Direct Fact Retrieval from Knowledge Graphs without Entity Linking

Jinheon Baek[1]* **Alham Fikri Aji**[2] **Jens Lehmann**[3] **Sung Ju Hwang**[1]
KAIST[1] MBZUAI[2] Amazon[3]
{jinheon.baek, sjhwang82}@kaist.ac.kr
alham.fikri@mbzuai.ac.ae jlehmnn@amazon.com

Abstract

There has been a surge of interest in utilizing Knowledge Graphs (KGs) for various natural language processing/understanding tasks. The conventional mechanism to retrieve facts in KGs usually involves three steps: entity span detection, entity disambiguation, and relation classification. However, this approach requires additional labels for training each of the three subcomponents in addition to pairs of input texts and facts, and also may accumulate errors propagated from failures in previous steps. To tackle these limitations, we propose a simple knowledge retrieval framework, which directly retrieves facts from the KGs given the input text based on their representational similarities, which we refer to as Direct Fact Retrieval (DiFaR). Specifically, we first embed all facts in KGs onto a dense embedding space by using a language model trained by only pairs of input texts and facts, and then provide the nearest facts in response to the input text. Since the fact, consisting of only two entities and one relation, has little context to encode, we propose to further refine ranks of top-k retrieved facts with a reranker that contextualizes the input text and the fact jointly. We validate our DiFaR framework on multiple fact retrieval tasks, showing that it significantly outperforms relevant baselines that use the three-step approach.

1 Introduction

Knowledge graphs (KGs) (Bollacker et al., 2008; Vrandecic and Krötzsch, 2014; Lehmann et al., 2015), which consist of a set of facts represented in the form of a (head entity, relation, tail entity) triplet, can store a large amount of world knowledge. In natural language applications, language models (LMs) (Devlin et al., 2019; Brown et al., 2020) are commonly used; however, their knowledge internalized in parameters is often incomplete, inaccurate, and outdated. Therefore, several recent works suggest augmenting LMs with facts from KGs, for example, in question answering (Oguz et al., 2022; Ma et al., 2022) and dialogue generation (Galetzka et al., 2021; Kang et al., 2022b).

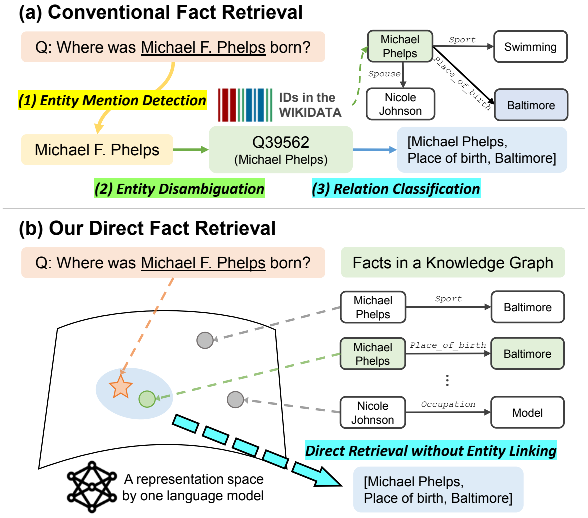
(a) Conventional Fact Retrieval
Q: Where was Michael F. Phelps born?
(1) Entity Mention Detection
IDs in the WIKIDATA
Michael F. Phelps
Q39562 (Michael Phelps)
[Michael Phelps, Place of birth, Baltimore]
(2) Entity Disambiguation
(3) Relation Classification
Michael Phelps
Sport
Swimming
Spouse
Nicole Johnson
Place_of_birth
Baltimore
(b) Our Direct Fact Retrieval
Q: Where was Michael F. Phelps born?
Facts in a Knowledge Graph
Michael Phelps
Sport
Baltimore
Michael Phelps
Place_of_birth
Baltimore
Nicole Johnson
Occupation
Model
Direct Retrieval without Entity Linking
A representation space by one language model
[Michael Phelps, Place of birth, Baltimore]

Figure 1: (a) A conventional fact retrieval from KGs involves three sequential steps: 1) entity mention detection to identify entities in queries; 2) entity disambiguation to match entities in input texts to KGs; 3) relation classification to select relevant relations. (b) Our fact retrieval directly retrieves relevant facts with their representational similarities to input queries.

However, despite the broad applications of the KGs, the existing mechanism for retrieving facts from them are, in many cases, unnecessarily complex. In particular, to retrieve facts from KGs, existing work (Fu et al., 2020; Lan et al., 2021; Wang et al., 2021) relies on three sequential steps, consisting of span detection, entity disambiguation, and relation classification, as illustrated in Figure 1a. For example, given an input text: "Where was Michael Phelps born?", they first detect a span of an entity within the input, which corresponds to "Michael Phelps". Then, they match the entity mention in the input to an entity id in the KG. Those two steps are often called entity linking. Finally, among 91 relations associated with the entity of Michael Phelps, they select one relation relevant to the input, namely "place of birth".

The aforementioned approach has a couple of

[*] Work done while interning at Amazon. Corresponding author: Jinheon Baek (jinheon.baek@kaist.ac.kr)

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 10038–10055
July 9-14, 2023 ©2023 Association for Computational Linguistics

drawbacks. First, all three sub-modules in the existing pipeline require module-specific labels in addition to query-triplet pairs for training. However, in real-world, high-quality training data is limited, and annotating them requires significant costs. Second, such a pipeline approach is prone to error propagation across steps (Singh et al., 2020; Han et al., 2020). For example, if the span detection fails, the subsequent steps, such as relation classification, are likely to make incorrect predictions as well. Third, certain modules, that match entities in queries to KGs or predict relations over KGs, are usually not generalizable to emerging entities and relations and cannot be applied to different KGs. It would be preferable to have a method that does not require KG-specific training and inference.

To tackle these limitations, we propose to directly retrieve the relevant triplets related to a natural language query by computing their similarities over a shared representation space (see Figure 1b). The design of our direct retrieval framework is motivated by a pioneering work of open-domain question answering with documents (Karpukhin et al., 2020), which showed the possibility of dense retrieval with simple vector similarities between the question and document embeddings. However, in contrast to the document retrieval scenario where documents have sufficient contexts to embed, it is unclear whether the LM can still effectively embed facts represented in the short triplet form for retrieval. Also, compared to the document retrieval which additionally requires a reader to extract only the relevant piece of knowledge, our fact retriever itself can directly provide the relevant knowledge.

To realize our fact retriever, we train it by maximizing similarities between representations of relevant pairs of input texts and triplets while minimizing irrelevant pairs, where we use LMs for encoding them. We note that this process requires only text-triplet pairs without using extra labels, unlike the conventional pipeline approach for fact retrieval. After training, we index all triplets in the KG with the trained encoder in an offline manner, and, given the input query, we return the nearest triplets over the embedding space. This procedure simplifies the conventional three steps for retrieving facts from KGs into one. To further efficiently search the relevant triplets, we approximate the similarity calculation with vector quantization and hierarchical search based on clustering (Johnson et al., 2021). We further note that, since we embed triplets using the LM, our retriever can generalize to different KGs without any modification, unlike some conventional retrieval systems that require additional training to learn new KG schema about distinct entities and relations types. We refer to our framework as **Di**rect **Fa**ct **R**etrieval (**DiFaR**).

