Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis[†‡], **Ethan Perez**[⋆],

Aleksandra Piktus[†], **Fabio Petroni**[†], **Vladimir Karpukhin**[†], **Naman Goyal**[†], **Heinrich Küttler**[†],

Mike Lewis[†], **Wen-tau Yih**[†], **Tim Rocktäschel**[†‡], **Sebastian Riedel**[†‡], **Douwe Kiela**[†]

[†]Facebook AI Research; [‡]University College London; [⋆]New York University;
plewis@fb.com

Abstract

Large pre-trained language models have been shown to store factual knowledge in their parameters, and achieve state-of-the-art results when fine-tuned on downstream NLP tasks. However, their ability to access and precisely manipulate knowledge is still limited, and hence on knowledge-intensive tasks, their performance lags behind task-specific architectures. Additionally, providing provenance for their decisions and updating their world knowledge remain open research problems. Pre-trained models with a differentiable access mechanism to explicit non-parametric memory can overcome this issue, but have so far been only investigated for extractive downstream tasks. We explore a general-purpose fine-tuning recipe for retrieval-augmented generation (RAG) — models which combine pre-trained parametric and non-parametric memory for language generation. We introduce RAG models where the parametric memory is a pre-trained seq2seq model and the non-parametric memory is a dense vector index of Wikipedia, accessed with a pre-trained neural retriever. We compare two RAG formulations, one which conditions on the same retrieved passages across the whole generated sequence, the other can use different passages per token. We fine-tune and evaluate our models on a wide range of knowledge-intensive NLP tasks and set the state-of-the-art on three open domain QA tasks, outperforming parametric seq2seq models and task-specific retrieve-and-extract architectures. For language generation tasks, we find that RAG models generate more specific, diverse and factual language than a state-of-the-art parametric-only seq2seq baseline.

1 Introduction

Pre-trained neural language models have been shown to learn a substantial amount of in-depth knowledge from data [41]. They can do so without any access to an external memory, as a parameterized implicit knowledge base [45, 46]. While this development is exciting, such models do have downsides: They cannot easily expand or revise their memory, can't straightforwardly provide insight into their predictions, and may produce "hallucinations" [34]. Hybrid models that combine parametric memory with non-parametric (i.e., retrieval-based) memories [18, 22, 42] can address some of these issues because knowledge can be directly revised and expanded, and its access can be inspected and interpreted. REALM [18] and ORQA [27], two recently introduced models that combine masked language models [8] with a differentiable retriever, have shown promising results,
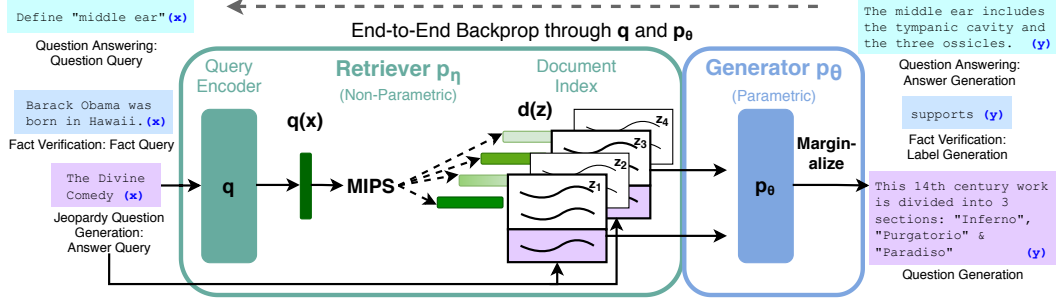
Figure 1: An overview of retrieval-augmented generation (RAG). We combine a pre-trained retriever (**Query Encoder + Document Index**) with a pre-trained encoder-decoder (**Generator**) and fine-tune end-to-end. For some query x, we use Maximum Inner Product Search (MIPS) to find the top-K most relevant documents of all documents z_i. To make the final prediction y, we treat z as a latent variable and marginalize over the encoder-decoder predictions given different documents.

but have only explored open-domain extractive question answering. Here, we bring hybrid parametric and non-parametric memory to the "workhorse of NLP," i.e. sequence-to-sequence (seq2seq) models.

