AttenWalker: Unsupervised Long-Document Question Answering via Attention-based Graph Walking

Yuxiang Nie[123], **Heyan Huang**[123]*, **Wei Wei**[4], **Xian-Ling Mao**[123]
[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications
[3]Beijing Institute of Technology Southeast Academy of Information Technology
[4]Huazhong University of Science and Technology
{nieyx,hhy63,maoxl}@bit.edu.cn,weiw@hust.edu.cn

Abstract

Annotating long-document question answering (long-document QA) pairs is time-consuming and expensive. To alleviate the problem, it might be possible to generate long-document QA pairs via unsupervised question answering (UQA) methods. However, existing UQA tasks are based on short documents, and can hardly incorporate long-range information. To tackle the problem, we propose a new task, named unsupervised long-document question answering (ULQA), aiming to generate high-quality *long-document* QA instances in an *unsupervised* manner. Besides, we propose AttenWalker, a novel unsupervised method to aggregate and generate answers with long-range dependency so as to construct long-document QA pairs. Specifically, AttenWalker is composed of three modules, i.e., span collector, span linker and answer aggregator. Firstly, the span collector takes advantage of constituent parsing and reconstruction loss to select informative candidate spans for constructing answers. Secondly, by going through the attention graph of a pre-trained long-document model, potentially interrelated text spans (that might be far apart) could be linked together via an attention-walking algorithm. Thirdly, in the answer aggregator, linked spans are aggregated into the final answer via the mask-filling ability of a pre-trained model. Extensive experiments show that AttenWalker outperforms previous methods on Qasper and NarrativeQA. In addition, AttenWalker also shows strong performance in the few-shot learning setting.[1]

1 Introduction

Textual question answering (QA) is the task of answering questions given textual documents as the context. Previous works can be divided into short-document QA[2] methods (Seo et al., 2017) and long-document QA methods (Nie et al., 2022b). Short-document methods approach, and even outperform humans due to the availability of large-scale short-document QA datasets (Rajpurkar et al., 2016). Despite that, long-document methods still lag behind humans by a large margin since annotating long-document QA datasets (Dasigi et al., 2021) is time-consuming and costly.

Intuitively, the high cost of annotating long-document QA pairs can be alleviated in an unsupervised manner. However, there are only short-document unsupervised question answering (UQA) works (Lewis et al., 2019; Pan et al., 2021), which aim to construct a large number of short-document QA pairs in an unsupervised manner and train a QA

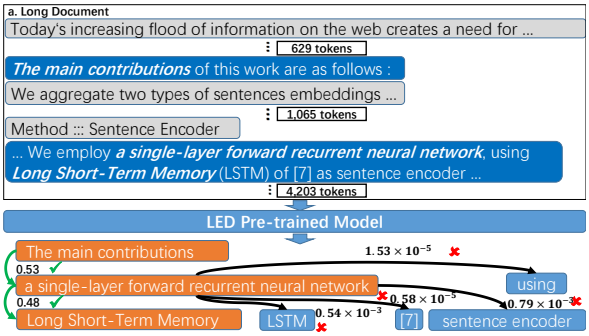

Figure 1: The long-range relation discovering process for a long document in Qasper dataset. The document is first fed into an LED pre-trained model (Beltagy et al., 2020) (the upper half). Then, the acquired token-level attention graph (not shown here) is converted into a span-level graph (the lower half) via the method described in Section 3.4. Spans (which might be far apart) are then linked if their edge weight is high. For example, the span "*The main contributions*" walks through 1,065 tokens and links with "*a single-layer forward recurrent neural network*", which is then linked with "Long Short-Term Memory" since their high weight edges (0.53 and 0.48). Other spans do not connect with them due to their low edge weights to these spans.

*Corresponding author

[1]We have released our codes and data in https://github.com/JerrryNie/Unsupervised-Long-Document-QA.

[2]Usually, the term 'short-document QA' is simplified as 'QA' in the literature, which refers to the QA task with a short context. We emphasize 'short-document' QA in this work to distinguish it with 'long-document' QA.

Findings of the Association for Computational Linguistics: ACL 2023, pages 13650–13663

model with these QA pairs. Lewis et al. (2019) first propose the UQA task and use unsupervised neural translation to construct QA pairs in a short passage. Pan et al. (2021) raise the unsupervised short-document multi-hop question answering (UMQA) task and design a question generation method to build multi-hop questions within two short passages. To break the document length limitation and incorporate long-range information, we propose a more challenging task, i.e. unsupervised long-document question answering (ULQA) task, to generate high-quality long-document QA pairs and train a competitive QA model without any human-labeled *long*-document QA pairs.

The core challenge of this task is in the modeling of long-range dependency without supervision. To address this issue, we study an *attention-driven* method to incorporate meaningful long-range information in the constructed QA pairs. Figure 1 illustrates a motivating example of the attention flow in a long document. It is observed that, by walking through the attention edges of a pre-trained model, related spans would be linked and long-range dependency in the document could be constructed. Therefore, long-range information could be also incorporated into QA pairs through these *walkable* attention patterns among text spans. Thus, we propose *AttenWalker*, a novel unsupervised framework to generate long-range dependent answers in long-document QA pairs. Specifically, AttenWalker comprises three modules: span collector, span linker and answer aggregator. Firstly, the span collector takes advantage of the constituent parsing and reconstruction ability of a pre-trained model to select informative candidate spans. Secondly, related spans that might be far apart could be connected through local or global attention edges of a long-document pre-trained model. Thirdly, collected spans are aggregated through the reconstruction ability of a pre-trained model.