We experimentally demonstrate that our direct retrieval on KGs works well; however, the fact represented in the triplet form has a limited context, since it consists of only two entities and one relation. Also, similarity calculation with the independently represented input text and triplets is arguably simple, and might be less effective. Therefore, to further improve the retriever performance, we additionally use a reranker, whose goal is to calibrate the ranks of retrieved triplets for the input text. In particular, we first retrieve k nearest facts with the direct retriever, and then use another LM which directly measures the similarity by encoding the input text and the triplet simultaneously. Moreover, another objective of the reranker is to filter out irrlevant triplets, which are the most confusing ones in the embedding space of the direct retriever. Therefore, to effectively filter them, we train the reranker to minimize similarities between the input text and the most nearest yet irrelevant triplets.

We evaluate our DiFaR framework on fact retrieval tasks across two different domains of question answering and dialogue, whose goals are to retrieve relevant triplets in response to the given query. The experimental results show that our DiFaR framework outperforms relevant baselines that use conventional pipeline approaches to retrieve facts on KGs, and also show that our reranking strategy significantly improves retrieval performances. The detailed analyses further support the efficacy of our DiFaR framework, with its great simplicity.

Our contributions in this work are as follows:

- We present a novel direct fact retrieval (DiFaR) framework from KGs, which leverages only the representational similarities between the query and triplets, simplifying the conventional three steps: entity detection, disambiguation, and relation classification, into one.
- We further propose a reranking strategy, to tackle a limitation of little context in facts, for direct knowledge retrieval, which is trained with samples confused by the direct retriever.
- We validate our DiFaR on fact retrieval tasks, showing that it significantly outperforms baselines on unsupervised and supervised setups.

2 Background and Related Work

Knowledge Graphs Knowledge Graphs (KGs) are factual knowledge sources (Bollacker et al., 2008; Vrandecic and Krötzsch, 2014), containing a large number of facts, represented in a symbolic triplet form: (head entity, relation, tail entity). Since some natural language applications require factual knowledge (Schneider et al., 2022), existing literature proposes to use knowledge in KGs, and sometimes along with language models (LMs) (Devlin et al., 2019). To mention a few, in question answering domains, facts in KGs can directly be answers for knowledge graph question answering tasks (Lukovnikov et al., 2017; Chakraborty et al., 2019), but also they are often augmented to LMs to generate knowledge-grounded answers (Zhang et al., 2019; Kang et al., 2022a). Similarly, in dialogue generation, some existing work augments LMs with facts from KGs (Galetzka et al., 2021; Kang et al., 2022b). However, prior to utilizing facts in KGs, fact retrieval – selection of facts relevant to the input context – should be done in advance, whose results substantially affect downstream performances. In this work, we propose a conceptually simple yet effective framework for fact retrieval, motivated by information retrieval.

Information Retrieval The goal of most information retrieval work is to retrieve relevant documents in response to a query (e.g., question). Early work relies on term-based matching algorithms, which count lexical overlaps between the query and documents, such as TF-IDF and BM25 (Robertson et al., 1994; Robertson and Zaragoza, 2009). However, they are vulnerable to a vocabulary mismatch problem, where semantically relevant documents are lexically different from queries (Nogueira et al., 2019; Jeong et al., 2021). Due to such the issue, recently proposed work instead uses LMs (Devlin et al., 2019; Liu et al., 2019) to encode queries and documents, and uses their representational similarities over a latent space (Karpukhin et al., 2020; Xiong et al., 2021; Qu et al., 2021). They suggest their huge successes are due to the effectiveness of LMs in embedding documents. However, they focus on lengthy documents having extensive context, and it is unclear whether LMs can still effectively represent each fact, succinctly represented with two entities and one relation in the triplet form, for its retrieval. In this work, we explore this new direction by formulating the fact retrieval problem as the information retrieval problem done for documents.

Knowledge Retrieval from KGs Since KGs have a large number of facts, it is important to bring only the relevant piece of knowledge given an input query. To do so, one traditional approach uses neural semantic parsing-based methods (Yih et al., 2015; Dong and Lapata, 2016; Bao et al., 2016; Luo et al., 2018) aiming to translate natural language inputs into logical query languages, such as SPARQL[1] and λ-DCS (Liang, 2013), executable over KGs. However, they have limitations in requiring additional labels and an understanding of logical forms of queries. Another approach is to use a pipeline (Bordes et al., 2014; Hao et al., 2017; Mohammed et al., 2018; Chen et al., 2019; Wang et al., 2021) consisting of three subtasks: entity span detection, entity disambiguation, and relation classification. However, they similarly require additional labels on training each subcomponent, and this pipeline approach suffers from errors that are propagated from previous steps (Singh et al., 2020; Han et al., 2020). While recent work (Oguz et al., 2022) proposes to retrieve textual triplets from KGs based on their representational similarities to the input text with the information retrieval mechanism, they still rely on entity linking (e.g., span detection and entity disambiguation) first, thus identically having limitations of the pipeline approach. Another recent work (Ma et al., 2022) merges a set of facts associated with each entity into a document and performs document-level retrieval. However, the document retrieval itself can be regarded as entity linking, and also the overall pipeline requires an additional reader to extract only the relevant entity in retrieved documents. In contrast to them, we directly retrieve facts from the input query based on their representational similarities, which simplifies the conventional three-step approach including entity linking into one single retrieval step.

3 DiFaR: Direct Fact Retrieval

3.1 Preliminaries

We formally define a KG and introduce a conventional mechanism for retrieving facts from the KG.

Knowledge Graphs Let $\mathcal{E}$ be a set of entities and $\mathcal{R}$ be a set of relations. Then, one particular fact is defined as a triplet: $t = (\mathtt{e_h}, \mathtt{r}, \mathtt{e_t}) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathtt{e_h}$ and $\mathtt{e_t}$ are head and tail entities, respectively, and $\mathtt{r}$ is a relation between them. Also, a knowledge graph (KG) $\mathcal{G}$ is defined as a set of fac-

[1] https://www.w3.org/TR/rdf-sparql-query/

tual triplets: $\mathcal{G} = \{(\mathtt{e_h}, \mathtt{r}, \mathtt{e_t})\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. Note that this KG is widely used as a useful knowledge source for many natural language applications, including question answering and dialogue generation (Oguz et al., 2022; Ma et al., 2022; Galetzka et al., 2021; Kang et al., 2022b). However, the conventional mechanism to access facts in KGs is largely complex, which may hinder its broad applications, which we describe in the next paragraph.

