Differentiable Open-Ended Commonsense Reasoning

Bill Yuchen Lin[1]*, **Haitian Sun**[2], **Bhuwan Dhingra**[2], **Manzil Zaheer**[2], **Xiang Ren**[1], **William W. Cohen**[2]
[1] University of Southern California
[2] Google Research
{yuchen.lin,xiangren}@usc.edu
{haitiansun,bdhingra,manzilzaheer,wcohen}@google.com

Abstract

Current commonsense reasoning research mainly focuses on developing models that use commonsense knowledge to answer *multiple-choice* questions. However, systems designed to answer multiple-choice questions may not be useful in applications that do not provide a small list of possible candidate answers to choose from. As a step towards making commonsense reasoning research more realistic, we propose to study *open-ended commonsense reasoning* (OpenCSR) — the task of answering a commonsense question *without* any pre-defined choices, using as a resource only a corpus of commonsense facts written in natural language. The task is challenging due to a much larger decision space, and because many commonsense questions require multi-hop reasoning. We propose an efficient **d**ifferentiable model for multi-hop **r**easoning over knowledge **fact**s, named DrFact. We evaluate our approach on a collection of re-formatted, open-ended versions of popular tests targeting commonsense reasoning, and show that our approach outperforms strong baseline methods by a large margin.

1 Introduction

The conventional task setting for most current commonsense reasoning research is *multiple-choice* question answering (QA) — i.e., given a question and a small set of pre-defined answer choices, models are required to determine which of the candidate choices best answers the question. Existing commonsense reasoning models usually work by scoring a question-candidate pair (Lin et al., 2019; Lv et al., 2020; Feng et al., 2020). Hence, even an accurate multiple-choice QA model cannot be directly applied in practical applications where answer candidates are not provided (e.g., answering a question asked on a search engine, or during conversation with a chat-bot).

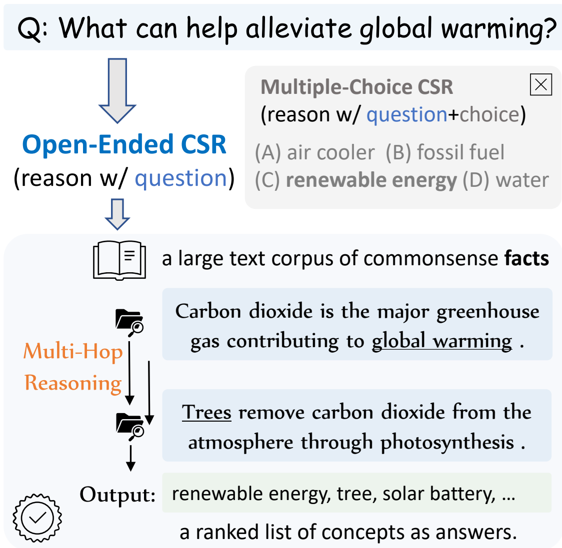

Figure 1: We study the task of open-ended commonsense reasoning (OpenCSR), where answer candidates are not provided (as in a multiple-choice setting). Given a question, a reasoner uses multi-hop reasoning over a knowledge corpus of facts, and outputs a ranked list of concepts from the corpus.

As we seek to advance commonsense reasoning towards practical applications, we propose to study **open-ended commonsense reasoning** (OpenCSR), where answers are generated efficiently, rather than selected from a small list of candidates (Figure 1). As a step toward this, here we explore a setting where a model produces a ranked list of answers from a large question-independent set of candidate concepts, which are extracted offline from a corpus of common-sense facts written in natural language.

The OpenCSR task is inherently challenging. One problem is that for many questions, finding an answer requires reasoning over two or more

* The work in progress was mainly done during Bill Yuchen Lin's internship at Google Research.

natural-language facts from the corpus. In the multiple-choice QA setting, as the set of candidates is small, we can pair a question with an answer, and use the combination to retrieve relevant facts and then reason with them. In the open-ended setting, this is impractical: instead one needs to retrieve facts from the corpus using the question alone. In this respect, OpenCSR is similar to multi-hop factoid QA about named entities, e.g. as done in HotpotQA (Yang et al., 2018).

However, the underlying reasoning chains of most multi-hop factoid QA datasets are relatively *clear* and *context-independent*, which are thus easier to infer. Commonsense questions, in contrast, exhibit more variable types of reasoning, and the connections between documents that support a reasoning chain are typically less clear. For example, in Fig. 1, a question "what can help alleviate X?" implicitly suggests a chain of Y="what contributes to X?" and "what can remove Y?" Furthermore, annotations are not available to identify which facts are needed in the latent reasoning chains needed to derive an answer — the only supervision is a set of questions and their answers. We discuss the formulation of OpenCSR and its challenges further in Section 3.

As shown in Fig. 1, many commonsense questions need *higher-order*, complex facts to reason about, such as "*trees remove carbon dioxide from the atmosphere through photosynthesis*", which are not stored in existing KGs. However, such facts have been collected in common-sense corpora, e.g., GenericsKB (Bhakthavatsalam et al., 2020). This motivates the question: how can we conduct *multi-hop* reasoning over such a knowledge corpus, just like traversing a KG? Moreover, can we achieve this in a *differentiable* way, to support end-to-end learning? To address the question, inspired by Seo et al. (2019) and Dhingra et al. (2020), we propose to construct an indexable representation of the corpus, and then reason over it via differentiable operations.

