QAMPARI: A Benchmark for Open-domain Questions with Many Answers

Samuel Joseph Amouyal **Ohad Rubin** **Ori Yoran** **Tomer Wolfson**
Jonathan Herzig **Jonathan Berant**
Blavatnik School of Computer Science, Tel Aviv University, Israel
{samuel.amouyal, ohad.rubin, joberant}@cs.tau.ac.il

Abstract

Existing benchmarks for open-domain question answering (ODQA) typically focus on questions whose answers are all in a single paragraph. By contrast, many natural questions, such as *"What players were drafted by the Brooklyn Nets?"* have a long *list of answers* extracted from multiple paragraphs. Answering such questions requires retrieving and reading many passages from a large corpus. We introduce QAMPARI, an ODQA benchmark, where question answers are lists of entities, spread across many paragraphs. We created QAMPARI by (a) generating questions with multiple answers from Wikipedia's knowledge graph and tables, (b) automatically pairing answers with supporting evidence in Wikipedia paragraphs, and (c) manually paraphrasing questions and validating each answer. Across a wide range of ODQA models, we find that QAMPARI is challenging in terms of both passage retrieval and answer generation, with models reaching an F_1 score of 31.0 at best. We view QAMPARI as a valuable resource for ODQA research, which will aid to develop models that handle a broad range of question types, including single and multi-answer questions.

1 Introduction

Open-domain question answering (ODQA) is a core language understanding task concerned with answering factoid questions over large document collections (Voorhees and Tice, 2000; Brill et al., 2002). Due to its wide applicability, ODQA has received substantial attention in recent years (Chen et al., 2017; Lee et al., 2019; Karpukhin et al., 2020). Typically, systems tackling ODQA tasks follow the "retrieve-and-read" paradigm, where *a retriever* first retrieves a set of candidate passages, followed by *a reader* which receives the retrieved passages and produces the final answer.

The retrieve-and-read paradigm has been effective for benchmarks such as Natural Questions

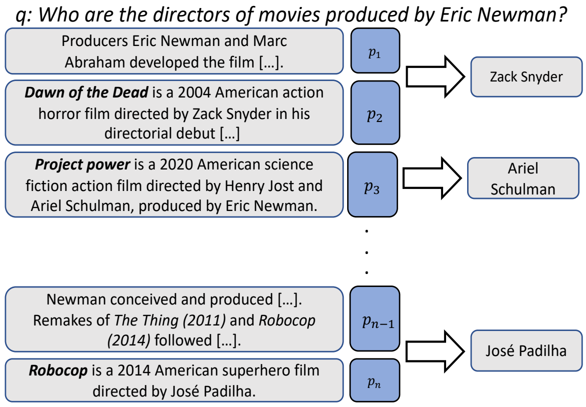


Figure 1: An example from QAMPARI with a generated question q, a subset of its evidence Wikipedia passages (left, p_i) and their corresponding answer.

(NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017), where the answer is a single phrase from a single passage. However, in many cases, a question might have *many* answers, spread across multiple passages. Consider the example in Fig. 1. Eric Newman produced multiple movies, so finding them, along with their directors, requires incorporating information from many passages. Such questions pose two main challenges to retrieve-and-read systems. First, as there are multiple answers that can be far apart, the reader must reason over a long text sequence to generate all correct answers. Second, since the reader is computationally constrained to process at most K passages, the retriever must score all necessary passages at its top-K results, which is challenging and even impossible when the number of relevant passages exceeds K.

While recent work explored questions that involve reading multiple passages, the number of passages was quite small. AMBIGQA (Min et al., 2020) studied ambiguous questions from NQ with several answers. However, as 70% of its questions have at most two answers, retrieve-and-read models can be adapted to AMBIGQA. HOTPOTQA (Yang et al., 2018) focused on multi-hop reasoning, but its questions require no more than two passages to answer. Last, WIKINLDB (Thorne et al., 2021) is a benchmark for testing reasoning over multiple facts. However, WIKINLDB restricted its text corpus to databases of 1,000 facts at most, making it significantly smaller than standard ODQA corpora. Moreover, these facts are model-generated utterances rather than natural language passages.

In this work, we present QAMPARI, a benchmark for **Q**uestions with many **A**nswers over **M**ultiple **Par**agraphs, **I**ndeed. All questions in QAMPARI have at least 5 answers, and an average of 13 answers. Examples are semi-automatically generated using two data sources, Wikidata (Vrandečić and Krötzsch, 2014) and Wikipedia tables. We automatically generate multi-answer questions of the form *"What/Who has [relation] with [entity]?"* and convert these into pseudo-language using manually defined templates. Then, we verify that our questions are answerable from Wikipedia by automatically extracting evidence passages for all their answers. Finally, we use crowdsourcing to validate example correctness, and paraphrase questions into natural language (Wang et al., 2015). To further enrich our data we also generate *composition* questions, that compose two relations (as in Fig. 1), and *intersection* questions, such as *"What movies were produced and directed by Clint Eastwood?"*. Overall, QAMPARI contains 2K development and test questions and more than 60K training examples – see Tab. 1 for some examples.

We evaluate models from the retrieve-and-read family as well-as a closed-book question answering model (Roberts et al., 2020), and find that they struggle on QAMPARI. Specifically, we experiment with BM25 (Robertson and Zaragoza, 2009) and DPR (Karpukhin et al., 2020) as retrievers, followed by either (a) a RAG-like reader (Lewis et al., 2020) that given each retrieved passage either decodes an answer or abstains, or (b) an FiD reader (Izacard and Grave, 2021) that takes the encoded representations of multiple passages and decodes the list of answers directly.