Existing Knowledge Graph Retrieval The input of most natural language tasks is represented as a sequence of tokens: $\boldsymbol{x} = [w_1, w_2, \ldots, w_{|\boldsymbol{x}|}]$. Suppose that, given the input $\boldsymbol{x}$, t^+ is a target triplet to retrieve[2]. Then, the objective of the conventional fact retrieval process for the KG $\mathcal{G}$ (Bordes et al., 2014; Wang et al., 2021) is, in many cases, formalized as the following three sequential tasks:

$$t^+ = \arg\max_{t \in \mathcal{G}} p_\theta(t|\mathtt{e}, \boldsymbol{x}, \mathcal{G}) p_\phi(\mathtt{e}|m, \boldsymbol{x}) p_\psi(m|\boldsymbol{x}), \quad (1)$$

where $p_\psi(m|\boldsymbol{x})$ is the model for mention detection with m as the detected entity mention within the input $\boldsymbol{x}$, $p_\phi(\mathtt{e}|m, \boldsymbol{x})$ is the model for entity disambiguation, and $p_\theta(t|\mathtt{e}, \boldsymbol{x}, \mathcal{G})$ is the model for relation classification, all of which are individually parameterized by ϕ, ψ, and θ, respectively.

However, there is a couple of limitations in such the three-step approaches. First, they are vulnerable to the accumulation of errors, since, for example, if the first two steps consisting of span detection and entity disambiguation are wrong and we are ending up with the incorrect entity irrelevant to the given query, we cannot find the relevant triplet in the final relation prediction stage. Second, due to their decomposed structures, three sub-modules are difficult to train in an end-to-end fashion, while requiring labels for training each sub-module. For example, to train $p_\psi(m|\boldsymbol{x})$ that aims to predict the mention boundary of the entity within the input text, they additionally require annotated pairs of the input text and its entity mentions: $\{(\boldsymbol{x}, m)\}$. Finally, certain modules are usually limited to predicting entities $\mathcal{E}$ and relations $\mathcal{R}$ specific to the particular KG schema, observed during training. Therefore, they are not directly applicable to unseen entities and relations, but also to different KGs.

3.2 Direct Knowledge Graph Retrieval

To tackle the aforementioned challenges of the existing fact retrieval approaches on KGs, we present the direct knowledge retrieval framework. In particular, our objective is simply formulated with the single sentence encoder model E_θ without introducing extra variables (e.g., m and $\mathtt{e}$), as follows:

$$t^+ = \arg\max_{t \in \mathcal{G}} f(E_\theta(\boldsymbol{x}), E_\theta(t)), \quad (2)$$

where f is a non-parametric scoring function that calculates the similarity between the input text representation $E_\theta(\boldsymbol{x})$ and the triplet representation $E_\theta(t)$, for example, by using the dot product. Note that, in Equation 2, we use the sentence encoder E_θ to represent the triplet t. To do so, we first symbolize the triplet as a sequence of tokens: $t = [w_1, w_2, \ldots, w_{|t|}]$, which is constructed by entity and relation tokens, and the separation token (i.e., a special token, [SEP]) between them. Then, we simply forward the triplet tokens to E_θ to obtain the triplet representation. While we use the single model for encoding both input queries and triplets, we might alternatively represent them with different encoders, which we leave as future work.

Training After formalizing the goal of our direct knowledge retrieval framework in Equation 2, the next step is to construct the training samples and the optimization objective to train the model (i.e., E_θ). According to Equation 2, the goal of our model is to minimize distances between the input text and its relevant triplets over an embedding space, while minimizing distances of irrelevant pairs. Therefore, following the existing dense retrieval work for documents (Karpukhin et al., 2020), we use a contrastive loss as our objective to generate an effective representation space, formalized as follows:

$$\min_\theta - \log \frac{\mathsf{exp}(f(E_\theta(\boldsymbol{x}), E_\theta(t^+)))}{\sum_{(\boldsymbol{x},t) \in \tau} \mathsf{exp}(f(E_\theta(\boldsymbol{x}), E_\theta(t)))}, \quad (3)$$

where τ contains a set of pairs between the input text and all triplets in the same batch. In other words, $(\boldsymbol{x}, t+) \in \tau$ is the positive pair to maximize the similarity, whereas, others are negative pairs to minimize. Also, $\mathsf{exp}(\cdot)$ is an exponential function.

Inference During the inference stage, given the input text $\boldsymbol{x}$, the model should return the relevant triplets, whose embeddings are closest to the input text embedding. Note that, since $E_\theta(\boldsymbol{x})$ and $E_\theta(t)$ in Equation 2 are decomposable, to efficiently do that, we represent and index all triplets in an offline manner. Note that, we use the FAISS library (Johnson et al., 2021) for triplet indexing and similarity

[2] For the sake of simplicity, we consider one triplet t^+ for each input; the retrieval target can be a set of triplets $\{t^+\}$.

calculation, since it provides the extremely efficient search logic, also known to be applicable to billions of dense vectors; therefore, suitable for our fact retrieval from KGs. Moreover, to further reduce the search cost, we use the approximated neighborhood search algorithm, namely Hierarchical Navigable Small World Search with Scalar Quantizer. This mechanism not only quantizes the dense vectors to reduce the memory footprint, but also builds the hierarchical graph structures to efficiently find the nearest neighborhoods with few explorations. We term our **Di**rect **Fa**ct **R**etrieval method as **DiFaR**.

3.3 Reranking for Accurate Fact Retrieval

The fact retrieval framework outlined in Section 3.2 simplifies the conventional three subtasks used to access the knowledge into the single retrieval step. However, contrary to the document retrieval case, the fact is represented with the most compact triplet form, which consists of only two entities and one relation. Therefore, it might be suboptimal to rely on the similarity, calculated by the independently represented input text and triplets as in Equation 2. Also, it is significantly important to find the correct triplet within the small k (e.g., $k = 1$) of the top-k retrieved triplets, since, considering the scenario of augmenting LMs with facts, forwarding several triplets to LMs yields huge computational costs.

To tackle such challenges, we propose to further calibrate the ranks of the retrieved triplets from our DiFaR framework. Specifically, we first obtain the k nearest facts in response to the input query over the embedding space, by using the direct retrieval mechanism defined in Section 3.2. Then, we use another LM, E_ϕ, that returns the similarity score of the pair of the input text and the retrieved triplet by encoding them simultaneously, unlike the fact retrieval in Equation 2. In other words, we first concatenate the token sequences of the input text and the triplet: $[\boldsymbol{x}, t]$, where $[\cdot]$ is the concatenation operation, and then forward it to $E_\phi([\boldsymbol{x}, t])$. By doing so, the reranking model E_ϕ can effectively consider token-level relationships between two inputs (i.e., input queries and triplets), which leads to accurate calibration of the ranks of retrieved triplets from DiFaR, especially for the top-k ranks with small k.