Specifically, we propose an efficient, differentiable multi-hop reasoning method for OpenCSR, named **DrFact** (for differentiable reasoning over facts). We formulate multi-hop reasoning over a corpus as a recursive process of *fact-following* operations over a hypergraph. We first encode all fact sentences within the corpus as *dense* vectors to form a *neural* fact index, such that a fast retrieval can be done via maximum inner product search (MIPS). Meanwhile, we use a *sparse* fact-to-fact matrix to store the *symbolic* links between facts (i.e., a pair of facts are linked if they share common concepts). DrFact merges both neural and symbolic aspects of the connections between facts to model the fact-following operation in an end-to-end differentiable framework. (Sections 4-5)

While the OpenCSR setting is more realistic, there is no existing dataset designed for evaluating this task. However, it is possible to construct meaningful evaluations in this setting from existing multiple-choice datasets, as we show in Section 6. We re-format three existing multiple-choice QA datasets for OpenCSR: QASC (Khot et al., 2020), OBQA (Mihaylov et al., 2018), and ARC (Clark et al., 2018). We compare with several strong baseline methods and find our proposed DrFact outperforms them by a large margin overall — 8.0% absolute improvement in Hit@100 accuracy over DPR (Karpukhin et al., 2020) (a state-of-the-art text retriever for QA) and 5.6% over DrKIT (Dhingra et al., 2020) (a strong method for entity-centric multi-hop reasoning).

2 Related Work

Commonsense Reasoning

Much recent CSR methods primarily focus on multiple-choice setting. For example, KagNet (Lin et al., 2019) and MHGRN (Feng et al., 2020) use an external commonsense knowledge graph to retrieve the connections between the question and each given candidate, and then use them as structural priors to individually score each choice. UnifiedQA (Khashabi et al., 2020), the state-of-the-art multi-choice QA model, instead simply concatenates the question with all answer candidates as a single input sequence to a T5 (Raffel et al., 2020) model for learning to generate the correct choice as extracting a span from the input. These methods, though powerful in determining the best choice for a multi-choice question, are less realistic for practical applications of NLP when answer candidates are usually not available.

Closed-book QA models (Roberts et al., 2020) are able to generate answers to given questions through simply fine-tuning a text-to-text transformer such as BART (Lewis et al., 2019) and T5. However, closed-book QA models do not provide intermediate explanations for their answers, i.e., the supporting facts, which makes them less trustworthy in downstream applications. Even augmented with an additional retrieval module (Lewis et al., 2020), they do not explicitly address multi-hop reasoning for OpenCSR-style questions.

Question Answering over Text Corpora

A conventional source of commonsense knowledge is triple-based symbolic commonsense knowledge graphs (CSKGs) such as ConceptNet (Speer et al., 2017). However, the binary relations in CSKGs greatly limit the types of the knowledge that can be encoded. Here we use a corpus of generic sentences about commonsense facts, in particular GenericsKB (Bhakthavatsalam et al., 2020), since text can represent more complex commonsense knowledge, including facts that relate three or more concepts. Formalized in this way, OpenCSR is a retrieval-based question answering task, similar to many open-domain QA tasks (Chen et al., 2017) such as HotpotQA (Yang et al., 2018) and Natural Questions (Kwiatkowski et al., 2019). However, the surface of commonsense questions in OpenCSR have fewer hints about kinds of multi-hop reasoning required to answer them than the factoid questions in open-domain QA, resulting in a particularly challenging reasoning problem (Sec. 3.2).

Multi-Hop Reasoning

Many recent models for open-domain QA tackle multi-hop reasoning through iterative retrieval, e.g., GRAFT-Net (Sun et al., 2018), MUPPET (Feldman and El-Yaniv, 2019), PullNet (Sun et al., 2019), and GoldEn (Qi et al., 2019). These models, however, are *not* end-to-end differentiable and thus tend to have slower inference speed, which is a limitation shared by many other works using reading comprehension for multi-step QA (Das et al., 2019; Lee et al., 2019). Neural Query Language (Cohen et al., 2019) designs differentiable multi-hop entity-following templates for reasoning over a symbolic KG, but the KG is limited to binary relations between entities from an explicitly enumerated set. As we want to reason over a textual knowledge corpus instead of a symbolic KG, DrKIT (Dhingra et al., 2020) is the most similar work to our proposed DrFact.

DrKIT computes a dense index of *entity mentions* (i.e., named-entity spans linked to a KG), and then models the entity-to-entity following operation in an end-to-end fashion. However, to date it has only been applied on *factoid* questions about named entities which have relatively clear multi-hop hints in the question itself. However, for commonsense facts and questions, such an entity-centric formulation of multi-hop reasoning can be sub-optimal. One problem is that concepts in commonsense facts are much more densely connected, unlike named entities in Wikipedia. DrKIT's entity following, i.e., hopping from a concept to another concept, so it can be much more expensive to traverse between mentions. More importantly, its mention-based index only keeps the partial context about each individual mention, while does not store the representations of the whole context of facts, which are essential for guiding a reasoner especially when multi-hop hints are relatively implicit as in the OpenCSR task.

3 Open-Ended Commonsense Reasoning

In this section, we first present our formulation of the open-ended setting for commonsense reasoning (OpenCSR), and then discuss the inherent challenges of this more realistic setting.