Extensive experiments on Qasper (Dasigi et al., 2021) and NarrativeQA (Kociský et al., 2018) show that the proposed AttenWalker can effectively model long-range dependency in long-document QA. Besides, AttenWalker also shows strong performance in the few-shot learning setting.

Our contributions are as follows:

- To the best of our knowledge, we are the first to explore unsupervised *long*-document QA.
- Without the human-annotated long-range knowledge, we propose AttenWalker, a novel unsupervised long-document QA framework, which can incorporate long-range reasoning via attention-based graph walking.
- Extensive experiments show that AttenWalker outperforms previous methods in unsupervised and few-shot settings.

2 Related Works

Unsupervised Question Answering Unsupervised question answering (UQA) (Lewis et al., 2019) targets at alleviating the data scarcity problem in QA datasets. It focuses on generating QA pairs without supervision and training a QA model on them. Lewis et al. (2019) firstly propose the UQA task. Based on a pure short document, they extract answers via named entity tools and propose a novel cloze translation method to make alignment between cloze question and natural question so as to generate plenty of natural questions. Then, the constructed (context, question, answer) triples are used to train a QA model. Li et al. (2020) use cited documents to generate questions so that the overlapping problem between the generated question and the raw context could be alleviated. Nie et al. (2022a) propose to mine answers beyond named entities in the synthetic QA dataset and improve the model's ability in dealing with diverse answers. Pan et al. (2021) propose the first unsupervised multi-hop QA framework via multi-hop question generation. However, most of these methods focus on the short-document scenario, while the long-document setting is still unexplored.

Long-document Question Answering Long-document question answering (long-document QA) aims to answer questions based on the understanding of a long sequence of text. Previous methods can be divided into end-to-end methods and select-then-read methods. End-to-end methods (Dasigi et al., 2021) apply sparse attention models to directly answer the question given a long document. Dasigi et al. (2021) uses the Longformer-Encoder-Decoder model to make long-range reasoning on a long document and then answer a question. Caciularu et al. (2022) uses a sequence-level objective to improve evidence verification. For the select-then-read methods, Nie et al. (2022b) propose a compressive graph selector network to select question-related snippets from the long document and then use the selected short snippets for answer generation. However, despite competitive performances

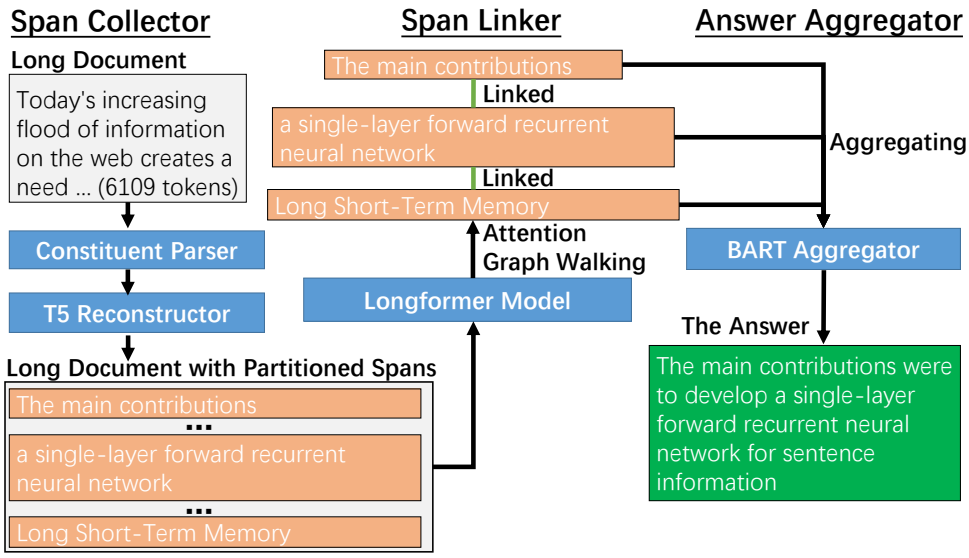

Figure 2: An overview of AttenWalker. It consists of three modules, including Span Collector, Span Linker, and Answer Aggregator.

on long-document QA, these methods heavily rely on supervised QA data and can hardly apply to the low-resource setting.

3 AttenWalker

In this section, we first formalize the task of long-document QA. After that, the proposed AttenWalker is described in detail.

3.1 Problem Formulation

The setup of long-document QA is as follows. Given a question q and a long document c, where c is often more than 10K tokens, the QA model $p_\theta(a|c,q)$ needs to produce a free-formed answer a by understanding the long document c and aggregating question-related snippets from c.

In this paper, we consider an unsupervised setting, where only long document c is available. Our aim is to generate synthetic QA pairs (q', a') with long-range information and train a competitive long-document QA model via (c, q', a') triples.

3.2 Overview of the Method

The proposed AttenWalker focuses on incorporating long-range information via a well-designed answer generator. Specifically, AttenWalker comprises three modules: Span collector, Span linker, and Answer Aggregator. As shown in Figure 2, the Span Collector first partitions the Long Document into different spans via Constituent Parsing and T5 Reconstructor. Secondly, a Span Linker is used to capture long-range dependency among these Partitioned Spans via Attention Graph Walking. This module aims to walk through local and global attention edges to link semantically related spans (which could be far apart in the long text) for aggregating answers. Thirdly, an Answer Aggregator combines all the Linked Spans via the reconstruction ability of a BART model to generate the answer.