For training, similar to the objective of DiFaR defined in Section 3.2, we aim to maximize the similarities of positive pairs: $\{(\boldsymbol{x}, t^+)\}$, while minimizing the similarities of irrelevant pairs: $\{(\boldsymbol{x}, t)\} \setminus \{(\boldsymbol{x}, t^+)\}$. To do so, we use a binary cross-entropy loss. However, contrary to the previous negative sampling strategy defined in Section 3.2 where we randomly sample the negative pairs, in this reranker training, we additionally manipulate them by using the initial retrieval results from our DiFaR. The intuition here is that the irrelevant triplets, included in the k nearest neighbors to the input query, are the most confusing examples, which are yet not filtered by the DiFaR model. Hereat, the goal of the reranking strategy is to further filter them by refining the ranks of the k retrieved triplets; therefore, to achieve this goal, we include them as the negative samples during reranker training. Formally, let $\tilde{\tau} = \{(\boldsymbol{x}, \tilde{t})\}$ is a set of pairs of the input query $\boldsymbol{x}$ and its k nearest facts retrieved from DiFaR. Then, the negative samples for the reranker are defined by excluding the positive pairs, formalized as follows: $\tilde{\tau} \setminus \{(\boldsymbol{x}, t^+)\}$. Note that constructing the negative samples with retrieval at every training iteration is costly; therefore, we create them at intervals of several epochs (e.g., ten), but also we use only a subset of triplets in KGs during retrieval. Our framework with the reranking strategy is referred to as **Di**rect **Fa**ct **R**etrieval with **R**eranking (**DiFaR$^\mathbf{2}$**).

4 Experimental Setups

We explain datasets, models, metrics, and implementations. For additional details, see Appendix A.

4.1 Datasets

We validate our **Di**rect **Fa**ct **R**etrieval (**DiFaR**) on fact retrieval tasks, whose goal is to retrieve relevant triplets over KGs given the query. We use four datasets on question answering and dialogue tasks.

Question Answering The goal of KG-based question answering (QA) tasks is to predict factual triplets in response to the given question, where predicted triplets are direct answers. For this task, we use three datasets, namely SimpleQuestions (Bordes et al., 2015), WebQuestionsSP (WebQSP) (Berant et al., 2013; Yih et al., 2016), and Mintaka (Sen et al., 2022). Note that SimpleQuestions and WebQSP are designed with the Freebase KG (Bollacker et al., 2008), ad Mintaka is designed with the Wikidata KG (Vrandecic and Krötzsch, 2014).

Dialogue In addition to QA, we evaluate our DiFaR on KG-based dialogue generation, whose one subtask is to retrieve relevant triplets on the KG that provides factual knowledge to respond to a

Table 1: **Main results on the question answering domain** for SimpleQuestions, WebQSP, and Mintaka datasets. We emphasize the best scores in bold, except for the incomparable model: Retrieval with Gold Entities, which uses labeled entities in inputs.

		SimpleQuestions			**WebQSP**			**Mintaka**		
Types	**Methods**	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
Unsupervised	Retrieval with Gold Entities	0.7213	0.5991	0.9486	0.5324	0.4355	0.7402	0.1626	0.0978	0.2969
	Retrieval with spaCy	0.3454	0.2917	0.4437	0.3530	0.2856	0.4863	0.0914	0.0585	0.1622
	Retrieval with GENRE	0.1662	0.1350	0.2234	0.3099	0.2498	0.4363	0.0935	0.0640	0.1540
	Retrieval with BLINK	0.5142	0.4276	0.6766	0.4853	0.3938	0.6694	0.1350	0.0850	0.2430
	Retrieval with ReFinED	0.4841	0.4047	0.6283	0.5008	0.4055	0.6953	0.1312	0.0831	0.2325
	Factoid QA by Retrieval	0.7835	0.6953	0.9304	0.3933	0.3089	0.5470	0.1350	0.0836	0.2344
	DiFaR (Ours)	0.7070	0.5872	0.9259	0.5196	0.4130	0.7352	0.1590	0.0895	0.3043
	DiFaR^2 (Ours)	**0.8361**	**0.7629**	**0.9470**	**0.5441**	**0.4321**	**0.7602**	**0.2077**	**0.1348**	**0.3595**
Supervised	Retrieval with Gold Entities	0.8007	0.7094	0.9477	0.6048	0.5079	0.7794	0.2705	0.1987	0.4070
	Retrieval with spaCy	0.3789	0.3380	0.4453	0.3963	0.3272	0.5162	0.1367	0.1019	0.2019
	Retrieval with GENRE	0.1921	0.1718	0.2255	0.3617	0.3014	0.4696	0.1346	0.1005	0.1964
	Retrieval with BLINK	0.5679	0.5008	0.6766	0.5483	0.4571	0.7052	0.2075	0.1530	0.3157
	Retrieval with ReFinED	0.5349	0.4765	0.6279	0.5707	0.4754	0.7377	0.2106	0.1562	0.3166
	Factoid QA by Retrieval	0.8590	0.8051	0.9293	0.5253	0.4546	0.6486	0.1548	0.1179	0.2179
	DiFaR (Ours)	0.7904	0.6986	0.9382	0.6102	0.5071	0.7927	0.3049	0.2138	0.4856
	DiFaR^2 (Ours)	**0.8992**	**0.8583**	**0.9576**	**0.7189**	**0.6528**	**0.8385**	**0.4189**	**0.3367**	**0.5847**

user's conversation query. We use the OpenDialKG data (Moon et al., 2019), designed with Freebase.

Knowledge Graphs Following Diefenbach et al. (2017) and Saffari et al. (2021), we use the Wikidata KG (Vrandecic and Krötzsch, 2014) for our experiments on QA, and use their dataset processing settings. For OpenDialKG, we use Freebase.

4.2 Baselines and Our Models

We compare our DiFaR framework against other relevant baselines that involve subtasks, such as entity detection, disambiguation, and relation prediction. Note that most existing fact retrieval work either uses labeled entities in queries, or uses additional labels for training subcomponents; therefore, they are not comparable to DiFAR that uses only pairs of input texts and relevant triplets. For evaluations, we include models categorized as follows:

Retrieval with Entity Linking: It predicts relations over candidate triplets associated with identified entities by the entity linking methods, namely **spaCy** (Honnibal et al., 2020), **GENRE** (De Cao et al., 2021), **BLINK** (Wu et al., 2020; Li et al., 2020), and **ReFinED** (Ayoola et al., 2022) for Wikidata; **GrailQA** (Gu et al., 2021) for Freebase.

Factoid QA by Retrieval: It retrieves entities and relations independently based on their similarities with the input query (Lukovnikov et al., 2017).

Our Models: Our **Di**rect **K**nowledge **R**etrieval (**DiFaR**) directly retrieves the nearest triplets to the input text on the latent space. **DiFaR with Reranking (DiFaR^2)** is also ours, which includes a reranker to calibrate retrieved results.

Retrieval with Gold Entities: It uses labeled entities in inputs and retrieves triplets based on their associated triplets. It is incomparable to others.

4.3 Evaluation Metrics

We measure the retrieval performances of models with standard ranking metrics, which are calculated by ranks of correctly retrieved triplets. In particular, we use **Hits@K** which measures whether retrieved Top-K triplets include a correct answer or not, and Mean Reciprocal Rank (**MRR**) which measures the rank of the first correct triplet for each input text and then computes the average of reciprocal ranks of all results. Following exiting document retrieval work (Xiong et al., 2021; Jeong et al., 2022), we consider top-1000 retrieved triplets when calculating MRR, since considering ranks of all triplets in KGs are computationally prohibitive.