When training models on QAMPARI alone, or in a multi-task setup with NQ, we observe that QAMPARI is challenging in terms of both passage retrieval and answer generation. Namely, the best model reaches an F_1 score of 31.0. Moreover, models return more than 80% of the correct answers in only 31.2% of the test examples, well below performance on single-answer datasets, such as NQ.

To summarize, QAMPARI is a challenging benchmark for evaluating the ability of ODQA models to handle questions with many answers over multiple passages. We advocate to evaluate ODQA models not on QAMPARI alone, but alongside benchmarks such as NQ and TriviaQA. Such joint evaluation will test models' ability to handle both single- and multi-answer questions, an evaluation that the community is currently lacking. The QAMPARI benchmark, models and relevant codebase are available at: `https://anonymized`.

2 Dataset Construction

Each example in QAMPARI is a triple $(q, \mathcal{A}, \mathcal{P})$, where q is a question, $\mathcal{A}$ is a set of answers and $\mathcal{P}$ is a set of passages from our target corpus. An answer $a \in \mathcal{A}$ has 1-2 evidence passages from $\mathcal{P}$ (see Fig. 1).

We define passages as consecutive sentences from our corpus (Wikipedia), that span at most 100 words. As our focus is multi-answer questions, examples in QAMPARI have $|\mathcal{A}| \geq 5$.

Overview We generate examples in two steps. First, we generate *simple questions* that involve a single entity and relation, e.g. *"Who was drafted by the Brooklyn Nets?"* (§2.1). Then, we expand such questions to generate *complex questions* with *intersection* and *composition* operations (§2.2).

To increase diversity, questions are generated from two data sources, Wikidata and Wikipedia tables. We first describe example generation over Wikidata, then briefly present the generation process from Wikipedia tables in §2.3. In both cases, we ensure answers can be derived from evidence passages in Wikipedia.[1] Tab. 1 presents examples from each data source and question type.

Notation We introduce notation for formal queries over Wikidata to explain example generation. Wikidata is a knowledge graph, $\mathcal{K}$, that can be viewed as a set of labeled edges (e_1, r, e_2). Graph nodes $e_1, e_2 \in \mathcal{EI}$ are entities connected by an edge labeled by the relation $r \in \mathcal{R}$. For example, a possible labeled edge is (`BarackObama`, `ReceivedAward`, `NobelPeacePrize`).

One can query $\mathcal{K}$ by applying a relation r over an entity e, resulting in a *simple query* $r(e)$ whose *denotation* (answer set) is $[\![r(e)]\!] = \{e_i \mid (e_i, r, e) \in \mathcal{K}\}$. *Composition queries* are formed

[1] Wikipedia dump: 2021-08-01

Data source	Question type	Question	Answer example
Wikidata	Simple	Who is or was a member of the Australian Army?	George Macarthur-Onslow
	Intersection	What movie produced by Jerry Ward was also directed by Vincent Sherman?	Hard Way
	Composition	From which country did Seattle Storm make draft selections?	Australia
Wiki. tables	Simple	What magazine is a satirical magazine?	The Clinic
	Composition	What are the museums found in Concord, Massachusetts?	The Wayside

Table 1: Example questions and one representative answer for all data sources and question types.

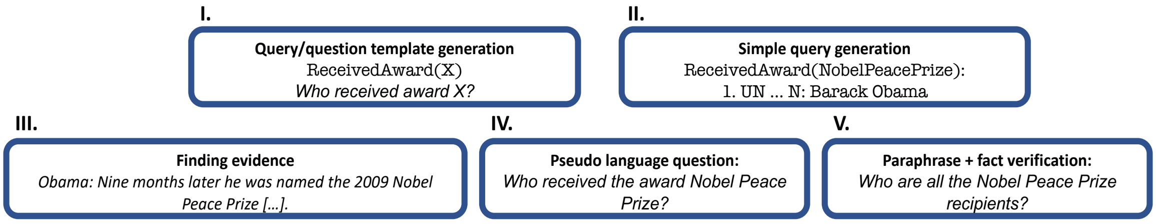


Figure 2: An overview of example generation for simple questions.

by applying a relation over the result of a simple query. We denote a composition query by $r_2(r_1(e))$, and its denotation is $[\![r_2(r_1(e))]\!] = \{e_i \mid \exists e_j \text{ s.t } (e_i, r_2, e_j) \in \mathcal{K} \wedge (e_j, r_1, e) \in \mathcal{K}\}$. Last, an *intersection query* $r_1(e_1) \sqcap r_2(e_2)$ corresponds to the intersection of two simple queries, $[\![r_1(e_1) \sqcap r_2(e_2)]\!] = \{e_i \mid (e_i, r_1, e_1) \in \mathcal{K} \wedge (e_i, r_2, e_2) \in \mathcal{K}\}$.

2.1 Simple Questions

Fig. 2 provides an overview of our procedure for creating *simple question* examples: (i) We manually define query templates, (ii) populate query templates using $\mathcal{K}$ to create queries with a sufficiently large number of answers in $\mathcal{K}$, (iii) automatically identify evidence passages for the answers and filter out noisy examples, (iv) map query templates to question templates to obtain pseudo-language questions, and (v) validate answers and paraphrase pseudo-language questions through crowdsourcing. Next, we describe each of these steps in detail.

