. In contrast, the extractive summaries may include repeated terms, high frequency of particular phrases, and redundancy in some sentences.[2]

We can diminish the weaknesses of the extractive and abstractive methods by proposing a hybrid framework that combines their strengths. Extractive methods excel at retaining the original source content, ensuring that crucial details and facts are not omitted in the summary.[14] Abstractive methods have the ability to rephrase and generate content creatively, providing summaries that are more concise and easier to understand for readers.[15] Abstractive methods can eliminate redundancy by rewriting similar sentences or phrases, making the summary more coherent and easier to read.[16] A hybrid approach can harness the content retention strength of extractive methods and the content generation ability of abstractive methods. This results in summaries that are not only comprehensive but also more concise and reader-friendly.[17]

Text summarizers have applications in different domains: generic (domain-independent); specific (domain-dependent); opinion/sentiment summarization; query-based summarization; and question-based summarization.[18] Generic (domain-independent) text summarization

provides a brief overview of a long document, conveying the core message of the document[2] and summarizes documents that are from different domains.[19] The domain-specific ATS systems, on the other hand, are designed to summarize documents within a specific domain (e.g., legal documents,[18,20] or medical reports[21–23]). Opinion summarization refers to the process of automatically summarizing multiple opinions that discuss the same subject.[24] A query-based summarization approach summarizes query-related content from the source document(s).[25] Question-driven summarization approaches answer a question and also provide additional informative content to that answer from the source document(s) to make it more understandable and convincing.[26] A question-driven summary must satisfy three goals: answerability, understandability, and persuasiveness. The primary goal of question-driven text summarization is to generate a concise summary of a given document in response to a specific question. The output in this case is a summary that not only contains the answer to the question but also contains additional relevant information about the answer and the question itself. It aims to provide a condensed version of the document that answers the question.

There have been several attempts to develop methods for question-driven automatic text summarization.[26–30] Examples of query-based and question-based text summarization are provided in Table 1. The bold text shows the relevant text to the proposed query and question, and the gold summary is the abstractive summary generated by a human. The query-based summary has summarized the text given the query "foods for lower blood sugar." It contains all the information about the diet for hyperglycemic people. Still, the question-driven summary is shorter and only contains specific information, the answer and its explanation, to the question.

Query-based summarization techniques use semantic relevance measurement to summarize the query-related content from the source document.[31–34] These approaches are not suitable for tackling question-driven summarization problems in Question Answering (QA) scenarios. For

**TABLE 1** Query-based and question-driven text summarization example.

| |
|---|
| Text: … **According to powers, your eating plan should focus on the type and amount of carbohydrates you eat throughout the day. Choose low-carb vegetables such as mushrooms, onions, eggplant, tomatoes, Brussels sprouts, and zucchini, as well as low-carb squashes. To add flavor and texture to a meal, serve them with low-fat souces, hummus, guacamole, and salsa, or roasted with herbs and spices like rosemary, cayenne pepper, and garlic. Sweet potatoes, when combined with other meals, can successfully slow food digestion, increase satiety, and moderate blood sugar swings. Thus, individuals who are hyperglycemic can have some sweet potatoes, which will not only not elevate blood sugar but will also aid in blood sugar control.** … |
| Query: Foods for lower blood sugar |
| Query-based gold summary: The amount and type of carbs you put in your diet throughout the day should be seriously considered. Low-carb and tasty veggies, like mushrooms, onions, eggplant, tomatoes, Brussels sprouts, and low-carb squashes, like zucchini with dips such as low-fat dressings, hummus, guacamole, and salsa, or roasted with different seasonings such as rosemary, cayenne pepper, or garlic could be included to the meal for better flavor and texture. Sweet potatoes can help to slow down food digestion, increase satiety, and stabilize blood sugar levels which not only do not raise blood sugar but also help to control blood sugar. |
| Question: How sweet potatoes helps people with hyperglycemic? |
| Question-driven gold summary: Sweet potatoes can help to slow down food digestion, increase satiety, and stabilize blood sugar levels. As a result, persons with hyperglycemia can eat sweet potatoes, which not only do not raise blood sugar but also help to control blood sugar. |

question-driven summarization, answer detection and the reasoning on the detected answer are needed.

To overcome the problems mentioned above and acquire a reliable summary of the document regarding a question, we propose a two-stage, hybrid extractive and abstractive summarization that combines the advantages of the two methods. First, the extractive model selects the answer sentence and its supporting sentences, which provide details or explanations for the answer sentence. After obtaining the question-driven extractive summary, a novel abstractive model transforms the extractive summary into an abstractive summary. The extractive phase reduces the amount of redundant information, which improves the effectiveness of the abstractive summarizer. We propose a question-driven abstractive summarization model, depicted in Figure 1 which we describe in detail later, and the main contributions can be summarized as follows:

1. A novel open-domain multi-hop QA model based on a convolutional neural network (CNN) and multi-head attention mechanism designed to comprehend the document and question for constructing the question-driven extractive summary. The answer selector module measures the semantic dependencies between the local features extracted from the document sentences and question to select the answer sentence.
2. A novel reasoning approach is proposed for analyzing the document regarding the detected answer sentence and searching for relevant supporting sentences based on lexical coverage and contextual semantic similarity.
3. A novel paraphrase framework based on general adversarial networks (GANs), Q-learning, and transformers to produce question-driven abstractive summaries from the generated extractive summaries. We showed that rewriting the extractive summaries using a paraphrase generation model helps us have abstractive summaries closer to gold summaries (human-generated summaries).

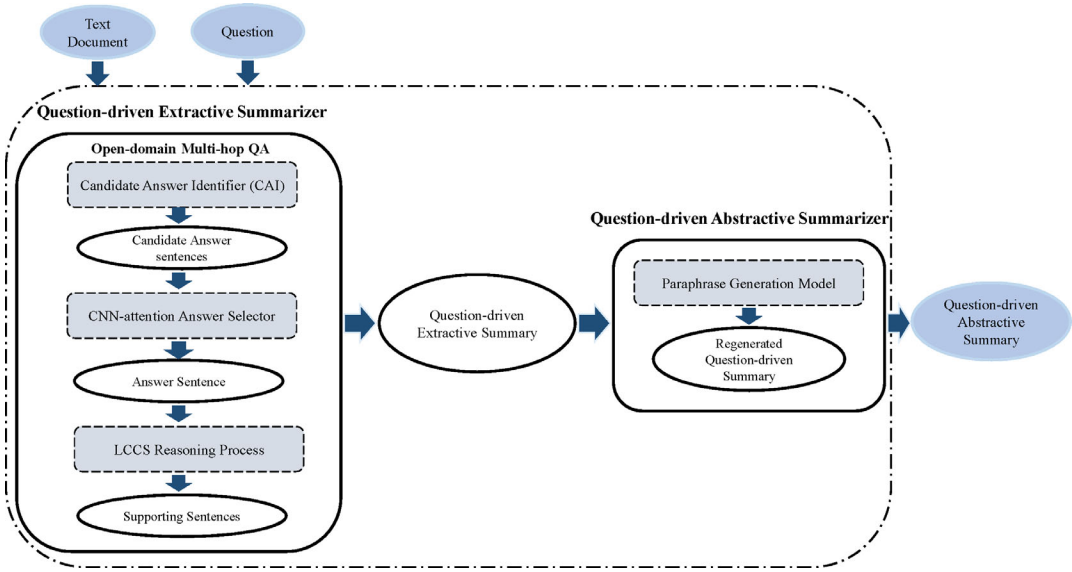


**FIGURE 1** Our proposed hybrid question-driven text summarization framework.

## 2 | RELATED WORK

In this section, we cover closely related literature. We also mention how our proposed method is different from the existing methods. Since our approach is a question-driven hybrid text summarization based on GANs for generating abstractive summaries, we have covered the existing GAN methods for text generation, query-based summarization, hybrid text summarization, abstractive summarization, and paraphrase generation approaches in the following subsections.

### 2.1 | Generative adversarial networks for text generation

GAN was first applied in the computer vision domain. Applying GAN to text generation is non-trivial because GAN is designed for generating real-valued, continuous data but struggles to generate discrete token sequences, such as texts. However, several studies are proposed to tailor this powerful network for text generation; therefore, we have summarized recent approaches and their advantages and drawbacks in this section. Generative adversarial nets (GAN) was proposed in Reference 35. It consists of two simultaneously trained models: one (the generator) trained to create new data, and the other (the discriminator) trained to distinguish the generated data (fake data) from real examples. Yu et al.,[36] introduced the first reinforcement learning (RL)-based[37] work, called sequence generative adversarial network (SeqGAN) which Monte Carlo tree search (MCTS) is used to calculate the reward at every generation step for evaluating a generated subsequence which is computationally expensive but in StepGAN proposed by Tuan et al.,[38] the discriminator is altered to automatically provide scores at each generation stage for assessing the quality of each subsequence. In StepGAN, a seq2seq generator and discriminator are designed, and the discriminator predicts the immediate rewards using Q-learning[39] without performing a tree search. TextGAIL [40] improves the discriminator's guidance by combining RoBERTa and GPT-2[41] with recent advances in RL. Zhang et al.,[42] enhance the standard actor-critic methodology[43] by designing a transformer-based generator and a CNN-based discriminator. A key barrier is a language's inherent characteristics, such as syntax, grammar, and semantic aspects. The model must learn the correct connection between words and characters to generate a viable text, commonly accomplished through various memories and situations (prior knowledge). Such issues can be addressed in a more robust pre-learning step, in which pre-trained embedding models BERT,[44] A lite bert for self-supervised learning of language representations (ALBERT),[45] ELECTRA,[46] or GPT-2 are combined with transformer-based seq2seq architectures to be capable of generating plausible "natural" language text. Transformer-based GANs incorporating contextualized pre-trained language models and stepwise evaluation are blank spots that still need to be appropriately addressed for text generation, which we have presented in this article.

### 2.2 | Query-based text summarization

As we present a deep-learning-based approach in this article, we describe recent approaches based on neural networks. Nema et al.,[33] introduced a typical encode-attend-decode model (based on LSTM) for query-based abstractive summarization, which first computes a vectorial

representation for the document and the query, and then the decoder produces a contextual summary one word at a time. Li et al.,[47] designed a bi-GRU sentence-level encoder is proposed to encode a sentence in a document, and then a query filter component attention model upon the sentence encoder is designed to inject such information into sentence encoding and computing the new sentence encoding, including the query information. In the end, a feed-forward neural network is applied to compute a salience score for each sentence. Ishigaki et al.,[34] introduced three copying mechanisms designed for query-based abstractive summarizers. In the copying mechanism, two different probabilities for every word in the vocabulary are considered, the copying probability and the generation probability. Zhao et al.,[48] have designed three solutions for Chinese query-based document summarization utilizing relevance ranking, dual attention and pre-trained word embeddings, BERT-based encoder and a text-pair classification which performed better than the other two methods. We have studied recent query-based text summarization approaches in this section to examine their ability to be used for question-driven text summarization. The main goal in query-based text summarization approaches is to summarize the retrieved relevant information to the query, but in the question-driven text summarization, answer detection and explaining that answer in a summarized form is desired. Furthermore, the query-based text summarization approaches are not adaptable for the question-driven summarization problem.