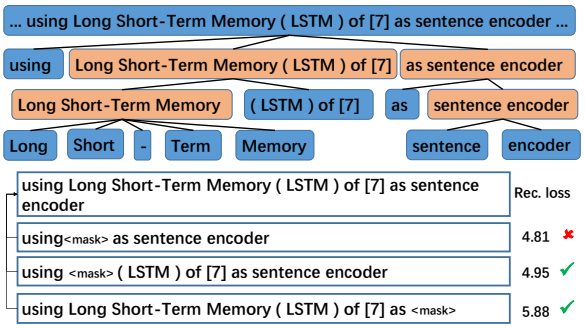

Figure 3: An illustration of the Span Collector of AttenWalker. To determine informative spans in the sentence "*using Long Short-Term Memory (LSTM) of [7] as sentence encoder*", the Constituent Parser first partitions the sentence into spans. These spans are then masked out and a pre-trained model (T5 Reconstructor) is used to make a reconstruction, where higher reconstruction loss could indicate a more informative span.

3.3 Span Collector

To determine the candidate spans for generating the answers, we propose a *Span Collector*. Specifically, as shown in Figure 3, it first seeks for candidate spans via constituent parsing and then reconstructs masked text via a pre-trained T5 model (Raffel

Figure 4: An attention heatmap from each token in span "*a single-layer forward recurrent neural network*" to each token in span "*Long Short-Term Memory*". Attention score values lower than 0.0001 are not displayed. The highest value 0.4762 is selected as the edge weight between these two spans.

et al., 2020) to select informative spans for answer generation. Each masked text serves as an input to the T5 model[3]. The reconstruction loss is:

$$\mathcal{L} = -\frac{1}{T}\sum_{i=1}^{T}\log(p(y_i)) \,, \tag{1}$$

where $\mathcal{L}$ is the reconstruction loss of the specific span. T is the number of tokens in the ground truth span and $p(y_i)$ is the T5 predicting probability of the i-th token y_i in the ground truth span. As shown in Figure 3, "*sentence encoder*" has the largest reconstruction loss. Thus, we select it as one of the candidate spans. Meanwhile, its parent spans (i.e. "*as sentence encoder*") and its child spans ("*sentence*" and "*encoder*") will not be selected for redundancy concern.

3.4 Span Linker

The proposed Span Linker is to incorporate long-range information in AttenWalker. It can effectively incorporate long-range dependency through attention-based graph walking. The Span Linker is composed of two sub-modules: a Span Graph Constructor and an Attention-based Graph Walker.

Span Graph Constructor To explore possible relations among spans, token-level attention scores[4] of the LED pre-trained model (Beltagy et al., 2020) can be used. As shown in Figure 1, based on the spans acquired in Section 3.3, we build a span graph $\mathcal{G}$ via attention scores between each pair of tokens as shown in Figure 4. For span i and span j, where $i, j \in \mathcal{G}$, if there are any attention edges from one of the tokens in span i to one of the tokens in span j, there is an edge from span i to span j. Motivated by the idea of max-pooling technique (Dumoulin and Visin, 2016), to obtain the most obvious relation in each pair of spans, the edge weight e_{ij} from span i to span j can be calculated by the maximum attention weight between any pair of tokens in between:

$$e_{ij} = \max_{m\in\mathcal{G}_i, n\in\mathcal{G}_j, (m,n)\in\mathcal{G}_t} w_{m,n} \,, \tag{2}$$

where $\mathcal{G}_i$ and $\mathcal{G}_j$ are tokens in span i and span j. (m, n) is an edge in token-level attention graph $\mathcal{G}_t$. $w_{m,n}$ is the attention weight of the edge (m, n).

In the LED encoder, there are local and global attention weights among the tokens in a long document. Both two types of weights can serve as the token-level edge weights $w_{m,n}$ in Eqn 2. In this work, we propose to consider both types for span graph construction. If there is a local attention weight $l_{m,n}$ from token m to token n, we directly assign the value to $w_{m,n}$. Otherwise, the global attention is considered: we insert a "</s>" at the beginning of each paragraph and set global attention for each of it (Appendix B). It means that each "</s>" can attend to every token in the long sequence and vice versa. Each "</s>" could serve as the representative of the paragraph that follows it. Therefore, "</s>" can be regarded as a bridge to two spans in different paragraphs, which could be far apart and could not be accessible to each other only through the local attention mechanism. To build the "bridge" from paragraph p_i to paragraph p_j, we first select one of the K tokens t_{p_i} with the maximum attention score to the representation of "</s>" s_{p_i}. Next, for the representation of s_{p_i}, L highest attention scores to other "</s>" tokens are selected. For one of the L "</s>" tokens s_{p_j} in paragraph p_j , we can access its maximum M attention weights to the corresponding M tokens (t_{p_j}) in paragraph p_j. For each t_{p_i}, its attention to the target token t_{p_j} can be:

$$g_{t_{p_i},t_{p_j}} = \sqrt[3]{w_{t_{p_i},s_{p_i}} \times w_{s_{p_i},s_{p_j}} \times, w_{s_{p_j},t_{p_j}}} \,, \tag{3}$$

where $g_{t_{p_i},t_{p_j}}$ is the global attention score from token t_{p_i} to token t_{p_j}. $w_{t_{p_i},s_{p_i}}, w_{s_{p_i},s_{p_j}}, w_{s_{p_j},t_{p_j}}$ are attention scores directly acquired according to the global attention in the LED model. Here,

[3]In practice, we use <extra_id_0> as the mask token. The <mask> token in Figure 3 is just for illustrative purpose.

[4]The token-level attention scores are acquired through the encoder part of the LED model. We consider each span graph for each Transformer layer and head.