3.1 Task Formulation

We denote a **corpus** of knowledge facts to be $\mathcal{F}$, and use $\mathcal{V}$ to denote the **vocabulary** of concepts; both are sets consisting of unique elements. A **fact** $f_i \in \mathcal{F}$ is a sentence that describes generic commonsense knowledge, such as "*trees* remove *carbon dioxide* from the *atmosphere* through *photosynthesis*." A **concept** $c_j \in \mathcal{V}$ is a noun or noun chunk (after lemmatization) mentioned frequently in these facts (e.g., 'tree' and 'carbon dioxide').

Given only a **question** q (e.g., "*what can help alleviate global warming?*"), an open-ended commonsense reasoner is supposed to **answer** it by returning a weighted set of concepts, such as $\{(a_1$=*'renewable energy'*, $w_1)$, $(a_2$=*'tree'*, $w_2)$, $(a_3$=*'solar battery'*, $w_3)$, $\ldots\}$, where $w_i \in \mathbb{R}$ is the weight of the predicted concept $a_i \in \mathcal{V}$. In contrast, closed-ended models for multiple-choice QA usually assume that a correct answer is present, along with several wrong distractor choices (e.g., *'air cooler'*, *'fossil fuel'*, and *'water'*) and can only reason in this limited context.

In order to learn interpretable, trustworthy reasoners, it is expected that models can output intermediate results that justify the reasoning process — i.e., the supporting facts from $\mathcal{F}$. For example, an **explanation** for 'tree' to be an answer to the question above can be the combination of two facts: f_1 = "*carbon dioxide* is the major *green-*

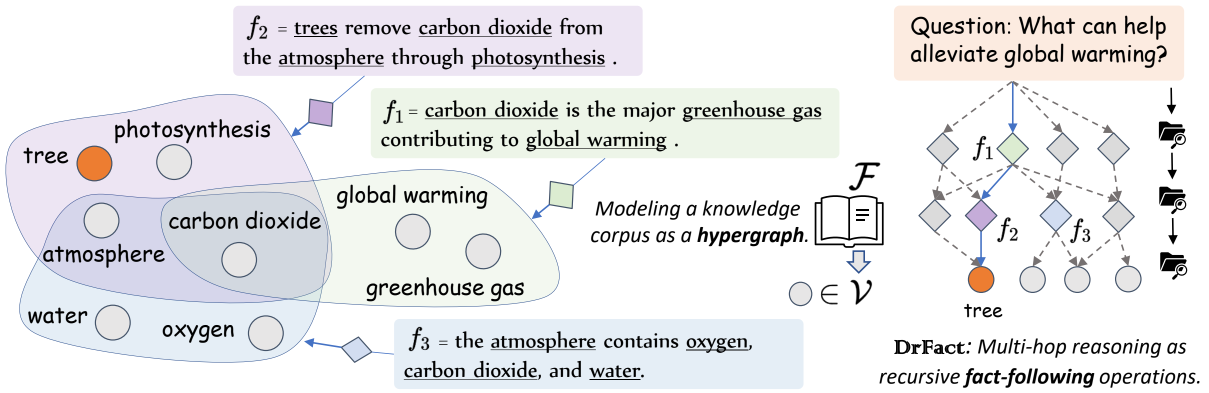

Figure 2: **A motivating example of how DrFact works for OpenCSR.** We model the knowledge corpus as a hypergraph consisting of *concepts* in $\mathcal{V}$ as *nodes* and *facts* in $\mathcal{F}$ as *hyperedges*. Then, we develop a differentiable reasoning method, DrFact, to perform *multi-hop reasoning* via *fact-following* operations (e.g., $f_1 \rightarrow f_2$).

house gas contributing to *global warming*" and f_2 = "*trees* remove *carbon dioxide* from the *atmosphere* through *photosynthesis*" (see Fig. 1).

3.2 Key Challenges

Though the open-ended setting is closer to the practical applications of commonsense reasoning, it is challenging due to the latent multi-hop structures of questions and the very large search space.

(C1) Latent Multi-Hop Structures. Commonsense questions (i.e., questions that need commonsense knowledge to reason) contrast with better-studied multi-hop factoid QA datasets, e.g., HotpotQA (Yang et al., 2018), which primarily focus on querying about **evident relations between named entities**. For example, an example multi-hop factoid question can be "which team does the player named 2015 Diamond Head Classic's MVP play for?" Its query structure is relatively clear and *self-evident* from the question itself: in this case the reasoning process can be decomposed into q_1 = "the player named 2015 DHC's MVP" and q_2 = "which team does q_1. answer play for".

The reasoning required to answer commonsense questions is usually more *implicit* and relatively unclear. Consider the previous example in Fig. 1, q = 'what can help alleviate global warming?' can be decomposed by q_1 = "what contributes global warming" and q_2 = "what removes q_1. answer from the atmosphere" — while many other decompositions are also plausible. Moreover, unlike HotpotQA, we assume that we have **no ground-truth justifications** in open-ended commonsense reasoning — i.e., the facts that justify an answer are unobserved and latent variables.

(C2) Very Large Search Space. In the multiple-choice setting, models only need to reason about a limited set of answer choices (usually less than 5), and one of them is guaranteed to be correct. In OpenCSR, we are given a large (e.g., $80k$) vocabulary $\mathcal{V}$ of concepts. The *efficiency* problem is thus especially serious when we want to perform multi-step retrieval to increase coverage and answer more complex questions. This poses a practical challenge of learning *differentiable* multi-hop reasoning approach for faster inference speed.