4.4 Implementation Details

We use a distilbert[3] as a retriever for all models, and a lightweight MiniLM model[4] as a reranker, both of which are pre-trained with the MSMARCO dataset (Nguyen et al., 2016). During reranking, we sample top-100 triplets retrieved from DiFaR. We use off-the-shelf models for unsupervised settings, and further train them for supervised settings.

5 Experimental Results and Analyses

Main Results We first conduct experiments on question answering domains, and report the results in Table 1. As shown in Table 1, our DiFaR with

[3] https://huggingface.co/sentence-transformers/msmarco-distilbert-base-v3

[4] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

Table 2: **Main results on the dialogue domain** for the OpenDialKG dataset. We emphasize the best scores in bold except for Retrieval with Gold Entities, which uses labeled entities.

		OpenDialKG		
Types	**Methods**	MRR	Hits@1	Hits@10
Unsupervised	Retrieval with Gold Entities	0.2511	0.1560	0.4683
	Retrieval with GrailQA	0.2051	0.1271	0.3745
	Factoid QA by Retrieval	0.1977	0.0892	0.4231
	DiFaR (Ours)	0.2396	0.1395	0.4424
	DiFaR^2 (Ours)	**0.2637**	**0.1603**	**0.4744**
Supervised	Retrieval with Gold Entities	0.2750	0.1495	0.5745
	Retrieval with GrailQA	0.2217	0.1198	0.4436
	Factoid QA by Retrieval	0.2042	0.1266	0.3587
	DiFaR (Ours)	0.2755	0.1405	0.5547
	DiFaR^2 (Ours)	**0.4784**	**0.3535**	**0.7380**

Table 3: **Zero-shot transfer learning results.** We use models trained on the WebQSP dataset with the Wikidata KG not only for SimpleQuestions and Mintaka datasets with the same KG, but also for the WebQSP dataset with the different Freebase KG. We use MRR as a metric, and N/A denotes not available.

	Wikidata		Freebase
Methods	SimpleQuestions	Mintaka	WebQSP
Retrieval with Gold Entities	0.7994	0.1950	0.6000
Retrieval with BLINK	0.5704	0.1617	N/A
Retrieval with ReFinED	0.5389	0.1591	N/A
Factoid QA by Retrieval	0.8014	0.1431	0.4239
DiFaR (Ours)	0.7812	0.2063	0.5913
DiFaR^2 (Ours)	**0.8244**	**0.2769**	**0.6324**

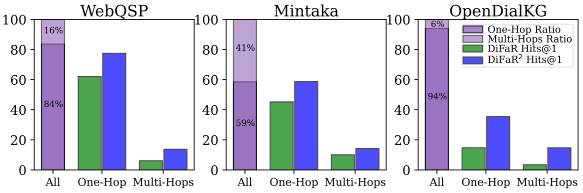

Figure 2: **Breakdown results by single and multi-hops.** We report ratios of single and multi-hops samples on the left side of each subfigure, and Hits@1 of DiFaR and DiFaR^2 across single and multi-hops on the middle and right. We exclude the SimpleQuestions dataset that consists of single-hop questions.

Reranking (DiFaR^2) framework significantly outperforms all baselines on all datasets across both unsupervised and supervised experimental settings with large margins. Also, we further experiment on dialogue domain, and report results in Table 2. As shown in Table 2, similar to the results on QA domains, our DiFaR^2 framework outperforms the relevant baselines substantially. These results on two different domains demonstrate that our DiFaR^2 framework is highly effective in fact retrieval tasks.

To see the performance gains from our reranking strategy, we compare the performances between our model variants: DiFaR and DiFaR^2. As shown in Table 1 and Table 2, compared to DiFaR, DiFaR^2 including the reranker brings huge performance improvements, especially on the challenging datasets: Mintaka and OpenDialKG. However, we consistently observe that our DiFaR itself can also show superior performances against all baselines except for the model of Factoid QA by Retrieval on the SimpleQuestions dataset. The inferior performance of our DiFaR on this SimpleQuestions dataset is because, its samples are automatically constructed from facts in KGs; therefore, it is extremely simple to extract entities and predict relations in response to the input query. On the other hand, our DiFaR framework sometimes outperforms the incomparable model: Retrieval with Gold Entities, which uses the labeled entities in the input queries. This is because this model is restricted to retrieve the facts that should be associated with entities in input queries; meanwhile, our DiFaR is not limited to query entities thanks to the direct retrieval scheme.

Analyses on Zero-Shot Generalization Our DiFaR can be generalizable to different datasets with the same KG, but also to ones with other KGs without any modifications. This is because it retrieves triplets based on their text-level similarities to input queries and does not leverage particular schema of entities and relations, unlike the existing entity linking methods. To demonstrate them, we perform experiments on zero-shot transfer learning, where we use the model, trained on the WebQSP dataset with the Wikidata KG, to different datasets with the same KG and also to ones with the different Freebase KG. As shown in Table 3, our DiFaR frameworks are effectively generalizable to different datasets and KGs; meanwhile, the pipeline approaches involving entity linking are not generalizable to different KGs, and inferior to ours.

Analyses on Single- and Multi-Hops To see whether our DiFaR frameworks can also perform challenging multi-hop retrieval that requires selecting triplets not directly associated with entities in input queries, we breakdown the performances by single- and multi-hop type queries. As shown in Figure 2, our DiFaR can directly retrieve relevant triplets regardless of whether they are associated with entities in input queries (single-hop) or not (multi-hop), since it does not rely on entities in queries for fact retrieval. Also, we observe that our reranking strategy brings huge performance gains, especially on multi-hop type queries. However, due to the intrinsic complexity of multi-hop retrieval, its performances are relatively lower than performances in single-hop cases. Therefore, despite the fact that the majority of queries are answerable with single-hop retrieval and that our DiFaR can handle multi-hop queries, it is valuable to further extend

Table 4: **Retrieval examples for complex questions**, on the challenging Mintaka dataset. We highlight the related phrases across the question and the triplet in yellow and green colors.

Question: What religion was the us president in 1963? **Retrieved Triplet**: (Robert F. Kennedy, religion, Catholicism) **Answer**: Catholicism
Question: Who commanded the allied invasion of western Europe at Normandy and was an American president? **Retrieved Triplet**: (Normandy landings, participant, Dwight D. Eisenhower) **Answer**: Dwight D. Eisenhower
Question: Which former Chicago Bull shooting guard was also selected to play on the 1992 US basketball team? **Retrieved Triplet**: (1992 US men's basketball team, has part, Michael Jordan) **Answer**: Michael Jordan

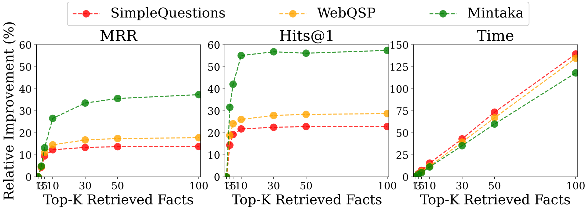

Figure 3: **Performances and efficiencies of DiFaR^2 with varying K**, where we change the number of Top-K retrieved triplets when leveraging the reranking strategy. We report results with the relative improvement (%) to our DiFaR without reranking. We report the time with average over 30 runs.