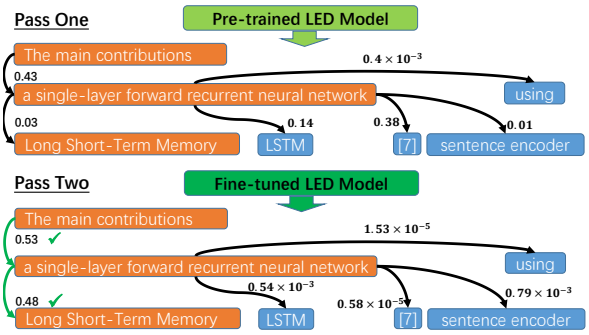

Figure 5: An example document (in Qasper training set) of edge weight changes from the first pass to the second pass.

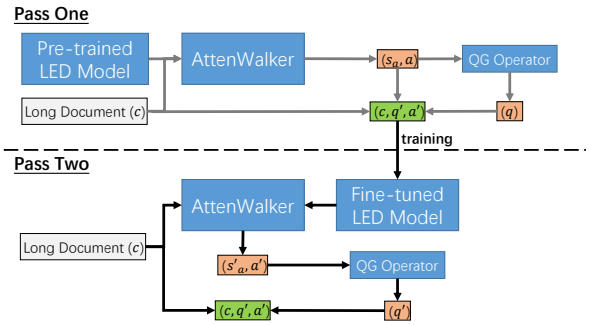

Figure 6: Overview of the proposed two-pass scheme. $s_a, s_a{}'$ are the sentences of the linked spans for a, a'

we use the geometric mean of the attention edge weights from t_{p_i} to t_{p_j} as the approximate attention weight of the edge (t_{p_i}, t_{p_j}). Thus, if there is no direct (local) attention from t_{p_i} to t_{p_j} but a global path, we can use $g_{t_{p_i},t_{p_j}}$ as the "lost" $w_{t_{p_i},t_{p_j}}$.

Attention-based Graph Walker Span linking can be done via attention-based graph walking on the constructed span graph. Essentially, the proposed graph walker collects interrelated spans via traversing the span graph. Its main algorithm is based on the Depth First Search (Even, 2011). As shown in the lower half of Figure 5, starting from the first span "*The main contributions*", graph walking continues searching for accessible span. Thus, it successfully links to the span "a single-layer forward recurrent neural network". Then, starting from this linked span, "Long Short-Term Memory" is also linked because of the high weight 0.48 between it and "*a single-layer forward recurrent neural network*". To decide whether the edge is of "high weight", we set a pre-defined threshold τ on the edge weight. In other words, the original span graph $\mathcal{G}$ can be pruned as a new graph $\mathcal{G}'$ via:

$$\mathcal{G}' = \{e | e \in \mathcal{G}, w_e > \tau\} \ , \tag{4}$$

where w_e is the weight of edge e. Finally, spans on the walking path are clustered together, which will be used in the following section.

3.5 Answer Aggregator

The proposed Answer Aggregator produces the final answer by aggregating the linked spans in Section 3.4. To achieve this goal, we take advantage of the reconstruction ability of a BART model (Lewis et al., 2020). For instance, the linked spans in the lower half of Figure 5 can be formalized into the input to BART: "`The main contributions <mask> a single-layer forward recurrent neural network <mask> sentence information`". Finally, the output can be an integral text as the answer: "`The main contributions were to develop a single-layer forward recurrent neural network for sentence information`".

3.6 Question Generation

Question generation (QG) is applied when we obtain the answer and all the sentences the linked spans from. We use the QG Operator in Unsupervised Multi-hop QA (Pan et al., 2021) as the QG module in our work. We concatenate the answer from Section 3.5 with all the aforementioned sentences into the QG module to generate a question.

3.7 Two-Pass Scheme for Long-Range Reasoning

In the pre-trained LED model, query, key, and value matrices of the global attention are just copied from the corresponding matrices in the local attention[5]. To further improve the ability of global attention in long-range reasoning, we design a two-pass scheme to construct long-document QA pairs as shown in Figure 6. In the first pass, only local attention is used in the proposed Span Graph Constructor. Then, an LED model is fine-tuned on these QA pairs with global and local attention as described in Appendix B. This step aims to improve the ability of the query, key, and value matrices, especially for global attention. In the second pass, based on the fine-tuned LED model, both local and global attention are considered to construct the span graph for attention walking. Hence, further knowledge with global attention is incorporated into the finally constructed QA pairs.

[5] https://github.com/allenai/longformer

Models	Qasper			NarrativeQA			
	Extractive	Abstractive	Overall	Bleu-1	Bleu-4	Meteor	Rouge-L
Supervised							
LED (Dasigi et al., 2021)	30.92	14.91	26.05	20.04	2.34	6.43	16.16
+ MQA-QG	28.98	13.87	24.42	20.88	**3.35**	6.99	17.38
+ AttenWalker	**32.44**	**15.41**	**27.08**	**21.15**	2.99	**7.03**	**18.07**
Human	58.92	39.71	52.80	44.43	19.65	24.14	57.02
Unsupervised							
UNMT (Lewis et al., 2019)	6.72	2.78	4.13	5.68	0.00	1.03	3.82
RefQA (Li et al., 2020)	3.08	0.63	2.26	0.95	0.00	1.02	0.96
DiverseQA (Nie et al., 2022a)	5.35	4.69	5.13	0.79	0.00	1.14	1.03
MQA-QG (Pan et al., 2021)	11.88	5.91	9.85	6.65	0.00	1.90	4.38
AttenWalker	**17.21**	**12.66**	**15.72**	**9.39**	**0.91**	**3.82**	**7.71**

Table 1: The performance on the test set of Qasper and NarrativeQA. In the second row, "Extractive, Abstractive, Overall" refer to Extractive F1, Abstractive F1 and Overall F1 in Qasper. In the "*Supervised*" block, the row "LED" denotes the performance of an LED model fine-tuned on the supervised dataset. "+MQA-QG" means that an LED model is first trained on the synthetic QA pairs from MQA-QG, and then continuously trained on supervised data. The meaning of "+AttenWalker" is similar. In the "*Unsupervised*" block, each unsupervised method generates long-document QA pairs and an LED model is fine-tuned on them without any supervised QA instances.