4 DrFact: An Efficient Approach for Differentiable Reasoning over Facts

To address the challenges of OpenCSR, we propose an efficient, **D**ifferentiable model for multi-hop **R**easoning over knowledge **Fact**s, named DRFACT. In this section we will describe the model at a high level, and the next section (Sec. 5) will present further details of the implementation.

4.1 Overview

As show in Figure 2, we propose to use a *hypergraph* to represent factual statements (i.e., sentences) about commonsense knowledge. A hypergraph, unlike a standard graph, allows a *hyperedge* to connect multiple nodes at a time. We here use the concept of hyperedge for representing facts ($\mathcal{F}$) and *node* for concepts ($\mathcal{V}$). A fact, as a hyperedge, connects multiple concepts that are mentioned in itself, while maintains the contextual information as a natural language statement. The concepts are connected with each other through such hyperedges, and the relation (for a particular hyperedge) is not assumed to be from any fixed

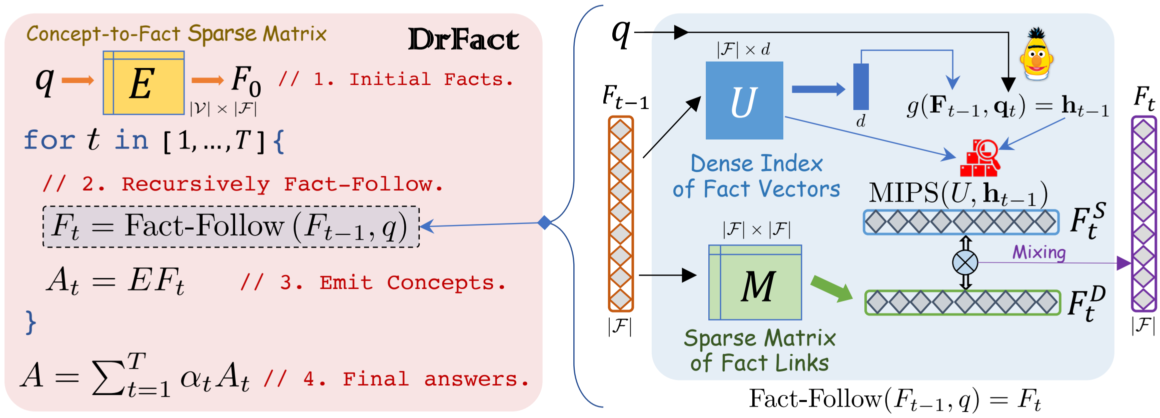

Figure 3: **The overall workflow of DrFact.** We encode the hypergraph (Fig. 2) with a concept-to-fact sparse matrix E and a fact-to-fact sparse matrix M. The dense fact index U is pre-computed with a pre-trained bi-encoder. A weighed set of facts is represented as a sparse vector F. The workflow (left) of DrFact starts mapping a question to a set of initial facts that have common concepts with it. Then, it recursively performs `Fact-Follow` operations (right) for computing F_t and A_t. Finally, it uses learnable hop-weights α_t to aggregate the answers.

set, but is open. Given such a hypergraph of commonsense knowledge, our open-ended reasoning model will then traverse the hypergraph starting from the question (concepts) and finally arrive at a set of concept nodes through multiple hops of fact-to-fact following steps. A probabilistic view of this process over T hops is shown below.

$$P(c \mid q) = P(c \mid q, F_T) \prod_{t=1}^{T} P(F_t \mid q, F_{t-1}) P\left(F_0 \mid q\right)$$

Intuitively, we want to model the distribution of a concept $c \in \mathcal{V}$ being an answer to a question q as $P(c \mid q)$. This answering process can be seen as a process of multiple iterations of "fact-following", or moving from one fact to using shared concepts (at the last hop). We use F_t to represent a weighted set of retrieved facts at the hop t, and F_1 is also called the initial facts below. Then, given the question information and the current retrieved facts, we iteratively retrieve the facts for the next hop. Finally, we score a concept using retrieved facts.

4.2 Differentiable Fact-Following Operation

The most important part in our framework is how to model the fact-following step in our formulation — $P\left(F_t \mid F_{t-1}, q\right)$. For modeling the translation from a fact to another fact under the context of a question q, we propose an efficient approach with a differentiable operation that uses both *neural* embeddings of the facts and their *symbolic* connections in the hypergraph.

The symbolic connections between facts is represented by a highly sparse fact-to-fact matrix M, which in our model is efficiently implemented with the `tf.RaggedTensor`[1] construct of TensorFlow. M stores a pre-computed dependency between pairs of facts, $M_{ij} = \text{follow\_score}(f_i, f_j)$. Intuitively, if we can traverse from a fact f_i to f_j, it usually means these fact should mention some common concepts, and also their semantics are related (e.g., they have high TF-IDF similarity), so our follow_score operator will reflect this intuition. The neural embeddings of the facts can be computed by a pre-trained BERT encoder, and forms a neural embedding index of fact vectors, which we denote with U. The embedding index contains richer semantic information about each fact, and helps measure the plausibility of a fact in the context of a question.

The proposed fact-follow operation has two parallel sub-steps: 1) sparse retrieval and 2) dense retrieval. The sparse retrieval uses the fact-to-fact sparse matrix to obtain possible next-hop facts. We can compute:

$$F_t^S = F_{t-1} M$$

efficiently thanks to the ragged representation of sparse matrices.