Generating query templates We manually select a set of 135 relations $\bar{\mathcal{R}} \subset \mathcal{R}$, which will be used in our query templates. We select frequent relations from Wikidata for which denotations contain many entities (e.g., `ReceivedAward`). The list of relations is in App. A. For each relation, we manually write a template to map queries to pseudo-language questions. For example, the template for `ReceivedAward` is *"Who received the award X?"*

Some relations are underspecified – for example, `LocatedIn` can describe the location of buildings, geographical features, and cities. When generating synthetic questions, this leads to vague questions such as *"What is located in Paris?"*. To address this, we manually split these to *typed relations* that specify the semantic type of their answers/denotations. This is done using the type hierarchy given in Wikidata and given the type t of answer entities. We denote typed relations by r_t, and the denotation of $r_t(e)$ comprises all entities of type t returned by $r(e)$. For example, the entity `The Louvre` has type `cultural organization`, and we can map the relevant query template to the pseudo-language question *"Which cultural organization is located in Paris?"*.

Simple query generation We instantiate all possible simple queries using all $r \in \bar{\mathcal{R}}$ and entities e in Wikidata. For a relation r (or r_t), we keep the query $r(e)$ iff $|r(e)| \geq 5$. We denote this set of instantiated simple queries by $\mathcal{S}$, which contains 1,431,268 simple queries.

Finding evidence sentences For an ODQA benchmark, we must verify that every answer is found in our target corpus. We do this by identifying candidate evidence sentences from Wikipedia, and verifying they entail the answer, using a Natural Language Inference (NLI) model.

Specifically, every simple query-answer pair can be viewed as a triple (e_1, r, e_2). We use a "distant supervision" approach (Mintz et al., 2009), similar to KELM (Agarwal et al., 2021), and define any sentence in the Wikipedia page of entity e_1 that contains the entity e_2, or one of its Wikidata aliases, as a candidate evidence sentence (and vice versa in the page of e_2). E.g., in Fig. 2, the evidence for (`BarackObama`, `ReceivedAward`, `NobelPeacePrize`) appears on the page `Barack Obama`, where *'Nobel Peace Prize'* appears.

Aligning Wikipedia sentences to Wikidata can lead to false positives. E.g., for the triple (`TheGoonies`, `HasScreenwriter`, `StevenSpielberg`), most mentions of Spielberg in the page `TheGoonies` are not as a screenwriter. To account for this, we use an off-the-shelf NLI model.[2] For every answer, we consider each candidate evidence sentence along with its two preceding sentences, and check whether they entail the hypothesis phrase describing the triple (e_1, r, e_2). We use templates to phrase triples as short declarative sentences (*"The Goonies has Steven Spielberg as screenwriter")*. An answer is *validated* if there is an evidence sentence that entails the triple. Manual analysis shows this process eliminates 70% of false positives, while removing only 7.5% of the correct alignments.

Query filtering After finding evidence sentences, we only keep queries that at least 80% of their answers were validated and their number of validated answers is between 5 and 200. The resulting set contains 60,792 simple queries, where each query has a set of validated answers, $\mathcal{A}$, and of passages $\mathcal{P}$ that contain the identified evidence sentences.[3]

2.2 Complex Questions

To increase diversity, we expand simple queries to composition and intersection queries, for which answers require reading two passages.

Intersection Intersection queries are generated by finding two simple queries such that the size of the intersection of their denotations is at least 5. To avoid improbable questions such as *"Which competition was won by Manchester City and had Manchester City as a participant?"*, we add a constraint that the denotation of one of the simple queries cannot be a subset of the other. Formally, the set of intersection queries are all queries $r_1(e_1) \sqcap r_2(e_2)$ such that $|[\![r_2(e_2) \sqcap r_1(e_1)]\!]| \geq 5$, $[\![r_1(e_1)]\!] \not\subseteq [\![r_2(e_2)]\!]$, and $[\![r_2(e_2)]\!] \not\subseteq [\![r_1(e_1)]\!]$.

Pseudo-language questions are generated by heuristically combining the two simple questions, for example *"Which television program had Chris Carter as screenwriter and had Frank Spotnitz as screenwriter?"*. There is no need to perform answer validation since all of the underlying intersecting answers were already validated.

Composition To create composition queries, we manually handpick a set of 423 relations $\mathcal{R}_{\text{comp}} \subset \mathcal{R}$ (list in our codebase), in a process similar to simple queries. Then, we generate all the possible composition queries $r_2(r_1(e))$ such that $r_1(e) \in \mathcal{S}$, $r_2 \in \mathcal{R}_{\text{comp}}$, and $|[\![r_2(r_1(e))]\!]| \geq 5$. An example composition query is *"What is the height of buildings located in Dubai?"*.

Unlike intersection queries, in composition queries we need to validate that our new triples (e_i, r_2, e_j), where $e_j \in [\![r_1(e)]\!]$, are indeed supported by Wikipedia sentences. We use the same procedure to find evidence sentences for triples (e_i, r_2, e_j), and consider an answer e_i as *validated* if both (e_i, r_2, e_j) and (e_j, r_1, e) can be aligned to Wikipedia. We keep all complex queries where 80% of the answers are validated. Finally, we manually define templates for relations in $\mathcal{R}_{\text{comp}}$ to generate pseudo-language questions.

2.3 Questions from Wikipedia Tables

To further diversify QAMPARI, we create an analogous pipeline for generating simple and composition questions from Wikipedia tables, with more open-ended relations compared to Wikidata. We briefly describe this pipeline.

We look at all Wikipedia tables with title *"List of X"* that have at least 5 rows, in total, 1,897 tables. We find the "key" column, c_{key} in each table using the table classifier from Talmor et al. (2021), which outputs the column of entities that the table describes. For example, in the table *List of nuclear whistle blowers*, c_{key} is *'name'* and specifies the whistle-blower names. This naturally creates simple questions of the form *"Who or what is X?"*.