## 2.3 | Extractive text summarization

In this section, we present both deep learning and graph-based extractive approaches for generic text summarization. The classic graph architecture involves a two-stage process of mapping a document into a graph network, where the vertices are sentences and the edges are the similarity between these sentences, and extracting the top-K sentences. The sentences are ranked based on the graph centrality scoring of each sentence.[49,50] Tohalino et al.[51] represented sentences as nodes and edges are established according to the lexical similarity between two sentences. A multilayer network is created by considering each document as a layer. As such, two types of links arises: those connecting sentences from the same documents and the links connecting sentence from distinct sources. They also used dynamical measurements to improve the characterization of the obtained networks. Amanico et al.[52] have extended the use of complex networks for developing extractive summarizers. Upon testing several new metrics, they found that vulnerability, closeness, and betweenness were not appropriate for selecting sentences for the summarizers. In contrast, diversity metrics proved excellent for this selection. Cui et al.[53] proposed a graph-based extractive model based on pre-trained BERT model to acquire contextual sentence representations and identify underlying topics through a joint neural topic model.[54] Then, the document is transformed into a graph network, where sentences represent the vertices and the connections denote the similarity between these sentences. Cao et al.,[55] among the supervised deep learning extractive approaches, suggested a recursive neural network for rating significant sentences in multi-document summarization. They generated summary prior features for extractive text summarization using improved CNNs. Cheng et al.[56] approach consists of a hierarchical document reader or encoder based on neural networks and an attention-based content extractor. The reader is responsible for deriving the document's meaning from its sentences and their constituent words. To extract sentences or words, their algorithms use a type of neural attention.

## 2.4 | Abstractive text summarization

Various approaches are proposed for abstractive summarization, but here we only consider those based on GANs, which are more relevant to our work. Liu et al.,[57] used RL (i.e., policy gradient) to optimize the bi-directional LSTM generator and implemented the discriminator as a trained text classifier to classify the generated summaries as a machine or human-generated. Scialom et al.,[58] introduced an approach using discriminative adversarial search (DAS) utilizing the Unified Language Model for natural language understanding and generation (UniLM)[59] based on BERT for generator. The seq2seq based discriminator is integrated into a beam search that obtains a label at each generation step to refine the probabilities and select the top candidate sequences. Rekabdar et al.,[60] designed a generator based on LSTM encoder–decoder with an attention mechanism which is modeled as a stochastic policy in RL, and the discriminator is based on CNN. Dang et al.,[61] presented a GAN model containing one generator and two discriminators. The generator is based on LSTM encoder–decoder and one of the discriminators is the similarity discriminator based on CNN text classifier with four classes (incomplete class, redundant class, similar class, irrelevant class). The other one is readability discriminator, a CNN-based model which tells whether the generator or human generates the summary.

Our method is significantly different from these methods in several ways, and it consists of two main components, extractive summarizer, and abstractive summarizer. The extractive component filters the irrelevant information and feeds the pruned information (extractive summary) to the abstractive component. We have designed a transformer-based GAN with Q-stepwise evaluation for abstractive part which regenerates and rewrites the generated extractive summaries and produces reliable abstractive summaries. Using transformers architectures with GAN and applying stepwise evaluation for generating text is an unexplored architecture which we proposed and studied in this article.

## 2.5 | Hybrid text summarization

The hybrid approaches combine the abstractive and extractive approaches and their advantages.[2] Wang et al.,[62] proposed a hybrid system "EA-LTS" comprises two stages, the extractive stage selects the key sentences using a graph model, and the abstractive stage is an RNN based encoder–decoder in addition to an attention and pointer mechanisms to generate summaries. Bhat et al.,[63] proposed "SumItUp" for single-document summarization consisting of an extractive sentence selection based on statistical features and an abstractive summary generation for converting extractive summary to abstractive using a language generator. Subramanian et al.,[64] created a basic extraction step utilizing a hierarchical bidirectional LSTM seq2seq sentence pointer. This phase minimizes the amount of context for a following abstractive step with a single trained transformer language model. Chen et al.,[65] presented a framework composed of five components: (1) word-level bidirectional GRU encoder for encoding the sentences word-by-word, (2) sentence-level bidirectional GRU encoder encodes the document sentences, (3) sentence extractor for labeling each sentence, (4) hierarchical attention facilitates generating the sentence-level and word-level context vectors to be consumed in the decoding steps, (5) a GRU-based decoder with a beam search algorithm decodes the output word sequence. Jin et al.,[66] unified extractive and abstractive summarization into one architecture based on attention mechanism. Extractive summarization works on sentence granularity and directly conducts the sentence representations, while abstractive summarization is designed for operating on word granularity and their

representations. Our work is different from the above approaches in several ways. First, we generate an extractive summary using the proposed multi-hop QA system and relevance ranking method, then a paraphrase generation model is designed for transforming the extractive summary to abstractive.

## 2.6 | Paraphrase generation

The task of paraphrase generation refers to generating one or multiple candidate paraphrases given the input sentence, which requires that the generated sentence and input sentence are different in expression form, but have the same expressed meaning.[67] Li et al.,[68] introduced DNPG as a way to decompose a sentence into sentence-level and phrase-level patterns in order to make neural paraphrase creation more intelligible and controlled, and they observed that DNPG could be applied to unsupervised domain adaptation for paraphrase production. Fu et al.,[69] suggested a novel model of paraphrasing based on a latent bag of words. Siddique et al.,[70] suggested an unsupervised paraphrase model using a variational autoencoder in a deep reinforcement learning framework. Liu et al.,[71] regarded paraphrase generation as an optimization issue and created a complex objective function. All of the strategies outlined above are concerned with the general quality of paraphrases and are unconcerned with their variety. Yang et al.,[72] Cao et al., [73] Vizcarra et al., [74] and Tuan et al., [38] proposed paraphrase generation models based on GAN which we consider them as our baseline methods and discussed them in detail in Section 4.2. Paraphrase generation is a fundamental task of natural language processing (NLP) that has been broadly used in many downstream applications, such as information retrieval, machine translation, question answering and so on. This is the first work that employs a paraphrase generation model for generating abstractive summaries to the best of our knowledge. To this end, we have proposed a novel paraphrase model based on transformers and Q-learning stepwise evaluation for text generation, which is an unexplored architecture.

# 3 | OUR NOVEL QUESTION-DRIVEN HYBRID TEXT SUMMARIZATION MODEL

We proposed a hybrid text summarization approach for generating question-driven extractive-abstractive summaries. Our inputs are a text document and a question as depicted in Figure 1. We developed an open-domain multi-hop QA system to select the answer sentence and extract the supporting sentences for generating the question-driven extractive summary. To generate high-quality summaries for human consumption, a novel paraphrase generation model is proposed to rewrite the sentences of the extractive summary and construct a question-driven abstractive summary. In a nutshell, in our novel framework, we exploit the advantages of both extractive and abstractive models. Our novel extractive model automatically selects the most appropriate sentences from the document that conveys non-redundant and important information to the question. Subsequently, our novel abstractive paraphrasing model uses GAN and transformers to generate high-quality abstractive summaries so that the resulting summaries are coherent and readable. As mentioned above, a key advantage of our model is that the extractive phase helps remove redundant information which not only helps improve the quality of the summary generated by our abstractive summarizer but also makes it efficient because the abstractive phase does not have to deal with a large amount of data.

In the subsection below, we describe our question-driven extractive summary model followed by the question-driven abstractive model.

## 3.1 | Question-driven extractive model

We have proposed a question-driven extractive summarizer based on an open-domain multi-hop QA system comprising candidate answer identifier (CAI), answer sentence selector, and reasoning process. Multi-hop QA is a type of natural language processing (NLP) system designed to answer complex questions that require gathering information from passages. As not all the relevant information about the question and its answer could not be found directly in a single passage and it needs reasoning and aggregation of information from several sources, passages, or intermediate steps (hops) to arrive at the final answer and information.[75] Answer selection, reasoning and information aggregation, and summary generation are the steps that our proposed multi-hop QA does and it's necessary for generating an extractive question-driven summary.

We have utilized CAI module introduced in Reference 76 which has six functions based on linguistic and syntactic features and patterns for reducing the document to sentences (candidate answer sentences) that could answer the given question. We designed a joint CNN and multi-head attention neural network to analyze and assign a score to each candidate answer sentence based on its relevance to the question. The CNN-attention layer calculates the relevance score based on the correlation of the semantic features extracted from the question and the candidate answer sentence.

After selecting the answer sentence, an unsupervised reasoning process (we call it *LCCS* reasoning process) based on ***L***exical ***C***overage and ***C***ontextualized ***S***imilarity for selecting supporting sentences (justification sentences). The reasoning process helps us select appropriate, relevant sentences explaining the answer sentence and then constructs the final extractive summary. The overall framework of our extractive model is shown in Figure 2. We rearrange the justification sentences and answer sentence according to their original indexes in the given document to bring coherence in the selected sequence of sentences and generate the question-driven extractive summary.

The generated question-driven extractive summary by the proposed multi-hop QA contains the answer sentence (the direct answer to the user's question) and key information or facts from the source text that are relevant to the question and/or answer sentence which is all the required information for a comprehensive question-driven extractive summary.

The example in Figure 2 shows a "what" question and the list of candidate answers for "what" questions generated by CAI are retrieved. Then the answer sentences in the list are forwarded to the model one by one. The question and the candidate answer are concatenated into a single sequence and the embedding for this sequence is obtained by pre-trained ALBERT contextual language model. The generated embedding is consumed by the CNN model for extracting significant n-gram features from the question and candidate answer. In the next layer, the multi-head attention mechanism is applied to capture more comprehensive semantic information and assess the semantic connection between the key features of the question and candidate answer sentence for determining the relevance score. The relevance score is calculated for all of the "what" candidate answers and the one with the highest score is selected. The selected answer sentence is the first sentence of the question-driven extractive summary, the other required information is acquired by analyzing other sentences in the document and their relevance to the question and the answer sentence. This process is accomplished by the LCCS reasoning component.