We endow pre-trained, parametric-memory generation models with a non-parametric memory through a general-purpose fine-tuning approach which we refer to as retrieval-augmented generation (RAG). We build RAG models where the parametric memory is a pre-trained generative seq2seq transformer, and the non-parametric memory is a dense vector index of Wikipedia, accessed using a pre-trained neural retriever. We combine these components in an end-to-end probabilistic model; the document retriever (Dense Passage Retriever [22], henceforth DPR) provides latent documents conditioned on the input, and the seq2seq model (BART [28]) then conditions on both these latent documents and the input to generate the output. We marginalize the latent variables through a top-K approximation, either on a per answer basis (assuming the same document is responsible for all tokens) or a per answer token basis (assuming different documents can be responsible for different tokens). Just like T5 [45] or BART, RAG can be fine-tuned on any seq2seq task, whereby both the sequence generator and retriever are jointly learned.

There has been extensive previous work proposing architectures to enrich systems with non-parametric memory which are trained from scratch for specific tasks—e.g. in memory networks [58, 49], stack-augmented networks [21] and memory layers for transformers [26]. In contrast, we explore a setting where both parametric and non-parametric memory components are pre-trained and pre-loaded with extensive knowledge. Crucially, by using pre-trained knowledge-access mechanisms, the ability to access knowledge is present without additional training.

Our results highlight the benefits of combining parametric and non-parametric memory with generation for knowledge-intensive tasks. Our RAG models achieve state-of-the-art results on open Natural Questions [25], WebQuestions [3] and CuratedTrec [2] and strongly outperform recent approaches that use specialised pre-training objectives on TriviaQA [20]. Despite these being extractive tasks, we find that unconstrained generation outperforms previous extractive approaches. For knowledge-intensive generation, we experiment with MS-MARCO [1] and Jeopardy question generation, and we find that our models generate responses that are more factual, specific, and diverse than a BART baseline. For the FEVER [50] fact verification task, we achieve results within 4% of sophisticated, state-of-the-art pipeline models which use strong supervision. Finally, we show that the non-parametric memory can be replaced in order to control generation, demonstrating a simple mechanism to update the knowledge that the model uses as facts about the world change.

2 Methods

We explore RAG models which use the input sequence x to retrieve text passages z and use these passages as additional context when generating the target sequence y. As shown in Figure 1, our models leverage two components: (i) a retriever $p_\eta(z|x)$ with parameters η that returns (top-K truncated) distributions over text passages given a query x and (ii) a generator $p_\theta(y_i|x, z, y_{1:i-1})$

parametrized by θ that generates a current token based on a context of the previous $i-1$ tokens $y_{1:i-1}$, the original input x and a retrieved passage z.

To train the retriever and generator end-to-end, we treat the retrieved document as a latent variable. We propose two models that marginalize over the latent documents in different ways to produce a distribution over generated text. In one approach, *RAG-Sequence*, the model uses the same document to predict each target token. In the other approach, *RAG-Token*, the model can predict each target token based on a different document. In what follows, we formally introduce both models and then describe the p_η and p_θ components, as well as the training and decoding procedure in more detail.

2.1 Models

RAG-Sequence Model   The RAG-Sequence model uses the same retrieved document to generate the complete *sequence*. Technically, it treats the retrieved passage as a single latent variable that is marginalized to get the seq2seq probability $p(y|x)$ via a top-K approximation,

$$p_{\text{RAG-Sequence}}(y|x) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) \prod_i^N p_\theta(y_i|x, z, y_{1:i-1}).$$

RAG-Token Model   In the RAG-Token model we can draw a different latent passage for each target *token* and marginalize accordingly. This allows the generator to choose content from several documents when producing an answer. Formally, we define:

$$p_{\text{RAG-Token}}(y|x) = \prod_i^N \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z_i|x) p_\theta(y_i|x, z_i, y_{1:i-1}).$$

Finally, we note that RAG can be used for sequence classification tasks by considering the target class as a target sequence of length one, in which case RAG-Sequence and RAG-Token are equivalent.

2.2 Retriever: DPR

The retrieval component $p_\eta(z|x)$ is based on DPR [22]. DPR follows a bi-encoder architecture:

$$p_\eta(z|x) \propto \exp \langle \mathbf{d}(z), \mathbf{q}(x) \rangle$$

where $\mathbf{d}(z)$ is a dense representation of the document produced by a BERT$_{\text{BASE}}$ transformer [8], and $\mathbf{q}(x)$ a representation of the query by another BERT$_{\text{BASE}}$ transformer with a different set of parameters.

To efficiently calculate top-k$(p_\eta(\cdot|x))$, the list of k elements z with highest prior probability $p_\eta(z|x)$, DPR employs a Maximum Inner Product Search (MIPS) index provided by the FAISS library [19].