Simple questions are expanded to composition questions by looking at non-key columns, $c_{\text{non-key}}$ and asking what rows in the table have the value v in column $c_{\text{non-key}}$. For example, what is the value in the column *'Year'* for nuclear whistle-blowers.

Questions from Wikipedia are validated using a procedure similar to Wikidata. For each answer entity e, we validate that the Wikipedia page for e contains the relevant words that are part of the name of the table as well as the value (for composition questions), and only keep questions where

[2] huggingface.co/ynie/roberta-large-snli_mnli_fever_anli_R1_R2_R3-nli

[3] We keep a single evidence passage for every triple.

80% of the table rows are validated and the number of validated answers is at least 5. Overall, we generate 170 simple questions and 6,036 composition questions using this process.

2.4 Data Split

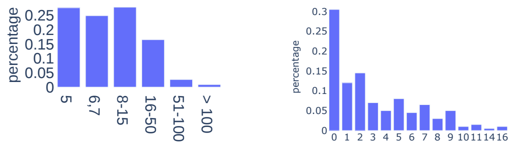
QAMPARI contains a training set, whose goal is to teach the model to handle multi-answer questions. However, we do not want the model to to memorize how particular Wikidata relations map to text patterns. Consequently, we perform a *relation split*, randomly splitting the set $\bar{\mathcal{R}}$ into two equally-sized sets $\bar{\mathcal{R}}_{\text{train}}$ and $\bar{\mathcal{R}}_{\text{test}}$. Simple queries are assigned to the train/test set based on their relation, composition queries $r_2(r_1(e))$ are assigned to the test set iff either r_1 or r_2 are in $\bar{\mathcal{R}}_{\text{test}}$, and intersection queries $r_1(e_1) \sqcap r_2(e_2)$ are placed in the test set iff both r_1 and r_2 are in $\bar{\mathcal{R}}_{\text{test}}$.

We now create the train/development/test split (Tab. 2). The main bottleneck in our example generation pipeline is validation of the test set through crowdsourcing (§2.5), since each question requires validating all of the answers. Thus, we pre-determine the test set to contain 1,000 simple questions (830 from Wikidata, 170 from Wikipedia tables) and 1,000 complex questions (400 Wikidata composition questions, 400 Wikidata intersection questions, 200 Wikipedia tables composition questions). For simple Wikidata questions, we sample 830 questions such that the distribution over relations from $\bar{\mathcal{R}}_{\text{test}}$ is roughly uniform. All Wikipedia tables simple questions are placed in the test set, and for complex questions we randomly sample the pre-determined number from the set of generated questions. Last, the test set is randomly split in half to a development set and test set. We also sub-sample training set examples, such that each relation appears in at most 1,000 examples.

2.5 Crowdsourcing

Correctness validation For every question and answer, we present a crowdsourcing worker with the question, the answer, and links to the Wikipedia page (or pages for complex questions) with the evidence passage. We ask the worker to check if the question can be answered from the given pages, using the text only (no infoboxes or tables).

Since the vast majority of examples are correct, we test worker performance by injecting wrong answers in 10% of the cases and reject workers that fail to identify wrong answers. Moreover, we manually verify 5% of examples marked as *correct*



(a) # answers (b) # added answers

Figure 3: Left: Distribution of the number of answers per example. Right: Proportion of questions per number of added answers in *ExtendedSet*.

and all examples marked as *incorrect*, and again reject low-performing workers. Overall, 24 annotators validated 30,259 answers for an average pay of 12.5$ per hour. We find that our process for generating examples is accurate, with 96.6% of the answers validated. Non-validated questions were replaced until 2,000 questions were validated. A question is defined non-validated if its number of distinct answers goes bellow 5. Snapshots from the presented tasks are in App. B.

Paraphrasing Since our questions are in pseudo-language, we follow past work (Wang et al., 2015) and ask workers to re-phrase 3,000 questions in the training set and the entire development/test set. We restrict this task to US or UK workers who pass a qualification test. We randomly verified half of the paraphrases for each worker for quality assurance.

3 Dataset Analysis

QAMPARI contains 61,911 training examples, 1,000 development examples and 1,000 test examples. Tab. 1 provides example questions of each question type and data sources. We describe key statistics in Tab. 2. Test examples in QAMPARI have 13.23 answers on average and a median of 7 answers. For comparison, the number of answers per question is substantially higher than in AmbigQA (Min et al., 2020), where the median is 2. On average, simple questions have more answers than complex ones while being shorter in length. We note that since test and development questions were manually re-phrased by annotators they are generally shorter than training questions.

Figure 3a presents a binned distribution of the number of answers per question on the development and test sets. Roughly half of the questions have over 8 answers, with 20% having more than 15 answers, and 3.5% with over 50 answer.