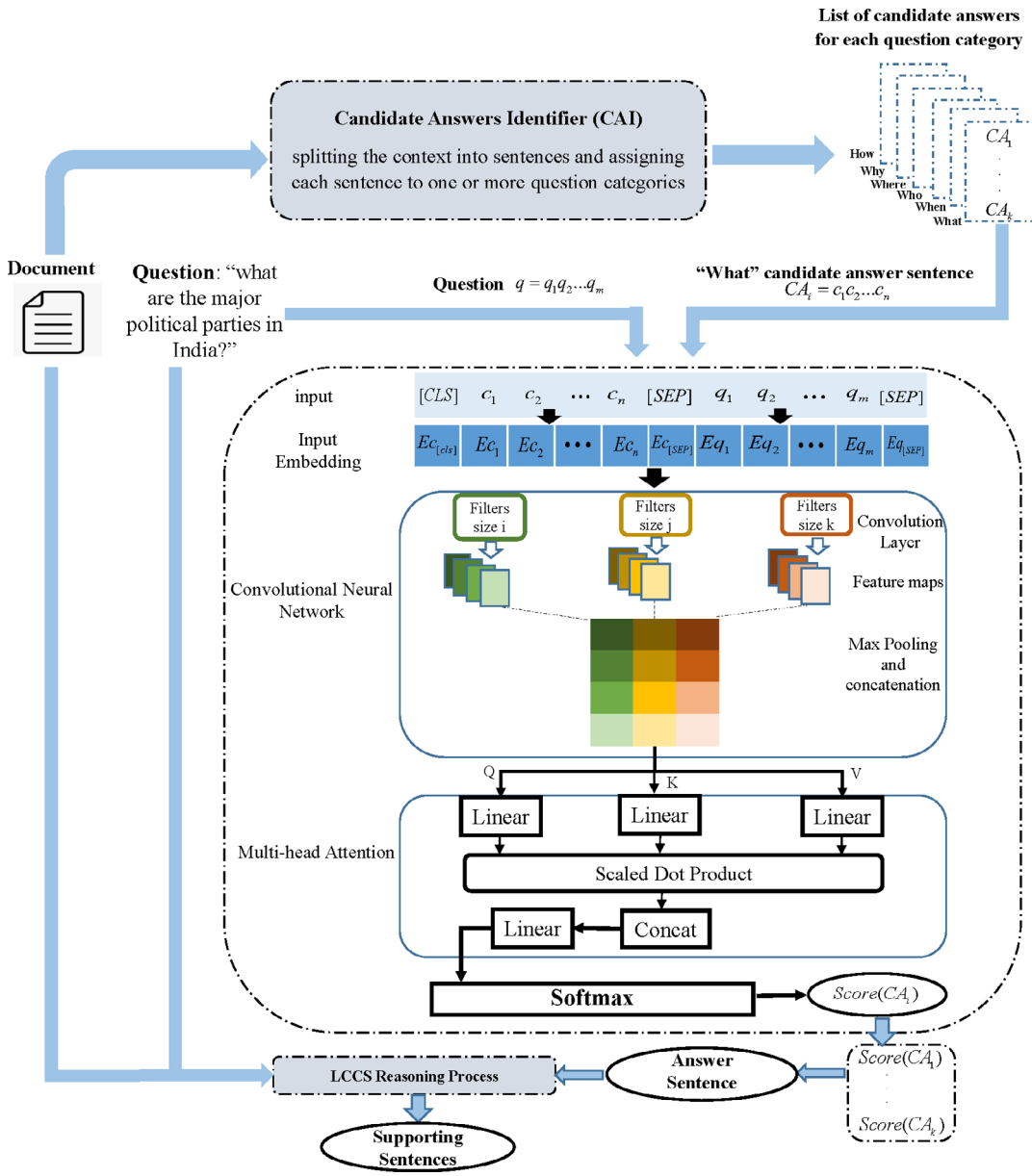


**FIGURE 2** The overall framework of our multi-hop QA model for an example "What" question. Answer sentence and supporting sentences are concatenated according to their original indexes in the given document for generating the question-driven extractive summary.

### 3.1.1 | Candidate answers identifier (CAI)

The candidate answer identifier (CAI) module limits the document to the sentences that are capable to answer the question based on its category (*When*, *Where*, *Who*, *What*, *Why*, *How*). CAI contains six functions for classifying the document sentences based on their linguistic and syntactic features. Preprocessing and splitting the document into sentences is the first step in the CAI

module. Then each sentence will be analyzed to be assigned to one or more question categories based on their linguistic features, shown in Figure 3. NN, NNP, NNS, NNPS, PRP, VB, VBG, and IN stand for a singular noun, singular proper noun, plural noun, plural proper noun, personal pronoun, verb, present participle verb, and preposition or subordinating conjunction respectively.

### 3.1.2 | CNN multi-head attention-based answer selector

There are $K$ possible candidate answers $CA_1, CA_2, \ldots, CA_k$ in the document $D$ for the given question $q$. The question $q$ and candidate answer $CA_i$ are concatenated into a single sequence with $m$ and $n$ tokens. The special token [*SEP*] is used to denote the distinction between the question and candidate answer in the CNN multi-head attention layer's input. For the embedding layer, we get the semantic representation of $q$ and $CA_i$ using a pre-trained ALBERT contextual language model. The objective is to obtain the most trustworthy answer sentence $CA_j$ to the $q$ in $D$. The output of ALBERT is taken only for the first token [*CLS*] which is used as the aggregate representation of the sequence. Untrained layers of CNN, pooling, and multi-head attention are added for fine-tuning the pre-trained ALBERT model. The CNN and attention mechanism constructs the question and sentence representation by focusing on the most significant features and their connections. The model produces a score for the candidate answer sentence utilizing the constructed semantic representation.

*Convolutional neural network*

CNNs are capable of learning and extracting significant n-gram features from the input text in order to generate a useful semantic representation for the subsequent tasks.[77] The convolution layer comprises of many convolution filters (also called kernel). For a sentence with $n$ words, $c_i$ is generated by applying the filter $\omega \in R^{ld}$ on a window of $l$ words where $l$ is the filter size.

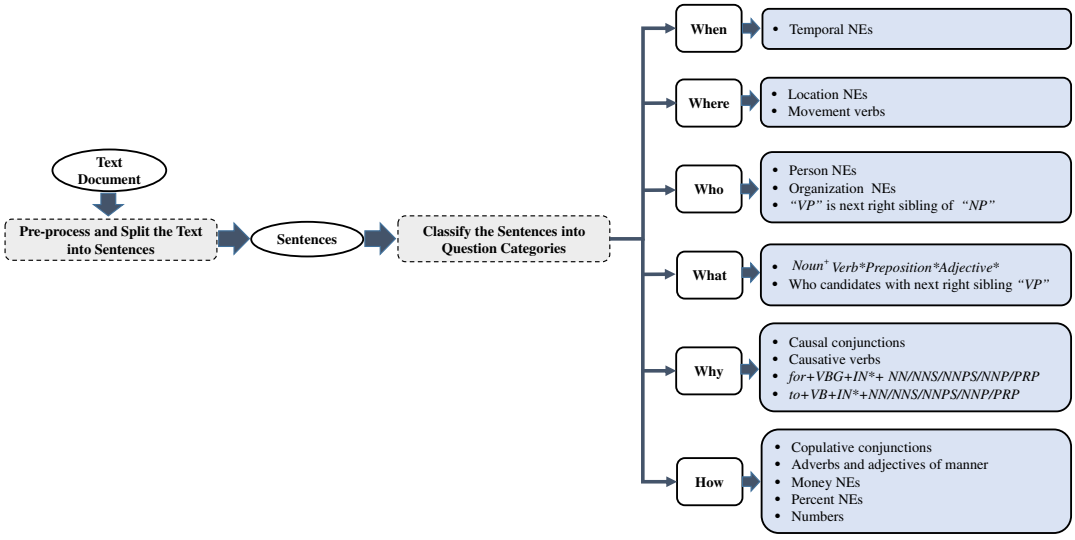


**FIGURE 3** The candidate answer identifier (CAI) module for identifying the candidate answer sentences for each question category (What, Where, When, Why, Who, How).

Here, $e_i \in R^d$ is the word embedding for the $i$th word in the sentence where the word embedding dimension is $d$, $f$ is a nonlinear activation function, and $b$ is the bias term.

$$c_i = f(e_{i:i+l-1}.\omega^T + b). \tag{1}$$

Using the same weights, the filter $\omega$ slides across the full sentence embedding matrix to construct the feature map $c = [c_1, c_2, \ldots, c_i, \ldots, c_{n-l+1}]$. Proposing a maximum pooling method after the convolution layer diminishes the output's dimension and gives us low dimension dominant features. All sampled feature maps generated by max pooling layer are combined into $\hat{C} = [\hat{c_1}, \hat{c_2}, \ldots, \hat{c_t}]$ as output of CNN ($\hat{c} = \max\{c\}$).

In our model, CNN extracts key local features from the aggregated question and candidate answer sentence. The feature vectors are concatenated to construct the matrix $Y$ as the multi-head attention layer input and global feature matrix.

*Multi-head attention*

The conventional attention mechanism is confined to acquiring attention information from a single level. Multiple linear transformations are performed to the input feature matrix in the multi-head attention mechanism to learn the attention representation of the text for obtaining more comprehensive semantic information.[78] We employed a multi-head attention comprising multiple self-attention mechanism (shown in Figure 4) to assess the semantic connection between the key features of the question and candidate answer sentence for determining the relevance score. The query matrix ($Q$), the key matrix ($K$), and the value matrix ($V$) in self-attention mechanism are initiated with the matrix $Y$, the CNN layer's output.

$$Q = K = V = Y. \tag{2}$$

Scaled dot-product attention (SDA) is one of the key concepts in the self-attention mechanism. To avoid an excessively large dot product, the dot product of $Q$ and $K$ is divided by $\sqrt{d_k}$ ($d_k$ is

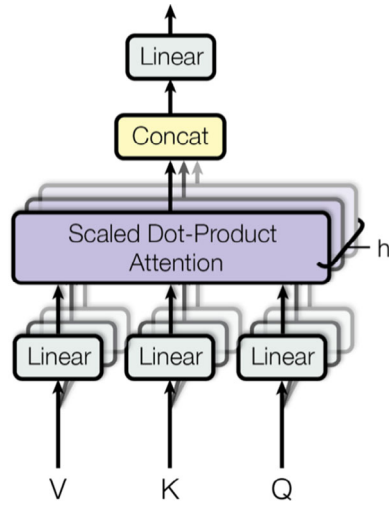


**FIGURE 4** Multi-head attention structure.

the matrix $K$ dimension). Multiplication by matrix $V$ is performed to capture the expression of attention after the normalization by Softmax.

$$\mathrm{SDA}(Q, K, V) = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{3}$$

Using different parameters $W_i^Q$, $W_i^K$, $W_i^V$ to perform linear transformation is the core idea of the multi-head attention mechanism. Applying the SDA on the linear transformation results is demonstrated by $\mathrm{head}_i$, as shown in (4).

$$\mathrm{head}_i = \mathrm{SDA}(QW_i^Q, KW_i^K, VW_i^V). \tag{4}$$

Concatenating the computed results $\mathrm{head}_1$ to $\mathrm{head}_h$ creates a matrix that is multiplied by the parameter $W$ to complete the final linear transformation. $H$ is the attention value of the entire sentence, depicted in (5), where $h$ is the number of heads in the multi-head attention mechanism.

$$H = \mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)W. \tag{5}$$

We perform average pooling to derive the feature vector $f$ for integrated $q$ and $\mathrm{CA}_i$ from the output matrix of the multi-head attention layer $H$ and then feed $f$ into the final softmax layer through the fully connected layer. The candidate answer score is defined based on the answer selection task where two labels are used to show the answer sentence. The $\mathrm{Score}(\mathrm{CA}_i)$ is only calculated for label 1 for all the candidate answer sentences. The candidate with the highest score is adopted as the answer sentence for the question $q$. Here, $C$ indicates the label, $w_c$ is the weight matrix, and the bias is $b_c$.

$$\mathrm{Score}(\mathrm{CA}_i) = P(C = 1|\mathrm{CA}_i, q), \tag{6}$$

$$P(C|\mathrm{CA}_i, q) = \mathrm{softmax}(w_c f + b_c). \tag{7}$$

### 3.1.3 | LCCS reasoning process

To tackle question-driven extractive summarization, the content selection process is not only determined by answer sentence selection to the given question. It also necessitates human-like reasoning for considering the content interrelationships thoroughly and meticulously across the whole document text. In other words, if we solely focus on the answer sentence for the given question, the resulting summary is likely to miss vital information. We have proposed a reasoning process based on ***L***exical ***C***overage and ***C***ontextualized ***S***imilarity for selecting justification sentences (LCCS reasoning process). We consider all the sentences in the document ($D$) as the candidate justifications sentences ($\mathrm{JC}_i$), and those candidates that are closest to the question ($q$), answer sentence (AS), and selected justification sentences ($\mathrm{JS}_i$) in the embedding space are selected. We utilized pre-trained BERT for generating the contextualized embedding for the candidate sentences, question, and AS, then the cosine similarity is calculated to generate a contextualized similarity score. Also, we measure the lexical coverage of the candidates with the $q$, AS, and $\mathrm{JS}_i$ keywords (unique terms) in (8) ($X = q, X = \mathrm{AS}, X = \mathrm{JS}_i$).

$$C(X, \mathrm{JC}_i) = \frac{|t(X) \cap t(\mathrm{JC}_i)|}{\max(|t(X)|, |t(\mathrm{JC}_i)|}. \tag{8}$$

$|t(X) \cap t(\mathrm{JC}_i)|$ is the size of common terms in $X$ and $\mathrm{JC}_i$ and $|t(X)|$, $|t(\mathrm{JC}_i)|$ are the size of unique terms of $X$ and $\mathrm{JC}_i$ (Algorithm 1).