Table 5: **Sensitivity analyses on architectures**, where we change the backbones of retriever and reranker in our DiFaR^2. MSMARCO in the model name indicates it is pre-trained by the MSMARCO dataset, and we report results on the WebQSP.

Types	**Models**	MRR	Hits@1	Hits@10
Retriever	DistilBERT	0.5983	0.4963	0.7810
	MSMARCO-TAS-B	0.6051	0.4963	0.7844
	MSMARCO-Distil	0.6102	0.5071	0.7927
Reranker	MiniLM	0.6675	0.5945	0.7927
	MSMARCO-TinyBERT	0.7068	0.6420	0.8177
	MSMARCO-MiniLM	0.7189	0.6528	0.8385

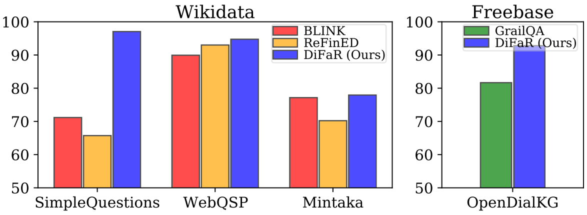

Figure 4: **Entity linking results,** where we measure the performances on benchmark datasets with Wikidata and Freebase KGs. Note that entity mentions of the SimpleQuestions dataset are not available; therefore, we cannot fine-tune existing entity linkers, which additionally require mention labels, unlike ours.

the model for multi-hop, which we leave as future work. We also provide examples of facts retrieved by our DiFaR framework in Table 4. As shown in Table 4, since LMs, that is used for encoding both the question and the triplets for retrieval, might learn background knowledge about them during pre-trainnig, our DiFaR framework can directly retrieve relevant triplets even for complex questions. For instance, in the first example of Table 4, the LM already knows who was the us president in 1963, and directly retrieves whose religion. Additionally, we provide more retrieval examples of our DiFaR framework in Appendix B.2 with Table 6 for both single- and multi-hop questions.

Analyses on Reranking with Varying K While we show huge performance improvements with our reranking strategy in Table 1 and Table 2, its performances and efficiencies depend on the number of retrieved Top-K triplets. Therefore, to further analyze it, we vary the number of K, and report the performances and efficiencies in Figure 3. As shown in Figure 3, the performances are rapidly increasing until Top-10 and saturated after it. Also, the time for reranking is linearly increasing when we increase the K values, and, in Top-10, the reranking mechanism takes only less than 20% time required for the initial retrieval. These results suggest that it might be beneficial to set the K value as around 10.

Sensitivity Analyses on Architectures To see different architectures of retrievers and rerankers make how many differences in performances, we perform sensitivity analyses by varying their backbones. We use available models in the huggingface model library[5]. As shown in Table 5, we observe that the pre-trained backbones by the MSMARCO dataset (Nguyen et al., 2016) show superior performances compared to using the naive backbones, namely DistilBERT and MiniLM, on both retrievers and rerankers. Also, performance differences between models with the same pre-trained dataset (e.g., MSMARCO-TAS-B and MSMARCO-Distil) are marginal. These two results suggest that the knowledge required for document retrieval is also beneficial to fact retrieval, and that DiFaR frameworks are robust across different backbones.

Analyses on Entity Linking While our DiFaR framework is not explicitly trained to predict entity mentions in the input query and their ids in the KG, during the training of our DiFaR, it might learn the knowledge on matching the input text to its entities. To demonstrate it, we measure entity linking performances by checking whether the retrieved triplets contain the labeled entities in the input query. As shown in Figure 4, our DiFaR surprisingly outperforms entity linking models. This might be because there are no accumulation of errors in entity linking steps, which are previously done with mention detection and entity disambiguation, thanks to direct retrieval with end-to-end learning; but also the fact in the triplet form has more beneficial information to retrieve contrary to the entity retrieval.

[5] https://huggingface.co/models

6 Conclusion

In this work, we focused on the limitations of the conventional fact retrieval pipeline, usually consisting of entity mention detection, entity disambiguation and relation classification, which not only requires additional labels for training each subcomponent but also is vulnerable to the error propagation across submodules. To this end, we proposed the extremely simple Direct Fact Retrieval (DiFaR) framework. During training, it requires only pairs of input texts and relevant triplets, while, in inference, it directly retrieves relevant triplets based on their representational similarities to the given query. Further, to calibrate the ranks of retrieved triplets, we proposed to use a reranker. We demonstrated that our DiFaR outperforms existing fact retrieval baselines despite its great simplicity, but also ours with the reranking strategy significantly improves the performances; for the first time, we revealed that fact retrieval can be easily yet effectively done. We believe our work paves new avenues for fact retrieval, which leads to various follow-up work.

Limitations

In this section, we faithfully discuss the current limitations and potential avenues for future research.

First of all, while one advantage of our Direct Fact Retrieval (DiFaR) is its simplicity, this model architecture is arguably simple and might be less effective in handling very complex queries (Sen et al., 2022). For example, as shown in Figure 2, even though our DiFaR framework can handle the input queries demanding multi-hop retrieval, the performances on such queries are far from perfect. Therefore, future work may improve DiFaR by including more advanced techniques, for example, further traversing over the KG based on the retrieved facts from our DiFaR. Also, while we use only the text-based similarities between queries and triplets with LMs, it is interesting to model triplets over KGs based on their graph structures and blend their representations with representations from LMs to generate more effective search space.

Also, we focus on retrieval datasets in English. Here we would like to note that, in fact retrieval, most datasets are annotated in English, and, based on this, most existing work evaluates model performances on English samples. However, handling samples in various languages is an important yet challenging problem, and, as future work, one may extend our DiFaR to multilingual settings.

Ethics Statement

For an input query, our Direct Fact Retrieval (DiFaR) framework enables the direct retrieval of the factual knowledge from knowledge graphs (KGs), simplifying the conventional pipeline approach consisting of entity detection, entity disambiguation, and relation classification. However, the performance of our DiFaR framework is still not perfect, and it may retrieve incorrect triplets in response to given queries. Therefore, for the high-risk domains, such as biomedicine, our DiFaR should be carefully used, and it might be required to analyze retrieved facts before making the critical decision.

Acknowledgements

We thank the members of the End-to-End Reasoning team of Alexa AI at Amazon and the anonymous reviewers for their constructive and insightful comments. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the previous and current funding agencies of the authors. The part of Jinheon Baek's graduate study and, accordingly, this work was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program (KAIST) and No.2021-0-02068, Artificial Intelligence Innovation Hub), and the Engineering Research Center Program through the National Research Foundation of Korea (NRF) funded by the Korea Government (MSIT) (NRF-2018R1A5A1059921).

References

Tom Ayoola, Shubhi Tyagi, Joseph Fisher, Christos Christodoulopoulos, and Andrea Pierleoni. 2022. Refined: An efficient zero-shot-capable approach to end-to-end entity linking. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track, NAACL 2022, Hybrid: Seattle, Washington, USA + Online, July 10-15, 2022*, pages 209–220. Association for Computational Linguistics.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2503–2514. ACL.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL.

Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*. ACM.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 615–620. ACL.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, Priyansh Trivedi, Jens Lehmann, and Asja Fischer. 2019. Introduction to neural network based approaches for question answering over knowledge graphs. *arXiv preprint arXiv:1907.09361*.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2019. Bidirectional attentive memory networks for question answering over knowledge bases. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2913–2923. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*. Association for Computational Linguistics.

Dennis Diefenbach, Thomas Pellissier Tanon, Kamal Deep Singh, and Pierre Maret. 2017. Question answering benchmarks for wikidata. In *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017*, CEUR Workshop Proceedings. CEUR-WS.org.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Bin Fu, Yunqi Qiu, Chengguang Tang, Yang Li, Haiyang Yu, and Jian Sun. 2020. A survey on complex question answering over knowledge base: Recent advances and challenges. *arXiv preprint arXiv:2007.13069*.

Fabian Galetzka, Jewgeni Rose, David Schlangen, and Jens Lehmann. 2021. Space efficient context encoding for non-task-oriented dialogue generation with graph attention transformer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 7028–7041. Association for Computational Linguistics.

Yu Gu, Sue Kase, Michelle Vanni, Brian M. Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond I.I.D.: three levels of generalization for question answering on knowledge bases. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3477–3488. ACM / IW3C2.

Namgi Han, Goran Topic, Hiroshi Noji, Hiroya Takamura, and Yusuke Miyao. 2020. An empirical analysis of existing systems and datasets toward general simple question answering. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 5321–5334. International Committee on Computational Linguistics.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 221–231. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. 2022. Augmenting document representations for dense retrieval with interpolation and perturbation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 442–452. Association for Computational Linguistics.

Soyeong Jeong, Jinheon Baek, ChaeHun Park, and Jong Park. 2021. Unsupervised document expansion for information retrieval with stochastic text generation. In *Proceedings of the Second Workshop on Scholarly Document Processing*, pages 7–17, Online. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Minki Kang, Jinheon Baek, and Sung Ju Hwang. 2022a. KALA: knowledge-augmented language model adaptation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5144–5167. Association for Computational Linguistics.

Minki Kang, Jin Myung Kwak, Jinheon Baek, and Sung Ju Hwang. 2022b. Knowledge-consistent dialogue generation with knowledge graphs. In *ICML 2022 Workshop on Knowledge Retrieval and Language Models*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4483–4491. ijcai.org.

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning dense representations of phrases at scale. In *ACL*, pages 6634–6647. Association for Computational Linguistics.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6433–6441. Association for Computational Linguistics.

Percy Liang. 2013. Lambda dependency-based compositional semantics. *arXiv preprint arXiv:1309.4408*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. 2017. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1211–1220. ACM.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Q. Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2185–2194. Association for Computational Linguistics.

Kaixin Ma, Hao Cheng, Xiaodong Liu, Eric Nyberg, and Jianfeng Gao. 2022. Open domain question answering with A unified knowledge interface. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1605–1620. Association for Computational Linguistics.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2018. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 291–296. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics, pages 845–854, Florence, Italy. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Rodrigo Frassetto Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2022. Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 1535–1546. Association for Computational Linguistics.

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

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. 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 5835–5847. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST).

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Amir Saffari, Armin Oliya, Priyanka Sen, and Tom Ayoola. 2021. End-to-end entity resolution and question answering using differentiable knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021 / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Phillip Schneider, Tim Schopf, Juraj Vladika, Mikhail Galkin, Elena Simperl, and Florian Matthes. 2022. A decade of knowledge graphs in natural language processing: A survey. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 601–614, Online only. Association for Computational Linguistics.

Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1604–1619, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Kuldeep Singh, Ioanna Lytra, Arun Sethupat Radhakrishna, Saeedeh Shekarpour, Maria-Esther Vidal, and Jens Lehmann. 2020. No one is perfect: Analysing the performance of question answering components over the dbpedia knowledge graph. *J. Web Semant.*, 65:100594.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. 2021. Retrieval, re-ranking and multi-task learning for knowledge-base question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 347–357. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6397–6407. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL*. The Association for Computer Linguistics.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *ACL*. The Association for Computer Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: enhanced language representation with informative entities. 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 1441–1451. Association for Computational Linguistics.

A Additional Experimental Setups

Here we provide additional experimental setups.

A.1 Datasets

Question Answering In KG-based question answering datasets, there exist pairs of questions and their relevant triplets, and we use them for training and evaluating models. We use the following three datasets: SimpleQuestions (Bordes et al., 2015), WebQuestionsSP (WebQSP) (Berant et al., 2013; Yih et al., 2016), and Mintaka (Sen et al., 2022), and here we describe them in details. First of all, the SimpleQuestions dataset is designed with the Freebase KG (Bollacker et al., 2008), which consists of 19,481, 2,821, and 5,622 samples on training, validation, and test sets. Similarly, the WebQSP dataset, which is a refined from the WebQuestions dataset by filtering out samples with invalid annotations, is annotated with the Freebase KG, consisting of 2,612 and 1,375 samples on training and test sets, and we further sample 20% of training samples for validation. Lastly, the Mintaka dataset is recently designed for complex question answering, which is collected from crowdsourcing and annotated with the Wikidata KG (Vrandecic and Krötzsch, 2014). Among eight different languages, we use questions in English, which consist of 14,000, 2,000, and 4,000 samples for training, validation, and test sets, respectively.

Dialogue Similar to the KG-based question answering datasets, the dataset on KG-based dialogue generation domain has pairs of the input query and its relevant triplets, where the input query consists of the user's utterance and dialogue history, and the annotated triplets are the useful knowledge source to answer the query. For this dialogue domain, we use the OpenDialKG dataset (Moon et al., 2019), which is collected with two parallel corpora of open-ended dialogues and a Freebase KG. We randomly split the dataset into training, validation, and test sets with ratios of 70%, 15%, and 15%, respectively, and preprocess it following Kang et al. (2022b), which results in 31,145, 6,722, and 6,711 samples on training, validation, and test sets.

Knowledge Graphs Following experimental setups of Diefenbach et al. (2017) and Saffari et al. (2021), we use the Wikidata KG (Vrandecic and Krötzsch, 2014) for our experiments on question answering, since the Freebase KG (Bollacker et al., 2008) is outdated, and the recently proposed entity linking models are implemented with the Wikidata, i.e., they are not suitable for the Freebase KG. Specifically, to use the Wikidata KG for datasets designed with the Freebase KG (e.g., SimpleQuestions and WebQSP), we use available mappings from the Freebase KG to the Wikidata KG (Diefenbach et al., 2017). Also, we use the wikidata dump of Mar. 07, 2022, and follow the dataset preprocessing setting from Saffari et al. (2021). For the OpenDialKG dataset, since it does not provide the Freebase entity ids, we cannot map them to the Wikidata entity ids using the available entity mappings. Therefore, for this dataset, we use original samples annotated with the Freebase KG.