For the neural dense retrieval, we use the maximum inner product search (MIPS) (Johnson et al., 2019) over the fact embedding index U:

$$\begin{aligned} \mathbf{F_{t-1}} &= F_{t-1} U \\ \mathbf{h_{t-1}} &= g(\mathbf{F_{t-1}}, \mathbf{q_t}) \\ F_t^D &= \text{MIPS}(\mathbf{h_{t-1}}, U) \end{aligned}$$

[1] https://www.tensorflow.org/guide/ragged_tensor

We first look up the embedding vectors of the facts in F_{t-1} and aggregate them into a dense vector $\mathbf{F_{t-1}}$. Then $\mathbf{F_{t-1}}$ is fed into a neural layer with the query embedding at the current step q_t to create a query vector $\mathbf{h_{t-1}}$. The function $g(\cdot)$ is an MLP that maps the concatenation of the two input vectors to a dense output with the same dimensionality as the fact vectors. Finally, we retrieve the next-hop top-K facts F_t^D.

To get the best of both symbolic and neural world, we compute the element-wise multiplication as a way to combine the sparse and dense retrieved results:

$$F_t = F_t^S \odot F_t^D$$

We can now summarize the fact-following operation as a series of differentiable operations:

$$\begin{aligned} F_t &= \text{Fact-Follow}(F_{t-1}, q) \qquad (1) \\ &= F_{t-1}M \odot \text{MIPS}(g(F_{t-1}U, \mathbf{q_t}), U) \end{aligned}$$

At each hop, we multiply F_t with a pre-computed fact-to-concept matrix E, thus generating α_t, a set of concept predictions. To aggregate the concept scores, we take the maximum score among the facts that mention a concept c. Finally we take the weighted sum of the concept predictions at all hops as the final weighted concept sets $A = \sum_{t=1}^{T} \alpha_t A_t$, where α_t is a learnable parameter.

4.3 Auxiliary Learning with Distant Evidence

Intermediate evidence, i.e., supporting facts, is significant for guiding multi-hop reasoning models during training. In a weakly supervised setting, however, we usually do not have ground-truth annotation as they are expensive to obtain. To get some noisy yet still helpful supporting facts, we propose to extract distant evidence from our dense-retrieved results of the training questions.

We concatenate the question and the answer to query our pre-trained index U, and then we group the results into four groups: 1) question-answer facts, 2) question-only facts, 3) answer-only facts, and 4) none-facts, depending on whether they contain question/answer concepts. Then, to get a 2-hop evidence chain, we can check if a question-only fact can be linked to an answer-only fact through our sparse fact-to-fact matrix M. Similarly, we can also get 3-hop distant evidence. Afterwards, we name the set of supporting facts at each hop-position as $\{F_1^*, F_2^*, \dots\}$.

The final learning objective is thus to optimize the sum of the cross-entropy loss l between the final weighed set of concepts A and the truth answer A^*, as well as the auxiliary loss from distant evidence — the mean of the hop-wise loss between the predicted facts F_t and the distant supporting facts at that hop F_t^*, defined as follows:

$$\mathcal{L} = l(A, A^*) + \frac{1}{T}\sum_{t=1}^{T} l(F_t, F_t^*)$$

5 Implementation Details

(1) Dense Neural Fact Index U. We pre-train a bi-encoder architecture over BERT (Devlin et al., 2019), which learns to maximize the score facts that contain correct answers to a given question, following the steps of Karpukhin et al. (2020), so that we can use MIPS to do dense retrieval. After pre-training, we embed each fact in $\mathcal{F}$ with a dense vector (using the `[CLS]` token representation) and U is thus a $|\mathcal{F}| \times d$ dense embedding matrix.

(2) Sparse Fact-to-Fact Index M. We pre-compute the sparse links between facts by a set of connection rules, such as $f_i \to f_j$ when f_i and f_j have at least one common concept and f_j introduces at least two more new concepts that are not in f_i. Hence M is a binary sparse, binary matrix of the dense shape as $|\mathcal{F}| \times |\mathcal{F}|$.

(3) Hop-wise Question Encoding $\mathbf{q_t}$. We encode the question q with BERT and then use its `[CLS]` token vector as the dense representation for $\mathbf{q}$. For each hop, we append a hop-specific layer to model how the question context changes over time, so $\mathbf{q_t} = \text{MLP}_{\theta_t}(\mathbf{q})$.

(4) Fact Translating Function g. The translating function accepts both the vector representation of previous-hop facts $\mathbf{F_{t-1}}$ and the hop-wise question vector $\mathbf{q_t}$ and uses an MLP to map the concatenation of them to a vector used for a MIPS query: $\mathbf{h_{t-1}} = \text{MLP}_{\theta_g}([\mathbf{F_{t-1}}; \mathbf{q_t}])$. Thus, $\mathbf{h_{t-1}}$ has the same dimension as a fact vector in U.

(5) Hop-wise Answer Weights α_t. We use the shared query vector to learn how to aggregate predictions at different hops. For a T-hop DrFact model, we learn to transform the $\mathbf{q}$ to a T-dim vector where α_t is the t-th component.

(6) Preparing Initial Facts F_0. The set of initial facts F_0 is computed differently, as they are produced using the question q, instead of a previous-hop F_{t-1}. We use our pre-trained bi-encoder and

MIPS query to finds facts related to q and then select from the retrieved set those facts that contain question concepts, using via the sparse concept-to-fact matrix E.