**Algorithm 1.** Reasoning process

**Input**: Question (q), Document (D), Answer Sentence (AS), size of justification set (J-num)
**Output**: Set of justification sentences (JS-list) with size J-num
k=1
**while** ($k$ <= J-num) **do**
  **for** sentence(JC) **in** D **do**
    ASq-score = C(AS,JC)+C(q,JC)+CosSimilarity(AS,JC)+CosSimilarity(q,JC)
    **if** ($k > 1$) **then**
      JS-score = $\sum_{i=1}^{|\text{JS-list}|} C(\mathrm{JS}_i, \mathrm{JC}) + \text{CosSimilarity}(\mathrm{JS}_i, \mathrm{JC})$
    **else**
      JS-score = 0
    **end if**
    Score(JC) = ASq-score + JS-score
  **end for**
  **return** JS = (JC with highest score)
  JS-list.Add(JS)
**end while**
**return** JS-list

## 3.2 | Question-driven abstractive model

We have proposed a paraphrase framework to transform the generated extractive summary to an abstractive summary. The input to this novel model is the extractive summary that we have obtained above. The details of this framework are presented in the following sections, and the trained paraphrase model is used for abstractive summary generation. Since our abstractive text summarizer is based on paraphrasing paradigm where the main goal is to effectively generate relevant abstractive summaries, we have found that GAN is a suitable framework to handle this task because of its ability to generate new samples.

In Figure 5, a paraphrase generation model is designed for generating an abstractive summary by rewriting the generated extractive summary. The aim of paraphrasing the extractive summary is to eliminate redundancy by rewriting similar sentences or phrases, making the summary more coherent and easier to read. In the proposed GAN model, the generator generates synthetic samples given a random noise (sampled from a latent space) and the discriminator is a binary classifier that discriminates between whether the input sample is real (output a scalar value 1) or fake (output a scalar value 0). As depicted in Algorithm 2, the generator and discriminator are pre-trained individuality, and then both of them are trained for $n$ rounds while the discriminator evaluates the generator with both real data (from the dataset) and fake data

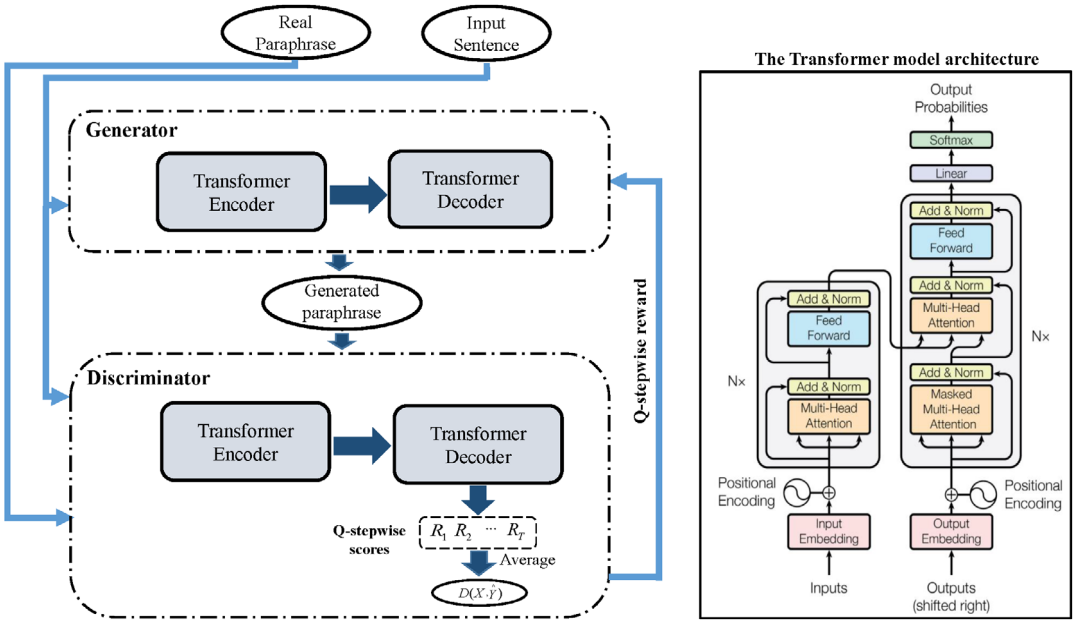


**FIGURE 5** The illustration of the proposed GAN for paraphrasing.

**Algorithm 2.** Training the paraphrasing model

**Result**: Trained $G_\theta$
Pre-train $G_\theta$
Generate samples using $G_\theta$
Pre-train $D_\phi$ with fake and real pairs
**for** $n$ rounds **do**
  **for** $i = 1$ to G-iteration **do**
    Sample $X$ from real data
    Generate a sequence $\hat{Y}$ using $G_\theta$
    Calculate $R$ for each sequence step
    Update $G_\theta$ using Equation (13)
  **end for**
  **for** $j = 1$ to D-iteration **do**
    Sample $(X, Y)$ from real data
    Sample $(X, \hat{Y})$ using $G_\theta$
    Update $D_\phi$ using Equation (12)
  **end for**
**end for**

(generated by the generator) by calculating rewards by utilizing the idea of Q-learning. The trained GAN paraphraser based on transformers is able to rewrite the sentences in the generated extractive summary and produce a question-driven abstractive summary which is less redundant and similar to the human-generated summary.

### 3.2.1 | Paraphrasing for question-driven abstractive summary generation

We begin by defining two sequences of tokens $X_{1:n} = \{x_1, \ldots, x_n\}$ and $Y_{1:T} = \{y_1, \ldots, y_T\}$, where the sequence $X$ represents an input sequence and $Y$ represents a paraphrase. We have designed a GAN model, depicted in Figure 5, for generating paraphrases. To this end, we have $G_\theta$ and $D_\phi$ to be a $\theta$ parameterized generator and a $\phi$ parameterized discriminator. We train $G_\theta$ to generate a sequence of tokens $\hat{Y}_{1:T} = \{\hat{y_1}, \ldots, \hat{y_t}\}$ that is similar to $Y$ for the given $X$. We train $D_\phi$ to discriminate between $Y$ and $\hat{Y}$ for input $X$. In the following sections, we will call $X$, $Y$, and $\hat{Y}$ as input sentence, target sentence, and generated sentence, respectively.

*Generator*: Generator is an encoder–decoder model based on transformers. It consists of an encoder and a decoder that are both stacks of residual attention blocks. The transformer-based encoder–decoder models process the input sequence $X_{1:n}$ of variable length $n$ with residual attention blocks without performing a recurrent structure, which is their main advantage and innovation. Transformer-based encoder–decoders are extremely parallelizable since they don't depend on a recurrent structure, which makes them more computationally efficient on modern hardware compared to RNN-based encoder–decoder models. The transformer-based encoder encodes the input sequence $X_{1:n}$ to a sequence of hidden states and the transformer-based decoder models the conditional probability distribution of the $\hat{Y}$ sequence given the sequence of encoded hidden states from the encoder.

*Discriminator*: The architecture of discriminator is similar to the generator, a transformer-based encoder–decoder model that accepts $X$ as encoder inputs, and $Y$ (either $\hat{Y}$ or $Y$) as decoder input. Rather of computing a scalar as the ultimate discriminator score $D(X, \hat{Y})$, we employ a stepwise evaluation.[38] After reading the input sentence $X$ and a portion of the output sequence $\hat{Y}_{1:t}$, the discriminator creates a scalar $R_t$. The ultimate discriminator score for the entire created sentence is the sum of all the scalars $R_{1:T}$ throughout the length $T$ of the generated sequence.

$$D(X, \hat{Y}) = \frac{1}{T}\sum_{t=1}^{T} R_t. \tag{9}$$

*Training*

At each generation step, the discriminator is customized to automatically allocate scores measuring the quality of each subsequence. Stepwise evaluation has substantially lower computational costs than MCTS, and the discriminator estimates instantaneous rewards by leveraging the idea of Q-learning and calculating state-action values without conducting tree search.

$$Q(s_t, \hat{y}_t) = \underset{z \sim P_G(.|x, \hat{y}_{1:t})}{E}[D(x, \hat{y}_{1:t}, z)], \tag{10}$$

$$R_t = Q(s_t, \hat{y}_t). \tag{11}$$

The current generator creates word sequence $z$ with input $X$ and generated prefix $\hat{Y}_{1:t}$. Thus, the anticipated return value of all the responses with the same prefix $\hat{y}_{1:t}$ is the state-action value $Q(s_t, \hat{y}_t)$. $s_t = (X, y_{1:t-1})$ and $y_t$ are discrete tokens which are the inputs of the Q-function. A Kronecker delta function (or a sharp distribution) can be used for $P_G$ in which all probabilities are zero except for the chosen sample. By this stepwise method a step dependent value, $R_t$, is calculated for each generation step which we call it Q-stepwise reward.

We design an approach for estimating $R_t$ value for generator while training the discriminator. For predicting the expected value $V(s_t)$, we train a value network V that has the same structure as discriminator. The value network is trained to approximate the predicted $R_t$ for every previous states $s_t$. As a result, the discriminator $D_\phi$ receives a pair of sentences and generates a score for each step. $D_\phi$ acquires knowledge using the following function:

$$J(\phi) = -\log D_\phi(X, Y) - \log(1 - D_\phi(X, Y)). \tag{12}$$

We train G with a stepwise evaluation technique, the objective function $J(G_\theta)$ of $G_\theta$ is:

$$J(\theta) = \sum_{t=1}^{T} R_t \nabla \log P_G(y_t | x, y_{1:t-1}). \tag{13}$$

As the first step, we use real data to pre-train $G_\theta$ using the maximum likelihood. We also apply supervised learning to pre-train $D_\phi$ using pairs composed of real and created data. Then we begin several rounds of adversarial training. First, we use real samples to train $G_\theta$ using (13). We use $G_\theta$ to output a generated sample for each input sentence once we have updated the settings. As a result, $D_\phi$ is fed a well-balanced set of real and fake (created) pairs. Finally, we use (12) to train $D_\phi$.

## 4 | EXPERIMENTS AND RESULTS

In this section, we present our detailed experimental study. Our goal through experiments is to demonstrate the performance of our model compared to different strong comparative models. As we use open-domain multi-hop QA system and a paraphrase generation model for producing abstractive summaries, we do need to train and evaluate them carefully due to their direct effect on summarization quality. To this end, we used three different sets of datasets and conducted an ablation analysis to evaluate our model's two components and the full question-driven abstractive text summarizer. The ablation analysis demonstrates that each of the components can produce reliable results and can be independently used. We have evaluated our approach in three different stages:

- We evaluate the proposed open-domain multi-hop QA system performance in Section 4.1.
- We evaluate the paraphrase model performance in Section 4.2.
- The full text summarization model is evaluated in the Section 4.3.