Extended set As discussed in §2.5, we manually validate each answer in QAMPARI is supported

		Total	Simp. WD	Simp. WP	Inter. WD	Comp. WD	Comp. WP
# Questions	train	61,911	28,574	-	2,301	25,200	5,836
	dev + test	2,000	830	170	400	400	200
Mean # Answers	train	13.25	16.65	-	9.19	9.74	13.35
	dev + test	13.23	15.69	23.84	8.94	8.77	11.51
Median # Answers	train	8.0	9.0	-	7.0	7.0	8.0
	dev + test	7.0	7.5	17.0	7.0	6.0	7.0
Mean Question len.	train	12.69	8.78	-	16.69	15.18	19.47
	dev + test	9.51	7.91	8.61	11.65	10.35	10.99

Table 2: QAMPARI questions breakdown by their type (**Simp**le, **Inter**section or **Compo**sition questions) and underlying data source (**WD** for Wikidata, **WP** for Wikipedia tables).

by sentences from Wikipedia. However, Wikipedia might contain additional correct answers. To alleviate this issue, we manually annotate additional gold answers for a subset of the questions, and name it the *ExtendedSet*. We randomly sampled 200 questions from the test set and had an author manually annotate as many additional answers as possible in 12 minutes per question. This annotation is not guaranteed to be complete, since that would require going over all of Wikipedia. Moreover, a small fraction of questions have hundreds of gold answers (*"Who worked for Burton F. C?"*) and would incur hours of annotation, which is too expensive. This is similar to work in open information extraction (Vo and Bagheri, 2017), where creating the full gold set of triples is not feasible.

Fig. 3 plots the number of added answers per question on the extended set. In 30% of the questions, we did not add any answer, and the median/average/maximum number of added answers are 2/3.13/16 respectively. Evaluation on the test set and the extended set in §4.3 shows that model precision on the extended set is somewhat higher, but does not alter model ranking, illustrating the reliability our test set.

4 Experimental Evaluation

4.1 Models

Retriever For retrieval, we experiment with both sparse and dense retrieval models on Wikipedia. As discussed in §2, we chunk Wikipedia into passages of consecutive sentences, using NLTK's sentence tokenizer, where each passage is 100 words at most. For both our retrievers, we evaluate retrieval accuracy of the top-200 passages returned per question.

We use BM25 (Robertson and Zaragoza, 2009) as a strong sparse retrieval model. BM25 scores question-passage pairs based on their lexical similarity. It has been shown that BM25 is notoriously hard to beat using unsupervised retrieval methods (Izacard et al., 2021; Ram et al., 2022), and achieves comparable performance to that of supervised methods (Thakur et al., 2021). As our dense retriever we use an off-the-shelf DPR (Karpukhin et al., 2020) trained on NQ. We leave training a retriever on QAMPARI for future work.

Reader We investigate two readers – a Passage-Independent Generator (PIG), which reads each passage independently (a-là RAG (Lewis et al., 2020)), and a Fusion-in-Decoder (FiD) model (Izacard and Grave, 2021), which reads multiple passages simultaneously.

PIG is an encoder-decoder model that takes each of the retrieved passages as input and decodes a single answer or outputs *"Not Relevant"* to indicate there is no answer. The final output is the union of all decoded answers across retrieved passages. We initialize PIG with T5-large (Raffel et al., 2019) and train with standard maximum likelihood. We use evidence passages as positive examples and the top scoring retrieved passage that is not an evidence passage that does not contain an answer or its aliases as a negative example.

FiD encodes each of the retrieved passages along with the input question. Its decoder then attends to the encoded representation and outputs a list of answers. We initialize FiD using a pretrained T5-Large model (Raffel et al., 2019) and train with standard maximum likelihood.

FiD is computationally expensive, as its decoder attends to a large number of encoded tokens and the generated output is long. Thus, we can only fit the top-50 passages returned by the retriever on a single A100 GPU.

Closed-book Question Answering We also experiment with a closed-book setting, where the QA

		QAMPARI					NQ
		Recall	Precision	F_1	%Recall≥0.8	%F_1 ≥0.5	EM
FiD-BM25	QO	25.1	36.8	28.3	6.8	24.2	30.4
	MT	26.9	**37.7**	29.7	7.4	25.6	47.2
FiD-DPR	QO	3.6	20.4	5.7	0	1.2	25.8
	MT	5.3	21.3	7.7	0.1	2.4	**52.8**
PIG-BM25	QO	43.1	30.7	**31.0**	26.7	**26**	13.1
	MT	**47.9**	28.2	30.5	**31.2**	22.3	31.9
PIG-DPR	QO	18.9	1.8	3.0	4.2	0	9.3
	MT	18.0	1.9	3.1	4.2	0	37.3
Closed book	QO	1.7	7.3	2.6	0	0.3	6.9

Table 3: QAMPARI & NQ test results. **QO**: models trained on QAMPARI only; **MT**: Multi-task training with NQ.

model generates answers from knowledge encoded in its parameters without any evidence passages. We initialize our closed-book QA model with T5-SSM with 3B parameters (Roberts et al., 2020), and train it with standard maximum likelihood – the question is provided as input, and the model is trained to generate the gold set of answers.

	ARecall@K		ERecall@K	
	BM25	DPR-NQ	BM25	DPR-NQ
K=10	24.6	11.1	11.1	2.0
K=25	37.4	16.1	28.4	3.4
K=50	46.6	20.7	38.7	4.5
K=100	54.6	25.2	47.6	5.8
K=200	61.0	29.7	55.6	7.3

Table 4: Retriever test results.

4.2 Experimental Setup

We created QAMPARI as a benchmark to be evaluated alongside other ODQA benchmarks such as NQ. Since QAMPARI is semi-automatically generated, one can develop models tailored for QAMPARI, but our goal is to have a single model that performs well across a wide variety of question types. Thus, we train and test models on QAMPARI only, but also in a multi-task setup, on both NQ and QAMPARI. See App. C for more details.