(7) Self-Following. Equation 1 defines a sort of random-walk process on the hypergraph associated with the corpus. We found that performance was improved by making this a "lazy" random walk—in particular by augmenting F_t with the facts in F_{t-1} which have a weight higher than a threshold τ:

$$F_t = \text{Fact-Follow}(F_{t-1}, q) + \text{Filter}(F_{t-1}, \tau).$$

This means that F_t contains highly-relevant facts for all distances $t' < t$, and makes the model perform better for datasets that include reasoning chains with different numbers of "hops".

6 Experiments

In this section, we first introduce our experimental setup, such as how we get our knowledge corpus, concept vocabulary, and the re-formatted OpenCSR version datasets as well as the evaluation metrics. We then discuss the baseline methods, and finally present our experimental results.

6.1 Experimental Setup

The knowledge corpus and concept vocab.

We use the GenericsKB-Best corpus (Bhakthavatsalam et al., 2020) as the main knowledge source. It was constructed from multiple commonsense knowledge corpora (e.g., SimpleWikipedia, the Waterloo corpus, the ARC Corpus) by only keeping naturally occurring generic statements, i.e., commonsense knowledge facts. Thus, it is a perfect fit for our goal towards open-ended commonsense reasoning. In total, we have **1,025,413** unique facts as our $\mathcal{F}$. We use spaCy[2] to prepossess all sentences in the corpus and then extract frequent noun chunks within them as our concepts. The vocabulary $\mathcal{V}$ has **80,524** concepts, and every concept is mentioned at least 3 times.

Reformatted datasets for OpenCSR.

To facilitate the research on open-ended commonsense reasoning, we reformatted several existing multi-choice QA datasets to allow evaluating OpenCSR methods. We choose three datasets: QASC (Khot et al., 2020), OBQA (Mihaylov et al., 2018), and ARC (Clark et al., 2018), as their questions require commonsense knowledge and are in natural language. The statistics are shown in Table 1. To understand the multi-hop nature and the difficulty of each dataset, we designed a simple way[3] to estimate the percentage of one-hop questions in each dataset. The ARC has about 80% one-hop questions and thus is the easiest dataset, while OBQA is the hardest, with only 40% one-hop questions.

Stat. \ Data	ARC	QASC	OBQA	**Overall**
# All Examples	6,308	8,155	5,057	19,520
# Training Set	5,058	6,577	4,164	15,800
# Validation Set	562	731	463	1,756
# Test Set	688	847	430	1,965
Single-answer%	53.34%	82.64%	49.53%	58.11%
One-hop%	79.36%	44.75%	**39.30%**	54.52%

Table 1: Statistics of reformatted data for OpenCSR.

To convert a multiple-choice question to an OpenCSR task, we first remove questions where the correct answer does not contain any concept in $\mathcal{V}$. We also removed the few questions that require comparisons between given choices, as they are designed only for closed-ended reasoning models, e.g., "Which of the following is the *most* ..."

We also record the concepts that appear in the correct answer for each question. Note that it is possible that correct answer mentions multiple concept: e.g., the answer 'gravitational pull from the sun' contains the concepts 'gravitational pull', 'pull', and 'sun'. We keep only the *longest* concepts in the answer for evaluation: e.g., for this case we include only 'gravitation pull' and 'sun', not 'pull'. The **only supervision** for our model is the questions and the concepts mentioned in their correct answers.

Automatic evaluation metrics.

In order to efficiently evaluate different OpenCSR methods, we use three automatic metrics. Recall that, given a question q, the final output of every method is a weighted set of concepts $A = \{(a_1, w_1), \dots\}$ (see Sec. 3.1). We denote the set of *true answer concepts*, as defined above, as $A^* = \{a_1^*, a_2^*, \dots\}$.

[2]https://spacy.io/

[3]Basically, we use DPR (Karpukhin et al., 2020), the state-of-the-art BERT-based learnable text retrieval, to get top-1000 facts for each question and then we consider a question as a one-hop question if there are more than 5 facts that have at least a pair of question concept and answer concept.

We define **Hit@K** accuracy to be the fraction of questions for which we can find *at least one* true answer concept $a_i^* \in A^*$ in the top-K concepts of A (sorted in descending order of weight). Hit@K accuracy is also named *top-K accuracy* in some information retrieval studies. However, only 58% of questions have a single truth answer concept. A more stringent criterion to use for ranking would be to see if it contains all true answer concepts. We thus use **FindAll@K** to denote teh fraction of questions for which *all* true answer concepts are in the top-K concepts of model outputs A.

We also make use of the distractors (i.e., wrong choices) of the original datasets to help evaluate the models. We denote the concepts in the distractors as $D = \{d_1, d_2, \dots\}$: intuitively, the concepts in D should be ranked lower than concepts in $A*$. To capture this intuition, we introduce **MC-Acc** as another metric, whichs represent the percentage of the questions for which any truth answer concept ranks higher than all distractor concepts in the proposed ranking A.

6.2 Baseline Methods

Text Retrieval Methods. The most intuitive and straightforward approach to the OpenCSR task is to retrieve relevant facts about given questions, and then use the concepts mentioned in the top-ranked facts as answer predictions. `BM25` is one of the most popular *unsupervised* method for retrieval[4]. The Dense Passage Retrieval (`DPR`) model is a state-of-the-art trainable, neural retriever (Karpukhin et al., 2020), which is designed for retrieval-based QA and thus fits OpenCSR. Follwing the instructions of the DPR authors, our training process of `DPR` uses BM25-retrieved facts to create positive and negative context for each training questions to create supervision for learning the DPR index. For both methods, we score a concept by the *max*[5] of the scores of the retrieved fact mention it.