For non-parametric pre-trained memory, we use a pre-trained bi-encoder from [22] to both initialize our retriever and to build the document index. This retriever was trained to retrieve documents which contain answers to TriviaQA [20] questions and Natural Questions [25].

2.3 Generator: BART

The generator component $p_\theta(y_i|x, z, y_{1:i-1})$ could be modelled using any encoder-decoder. We use BART-large [28], a pre-trained seq2seq transformer [52] with 400M parameters. To combine the input x with the retrieved content z when generating from BART, we simply concatenate them.

BART was pre-trained using a denoising objective and a variety of different noising functions. It has obtained state-of-the-art results on a diverse set of generation tasks and outperforms comparably-sized T5 models [28]. We refer to the BART generator parameters θ as the *parametric memory* henceforth.

2.4 Training

We jointly train the retriever and generator components without any direct supervision on what document should be retrieved. Given a fine-tuning training corpus of input/output pairs (x_j, y_j), we

minimize the negative marginal log-likelihood of each target, $\sum_j - \log p(y_j | x_j)$ using stochastic gradient descent with Adam [24]. Updating the document encoder during training is costly as it requires the document index to be periodically updated as REALM does during pre-training [18]. We do not find this step necessary for strong performance, and we keep the document encoder (and index) fixed, only fine-tuning the query encoder and the generator.

2.5 Decoding

At test/decoding time, RAG-Sequence and RAG-Token require different ways to approximate $\arg\max_y p(y|x)$.

RAG-Token The RAG-Token model can be seen as a standard, autoregressive, seq2seq generator with transition probability:

$$p'_\theta(y_i | x, y_{1:i-1}) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z_i|x) p_\theta(y_i|x, z_i, y_{1:i-1})$$

To decode, we can plug $p'_\theta(y_i | x, y_{1:i-1})$ into a standard beam decoder.

RAG-Sequence The likelihood $p(y|x)$ does not break into a conventional per-token likelihood for the RAG-Sequence, and hence we cannot solve it with a single beam search pass. Instead, we run beam search for each candidate document z, scoring each hypothesis using $p_\theta(y_i|x, z, y_{1:i-1})$. This yields a set of hypotheses Y of which some might not have appeared in the beams of all documents. To estimate the probability of an hypothesis y across all beams, we run an additional forward pass for each document z for which y does not appear in the beam, multiply the generator score with $p_\eta(z|x)$ and then sum up the probabilities across beams for the marginals. We refer to this decoding procedure as "Thorough Decoding."

For longer output sequences, $|Y|$ can become large, requiring many forward passes. For more efficient decoding, we can make a further approximation that $p_\theta(y|x, z_i) \approx 0$ where y was not generated during beam search from x, z_i. This avoids the need to run additional forward passes once the candidate set Y has been generated. We refer to this decoding procedure as "Fast Decoding".

3 Experiments

We experiment with RAG in a wide range of knowledge-intensive tasks. For all experiments, we use a single Wikipedia dump for our non-parametric knowledge source. Following Lee et al. [27] and Karpukhin et al. [22], we use the December 2018 dump. Each Wikipedia article is split into disjoint 100-word chunks, to make a total of 21,015,324 documents.[1] We use the DPR document encoder to compute document embeddings for each document, and we build a single MIPS index using FAISS [19] using Hierarchical Navigable Small World approximation for efficient retrieval [33], which is then used for all experiments. During training, we retrieve the top k documents for each query, where we consider $k \in \{5, 10\}$. We determine k for test time using validation data. In the remainder of this section, we will discuss the experimental details for each of these task settings.

3.1 Open-domain Question Answering

Open-domain QA is an important real-world NLP application and is often used as test-bed for knowledge-intensive tasks [18]. We tackle open-domain QA by treating questions and answers as simple input-output text pairs (x, y), and we train RAG by directly minimizing the negative log-likelihood of answers. We compare our results to the popular extractive QA paradigm [5, 7, 27, 22], where answers are extracted as spans from retrieved documents, relying primarily on non-parametric knowledge. In addition, we also compare to "Closed-Book QA" approaches [46], which, like RAG, generate answers, but do not exploit latent retrieval, instead relying purely on parametric knowledge.