4 Experimental Setup

We evaluate the proposed AttenWalker on Qasper (Dasigi et al., 2021) and NarrativeQA (Kociský et al., 2018). In particular, for Qasper, the answer types in this dataset can be extractive, abstraction, yes/no, or unanswerable. Yet, according to our analysis (Appendix A), QA instances with yes/no or unanswerable answers cannot properly evaluate the ability of long document reasoning. Therefore, we only focus on the extractive and abstractive QA instances in this work. The datasets splitting and processing details are in appendix C.1.

We use the documents in the Qasper training set to construct QA pairs for training the QA model and do Qasper-related experiments. The long documents in the training set of NarrativeQA are used similarly. The dataset construction details can be found in Appendix C.2. What's more, the setting of the long document QA model trained on the constructed dataset can be referred to C.3.

5 Experiment

In this section, we first discuss the main results of AttenWalker on Qasper and NarrativeQA, and then further analyze the proposed method.

5.1 Main Results

Since there is no direct unsupervised method for long documents, we select competitive baselines from unsupervised short-document QA (UQA) and unsupervised short-document multi-hop QA (UMQA). The UQA works include UNMT (Lewis et al., 2019), RefQA (Li et al., 2020), DiverseQA (Nie et al., 2022a). The UMQA work is MQA-QG (Pan et al., 2021). The adaptation of them to long documents is described in Appendix E. Following Dasigi et al. (2021) and Kociský et al. (2018), we use answer F1 score (including extractive F1, abstractive F1 and overall F1 in this paper) as the evaluation metrics on Qasper dataset, while we use Bleu-1/4 (Papineni et al., 2002), Meteor (Denkowski and Lavie, 2011) and Rouge-L (Lin, 2004) for evaluation on NarrativeQA dataset.

As shown in Table 1, in the *Supervised* block, it can be found that an LED model trained on the synthetic dataset of AttenWalker can further make improvements when it is continuously fine-tuned on the supervised data, especially on Qasper, showing that the proposed method can effectively alleviate the data scarcity problem in Qasper. In the *Unsupervised* block, the proposed AttenWalker outperforms all baselines by a large margin in the fully unsupervised setting. showing a competitive performance of AttenWalker.

5.2 Ablation Study

We conduct an extensive ablation study on different components of AttenWalker. As shown in Table 2, the effectiveness of each component can be shown according to four observations.

Effects of the span collector. As shown in Table 2, the performance drop of "w/ Random Span Collector" illustrates that randomly selecting candidate spans could introduce much noise and harm the quality of the generated QA pairs.

Datasets	Qasper			NarrativeQA			
	Extractive	Abstractive	Overall	Bleu-1	Bleu-4	Meteor	Rouge-L
AttenWalker	12.13	**15.57**	**13.28**	**9.62**	**1.11**	**3.83**	**7.39**
w/ Random Span Collector	9.06	8.65	8.93	8.40	0.67	2.67	6.25
w/ Un-pre-trained LED	9.39	7.80	8.90	0.59	0.00	1.11	0.93
w/ Embedding Linker	11.69	9.04	10.87	6.33	0.24	2.87	5.60
w/o Global	11.36	10.75	11.16	7.23	0.38	3.10	6.06
w/ Answer Connector	9.48	10.00	9.66	6.66	0.00	3.13	5.99
w/ Single Pass	**12.52**	10.99	12.02	7.77	0.62	3.33	6.60
w/ Single Pass + Global	12.07	11.25	11.81	7.55	0.34	2.94	5.66

Table 2: Ablation study of AttenWalker, evaluating on the dev set of Qasper and NarrativeQA. "w/ Random Span Collector" denotes that candidate spans are randomly selected. "w/ Un-pre-trained LED" uses an LED model with randomly initialized parameters in the Span Linker. "w/ Embedding Linker" calculates attention scores only by the inner-product values between each pair of input embeddings. "w/o Global" does not consider the global attention in AttenWalker. "w/ Answer Connector" directly connects linked spans to form the answer. "w/ Singe Pass" only uses the pass-one in the proposed Two-Pass Scheme, while "w/ Single Pass + Global" further add global attention in it.

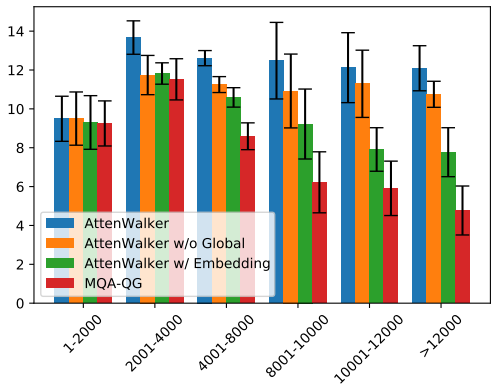

Figure 7: The mean and variance of Overall F1 (with 5 random seeds) for AttenWalker, two ablated versions "w/o Global""w/ Embedding" and MQA-QG. The dev set of Qasper is divided based on document length.