At each section we described the relevant datasets, evaluation metrics and baseline methods for each stage.

**Hyperparameter settings**

We used Stanford CoreNLP[79] and settings provided in Reference 76 for document analysis and candidate answers selection in CAI module. We have utilized the Answer Sentence Natural Questions (ASNQ)[80] derived from the Google Natural Questions (NQ) dataset[81] for training the the CNN and multi-head attention based answer selector component. ASNQ dataset contains the question, candidate answer pairs with labels, in each pair if the candidate sentence contains answer the label is 1 otherwise the label is 0. For token embeddings, we utilized the pre-trained ALBERT basic model, which consists of 12 Transformer blocks with 12 self-attention heads and a hidden size of 768. There is no mathematical procedure for determining the hyperparameters' optimal values in order to acquire the best model performance.

The choice of parameters in machine learning models can have a profound impact on the performance of a given task. These parameters control various aspects of the model's behavior, and selecting the right values is crucial for achieving optimal results. The choice of parameters can affect task performance by identifying the best values for model architecture parameters such as the number of layers, hidden units, and activation functions which influence the model's capacity to capture complex patterns.[82] The learning rate determines how quickly a model adapts to the training data. A high learning rate may cause the model to overshoot optimal weights, making it converge slowly or even diverge. A low learning rate can lead to slow convergence or getting stuck in local minima. Thus, the learning rate must be tuned to balance between rapid learning and stable convergence and it could be efficiently done by utilizing a hyperparameter optimization algorithm.[82]

As a result, we employed tools for tuning the model hyperparameters automatically. We optimized the hyperparameters' value using the Ray Tune Python library* with Hyperband algorithm.[83] Hyperband is a variation of random search employing some explore-exploit theory to determine the optimal time allocation for each configuration. It is designed to efficiently search for optimal hyperparameters for deep learning models. The search spaces are {2, 3, 4, 5}, {10, 20, 30, 50, 100}, {1e−5, 1e−7, 2e−7, 1e−8}, {16, 32, 64} for filter size, number of filters, learning rate, and batch size respectively. The Hyperband algorithm trains and evaluates all $n$ possible configurations for the hyperparameters and it keeps the top-performing configurations. Hyperband's efficiency makes it a popular choice for hyperparameter optimization, especially in situations where computational resources are limited or expensive, such as deep learning experiments. Hyperband efficiently allocates resources by eliminating poor-performing configurations early in the process. This makes it faster than a traditional grid or random search. Hyperband adapts to the available resources and can be used with different budgets, making it suitable for various computing environments. It balances exploration (evaluating a variety of hyperparameter settings) with exploitation (allocating more resources to promising configurations) to find a good trade-off.

Number of filters, filter size, batch size, and learning rate were optimized for training the answer selector component and their optimal values are as follows: 100, {2, 3, 4}, 64, 1e−5. We limited the sequence length to 128 tokens for ALBERT. We updated the parameters using the Adam optimization technique.[84] We calculated the loss using the cross-entropy loss function. We employed early stopping on the loss value on the development set and the maximum number of epochs is set to 10. We have utilized the pre-trained BERT basic model for generating the sentence embedding for calculating the cosine similarity in the reasoning process. The input representation for our paraphrase model is the pre-trained wordpiece embeddings from ALBERT. For training the paraphrase model, we trained the model for 10 epochs by Q-stepwise evaluation method after pre-training the generator by MLE. The discriminator is pre-trained on the generated samples from the pre-trained generator and real data. We adopted Adam optimization algorithm

to pre-train the generator and train the discriminator. The optimal learning rates for $G_\theta$, $D_\phi$ are 2e−6, 5e−6 calculated by Hyperopt algorithm. Hyperopt determines the optimal batch size 32 and 64 for Quora (100 K, 150 K) and MSCOCO[85] datasets to feed our generator and discriminator, and we performed 20 rounds of adversarial training.

## 4.1 | Open-domain multi-hop QA

We used MultiRC dataset for evaluating the proposed open-domain multi-hop QA model. Multi-sentence reading comprehension (MultiRC) is a reading comprehension dataset administered via a multiple-choice QA task.[86] Each question is based on a paragraph that comprises the question's gold justification sentences.

We used $F1_m$, $F1_a$, and EM evaluation metrics introduced in Reference 86. Table 2 summarizes the experimental results for open-domain multi-hop QA and four baseline methods which are described below.

WAIR[87] utilizes the alignment IR approach[88] to retrieve justification sentences and a RoBERTa binary classifier for answer selection. The WAIR technique, in two iterations, reduces the weights of question terms that have already been addressed by previously retrieved sentences and increases the weights of reformulated question terms that have not yet been covered. The second iteration reranks the clusters of evidence sentences using a regression task, with each sentence cluster allocated an F1 score generated from the gold annotated evidence sentences.

AIR[89] discovers justification sentences by an unsupervised strategy based on GloVe embeddings and an alignment model. To choose answers, a RoBERTa binary classifier is utilized. The question and candidate answer text are used to initiate the query. AIR adjusts its query after each repetition to focus on the missing information in the current set of justifications. The alignment approach computes the cosine similarity between each token's word embeddings in the query and the provided text sentence, resulting in a matrix of cosine similarity scores.

ROCC[90] presented an unsupervised technique for maximizing the relevance of selected sentences, minimizing the overlap between selected facts, and maximizing both question and answer coverage. The relevance, coverage, and overlap scores of candidate justification sets are calculated. They used BERT as a binary classifier to choose answers.

**TABLE 2** $F1_m$, $F1_a$, and EM score for our method and open-domain multi-hop QA baseline methods on MultiRC dataset.

| | MultiRC dataset | | |
|---|---|---|---|
| **Model** | **$F1_m$** | **$F1_a$** | **EM** |
| WAIR | 79.5 | 76.5 | 35.4 |
| AIR | 79.0 | 76.4 | 36.3 |
| ROCC | 73.8 | 70.6 | 26.1 |
| Multee | 71.7 | 68.3 | - |
| CNN-Att-MhopQA (ours) | 82.2 | 79.8 | 40.3 |
| CNN-Att-MhopQA (with self-attention) | 80.7 | 77.9 | 37.6 |
| CNN-Att-MhopQA (without Attention) | 78.8 | 76.0 | 35.7 |
| CNN-Att-MhopQA (without CNN) | 77.3 | 75.5 | 34.5 |

Multee[91] presented models of entailment for multi-hop QA composing a relevance module and multi-layer aggregation module. Both modules make use of ESIM,[92] a recently developed sentence-level entailment model that has been trained on the SNLI and MultiNLI datasets.

We have reported the results for the baseline methods from their paper. It is evident that WAIR outperformed other baselines since it introduced several attention and embedding-based analyses. It demonstrates that by combining retrieval and reranking techniques, it is possible to acquire the compositional knowledge necessary for multi-hop reasoning. AIR is the second-best baseline and outperformed ROCC and Multee due to the iterative method used to reformulate queries and focus on words not covered by existing justifications. AIR is an unsupervised alignment technique that uses only GloVe embeddings to soft-align questions and answers with justification sentences. ROCC outperformed Multee because it is an unsupervised strategy that utilizes a BERT answer classifier and three scoring functions to rank candidate reason sets. In compared to Multee's entailment technique, the ranking functions improve ROCC performance by increasing the relevance of the selected sentences and decreasing lexical overlap between the selected facts. Our model (CNN-Att-MhopQA) outperformed all baselines since it investigates the the semantic correlations between the features extracted from the question and relevant sentences in document. The semantic correlations between features are obtained by applying CNN and multi-head attention to the combined representation of the question and candidate answer sentences. Also, the reasoning process based on lexical coverage and BERT embedding is a complement to answer selector module for selecting justification sentences. We have added the results of the proposed model with self-attention and compared the performance with multi-head attention. As it is shown, the multi-head attention variant performs better due to training stability in comparison to self-attention variant.

We also included an ablation study in Table 2, we systematically evaluate the performance of our proposed CNN-Att-MhopQA model under various configurations to gain insights into the contributions of its components. The default model "CNN-Att-MhopQA" outperformed other variations. In the CNN-Att-MhopQA (with self-attention) variant, we replaced the multi-head attention mechanism with single-head attention. The experiments show utilizing multi-head attention obtains better results and it depicts that including both CNN and attention layer improves the performance (when we compare it to other variants). In the CNN-Att-MhopQA (without CNN) variant, we remove the attention mechanism while keeping the CNN architecture intact. This variant is superior to the CNN-Att-MhopQA (without attention) and it highlights the impact of the attention layer in our model performance.

Figure 6 shows the CNN-Att-MhopQA performance on each question category for MultiRC dataset. The multi-head attention variant outperformed the self-attention variant on all question categories since multi-level attention information is acquired.

## 4.2 | Paraphrase generation model

At the second stage of our experiments, we implement and evaluate our paraphrase generation model (QTrans-GAN) independently to assess its capability for paraphrasing extractive summaries. We choose the two most widely used datasets, Quora[†] and MSCOCO [85] for paraphrase generation experiments.

- **Quora** dataset consists of over 400 K candidate question paraphrase pairs with manually annotated labels. Two questions are paraphrasing each other only when the question pair's label is 1.

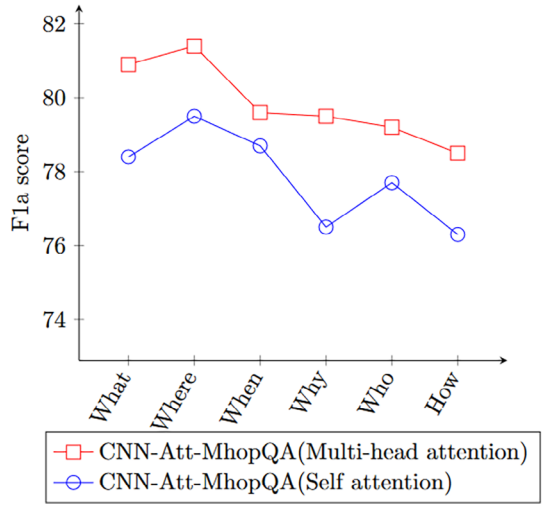


**FIGURE 6** CNN-Att-MhopQA multi-head attention and self-attention variants' performance on each question category for the MultiRC dataset.

We have used two different training sizes (100 K and 150 K) from Quora to have the same setting with baseline methods and show how the size of the dataset can affect the results of paraphrase generation.

- **MSCOCO** is a benchmark for the task of image captioning which contains over 82 K training and 42 K validation images, and at most five human-labeled caption are provided for each image. Similar to the previous works on paraphrase generation, different captions of the same image are considered as paraphrases.

We used some automatic metrics to evaluate QTrans-GAN framework and compare it with other methods.

- BLEU4 [93] is the most widely used evaluation metric in paraphrase generation. This approach works by counting matching n-grams in the generated sentence and the reference sentence.
- METEOR [94] metric is based on the harmonic mean of unigram precision and recall, with recall taking precedence over precision. Along with the basic precise word matching, it also offers other features which are not found in other measures, such as stemming and synonym matching. METEOR was designed to address some of the flaws in the BLEU metric while also producing a high level of correlation with human judgement at the segment or sentence level.