We consider four popular open-domain QA datasets: Natural Questions (NQ) [25], TriviaQA (TQA) [20]. WebQuestions (WQ) [3] and CuratedTrec (CT) [2]. The answers for CuratedTrec are given in the form of regular expressions, which has been cited as a reason why it is unsuitable

---

[1]The reader is referred to Karpukhin et al. [22] for further details on how Wikipedia is pre-processed.

for answer-generation models [18]. To overcome this, we use a pre-processing step where we first retrieve the top 1000 documents for each query, and use the answer that most frequently matches the regex pattern as the supervision target. If no matches are found, we resort to a simple heuristic: generate all possible permutations for each regex, replacing non-deterministic symbols in the regex nested tree structure with a whitespace. As CuratedTrec and WebQuestions are small datasets, we follow DPR [22] by initializing CuratedTrec and WebQuestions models with our Natural Questions RAG model.

We use the same training/dev/testing splitting method as in previous work [27, 22] and report the standard Exact Match (EM) metric. For TriviaQA, in order to compare to T5 [46], we do an additional test evaluation on the TriviaQA Wiki test set.

3.2 Abstractive Question Answering

Because RAG leverages an encoder-decoder model, it can go beyond extractive question-answering and answer questions with free-form, abstractive text generation. To test RAG's ability to generate natural language responses in a knowledge-intensive setting, we use the MS-MARCO Natural Language Generation task v2.1 [38]. This task consists of natural language questions submitted to a search engine, ten snippets retrieved from a search engine for each question, and a full sentence natural language answer annotated from these retrieved passages.

As we are interested in models that can perform their own latent retrieval, we do not use the supplied passages, only the questions and answers, thus treating MS-MARCO as an open-domain abstractive question answering task. MS-MARCO does contain some questions that cannot be answered in a way that matches the reference answer without access to the context passages, such as "What is the weather in volcano, CA?" so we note that performance on Open-MSMARCO will be lower than models that do use these gold context passages.

We further note that there are questions in MS-MARCO that cannot be answered using a Wikipedia knowledge source alone. In these cases, RAG can rely on the parametric implicit knowledge in its BART parameters in order to generate commonsense responses.

3.3 Jeopardy Question Generation

In order to further evaluate RAG's generation abilities in a non-question answering setting, we propose to study Open-domain question generation. Rather than repurpose questions from standard open-domain QA tasks, which typically consist of short and simple questions, we instead propose to study the more demanding task of generating of Jeopardy questions. Jeopardy is an unusual format that consists of trying to guess an entity from a fact about that entity. For example, "The World Cup" is the answer to the jeopardy question "In 1986 Mexico scored as the first country to host this international sports competition twice." As Jeopardy "questions" are precise, factual statements, generating Jeopardy-style questions conditioned on the answer entity they refer to constitutes a challenging knowledge-intensive generation task.

We use the raw Jeopardy data and splits from SearchQA [10], consisting of 97,391 training, 13,713 development, and 26,848 test datapoints. As this is a new task, we also train a BART system to compare RAG to. Following [60], we evaluate generations using the SQuAD-tuned Q-BLEU-1 metric [37]. Q-BLEU-1 is a variant of BLEU-1 which puts a higher weight on the matching entities, and has higher correlation with human judgment for question generation compared to standard word overlap metrics.

As automatic metrics can be unreliable, especially on such open-ended tasks, we also perform a human evaluation of generations. We run two evaluations, one to assess the factuality of generations, and one to assess specificity. We follow the recent best-practice of performing a pairwise comparative evaluation between two systems [30]. Assessors are shown an answer entity and two generated questions about that entity, one from BART and one from RAG. They are then asked to pick one of four possible options—Sentence A is better, Sentence B is better, both are correct or neither is good.

3.4 Fact Verification

FEVER [50] is a fact verification dataset that involves classifying whether a natural language claim is supported or refuted by Wikipedia, or whether there is not enough information to decide. The

5

	Model	NQ	TQA	WQ	CT
Closed-Book	T5-11B [46]	34.5	- /50.1	37.4	-
	T5-11B + SSM [46]	36.6	- /60.5	44.7	-
Open-Book	REALM [18]	40.4	- / -	40.7	46.8
	DPR [22]	41.5	**57.9**/ -	41.1	50.6
	RAG-Token	44.1	55.2/66.1	**45.5**	50.0
	RAG-Sequence	**44.5**	56.1/**68.0**	45.2	**52.2**

Table 1: Open-Domain QA Test Scores. For TQA, the left column uses the test split commonly used in Open-Domain QA. The right column uses the hidden TQA Wiki test split. See Appendix B for further information.