Reading Comprehension Module. Although retriever-only methods are efficient, the aggregated concept scores can be sub-optimal, as they do not consider the context (i.e., the combination of the question and a particular fact) in evaluating concepts. Inspired by the conventional open-domain QA methods and following the experiments of Lee et al. (2019) and Karpukhin et al. (2020), we train a BERT-based neural reader module. Its input is a question and a particular retrieved fact by BM25/DPR. Then, it concatenates them as a single input sequence to BERT and finally scores any span in the input as an answer.

Specifically, we use the BM25/DPR retrieved facts to create positive and negative training examples in the SQuAD format[6]. The learned reader module can thus judge if the question is answerable given the retrieved fact, and (if answerable) what spans should be the answer, and provide a confidence for a span. We use the reader modules over BM25/DPR retrieved facts and re-score concepts with the product of retrieval scores and span scores. However, note that the computation cost of such reader modules is very sensitive to the number of retrieved facts which we get from BM25/DPR, and it can be very expensive in practice.

DrKIT. Following Dhingra et al. (2020), we use DrKIT for OpenCSR. DrKIT is also an efficient multi-hop reasoning model that reasons over a pre-computed indexed corpus. The main difference[7] between DrKIT and DrFact is that DrKIT uses entity mentions (concept-mention) as the basic unit in its multi-hop following steps, instead of fact vectors: i.e. DrKIT traverses a graph of entities and entity mentions, while DrFact traverses a hypergraph of facts.

6.3 Results and Analysis

Main results. We train and test the baseline models (excluding BM25) and our DrFact on each OpenCSR-version dataset individually[8]. Each method will produce a ranked list of concepts (i.e., the weighted set of concepts A). For a comprehensive understanding, we report the Hit@K and FindAll@K accuracy of all methods, at K=50 and K=100, in Table 2. We also show the MC-Acc results in Table 3. The **overall** results are the weighted average of performance over the three datasets, weighted by their sizes of test sets.

[4] Our implementation is based on `https://github.com/elastic/elasticsearch`

[5] We also tried *mean* and *sum*, and *max* performs the best.

[6] We use the SQuAD 2.0 format. Positive examples are the top facts with truth answer concepts and questions are *answerable* in these examples, while negative, unanswerable examples are retrieved facts that do not contain any answers.

[7] Please find more details in Section 2 and Section 4.

[8] For all methods, we tune their hyper-parameters on the validation set and then use the same configurations to train them with the combination of the training and validation sets.

Methods / Metrics \ Data	ARC		QASC		OBQA		**Overall**	
Metric = **Hit@K**	@50	@100	@50	@100	@50	@100	@50	@100
BM25+Reader (BERT-Large)	47.67%	55.67%	**38.91%**	46.55%	21.16%	29.30%	38.09%	45.97%
DPR+Reader (BERT-Large)	48.40%	56.10%	37.82%	47.39%	21.63%	30.93%	37.98%	46.84%
BM25 (Off-the-shelf)	33.58%	41.28%	32.11%	38.96%	20.23%	26.05%	30.02%	36.95%
DPR (Karpukhin et al., 2020)	44.91%	54.36%	35.06%	45.34%	21.86%	29.77%	35.62%	45.09%
DrKIT (Dhingra et al., 2020)	46.22%	61.48%	34.46%	42.37%	30.00%	36.28%	37.60%	47.73%
DrFact (Ours)	**49.42%**	**61.77%**	38.72%	**51.59%**	**33.26%**	**42.09%**	**41.27%**	**53.08%**
Metric = **FindAll@K**	@50	@100	@50	@100	@50	@100	@50	@100
BM25+Reader (BERT-Large)	33.28%	37.94%	33.45%	41.09%	10.70%	15.12%	28.41%	34.30%
DPR+Reader (BERT-Large)	33.43%	38.81%	32.73%	41.70%	10.70%	16.28%	28.15%	35.13%
BM25 (Off-the-shelf)	22.24%	26.74%	27.98%	33.77%	10.00%	12.56%	22.04%	26.67%
DPR (Karpukhin et al., 2020)	30.81%	36.77%	30.34%	40.14%	11.40%	16.05%	26.36%	33.69%
DrKIT (Dhingra et al., 2020)	32.85%	**43.31%**	29.38%	36.72%	14.19%	18.14%	27.27%	34.96%
DrFact (Ours)	**34.45%**	43.02%	**33.53%**	**45.81%**	**18.14%**	**23.72%**	**30.48%**	**39.99%**

Table 2: Experimental results of the Hit@K and FindAll@K accuracy for evaluating OpenCSR methods. The BM25, DPR, DrKIT, and DrFact are all retriever-only methods, and their efficiency is at the same level (the last three use BERT-base). BM25/DPR+Reader, however, uses a BERT-Large model to read the combination of a question and each retrieved fact by BM25/DPR, thus resulting in a very low inference speed for practical use.