We employ four recent paraphrase generation approaches based on GANs as baseline methods, which are described in detail below.

EndtoEnd-GAN [72] regarded the generator (two stacked LSTMs encoder and decoder) as the stochastic policy and the output of discriminator (one LSTM) as its reward. In this way, they propagated the gradients from the discriminator to both the generator models and encoder models.

**TABLE 3** Experimental results of paraphrase generation on Quora (with 100 K and 150 K training set size) and MSCOCO datasets.

| | Quora-100K | | Quora-150K | | MSCOCO | |
|---|---|---|---|---|---|---|
| **Method** | **BLEU4** | **METEOR** | **BLEU4** | **METEOR** | **BLEU4** | **METEOR** |
| EndtoEnd-GAN[72] | 41.33 | 28.46 | 43.31 | 28.25 | 42.53 | 32.77 |
| Div-GAN[73] | - | - | 28.49 | - | 20.63 | - |
| Pen-GAN[74] | 29.07 | 31.27 | - | - | - | - |
| SE-GAN[38] | 41.96 | 30.36 | 43.62 | 31.04 | 42.70 | 32.89 |
| **QTrans-GAN (ours)** | **43.79** | **32.41** | **44.71** | **33.23** | **45.03** | **33.97** |

*Note*: The results for EndtoEnd-GAN, Div-GAN, and Pen-GAN are reported from their paper. Bold values indicate better performance in comparison to the baseline models.

Div-GAN [73] proposed a conditional GAN-based framework consisting a GRU-based generator and a CNN-based discriminator. They adopted the policy gradient and early feedback techniques described in Reference 36 for training.

Pen-GAN [74] utilized a convolutional seq2seq model for both generator and discriminator. They engage the discriminator output as penalization rather than using policy gradients, and they avoid the Monte-Carlo search by proposing a global discriminator.

SE-GAN [38] proposed the stepwise evaluation for chit-chat dialogue generation using GRU encoder decoder for both generator and discriminator and estimated state-action values for each generation step by modifying the architecture of the discriminator. We have applied this approach for paraphrase generation as a baseline method.

Table 3 summarizes the experimental results for paraphrasing on Quora (with 2 training sizes, 100 K and 150 K) and MSCOCO datasets. We reported the results for EndtoEnd-GAN, Div-GAN, and Pen-GAN from their paper. SE-GAN outperformed on all datasets compared to other baseline methods due to employing stepwise evaluation. Div-GAN has the worst performance on Quora-150K and MSCOCO datasets because of using policy gradient. EndtoEnd-GAN and Pen-GAN are in the second and third places respectively regarding their BLEU scores; however, Pen-GAN has a better METEOR score on Quora-100K dataset. EndtoEnd-GAN outperformed Pen-GAN because of proposing a generator based on stacked LSTMs and applying stochastic policy. Our model improved the BLEU and METEOR scores compared to all these baseline methods because of using transformers and Q-stepwise rewarding jointly in the discriminator. In detail, stacks of residual attention blocks in transformer, not relying on a recurrent structure, and reward calculation based on Q-learning for each generation step are the reasons for better performance in QTrans-GAN.

## 4.3 | Question-driven abstractive text summarization

In the third stage, we evaluate our hybrid summarization framework (QParaSum) using Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metric,[95] it compares an automatically generated summary with a set of human-produced summaries. ROUGE-N measures unigrams, bigrams, trigrams, and higher-order n-grams overlap. ROUGE-L utilizes the longest common subsequence method (LCS) to determine the longest matching sequence of words. We

don't require a predefined n-gram length since it automatically contains the longest in-sequence common n-grams. We evaluate QParaSum model on three large-scale summarization datasets, WikiHow,[96] PubMedQA,[97] and MEDIQA dataset.[98]

- **WikiHow** is a dataset accumulated from the WikiHow community-based QA website for abstractive text summarization task. Each sample in WikiHow dataset consists of a lengthy article, a non-factoid question, and the associated summary as the answer to the question.
- **PubMedQA** is a biomedical QA dataset derived from PubMed2 abstracts. Each sample includes a question, an article, and an abstractive answer which summarizes the context corresponding to the question.
- **MEDIQA** is a dataset comprising 156 consumer-submitted health questions, corresponding articles to these questions, and expert-written summaries of the answers.

At the final phase of our experiments, we consider four recent question-driven text summarization and three query-based[‡] baseline methods for evaluating QParaSum model.

HSCM [28] presented an approach for extractive answer summarization consisting of three components. In the first and second components (word-level and sentence-level compare-aggregate), an attention operation is used to align the word-level and sentence-level information between the answer sentence and question. In the third component, question-aware sequential extractor, a RNN decoder is designed to label each sentence consecutively and construct the answer summary for the target question.

MSG [26] proposed multi-hop selective generator (MSG), a question-driven abstractive summarization approach that integrates multi-hop reasoning to identify the key content for assisting the answer generation. In addition to the multi-view pointer network, they introduced a multi-view coverage technique to overcome the duplication issue and generate informative and precise answers.

QPGN [29] presented a question-driven pointer-generator network that utilizes the correlation information between question-answer pairs to add substantial information when generating abstractive answer summaries. Their framework consists of four components: Bi-LSTM encoder, seq2seq model joint with question-aware attention, question-answer alignment with summary representations, question-driven pointer-generator network.

Trans [30] has studied the capability of three state-of-the-art transformers for question-driven text summarization: BART, T5, and PEGASUS in both zero-shot and few-shot learning settings for question-driven abstractive text summarization on MEDIQA dataset. T5 outperformed the others thus we consider it as a baseline method.

Div-qsum[33] introduced a typical encode-attend-decode model (based on LSTM) for query-based abstractive summarization, which first computes a vectorial representation for the document and the query, and then the decoder produces a contextual summary one word at a time.

PGRU-qsum[99] is a pointer-generator model based on GRU encoder–decoder with attention and a pointer mechanism, for generating query-based summaries.

SummerTime[100] is a comprehensive text summarizing toolkit that interfaces with libraries built for NLP researchers and provides simple-to-use APIs to users. For query-based summarization, the top-k query-relevant phrases are retrieved using TF-IDF and BM25.

Table 4 shows the experimental results for one extractive (HSCM), three abstractive question-driven summarization approaches (MSG, QPGN, Tran(T5)), and three query-based

**TABLE 4** Results on WikiHow, PubMedQA, and MEDIQA.

| | WikiHow | | | PubMedQA | | | MEDIQA | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | **R1** | **R2** | **RL** | **R1** | **R2** | **RL** | **R1** | **R2** | **RL** |
| HSCM[28] | 27.84 | 7.75 | 25.85 | 32.34 | 10.07 | 25.98 | - | - | - |
| QPGN[29] | 28.8 | 9.7 | 27.7 | 34.2 | 12.8 | 28.7 | - | - | - |
| MSG[26] | 30.5 | 10.5 | 29.3 | 37.2 | 14.8 | 30.2 | - | - | - |
| Trans (T5)[30] | - | - | - | - | - | - | 38.56 | 18.52 | 26.00 |
| PGRU-qsum | 19.82 | 6.41 | 17.96 | 24.58 | 8.13 | 17.67 | 25.32 | 8.98 | 18.42 |
| Div-qsum | 18.56 | 5.10 | 15.81 | 22.67 | 7.55 | 16.80 | 23.56 | 7.08 | 17.67 |
| SummerTime | 15.43 | 4.67 | 13.78 | 19.67 | 5.98 | 14.60 | 20.42 | 6.31 | 15.66 |
| **QParaSum-Extractive (ours)** | **31.71** | **11.23** | **30.09** | **38.89** | **14.81** | **30.76** | **40.94** | **20.11** | **27.46** |
| **QParaSum-Abstractive (ours)** | **33.69** | **12.05** | **31.79** | **41.00** | **16.42** | **32.93** | **44.32** | **23.02** | **29.81** |

*Note*: The results for HSCM, QPGN, MSG, and Trans(T5) are reported from their paper. Bold values indicate better performance in comparison to the baseline models.

summarization approaches (Div-qsum, PGRU-qsum, SummerTime) on WikiHow, PubMedQA, MEDIQA datasets. The results for HSCM, MSG, QPGN, Tran(T5) are reported from their papers, and the results for query-based baselines are generated by using their public code for our datasets. ROUGE-1 (R1), ROUGE-2 (R2), and ROUGE (RL) are considered to evaluate the quality of our extractive and abstractive summaries. We have included the evaluation for our question-driven extractive summary to assess the impact of the paraphrasing process for generating abstractive summaries. HSCM generated extractive answer summaries and as it is shown in Table 4 the R1, R2, and RL for our extractive summaries and other abstractive baseline methods are superior to HSCM. Employing the GloVe language model and relying on a recurrent structure decoder for generating extractive answer summaries caused this inefficiency and poor performance in HSCM. MSG achieves relatively better performance than QPGN because of incorporating multi-hop reasoning for abstractive summarization. MSG and QPGN have used the pre-trained Glove model [101] which is not a very efficient model because of the co-occurrence matrix of words that consumes a considerable amount of memory. Besides using an inefficient language model, having multi-stages of training is another problem of these baseline methods. In Trans, each language generation model (BART, T5, PEGASUS) is pre-trained with different strategies which are unclear whether these strategies are the optimal ones. The query-based baselines have poor performances compared to question-driven baselines since answer selection and justification are not considered in query-based summarization. PGRU-qsum outperformed Div-qsum and SummerTime because of utilizing attention and pointer mechanism. Div-qsum outperformed SummerTime due to using an attention mechanism for encoding documents and queries.

Favorably QParaSum model obtains the state-of-the-art results for all three datasets with the generated extractive and abstractive summaries. The results indicate that the generated extractive summary covers the essential information for satisfying answerability, understandability, and persuasiveness measures by finding the AS and its supporting sentences using the proposed multi-hop QA system. Also, the extractive stage prunes the text (the input) for abstractive stage by removing the irrelevant and redundant information regarding the question. It was shown that the idea of exploiting an appropriate paraphrasing model for transforming the extractive summaries to abstractive is feasible since for all the three datasets the abstractive summaries obtain

**TABLE 5** An example from MEDIQA dataset for extractive and abstractive summaries generated by our framework that are evaluated by the gold summary.

**Question**: I have an hernia I would love to take care of it ASAP I was wondering if you guys could help and tell me what should I do?

**Article**: Hiatal hernia (Treatment): The majority of patients who have a hiatal hernia will exhibit no signs or symptoms and will not require treatment. If you have persistent heartburn or acid reflux, you may require medication or surgery. If you suffer from heartburn or acid reflux, your doctor may prescribe the following medications: Antacids that act as a buffer for stomach acid. Anti-acid medications such as Mylanta, Rolaids, and Tums may give immediate relief. Certain antacids may have adverse effects such as diarrhea or kidney problems if used in excess. Medications that inhibit acid production. Cimetidine (Tagamet), famotidine (Pepcid), nizatidine (Axid), and ranitidine are all H-2 receptor antagonists (Zantac). Prescriptions are required for stronger versions. Anti-acid medications that aid in the healing of the esophagus. Proton pump inhibitors are more effective acid blockers than H-2 receptor antagonists, and they allow a longer time for injured esophageal tissue to repair. Lansoprazole (Prevacid 24HR) and omeprazole are two proton pump inhibitors available over-the-counter (Prilosec, Zegerid). Surgery is normally reserved for those who are unable to control their heartburn or acid reflux with medicines or who have problems such as significant inflammation or esophageal constriction. Surgery to repair a hiatal …

**Question-driven extractive summary**: (Answer sentence) Hiatal hernia (treatment): The majority of patients who have a hiatal hernia will exhibit no signs or symptoms and will not require treatment. (IS1) If you have persistent heartburn or acid reflux, you may require medication or surgery. (IS2) If you suffer from heartburn or acid reflux, your doctor may prescribe the following medications:- Antacids that act as a buffer for stomach acid. (IS3) Surgery is normally reserved for those who are unable to control their heartburn or acid reflux with medicines or who have problems such as significant inflammation or esophageal constriction.