Model	Jeopardy QGen		MS-MARCO		FEVER-3	FEVER-2
	B-1	QB-1	R-L	B-1	Label Accuracy	
SotA	-	-	**49.8***	**49.9***	**76.8**	**92.2***
BART	15.1	19.7	38.2	41.6	64.0	81.1
RAG-Token	**17.3**	**22.2**	40.1	41.5	72.5	89.5
RAG-Sequence	14.7	21.4	<u>40.8</u>	<u>44.2</u>		<u>89.5</u>

Table 2: Generation and classification task Test Scores. SotA for MS-MARCO is [4], FEVER-3 is [61] and FEVER-2 is [51] * Uses gold context/evidence, best-performing model without gold access underlined. As FEVER is a classification dataset, RAG-Token and RAG-Sequence are equivalent.

task requires retrieving evidence from Wikipedia relating to the claim and then reasoning about the retrieved evidence to classify whether the claim is true, false, or unverifiable from Wikipedia alone. FEVER is a retrieval problem coupled with an entailment reasoning task. It also provides a good test bed for exploring the RAG models' ability to handle classification rather than generation.

We map FEVER class labels (supports, refutes, or not enough info) to single output tokens and directly train with claim-class pairs. Crucially, unlike most other approaches to FEVER, we do not use supervision on retrieved evidence. We explore two different FEVER variants: the standard 3-way classification task (supports/refutes/not enough info) and the 2-way FEVER (supports/refutes) task studied in Thorne and Vlachos [51]. In both cases we report label accuracy.

3.5 Implementation Details

For Open-domain QA we report test numbers using 15 retrieved documents for RAG-Token models. For RAG-Sequence models, we report test results using 50 retrieved documents, and we use the Thorough Decoding approach since answers are generally short. We use greedy decoding for QA as we did not find beam search improved results. For Open-MSMarco and Jeopardy question generation, we report test numbers using ten retrieved documents for both RAG-Token and RAG-Sequence, and we also train a BART-large model as a baseline. We use a beam size of four, and use the Fast Decoding approach for RAG-Sequence models, as Thorough Decoding did not improve performance.

4 Results

4.1 Open-domain Question Answering

Table 1 shows results for RAG along with recent state-of-the-art models. On all four open-domain QA tasks, RAG sets a new state-of-the-art (in the case of TQA only on the T5-comparable split).

RAG combines the generation flexibility of the "closed-book" (parametric only) approaches and the performance of "open-book" retrieval-based approaches. Unlike REALM and T5+SSM, RAG enjoys strong results without expensive specialized "salient span masking" pre-training [18], relying on off-the-shelf components. It is worth noting that RAG's retriever is initialized using DPR's retriever,

	BART better	RAG-Token better	Both good	Both poor	No Majority
Factuality	7.1%	**42.7%**	11.7%	17.7%	20.8%
Specificity	16.8%	**37.4%**	18.8%	6.9%	20.1%

Table 3: Human assessments for the Jeopardy Question Generation Task.

which does use retrieval supervision on Natural Questions and TriviaQA. RAG compares favourably to DPR QA system on open-domain QA, which uses a BERT-based cross-encoder system to re-rank documents, along with an extractive reader. RAG demonstrates that neither a re-ranker nor extractive reader is necessary for state-of-the-art machine reading performance. Generating answers even when it is possible to extract them has a number of advantages. Documents which contain clues as to the correct answer but do not contain the correct answer verbatim themselves can still contribute probability mass towards a correct answer being generated, which is not possible with standard extractive approaches, leading to more effective marginalization across documents. Furthermore, RAG is able to generate correct answers even when the correct answer is not present in any of the retrieved documents, achieving an accuracy of 11.8% in such cases for Natural Questions, whereas an extractive model would score 0%.

4.2 Abstractive Question Answering

As shown in Table 2, RAG-Sequence outperforms BART on Open MS-MARCO generation by 2.6 Bleu points and 2.6 Rouge-L points. It approaches the performance of state-of-the-art models, which is impressive considering that (i) these models have access to passages that contain the specific information required to generate the reference answer, (ii) many questions are unanswerable without access to gold passages, and (iii) other questions are unanswerable from Wikipedia alone. Table 4 shows some generated answers from our models. Qualitatively, we find that RAG models hallucinate less and generate factually correct text more often than BART. Later we also show that RAG generations are more diverse than BART generations (see Section 4.6).