Methods \ Data	ARC	QASC	OBQA	Overall
BM25+Reader	41.72%	36.48%	23.26%	35.42%
DPR+Reader	**43.75%**	35.27%	26.05%	36.22%
BM25	36.63%	32.47%	22.79%	31.81%
DPR	42.30%	35.18%	23.49%	35.11%
DrKIT	42.15%	28.81%	33.72%	34.56%
DrFact (Our	43.46%	**39.20%**	**39.77%**	**40.82%**

Table 3: Experiments on the **MC-Acc** metric. The reader modules are based on BERT-Large; DPR, DrKIt, and DrFact are using BERT-base.

Interestingly, we find that with a powerful reader such as BERT-Large for re-scoring answers, DPR-Reader does not yield a large improvement over BM25-Reader, especially for datasets with fewer one-hop questions.

We can see that DrFact outperforms all baseline methods for most datasets and metrics (the only exception being the ARC dataset for the MC-Acc metric, where its performance is slightly below the DPR+Reader baseline). Comparing with the state-of-the-art text retriever DPR, DrFact improves by about 5.5% absolute points in Hit@50 and 8% in Hit@100 overall. Even compared with the reader-augmented methods, which are much more *computational expensive*, DrFact still has about 3% overall improvement in Hit@50 accuracy. The largest performance gain is on the OBQA dataset, which has the *least* one-hop questions and is thus the most challenging.

The results in FindAll@K accuracy are much lower than Hit@K accuracy. This is expected, as it is more challenging to retrieval all truth concepts than to retrieve at least one. However, DrFact will outperforms the baseline consistently. Table 3 shows that this also holds for the MC-Acc metric.

Run-time efficiency. We use BERT-Large for the reader module because we would like to see how well we can do with the conventional open-domain QA pipeline, without considering efficiency. The results of DPR, DrKIT and DrFact in all experiments are all done with BERT-base, since efficiency is a key issue in the OpenCSR task. Therefore, for each question, at inference time, DPR will make one call to BERT-base, for encoding the question and do one MIPS search. Similarly, DrKIT and DrFact (with $T = 3$ hops) will make one call to BERT-base for query encoding and do three MIPS searches (which are very efficient in GPU). That being said, as the entity-to-mention sparse matrix of DrKIT is much larger than the fact-to-fact matrix of DrFact, DrKIT is significantly slower than DrFact. However, BM25/DPR+Reader are much more computationally expensive as they read all top-K facts retrieved by BM25/DPR, and make K calls to BERT-Large. Note that $K = 100$ here.

Ablation study. We first use different maximum hops ($T = \{1, 2, 3\}$) — i.e., the number of the

	ARC	QASC	OBQA	Overall
T=1	59.46%	46.66%	30.14%	47.53%
T=2	61.19%	50.30%	43.02%	52.52%
T=3 ✓↓	61.77%	51.59%	42.09%	53.08%
Self-follow ✗	61.19%	50.41%	40.70%	52.06%
Aux. loss ✗	59.45%	50.82%	41.56%	51.82%

Table 4: Ablation study of DrFact (in Hit@100 Acc).

calls to `Fact-Follow`, as shown in Table 4. The overall performance is the best when $T = 3$ while the performance drops nearly 1% point on OBQA, comparing with $T = 2$. We conjecture this is due to nature of the datasets, such as the percentage of underlying hard questions. We keep $T = 3$ when testing the situations without the *auxiliary learning loss* (Sec. 4.3) or the *self-following* trick (Sec. 5 (7)). Both are shown to be important to DrFact. Self-following is especially helpful for QASC and OBQA, where there are more multi-hop questions, by learning to directly keep important facts from previous hops.

7 Conclusion

We study a realistic (yet challenging) task setting — open-ended commonsense reasoning. We present DrFact, a scalable multi-hop reasoning method that traverses a corpus (as a hypergraph) via differentiable fact-following module. It efficiently uses both a neural dense index of facts and sparse tensors of symbolic links between facts, through MIPS and sparse-matrix computation. DrFact outperforms several strong baseline methods on three reformatted, open-ended version of datasets targeting commonsense knowledge. Our contribution is a step towards advancing commonsense reasoning for more practical applications.

As OpenCSR is still an underexplored area, important future directions include: 1) developing new evaluation protocols (including both annotations and evaluation metrics) that can explicitly evaluate multiple correct answers, 2) enriching the vocabulary V to include more diverse concepts, and 3) incorporating inference logic rules as typed symbolic links between facts in M.

References

Sumithra Bhakthavatsalam, Chloe Anastasiades, and Peter Clark. 2020. Genericskb: A knowledge base of generic statements. *arXiv preprint arXiv:2005.00660*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

William W Cohen, Matthew Siegler, and Alex Hofer. 2019. Neural query language: A knowledge base query language for tensorflow. *arXiv preprint arXiv:1905.06209*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. *arXiv preprint arXiv:1905.05733*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base. In *International Conference on Learning Representations*.

Yair Feldman and Ran El-Yaniv. 2019. Multi-hop paragraph retrieval for open-domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2296–2309, Florence, Italy. Association for Computational Linguistics.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. *arXiv preprint arXiv:2005.00646*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *arXiv preprint arXiv:2005.00700*.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. Qasc: A dataset for question answering via sentence composition. In *AAAI*, pages 8082–8090.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839, Hong Kong, China. Association for Computational Linguistics.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *AAAI*, pages 8449–8456.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D Manning. 2019. Answering complex open-domain questions through iterative query generation. *arXiv preprint arXiv:1910.07000*.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur P Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. *arXiv preprint arXiv:1906.05807*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.