**Question-driven abstractive summary**: Hiatal hernia (treatment): Most individuals with a hiatal hernia don't have any signs or symptoms and will not require treatment. If you have signs like repetitive acid reflux and heartburn, you may require medication or surgery. Your doctor may recommend Antiacids to neutralize stomach acid if you experience acid reflux and heartburn. Surgery is recommended if the medications do not help the individual to soothe acid reflux and heartburn, or have complexities like serious inflammation or narrowing of the esophagus.

**Gold summary**: If a hiatal hernia does not have any symptoms, it won't require treatment. If the hernia causes heartburn and acid reflux, your doctor may recommend antacids. If the medications do not help or hiatal hernia causes inflammation or narrowing of the esophagus, your doctor might recommend surgery.

**Extractive summary evaluation**:{R1:0.406, R2: 0.148, RL: 0.260} **Abstractive summary evaluation**:{R1: 0.589, R2: 0.314, RL: 0.434}

higher R1, R2, and RL. The paraphrase model makes our hybrid summarizer capable to generate high quality abstractive summaries that are more close to the human generated ones. In Table 5, we show a practical example of our framework outputs for an instance from MEDIQA dataset. At first stage, the AS is selected by the proposed answer selector module based on CNN and multi-head attention and then IS1 is detected by the proposed LCCS reasoning method as the most semantically relevant sentence to AS. IS2 and IS3 are selected as the next supporting sentences for IS1 and AS. Since the IS1 contains general and important information about the AS (it contains the symptoms, medication and surgery as the treatment for hernia), we need to find sentences explaining and extending this information. The extractive summary is constructed by concatenating AS, IS1, IS2, and IS3 with the same order that they have in the article. At the second stage, the trained paraphrase model (the generator) is used for rewriting the sentences in the extractive

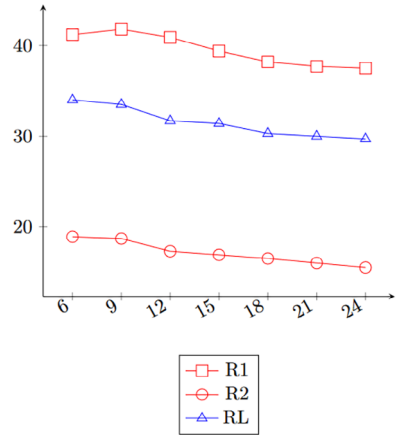


**FIGURE 7** QParaSum-Abstractive performance (average R1, R2, RL across all datasets) with different question lengths.

summary to improve it and make it more similar to the human generated summary. After generating the extractive summary, it is evident that we have tried to simulate the human action for text summarization, regenerating and rewriting the sentences regarding to their understanding from text, using a paraphrase model on the extractive summary. To evaluate the generated abstractive summary and compare its quality to the extractive summary, we have used the human generated summary (gold summary) and R1, R2, RL metrics to demonstrate whether our abstractive summary is similar to the gold summary. The generated abstractive summary obtained higher R1, R2, and RL, and it shows that our abstractive summary has more in common sequences of words with the gold summary. Figure 7 shows the average of R1, R2, and RL scores for QParaSum-Abstractive model across all datasets with different question lengths. It is evident that the model performance is not impacted by the length of the input question since an insignificant performance degradation (R1, R2, RL) is observed when the question length increases. However, for generating shorter abstractive summaries, merging the sentences could be considered during the paraphrase stage, which is the goal for our future work.

## 4.4 | Discussion

We have proposed a novel paraphrasing model for generating abstractive summaries. To begin with, the most relevant pieces of information are selected from the text regarding the target question for constructing the question-driven extractive summary. The trained paraphrase generation model is applied to the selected sentences to rewrite them and generate the abstractive summary. The results show that the generated abstractive summary is closer to the gold summary compared to the generated extractive summary. The reason for this quality improvement in the abstractive summary is empowering the GAN with transformers and Q-learning stepwise evaluation for paraphrase generation which is applied on the extractive summary generated by an open-domain multi-hop QA system. We have explored the previous question-driven text summarization approaches shortcomings and considered them while proposing our model. HSCM,[28] MSG,[26] and QPGN [29] have used the pre-trained Glove model which is not efficient because of the co-occurrence matrix of words that takes a lot of memory for storage. Besides, having multiple

steps of training is another problem in these approaches. In Trans [30] each language generation model (BART, T5, PEGASUS) is pre-trained with different strategies which are unclear whether these strategies are the optimal ones. We have designed two stages of training which are done once for open-domain question-driven text summarization while it could be even tuned for a specific domain. In our model, the extractive model is designed for producing extractive summaries with four sentences, while the average length of the gold summaries is four sentences, and calculating the optimal summary length for each instance is considered as one of our future works. In the paraphrase generation stage, we have proposed a sentence-level paraphrase model that processes the extractive summary sentences one by one and generates an abstractive summary which is more close to the gold summary as the results show in Section 4.3. Although considering the complete extractive summary is more desirable, proposing a framework capable of paraphrasing and merging some sentences at the same time will be considered in our future work. In other words, generating the correct link between the sentences and condensing them to improve the summary coherence is the complementary idea for this article which we will study in the future.

## 5 | CONCLUSION

In this article, we proposed a novel open-domain question-driven hybrid text summarization method. The question-driven text summarization poses unique challenges compared to generic or query-based summarization tasks. Ensuring that the generated summary is directly relevant to the user's question and covers all aspects of the question and relevant information from the source text are the main challenges that we attempted to overcome in this article. We designed an open-domain multi-hop QA system for question-driven extractive summarization and a paraphrase model to regenerate the extractive summaries and construct the abstractive ones. The idea of transforming extractive summary to abstractive summary is investigated in this research since abstractive summaries tend to be more coherent and fluent, making them easier for readers to grasp. Paraphrasing can help create summaries that flow smoothly and maintain a consistent narrative while enhancing readability. We showed that the proposed paraphrase model based on GANs and transformers with Q-learning stepwise evaluation can operate this transformation proficiently. To the best of our knowledge, this is the first work that employs a paraphrase generation model to generate abstractive summaries. We have evaluated our results on different benchmark datasets and compared our results to several baseline methods, and we can conclude that our hybrid framework is more effective for question-driven text summarization.

### DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in GitHub at https://github.com/mahnazkoupaee/WikiHow-Dataset.

### ENDNOTES

*https://docs.ray.io/en/latest/tune/index.html.

†https://www.quora.com/share/First-Quora-Dataset-Release-Question-Pairs.

‡We used their public code to apply their model to the datasets and generate question-driven summaries.

## REFERENCES


1. Kryściński W, Paulus R, Xiong C, Socher R. Improving abstraction in text summarization. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics; 2018:1808-1817.
2. El-Kassas WS, Salama CR, Rafea AA, Mohamed HK. Automatic text summarization: a comprehensive survey. *Expert Syst Appl*. 2021;165:113679.
3. Suleiman D, Awajan A. Deep learning based abstractive text summarization: approaches, datasets, evaluation measures, and challenges. *Math Probl Eng*. 2020;2020:1-29.
4. Li P, Lam W, Bing L, Wang Z. Deep recurrent generative decoder for abstractive text summarization. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics; 2017:2091-2100.
5. Kirmani M, Hakak NM, Mohd M, Mohd M. Hybrid text summarization: a survey. *Soft Computing: Theories and Applications*. Springer; 2019:63-73.
6. Aksenov D. *Abstractive Text Summarization with Neural Sequence-to-Sequence Models*. Master's Thesis. 2020.
7. Zhang J, Zhao Y, Saleh M, Liu P. Pegasus: pre-training with extracted gap-sentences for abstractive summarization. *International Conference on Machine Learning*. PMLR; 2020:11328-11339.
8. Wang L, Yao J, Tao Y, Zhong L, Liu W, Du Q. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. AAAI Press; 2018:4453-4460.
9. Song K, Wang B, Feng Z, Liu R, Liu F. Controlling the amount of verbatim copying in abstractive summarization. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2020:8902-8909.
10. Shi T, Keneshloo Y, Ramakrishnan N, Reddy CK. Neural abstractive text summarization with sequence-to-sequence models. *ACM Trans Data Sci*. 2021;2(1):1-37.
11. Kouris P, Alexandridis G, Stafylopatis A. Abstractive text summarization: enhancing sequence to sequence models using word sense disambiguation and semantic content generalization. *Comput Linguist*. 2021;47:813-859.
12. Cao Z, Li W, Wei F, Li S. Retrieve, rerank and rewrite: soft template based neural summarization. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2018:152-161.
13. Yang M, Qu Q, Tu W, Shen Y, Zhao Z, Chen X. Exploring human-like reading strategy for abstractive text summarization. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2019:7362-7369.
14. Turky SN, Al-Jumaili ASA, Hasoun RK. Deep learning based on different methods for text summary: a survey. *J Al-Qadisiyah Comput Sci Math*. 2021;13(1):26-35.
15. Fabbri AR, Li I, She T, Li S, Radev D. Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2019:1074-1084.
16. Nan F, Nallapati R, Wang Z, et al. Entity-level factual consistency of abstractive text summarization. *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics; 2021:2727-2733.
17. Sarker IH. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Comput Sci*. 2021;2(6):420.
18. Kanapala A, Pal S, Pamula R. Text summarization from legal documents: a survey. *Artif Intell Rev*. 2019;51(3):371-402.
19. Abdelaleem NM, Kader HA, Salem R. A brief survey on text summarization techniques. *IJ Electron Inf Eng*. 2019;10(2):103-116.
20. Kornilova A, Eidelman V. BillSum: a corpus for automatic summarization of US legislation. *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. Association for Computational Linguistics; 2019:48-56.
21. Kieuvongngam V, Tan B, Niu Y. Automatic text summarization of COVID-19 medical research articles using BERT and GPT-2. arXiv preprint arXiv:2006.01997, 2020.
22. Afzal M, Alam F, Malik KM, Malik GM. Clinical context–aware biomedical text summarization using deep neural network: model development and validation. *J Med Internet Res*. 2020;22(10):e19810.