Effects of the span linker. From the performance drop in setting "w/ Un-pre-trained LED" and "w/ Embedding Linker" as shown in Table 2, it can be known that the attention information stored in the LED parameters is rather useful for constructing high-quality long-document QA pairs. Besides, the competitive result of "w/ Embedding Linker" suggests that embedding information can benefit the QA pair construction. In addition, the performance of "w/o Global" illustrates that global attention is also an essential factor in improving the quality of the generated long-document QA pairs.

Effects of the answer aggregator. According to "w/ Answer Connector" in Table 2, the performance drops when simply connecting spans. It shows that connecting spans with proper transition words is crucial for generating a high-quality answer.

Effects of the two-pass scheme. The Two-Pass Scheme is helpful in improving the performance of the model as shown in the "w/ Single Pass" and "w/ Single Pass + Global" setting from Table 2. It suggests that local and global attention can benefit from the parameters of a fine-tuned LED model.

5.3 Effects on Long-Range Modeling

AttenWalker aims to incorporate long-range information in the QA pair construction. To further understand it, an experiment with varied document lengths is conducted. As shown in Figure 7, in essence, "w/o Global" is only to use local attention while "w/ Embedding" denotes a situation that both global and local are not used. When the document length is small (1-2,000), the performances of different methods are comparable. However, with the increasing document length, the gap among methods becomes larger. It shows that AttenWalker can model long-range dependency effectively. Furthermore, it is observed that MQA-QG performs worse than "w/ Embedding" when the document length is large. It can be explained in two aspects. Firstly, MQA-QG could hardly capture long-range information. Secondly, MQA-QG is only a reduced version of "w/ Embedding", which can only link two spans via literal matching (Section 5.6).

5.4 Effects of Attention Weights

We design three different span graph construction strategies to further investigate their influences on the proposed method. As shown in Table 3, the "Max-Pooling" strategy outperforms the other two strategies by large margins. It can be explained that the "Max-Pooling" strategy can capture the most obvious (and probably important) relation between two spans, which is useful in QA pair construction.

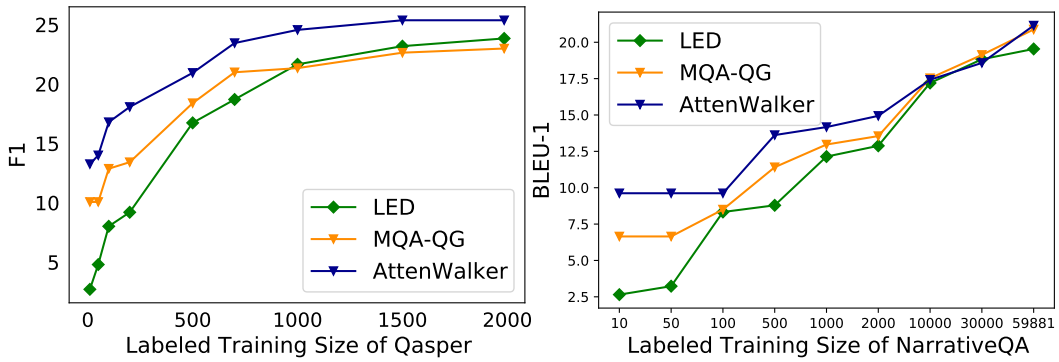

Figure 8: The few-shot learning of three methods on different sizes of labeled training data, evaluated on the dev set.

	Extractive	Abstractive	Overall
Max-Pooling*	**12.13**	**15.57**	**13.28**
Min-Pooling	6.79	5.78	6.47
Mean-Pooling	6.81	6.42	6.54

Table 3: Overall F1 of several methods with different strategies to build span graph, on the Qasper dev set. "Max-Pooling*" is used in AttenWalker, where the maximum attention score between tokens of two spans is selected as the edge weight. Similarly, "Min-Pooling" uses the minimum attention score, while "Mean-Pooling" uses the average of attention scores.

5.5 Few-Shot Learning

We conduct the few-shot learning experiment to explore the effectiveness of AttenWalker in different low-resource settings. As shown in Figure 8, with the increasing of the labeled training size, the performance of the model trained on the synthetic QA pairs from AttenWalker is consistently better than that of MQA-QG in Qasper and an LED model. It is because the Qasper dataset is quite small, which makes the synthetic dataset rather beneficial. Besides, in the NarrativeQA, AttenWalker reaches the best performance from 10 to 10,000 training sizes and then becomes comparable with MQA-QG. It can be explained that a large number of training sizes would narrow the gaps between them.

5.6 Case Study

In this section, we first analyze an example with the proposed two-pass scheme to explore the benefits of attention changes. Then, we compare two QA examples between AttenWalker and MQA-QG.

As shown in Figure 5, with an LED model, the spans "The main contributions" can be connected with "a single-layer forward recurrent neural network" and "[7]". Yet, after fine-tuning the model with generated QA instances, a more reasonable path "The main contributions" -> "a single-layer forward recurrent neural network" -> "Long Short-Term Memory" is strengthened and the link to the trivial span "[7]" is weakened. It can be explained that after fine-tuning, noise in the LED attention edges is reduced, further improving the span linking and the quality of the generated QA instances.

In addition, as shown in Table 4, we compare two QA pairs generated by AttenWalker and the best-performed baseline, MQA-QG. There are three key observations from the table. Firstly, AttenWalker can synthesize multiple spans into an answer whereas MQA-QG can only link the repeated text. Secondly, MQA-QG fails in long-range modeling since repeated spans could probably be in a short distance. Thirdly, the generated answer by AttenWalker is much more informative than MQA-QG's. In the long-document setting, answering a question might need synthesizing many pieces of information from different parts of the document. Therefore, the informativeness property of AttenWalker can be a better method for this setting.