Our main metrics are recall, precision, and F_1. Specifically, for a test example $(q, \mathcal{P}, \mathcal{A})$, and a predicted set of answers $\mathcal{A}_{\text{pred}}$, recall, precision, and F_1 are computed in the typical manner, comparing $\mathcal{A}$ and $\mathcal{A}_{\text{pred}}$, allowing for aliases, i.e., a gold answer is considered covered if it or one of its aliases appears as a predicted answer. A model score is average recall, precision, and F_1 across examples. To get a sense of the average accuracy across examples we measure the fraction of examples where F_1 is at least 0.5 (%F_1 ≥0.5) and the fraction where recall is at least 0.8 (%Recall≥0.8). For NQ, we use the standard exact match (EM) metric.

We evaluate the retriever with RECALL@K, that is, the fraction of answers that appear in the top-K retrieved passages, averaged across examples. This metric comes in two flavors: (a) Answer RECALL@K (ARECALL@K): for every gold answer whether it or one of its aliases appear in the top-K retrieved passages. This is a loose metric since an answer can appear even if the evidence does not support the answer; (b) Evidence RECALL@K (ERECALL@K): since we have evidence paragraphs for every answer, we consider for every gold answer the fraction of evidence passages in the top-K retrieved passages. This is a strict metric since an answer can sometimes be answered by passages other than the ones we identified.

4.3 Results

Tab. 4 presents the passage retrieval results on QAMPARI test set. For our loose metric (ARecall@K), even when K=200 the average recall for BM25 and DPR-NQ are 61.0 and 29.7, respectively. As for our strict evaluation (ERecall@K), results are unsurprisingly lower. BM25 retrieves 55.6 of the evidence passages when K=200, while DPR-NQ retrieves just 7.3 of the evidence passages. Overall, DPR trained on NQ does not generalize well to QAMPARI compared to BM25. Sciavolino et al. (2021a) have shown that, when tested on questions with *rare entities*, DPR-NQ performs much worse than BM25. We hypothesize that rare entities in QAMPARI questions may account for DPR-NQ's lesser performance.

Tab. 3 lists results on the test sets of QAMPARI and NQ. Overall, performance on QAMPARI is low. FiD is precision-oriented with FiD-BM25

	Recall	Precision	F_1
Wikidata simple	21.3	30.7	23.1
Wikidata intersection	37.0	47.1	40.0
Wikidata composition	18.6	32.4	22.2
Wikipedia simple	9.1	20.6	11.5
Wikipedia composition	31.2	37.4	32.7

Table 5: Question type analysis of FiD-BM25, trained in MT setup on QAMPARI development set.

achieving precision of 37.7. PIG is recall-oriented, with PIG-BM25 achieving recall of 47.9. F1 results are relatively similar across PIG and FiD with a slight advantage for PIG-BM25 with an F1 of 31.0. When training on both NQ and QAMPARI in a multi-task (MT) setting, model results on NQ (both with BM25 and DPR) are similar to those reported by Izacard and Grave (2021). Results on QAMPARI are much lower than NQ, despite the fact that NQ's EM evaluation metric is much more strict than the soft metrics used for QAMPARI, further illustrating the challenge in answering multi-answer questions. Results using DPR are significantly lower than those using BM25. This corresponds to our findings in Tab. 4 that DPR-NQ retriever performs significantly worse than BM25.

Closed book performance is low with an F_1 of 2.6 for QAMPARI. We believe this is due to the train/test relation-based split (§2.4). This guarantees that there is no overlap between train and test questions. Lewis et al. (2021) have shown that mitigating such train-test overlap causes a drop in QA performance, with a drastic drop being observed in closed-book models.

***ExtendedSet* results** We report results for FiD and PIG on the *ExtendedSet* (see §3) in §D. As expected, considering additional correct answers improves the precision of all models. Since changes to recall are small, the overall F_1 is higher when considering manual annotations. Importantly, ranking across models does not change, and the absolute performance remains low, suggesting that our test set can be safely used for evaluation.

Question type analysis We break test performance of FiD-BM25 (MT) by question type (Tab. 5). Surprisingly, performance on simple questions is lower than complex questions, and intersection questions seem easiest. Possible explanations are: (a) simple questions have more answers (see Tab. 2), which makes them harder, and (b) models can predict the right answer with only one of the evidence passages, due to "shortcuts" (Chen and Durrett, 2019), or parametric knowledge (Longpre et al., 2021).

5 Related work

ODQA tasks have largely been dedicated to questions whose answer is located in a single phrase (Berant et al., 2013; Baudiš and Šedivý, 2015; Rajpurkar et al., 2016; Joshi et al., 2017; Kwiatkowski et al., 2019; Sciavolino et al., 2021b). Questions in multi-hop QA datasets require reasoning over multiple evidence of text in order to retrieve their answer. However, the answers to such datasets are short, no longer than a single phrase (Welbl et al., 2018; Yang et al., 2018; Thorne et al., 2018). Recently, the AmbigQA dataset (Min et al., 2020) introduced ambiguous questions that have multiple answers depending on their interpretation. In contrast, questions in QAMPARI are unambiguous and contain significantly more answers. Similar to our work, questions in WikiNLDB (Thorne et al., 2021) require that models reason over sets of facts. However, its retrieval is restricted to a corpus of 1,000 model-generated sentences. Contrastly, we use Wikipedia as our open-domain corpus, making our setup more realistic and challenging.

Currently, retrieve-and-read models are the prevailing approach in ODQA (Chen et al., 2017; Yang et al., 2019; Lee et al., 2019; Izacard and Grave, 2021; Sachan et al., 2021). When the number of evidence passages is large, such models must fetch all relevant passages in order to generate the answer. Closed-book QA is an alternative approach, where information is encoded in model parameters (Roberts et al., 2020; Tay et al., 2022) however, it entails using high-capacity models. A less explored approach, potentially suitable for large answer sets, is that of virtual knowledge-bases (KBs), which encode a corpus into a differentiable KB that is amenable for retrieval and logical operations (Sun et al., 2021; Dhingra et al., 2020).