A.2 Baselines and Our Model

In this subsection, we provide the detailed explanation of models that we use for baselines. Note that entity linking models are further coupled with the relation classification module to predict triplets based on identified entities in input queries. We begin with the explanations of entity linkers.

spaCy This model (Honnibal et al., 2020) sequentially predicts spans and KG ids of entities based on the named entity recognition and entity disambiguation modules. We use the spaCy v3.4[6].

GENRE This model (De Cao et al., 2021) first predicts the entity spans and then generates the unique entities in an autoregressive manner. Note that this model is trained for long texts; therefore, it may not be suitable for handling short queries.

BLINK This model (Wu et al., 2020) retrieves the entities based on their representational similarities with the input queries, and, before that, entity mentions in the input should be provided. We use a model further tuned for questions (Li et al., 2020).

ReFinED This model (Ayoola et al., 2022) performs the entity mention detection and the entity disambiguation in a single forward pass. We use a model further fine-tuned for questions.

GrailQA Unlike the above entity linkers that are trained for the Wikidata KG, this model (Gu et al., 2021) is trained to predict entities in the Freebase KG. This model performs the entity detection and the disambiguation sequentially, which is similar to the entity linking mechanism of spaCy.

[6] https://spacy.io/api/entitylinker

Factoid QA by Retrieval This model is a baseline (Lukovnikov et al., 2017) that individually retrieves the entities and relations based on their embedding-level similarities to input queries. Then, it merges the retrieved entities and relations with the KG-specific schema to construct the triplets.

DiFaR This is our fact retrieval framework that directly retrieves the facts on KGs based on their representational similarities to the input queries.

DiFaR^2 This is our fact retrieval framework with the proposed reranking strategy, where we further calibrate the retrieved results from DiFaR.

Retrieval with Gold Entities This is an incomparble model to others, which uses labeled entities in input queries to predict relations based on them.

A.3 Implementation Details

In this subsection, we provide additional implementation details that are not discussed in Section 4.4. In particular, we use the distilbert (Sanh et al., 2019)[7] as the retriever, and it consists of the 66M parameters. Also, for the reranker, we use the MiniLM model (Wang et al., 2020)[8], which consists of the 22M parameters. For supervised learning experiments, we train all models for 30 epochs, with a batch size of 512 for question answering and 32 for dialogue, and a learning rate of 2e-5. Also, we optimize all models using an AdamW optimizer (Loshchilov and Hutter, 2019). We implement all models based on the following deep learning libraries: PyTorch (Paszke et al., 2019), Transformers (Wolf et al., 2020), Sentence-Transformers (Reimers and Gurevych, 2019), and BEIR (Thakur et al., 2021). For computing resources, we train and run all models with four GeForce RTX 2080 Ti GPUs and with Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz having 72 processors. Also, training of our DiFaR framework takes less than one day. Note that we report all results with the single run, since our DiFaR framework significantly outperforms all baselines, but also it is costly to conduct multiple run experiments in the information retrieval experiment setting.

B Additional Experimental Results

Here we provide additional experimental results.

B.1 Running Time Efficiency

Note that, while we provide running time comparisons between our DiFaR and DiFaR^2 in Figure 3, it might be interesting to see more detailed running costs required for our dense fact retriever. As described in the Inference paragraph of Section 3.2, we index dense vectors with the Faiss library (Johnson et al., 2021) that supports vector quantization and clustering for highly efficient search. Specifically, following the common vector index setting in previous document retrieval work (Karpukhin et al., 2020; Lee et al., 2021), we use the HNSW index type. Please refer to the documentation of the Faiss library[9][10], if you want to further explore different index types and their benchmark performances.

We report running time efficiencies on the OpenDialKG dataset, which are measured on the server with Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz having 72 processors (See Section A.3). First of all, during inference, we can process about 174 queries per second where we return the top 1,000 facts for each query. Also, the average time for encoding and indexing one fact takes about 1 ms, which can be not only boosted further with more parallelization but also done in an online manner. Lastly, the performance drop of the approximation search with Faiss from the exact search is only 0.0098 on MRR.

B.2 Additional Retrieval Examples

In this subsection, on top of the retrieval examples provided in Table 4, we provide the additional examples of our DiFaR framework in Table 6.

[7] https://huggingface.co/sentence-transformers/msmarco-distilbert-base-v3

[8] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

[9] https://github.com/facebookresearch/faiss

[10] https://github.com/facebookresearch/faiss/wiki/Indexing-1M-vectors

Table 6: **Retrieval examples of our DiFaR^2** on the Mintaka dataset for both single- and multi-hop questions.

Index	**Question**	**Question Entities**	**Retrieved Fact**	**Answer Entity**
1	Which a series of unfortunate events books were not published in the 2000s?	A Series of Unfortunate Events	(A Series of Unfortunate Events, has part, The Bad Beginning)	The Bad Beginnin
2	Who was the last leader of the soviet union?	Soviet Union	(Soviet Union, head of state, Mikhail Gorbachev)	Mikhail Gorbachev
3	Who was the only u.s. vice president who is not male?	U.S. vice president	(Vice President of the United States, officeholder, Kamala Harris)	Kamala Harris
4	Which author has won the most national book awards for fiction?	National Book Awards for Fiction	(National Book Award for Fiction, winner, Saul Bellow)	Saul Bellow
5	Angkor wat can be found in which country?	Angkor Wat	(Angkor Wat, country, Cambodia)	Cambodia
6	Albany is the capital of what state?	Albany	(Albany, capital of, New York)	New York
7	Which u.s. president served the longest in office?	U.S.	(United States of America, head of government, Franklin Delano Roosevelt)	Franklin Delano Roosevelt
8	Which state has the four largest cities in the united states and also does not share any borders with any other u.s. states?	United States	(United States of America, contains administrative territorial entity, Alaska)	Alaska
9	What man was a famous american author and also a steamboat pilot on the mississippi river?	Mississippi River, American	(Life on the Mississippi, author, Mark Twain)	Mark Twain
10	What country participated in ww ii and also used nuclear weapons in combat?	WW II	(Allies of the Second World War, has part, United States of America)	United States of America

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Yes, see the Limitations section after the Conclusion section.

☑ A2. Did you discuss any potential risks of your work?
Yes, see the Ethics Statement section after the Conclusion section.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Yes, see the Abstract and Introduction sections.

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

B ☑ Did you use or create scientific artifacts?

Yes, we describe them in Section 4 and Appendix A.

☑ B1. Did you cite the creators of artifacts you used?
Yes, we cite them in Section 4 and Appendix A.

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No, but we instead follow the licenses and cite the original papers that released artifacts.

☒ 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)?
No, but we instead cite the original papers for artifacts, and follow their licenses.

□ 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.
Yes, we provide them in Section 4.1 and Appendix A.

C ☑ Did you run computational experiments?

Yes, see Section 5.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Yes, we report them in Section 4, and Appendix A.

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?
Yes, we describe them in Section 4 and Appendix A.

☑ 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?
Yes, we clearly provide them in Table 1 and Table 2, as well as in Appendix A.

☑ 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.)?
Yes, we report them in Section 4 and Appendix A.

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.