6 Conclusion

We study a new task, named unsupervised long-document question answering, and propose AttenWalker, an unsupervised method to incorporate long-range information in QA pairs via graph walking. Extensive experiments show the strong performance of the proposed method. We believe that this work can be an important step in the long-document reasoning with a low-resource setting.

Limitations

Despite the strong performance of the proposed AttenWalker. There is still large room for improving efficiency. For example, the time cost of our method is still high. Since we need to search for all Transformer layers and heads to find potentially re-

AttenWalker
Related Context: ..... QG research traditionally considers ...(1,909 tokens)... most commonly considered factor by current NQG systems is the target answer ...(1,919 tokens)... the answer also deserves more attention from the model... **Generated Answer**: QG research shows the target answer deserves more attention **Generated Question**: What is the most commonly considered factor by current NQG systems?
MQA-QG
Related Context: ... They both follow the traditional decomposition of QG into content selection and question construction ...(8 tokens)... For content selection, [58] learn a sentence selection task to identify question-worthy sentences ... **Generated Answer**: content selection **Generated Question**: What is the task of identifying question-worthy parts in traditional the question that is the purpose of Question Generation synonymous with?

Table 4: Examples of the generated QA instances from AttenWalker and MQA-QG given the same long document. Blue texts are selected spans for answer generation.

lated spans, the dataset construction could be quite time-consuming. Therefore, an algorithm could be designed in the future to *pre-select* proper layers and heads for attention-based graph walking, which would save much time in dataset construction.

Acknowledgements

The work is supported by National Key R&D Plan (No. 2020AAA0106600), National Natural Science Foundation of China (No.62172039, U21B2009 and 62276110), in part by CCF-AFSG Research Fund under Grant No.RF20210005, and in part by the fund of Joint Laboratory of HUST and Pingan Property & Casualty Research (HPL). We thank the ACL reviewers for their helpful feedback. We would like to acknowledge Rong-Cheng Tu for the helpful discussions.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Avi Caciularu, Ido Dagan, Jacob Goldberger, and Arman Cohan. 2022. Long context question answering via supervised contrastive learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2872–2879, Seattle, United States. Association for Computational Linguistics.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4599–4610. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the sixth workshop on statistical machine translation*, pages 85–91.

Vincent Dumoulin and Francesco Visin. 2016. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*.

Shimon Even. 2011. *Graph algorithms*. Cambridge University Press.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Tomás Kociský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Trans. Assoc. Comput. Linguistics*, 6:317–328.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Patrick S. H. Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4896–4910. Association for Computational Linguistics.

Zhongli Li, Wenhui Wang, Li Dong, Furu Wei, and Ke Xu. 2020. Harvesting and refining question-answer pairs for unsupervised QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July*

5-10, 2020, pages 6719–6728. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yuxiang Nie, Heyan Huang, Zewen Chi, and Xian-Ling Mao. 2022a. Unsupervised question answering via answer diversifying. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 1732–1742. International Committee on Computational Linguistics.

Yuxiang Nie, Heyan Huang, Wei Wei, and Xian-Ling Mao. 2022b. Capturing global structural information in long document question answering with compressive graph selector network. *arXiv preprint arXiv:2210.05499*.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Unsupervised multi-hop question answering by question generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5866–5880. Association for Computational Linguistics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Appendix

A Analysis of Qasper Question Types

In this section, we analyze the contributions of the long document to different question types in the original Qasper dataset. As shown in Table 5, when the full text is absent from the input, the performance drops dramatically on the "Extractive" and "Abstractive" answer types. However, for "Yes/No" answers, the performance only drops a little, also keeping a competitive F1 score of 64.84. Besides, the performance of "Unanswerable" answers become unexpectedly better. Based on these observations, we argue that "Yes/No" and "Unanswerable" types are not suitable for testing the ability of long-range reasoning. Therefore, we only use "Extractive" and "Abstractive" in our experiments.

B Details in Fine-Tuning the LED Model

Similar to the input setting in Dasigi et al. (2021), for a long document, we prepend a special token `</s>` before each paragraph. And then we send the preprocessed long document into an LED model. For example, assume that there is a long document: $[t_{1,1}, t_{1,2}, ..., t_{p,1}, t_{p,2}, ..., t_{P,P_N-1}, t_{P,P_N}]$, where $t_{i,j}$ is the i-th token in paragraph j, P is the number of paragraphs, P_N is the number of tokens in paragraph P. After inserting the special token `</s>`, the input can be $[$`</s>`$, t_{1,1}, t_{1,2}, ...,$ `</s>`$, t_{p,1}, t_{p,2}, ..., t_{P,P_N-1}, t_{P,P_N}]$.

C Preprocessing Details of Qasper and NarrativeQA

C.1 Datasets

We evaluate the proposed AttenWalker framework on two long-document QA datasets[6]: Qasper (Dasigi et al., 2021) and NarrativeQA (Kociský et al., 2018). Qasper [7] is a dataset (license: CC BY 4.0) for answering questions based on long scientific papers. The questions are annotated based on the abstract of a scientific paper and the answer is annotated by understanding the entire paper's content. The answer types in this dataset can be extractive, abstraction, yes/no or unanswerable. Yet, according to our analysis (Appendix A), QA instances with yes/no or unanswerable answers cannot properly evaluate the ability of long document reasoning. Therefore, we only focus on the extractive and abstractive QA instances in this work. NarrativeQA (license: Apache-2.0) is a QA dataset established upon books and movie scripts of long text sequences. Given summaries of the books/scripts, annotators need to generate corresponding QA pairs where answers are free-formed. Table 6 shows the statistics of these two datasets. We use version 0.3 of Qasper dataset[8] for our experiment, where empty documents are removed. For NarrativeQA, we use the dataset[9] provided in Huggingface, which is a well-formed dataset. Thus, no extra cleaning step is needed.