6 Conclusions

We release QAMPARI, a dataset targeting ODQA models ability to answer questions with many answers, and show that it is challenging for current state-of-the-art models. Multi-answer questions are an integral part of ODQA that has thus far been neglected. QAMPARI will aid to develop models that answer a wide range of question types, including single- and multi-answer questions.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Petr Baudiš and Jan Šedivý. 2015. Modeling of the question answering task in the YodaQA system. In *Proceedings of the 6th International Conference on Experimental IR Meets Multilinguality, Multimodality, and Interaction - Volume 9283*, CLEF'15, page 222–228, Berlin, Heidelberg. Springer-Verlag.

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*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Eric Brill, Susan Dumais, and Michele Banko. 2002. An analysis of the AskMSR question-answering system. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 257–264. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4026–4032, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *ArXiv preprint*, abs/2112.09118.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick 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)*, pages 6769–6781, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference* on Empirical Methods in Natural Language Processing, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv preprint*, abs/1910.10683.

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

Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. Learning to retrieve passages without supervision. In *North American Association for Computational Linguistics (NAACL)*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

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

Devendra Singh Sachan, Siva Reddy, William L. Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. In *Advances in Neural Information Processing Systems*.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021a. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021b. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haitian Sun, Patrick Verga, Bhuwan Dhingra, Ruslan Salakhutdinov, and William W. Cohen. 2021. Reasoning over virtual knowledge bases with open predicate relations. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9966–9977. PMLR.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodalqa: complex question answering over text, tables and images. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yi Tay, Vinh Q Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. 2022. Transformer memory as a differentiable search index. *ArXiv preprint*, abs/2202.06991.

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)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, and Alon Halevy. 2021. Database reasoning over text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3091–3104, Online. Association for Computational Linguistics.

Duc-Thuan Vo and Ebrahim Bagheri. 2017. Open information extraction. *Encyclopedia with semantic computing and Robotic intelligence*, 1(01):1630003.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece. European Language Resources Association (ELRA).

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342, Beijing, China. Association for Computational Linguistics.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

A Simple Relations

In Tab. 6. we gathered all the 135 relations we used to create our simple questions. The 423 relations used to create our composition questions can be found in our code base.

B Crowdsourcing Validation

Fig. 4 shows two screenshots of the task crowdsourcing workers performed.

C Experimental setup details

For both readers (FiD and PIG), we used T5-large which has 770 million parameters. We used an A100 to train both of them, FiD with a batch size of 8 and PIG with a batch size of 512 for a single GPU. We trained each of them for around 48 hours on two GPUs.

We performed an hyper parameter search around the learning rate, the number of training steps, the ratio of positive to negative (for PIG) and the number of times an NQ example will appear in each epoch (for multi task). Tab. 7 presents the parameters of the reported results.

We report the results of a single run with seed 0.

D *ExtendedSet* Results

In Tab. 8 we present results analogous to those in Tab. 3 for the *ExtendedSet* with BM25. Precision improves by 5-6 points across models, while recall changes are smaller leading to an overall increase in F_1. Nevertheless changes are not dramatic and model ranking remains constant, suggesting the full test set can be safely used.

E Development Set Results

In Tab. 9 we present results analogous to those in Tab. 3 for the development set.

is a	has author	located in	language	occupation
sex or gender	country of citizenship	part of	place of birth	located in
educated at	language spoken, written or signed	has part	played the sport	employer
genre	position held	cast member	country of origin	award received
place of death	made from material	creator	has participant	depicts
maintained by	operator	performer	member of political party	owned by
religion	headquarter location	participant	member of	position played
original language	competition class	publisher	role	record label
work location	director	doctoral advisor	residence	native language
place of publication	medical condition	winner	field of work	form or work
conflict	place of burial	instrument	composer	league
screenwriter	distribution format	producer	sponsor	ethnicity
voice actor	distributed by	participating team	academic degree	manufacturer
architectural style	fabrication method	present in work	production company	cause of death
military branch	manner of death	industry	director of photography	narrative location
original broadcaster	organizer	student of	location of creation	located in or next to body of water
architect	archives at	nominated for	country of registry	allegiance
movement	voice actor	noble title	based on	dedicated to
legislated by	location of formation	developer	contributor to creative work or subject	lyrics written by
located in protected area	tracklist	editor	presenter	religious order
from narrative universe	location of discovery	media franchise	commissioned by	political ideology
commemorates	port of registry	influenced by	indigenous to	operating area
translator	brand	interested in	designed by	illustrator
vessel class	costume designer	drafted by	coach of sports team	convicted of
scenographer	culture	significant place	executive producer	represented by
broadcast by	investor	cover art by	home port	collection creator
armament	inspired by	first appearance	choreographer	animator
source of energy	musical conductor	adapted by	sound designer	has written for
academic major	ratified by	business model	worshipped by	narrator
partnership with	colorist	art director	has work in the collection	military rank