23. Gharebagh SS, Goharian N, Filice R. Attend to medical ontologies: content selection for clinical abstractive summarization. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2020:1899-1905.
24. Abdi A, Shamsuddin SM, Aliguliyev RM. QMOS: query-based multi-documents opinion-oriented summarization. *Inf Process Manag*. 2018;54(2):318-338.
25. Rahul, Adhikari S, Monika. NLP based machine learning approaches for text summarization. *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE; 2020:535-538.
26. Deng Y, Zhang W, Lam W. Multi-hop inference for question-driven summarization. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics; 2020:6734-6744.
27. Song H, Ren Z, Liang S, Li P, Ma J, Rijke M. Summarizing answers in non-factoid community question-answering. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery; 2017:405-414.
28. Deng Y, Zhang W, Li Y, Yang M, Lam W, Shen Y. Bridging hierarchical and sequential context modeling for question-driven extractive answer summarization. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery; 2020:1693-1696.
29. Deng Y, Lam W, Xie Y, et al. Joint learning of answer selection and answer summary generation in community question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2020:7651-7658.
30. Goodwin TR, Savery ME, Demner-Fushman D. Flight of the PEGASUS? Comparing transformers on few-shot and zero-shot multi-document abstractive summarization. *Proceedings of COLING. International Conference on Computational Linguistics*. NIH Public Access; 2020:5640.
31. Afsharizadeh M, Ebrahimpour-Komleh H, Bagheri A. Query-oriented text summarization using sentence extraction technique. *2018 4th International Conference on Web Research (ICWR)*. IEEE; 2018:128-132.
32. Van Lierde H, Chow TW. Query-oriented text summarization based on hypergraph transversals. *Inf Process Manag*. 2019;56(4):1317-1338.
33. Nema P, Khapra MM, Laha A, Ravindran B. Diversity driven attention model for query-based abstractive summarization. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics; 2017:1063-1072.
34. Ishigaki T, Huang HH, Takamura H, Chen HH, Okumura M. Neural query-biased abstractive summarization using copying mechanism. *Advances in Information Retrieval*. Vol 12036. Springer; 2020:174.
35. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks. *Commun ACM*. 2020;63(11):139-144.
36. Yu L, Zhang W, Wang J, Yu Y. SeqGAN: sequence generative adversarial nets with policy gradient. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2017.
37. Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*. MIT Press; 2018.
38. Tuan YL, Lee HY. Improving conditional sequence generative adversarial networks by stepwise evaluation. *IEEE/ACM Trans Audio Speech Lang Process*. 2019;27(4):788-798.
39. Watkins CJ, Dayan P. Q-learning. *Mach Learn*. 1992;8(3-4):279-292.
40. Wu Q, Li L, Yu Z. TextGAIL: generative adversarial imitation learning for text generation. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2021:14067-14075.
41. Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners. *OpenAI Blog*. 2019;1(8):9.
42. Zhang C, Xiong C, Wang L. A research on generative adversarial networks applied to text generation. *2019 14th International Conference on Computer Science Education (ICCSE)*. IEEE; 2019:913-917.
43. Konda VR, Tsitsiklis JN. Actor-critic algorithms. Advances in Neural Information Processing Systems; 2000: 1008–1014.
44. Devlin J, Chang MW, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.
45. Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R. ALBERT: a lite BERT for self-supervised learning of language representations. International Conference on Learning Representations; 2019.

46. Clark K, Luong MT, Le QV, Manning CD. Electra: pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555, 2020.
47. Li W, Zhang X, Wu Y, Wei F, Zhou M. Document-based question answering improves query-focused multi-document summarization. *CCF International Conference on Natural Language Processing and Chinese Computing*. Springer; 2019:41-52.
48. Zhao M, Yan S, Liu B, et al. QBSUM: a large-scale query-based document summarization dataset from real-world applications. *Comput Speech Lang*. 2021;66:101166.
49. Erkan G, Radev DR. Lexrank: graph-based lexical centrality as salience in text summarization. *J Artif Intell Res*. 2004;22:457-479.
50. Mihalcea R, Tarau P. TextRank: bringing order into text. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics; 2004:404-411.
51. Tohalino JV, Amancio DR. Extractive multi-document summarization using multilayer networks. *Phys A Stat Mech Appl*. 2018;503:526-539.
52. Amancio DR, Nunes MG, Oliveira ON Jr, Costa LF. Extractive summarization using complex networks and syntactic dependency. *Phys A Stat Mech Appl*. 2012;391(4):1855-1864.
53. Cui P, Hu L, Liu Y. Enhancing extractive text summarization with topic-aware graph neural networks. *Proceedings of the 28th International Conference on Computational Linguistics*. NIH Public Access; 2020:5360-5371.
54. Srivastava A, Sutton C. Autoencoding variational inference for topic models. arXiv preprint arXiv:1703.01488, 2017.
55. Cao Z, Wei F, Li S, Li W, Zhou M, Wang H. Learning summary prior representation for extractive summarization. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics; 2015:829-833.
56. Cheng J, Lapata M. Neural summarization by extracting sentences and words. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics; 2016:484-494.
57. Liu L, Lu Y, Yang M, Qu Q, Zhu J, Li H. Generative adversarial network for abstractive text summarization. *Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press; 2018.
58. Scialom T, Dray PA, Lamprier S, Piwowarski B, Staiano J. Discriminative adversarial search for abstractive summarization. *International Conference on Machine Learning*. PMLR; 2020:8555-8564.
59. Dong L, Yang N, Wang W, et al. Unified language model pre-training for natural language understanding and generation. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc; 2019:13063-13075.
60. Rekabdar B, Mousas C, Gupta B. Generative adversarial network with policy gradient for text summarization. *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*. IEEE; 2019:204-207.
61. Dang N, Khanna A, Allugunti VR. TS-GAN with policy gradient for text summarization. *Data Analytics and Management*. Springer; 2021:843-851.
62. Wang S, Zhao X, Li B, Ge B, Tang D. Integrating extractive and abstractive models for long text summarization. *2017 IEEE International Congress on Big Data (BigData Congress)*. IEEE; 2017:305-312.
63. Bhat IK, Mohd M, Hashmy R. SumitUp: a hybrid single-document text summarizer. *Soft Computing: Theories and Applications*. Springer; 2018:619-634.
64. Subramanian S, Li R, Pilault J, Pal C. On extractive and abstractive neural document summarization with transformer language models. arXiv preprint arXiv:1909.03186, 2019.
65. Chen Y, Ma Y, Mao X, Li Q. Multi-task learning for abstractive and extractive summarization. *Data Sci Eng*. 2019;4(1):14-23.
66. Jin H, Wang T, Wan X. Multi-granularity interaction network for extractive and abstractive multi-document summarization. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2020:6244-6254.
67. Gupta A, Agarwal A, Singh P, Rai P. A deep generative framework for paraphrase generation. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2018.

68. Li Z, Jiang X, Shang L, Liu Q. Decomposable neural paraphrase generation. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2019:3403-3414.
69. Fu Y, Feng Y, Cunningham JP. Paraphrase generation with latent bag of words. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Vol 32. Curran Associates Inc; 2019:13645-13656.
70. Siddique A, Oymak S, Hristidis V. Unsupervised paraphrasing via deep reinforcement learning. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM; 2020:1800-1809.
71. Liu X, Mou L, Meng F, Zhou H, Zhou J, Song S. Unsupervised paraphrasing by simulated annealing. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2020:302-312.
72. Yang Q, Huo Z, Shen D, et al. An end-to-end generative architecture for paraphrase generation. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics; 2019:3132-3142.
73. Cao Y, Wan X. DivGAN: towards diverse paraphrase generation via diversified generative adversarial network. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*. Association for Computational Linguistics; 2020:2411-2421.
74. Vizcarra G, Ochoa-Luna J. Paraphrase generation via adversarial penalizations. *Proceedings of the Sixth Workshop on Noisy User-Generated Text (W-NUT 2020)*. Association for Computational Linguistics; 2020:249-259.
75. Mohammadi A, Ramezani R, Baraani A. A comprehensive survey on multi-hop machine reading comprehension approaches. arXiv preprint arXiv:2212.04072, 2022.
76. Kia MA, Garifullina A, Kern M, Chamberlain J, Jameel S. Adaptable closed-domain question answering using contextualized CNN-attention models and question expansion. *IEEE Access*. 2022;10:45080-45092. doi:10.1109/ACCESS.2022.3170466
77. Young T, Hazarika D, Poria S, Cambria E. Recent trends in deep learning based natural language processing. *IEEE Comput Intell Mag*. 2018;13(3):55-75.
78. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Curran Associates Inc; 2017:5998-6008.
79. Manning CD, Surdeanu M, Bauer J, Finkel JR, Bethard S, McClosky D. The Stanford CoreNLP natural language processing toolkit. *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. Association for Computational Linguistics; 2014:55-60.
80. Garg S, Vu T, Moschitti A. Tanda: transfer and adapt pre-trained transformer models for answer sentence selection. *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press; 2020:7780-7788.
81. Kwiatkowski T, Palomaki J, Redfield O, et al. Natural questions: a benchmark for question answering research. *Trans Assoc Comput Linguist*. 2019;7:453-466.
82. Yang L, Shami A. On hyperparameter optimization of machine learning algorithms: theory and practice. *Neurocomputing*. 2020;415:295-316. doi:10.1016/j.neucom.2020.07.061
83. Li L, Jamieson K, DeSalvo G, Rostamizadeh A, Talwalkar A. Hyperband: a novel bandit-based approach to hyperparameter optimization. *J Mach Learn Res*. 2017;18(1):6765-6816.
84. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
85. Lin TY, Maire M, Belongie S, et al. Microsoft COCO: common objects in context. *European Conference on Computer Vision*. Springer; 2014:740-755.
86. Khashabi D, Chaturvedi S, Roth M, Upadhyay S, Roth D. Looking beyond the surface: a challenge set for reading comprehension over multiple sentences. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics; 2018:252-262.
87. Yadav V, Bethard S, Surdeanu M. If you want to go far go together: unsupervised joint candidate evidence retrieval for multi-hop question answering. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics; 2021:4571-4581.

88. Yadav V, Bethard S, Surdeanu M. Alignment over heterogeneous embeddings for question answering. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics; 2019:2681-2691.
89. Yadav V, Bethard S, Surdeanu M. Unsupervised alignment-based iterative evidence retrieval for multi-hop question answering. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2020:4514-4525.
90. Yadav V, Bethard S, Surdeanu M. Quick and (not so) dirty: unsupervised selection of justification sentences for multi-hop question answering. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics; 2019:2578-2589.
91. Trivedi H, Kwon H, Khot T, Sabharwal A, Balasubramanian N. Repurposing entailment for multi-hop question answering tasks. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics; 2019:2948-2958.
92. Chen Q, Zhu X, Ling ZH, Wei S, Jiang H, Inkpen D. Enhanced LSTM for natural language inference. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics; 2017:1657-1668.
93. Papineni K, Roukos S, Ward T, Zhu WJ. BLEU: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics; 2002:311-318.
94. Lavie A, Agarwal A. METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*. Association for Computational Linguistics; 2007:228-231.
95. Lin CY. ROUGE: a package for automatic evaluation of summaries. *Text Summarization Branches Out*. Association for Computational Linguistics; 2004:74-81.
96. Koupaee M, Wang WY. Wikihow: a large scale text summarization dataset. arXiv preprint arXiv:1810.09305, 2018.
97. Jin Q, Dhingra B, Liu Z, Cohen W, Lu X. PubMedQA: a dataset for biomedical research question answering. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics; 2019:2567-2577.
98. Savery M, Abacha AB, Gayen S, Demner-Fushman D. Question-driven summarization of answers to consumer health questions. *Sci Data*. 2020;7(1):1-9.
99. Hasselqvist J, Helmertz N. *Query-Based Abstractive Summarization Using Neural Networks*. Master's Thesis. 2017.
100. Ni A, Azerbayev Z, Mutuma M, et al. SummerTime: text summarization toolkit for non-experts. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics; 2021:329-338.
101. Pennington J, Socher R, Manning CD. Glove: global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics; 2014:1532-1543.