C.2 Unsupervised Long-Document QA Dataset Construction

The datasets constructing process is shown in Figure 6. Specifically, we first extract sentence constituents from a long document using Berkeley Neural Parser (Kitaev et al., 2019). Then, a `t5-small` model is used in reconstruction-based span selection. In the span linker, we use `led-base-16384` to acquire the token-level attention graph for span linking. The threshold τ is set to 0.45. In the answer aggregator, we use the `bart-large` model to convert spans into an integral answer. Then, an operator [10] is used to generate questions. In the first pass, the generated dataset is used to train an `led-base-16384` model. In the second pass, the trained LED model is first used to provide the token-level attention graph as mentioned above. Besides, the global attention scores are also used to complete the attention graph (described in the paragraph "*Span Graph Constructor*"). The global-attention-related hyperparameters K, L, M. are all set as 3. The construction of the Qasper-document-based dataset costs 12 hours on 4 11GB GPUs while 15 hours on the NarrativeQA-document-based dataset.

C.3 Long-Document QA Model Setting

We use `led-base-16384` as the QA model throughout all of our experiments. The input format is described in Appendix B. We searched over batch sizes {2, 4, 8, 16, 32}, learning rates {3e-5, 5e-5, 8e-5, 1e-4}, warmup proportions {10%, 20%, 30%, 40%, 50%}, epochs {2, 4, 5, 6, 8, 10}. And the final batch size is 16, the learning rate is 5e-5, the

[6]The datasets used are originally created for research, which is consistent with our purpose.

[7]https://allenai.org/data/qasper

[8]https://allenai.org/data/qasper

[9]https://huggingface.co/datasets/narrativeqa

[10]https://github.com/teacherpeterpan/Unsupervised-Multi-hop-QA

Models	Extractive	Abstractive	Yes/No	Unanswerable	Overall
LED +Q +Full Text	32.49	13.40	68.90	39.22	34.23
LED +Q	3.45	4.05	64.84	78.95	22.75

Table 5: The performance of F1 scores on the dev set of Qasper. In the first row, "Extractive, Abstractive, Yes/No, Unanswerable" are four types of answers. "Overall" is the F1 score of all the answers. "LED+Q+Full Text" denotes training an LED model with a question and the long document as the input. "LED+Q" denotes a setting when the question but the long document is not provided for training the QA model.

	#Examples	Avg. #Tokens Input	Avg. #Tokens Output
Qasper			
Train	1985	5438.6	25.8
Dev	1393	4963.3	23.5
Test	2695	4864.7	23.3
NarrativeQA			
Train	59881	74420.1	6.0
Dev	3461	74749.7	6.0
Test	10557	68642.6	6.1

Table 6: Statistics of Qasper and NarrativeQA.

	Qasper	NarrativeQA
Overall	22,557	25,513
w/ Global Attention	5,505	1,370
Multi-Spans	10,754	8,361

Table 7: The statistics of QA pairs in the synthetic dataset constructed by AttenWalker.

warmup proportion is 30% and the epoch number is 5. We chunk the maximum input length into 13,000 tokens and set the attention window size to 640 so that the LED model in this configuration can be trained on four 11GB GPUs in 3 hours. Despite this relatively limited setting, we find that the performance of the LED model is comparable to the default configuration.

D Statistics of the Generated Datasets

In this section, we summarize the long-document QA datasets generated by AttenWalker. For saving time in QA pair generation, for each document, we randomly sample at most 32 linked span sets for QA-pair generation. The final generated results are shown in Table 7.

E Details in the Implementing of Baselines

Since current UQA methods cannot directly apply to the ULQA setting, we make further modifications and describe our implementation in detail.

UNMT (Lewis et al., 2019) To generate QA pairs with UNMT, each paragraph in the long document is used as a short context for QA generation. When training the LED model, the question generated by UNMT and the full long document is concatenated into a full sequence so as to train the model.

RefQA (Li et al., 2020) Similar to UNMT, each paragraph in the long document is separately used to generate QA pairs.

DiverseQA (Nie et al., 2022a) Similar to UNMT and RefQA, each paragraph is selected as a short document. And then, answers of diverse types are extracted from the document. Finally, each question is generated based on the answer and the short document.

MQA-QG (Pan et al., 2021) For MQA-QG, in a long document, two paragraphs are randomly sampled. These two paragraphs are then input into the MQA-QG for generating multi-hop QA pairs. Finally, the generated question is concatenated with the long document as the input to train the LED model.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section "Limitations".

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section "Abstract" and "1. Introduction".

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section "G.1 Datasets".

☑ B1. Did you cite the creators of artifacts you used?
Section "G.1 Datasets".

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section "G.1 Datasets".

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section "G.1 Datasets".

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section "G.1 Datasets".

C ☒ Did you run computational experiments?

Left blank.

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Not applicable. Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section "G.2 Unsupervised Long-Document QA Dataset Construction" and "G.3 Long-Document QA Model Setting".

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section "5.3 Effects on Long-Range Modeling", "A. Maximum Evidence Span Range Analysis" and "B. Multi-Hop Analysis"

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section "5.1 Main Results".

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.