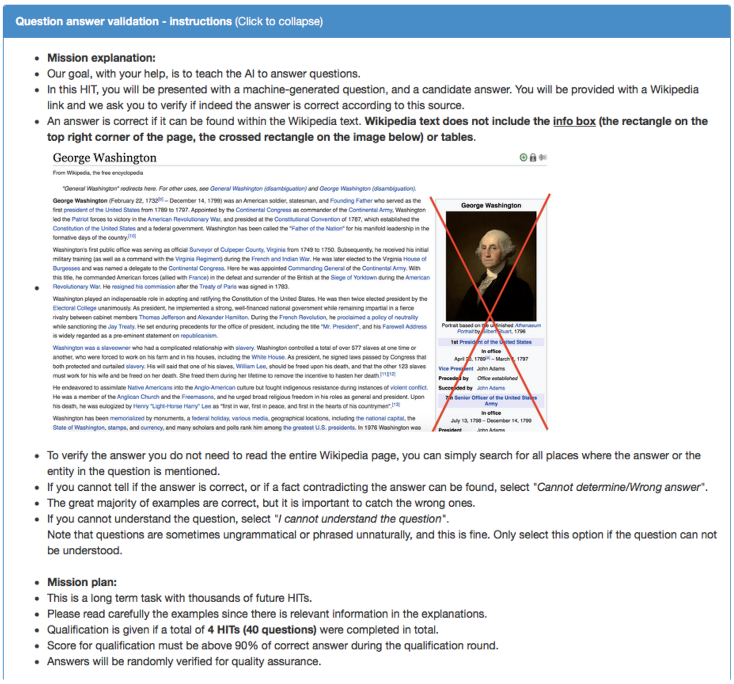
Table 6: Simple relations

		Learning rate	# steps	pos. to neg.	# NQ examples
FiD-BM25	QO	0.00005	90k	-	-
	MT	0.00005	190k	-	2
FiD-DPR	QO	0.00005	85k	-	-
	MT	0.00005	190k	-	2
PIG-BM25	QO	0.000001	60k	1	-
	MT	0.000001	75k	1	1
PIG-DPR	QO	0.000001	60k	1	-
	MT	0.000001	75k	1	1
Closed book	QO	0.0001	95k	-	-

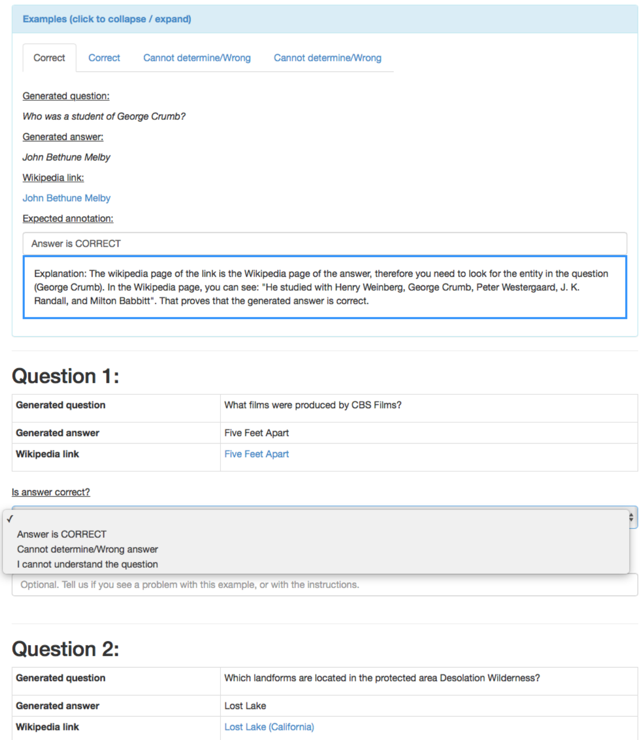
Table 7: Hyper parameters used for reported results.

			QAMPARI				
			Recall	**Precision**	**F_1**	**%Recall≥0.8**	**%F_1 ≥0.5**
FiD-BM25	**QO**	w.o. annotations	20.5	34.6	24.3	4.0	19.6
		w. annotations	23.3	40.6	27.8	4.5	25.1
FiD-BM25	**MT**	w.o. annotations	22.8	**37.0**	26.8	4.5	20.6
		w. annotations	25.7	**42.9**	30.6	5.0	24.6
PIG-BM25	**QO**	w.o. annotations	45.1	28.9	**30.7**	27.5	**23**
		w. annotations	42.7	33.6	32.8	24	**29.5**
PIG-BM25	**MT**	w.o. annotations	**49.3**	27.9	**30.7**	**31.5**	20.5
		w. annotations	**47.1**	33.1	**33.2**	**27**	26

Table 8: QAMPARI *ExtendedSet* results with (w.) and without (w.o.) the additional manual annotations. The best results with and without annotations are bolded. **QO**: models trained on QAMPARI only; **MT**: Multi-task training with NQ.



(a) Instructions



(b) Task

Figure 4: Screenshots from crowdsourcing task.

		QAMPARI				
		Recall	**Precision**	$\mathbf{F_1}$	**%Recall≥0.8**	$\mathbf{\%F_1 \geq 0.5}$
FiD-BM25	QO	23.3	35.6	26.3	5.9	22.7
	MT	23.9	34.2	26.3	6.0	22.4
FiD-DPR	QO	3.6	19.9	5.7	0	1.4
	MT	4.7	21.1	7.0	0.1	1.6
PIG-BM25	QO	41.4	26.4	28.0	25.3	21.0
	MT	43.7	26.9	28.9	26.6	22.0
FiD-DPR	QO	20.3	2.1	3.4	4.2	0
	MT	19.8	2.3	3.8	3.6	0.1
Closed book	QO	2.4	7.2	3.1	0.1	0.7

Table 9: QAMPARI development results. **QO**: models trained on QAMPARI only; **MT**: Multi-task training with NQ.