4.3 Jeopardy Question Generation

Table 2 shows automatic metric results on the Jeopardy question generation task. We find that RAG-Token performs better than the RAG-Sequence model in this setting, with both models outperforming BART using the Q-BLEU-1 metric.

Table 3 shows the results from the human evaluation. The human evaluation was carried out with 452 pairs of generations from BART and RAG-Token. The annotators indicated that BART was more factual than RAG in only 7.1% of cases, while RAG was more factual in 42.7% of cases and both RAG and BART were factual in a further 17% of cases, clearly demonstrating the comparative effectiveness of RAG on the task over a state-of-the-art conditional generation model. The annotators also strongly prefer RAG generations in terms of specificity.

Typical example of generations from each model are shown in Table 4. BART generates a more generic response (which is incorrect), whereas the RAG models generate specific and correct facts about Washington state.

We hypothesise that RAG-Token performs best for this task as Jeopardy questions often contain two separate pieces of information about the entity, and RAG-Token is able to synthesize a response by combining disparate information from different retrieved documents in one generation. Figure 2 shows an example where content from two documents has been combined to produce the generated question. Document 2 contains information about Hemingway's "The Sun also rises," and the contribution for "Sun" is very high for document 2. Similarly, "A Farewell to Arms" is mentioned in Document 1, which dominates the posterior when this title is generated. Intriguingly, after the first token of these book titles are generated, the distribution over documents flattens again. This observation suggests that the generator completes the book titles without depending on specific documents. In other words, the model's parametric knowledge is sufficient to complete the titles.

We show evidence for the above interpretation by feeding the BART-only baseline with the partial decoding `"The Sun`. BART completes the generation `"The Sun Also Rises" is a novel by this`

Document 1: his works are considered classics of American literature ... His wartime experiences formed the basis for his novel **"A Farewell to Arms"** (1929) ...

Document 2: ... artists of the 1920s "Lost Generation" expatriate community. His debut novel, **"The Sun Also Rises"**, was published in 1926.
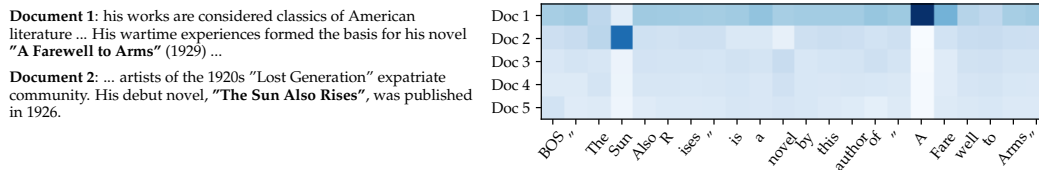

Figure 2: RAG-Token document posterior $p(z_i|x, y_i, y_{-i})$ for each generated token for input "Hemingway" for Jeopardy generation with 5 retrieved documents. The posterior for document 1 is high when generating "A Farewell to Arms" and for document 2 when generating "The Sun Also Rises"

Task	Input	Model	Generation
MS-MARCO	define middle ear	BART	[?] The middle ear is the part of the ear between the middle ear and the nose.
		RAG-T	The middle ear is the portion of the ear internal to the eardrum.
		RAG-S	The middle ear includes the tympanic cavity and the three ossicles.
	what currency needed in scotland	BART	The currency needed in Scotland is Pound sterling.
		RAG-T	Pound is the currency needed in Scotland.
		RAG-S	The currency needed in Scotland is the pound sterling.
Jeopardy Question Gener -ation	Washington	BART	[?] This state has the largest number of counties in the U.S.
		RAG-T	It's the only U.S. state named for a U.S. president
		RAG-S	It's the state where you'll find Mount Rainier National Park
	The Divine Comedy	BART	[*] This epic poem by Dante is divided into 3 parts: the Inferno, the Purgatorio & the Purgatorio
		RAG-T	Dante's "Inferno" is the first part of this epic poem
		RAG-S	This 14th century work is divided into 3 sections: "Inferno", "Purgatorio" & "Paradiso"

Table 4: Example Generations for MS-MARCO and Jeopardy Question generation. RAG models generate mpre specific and factually accurate responses, whereas BART generate more factually incorrect (marked by '?'), or partially correct (marked by *) and more generic responses.

`author of "The Sun Also Rises"` indicating the title "The Sun Also Rises" is stored in BART's parameters. Similarly, feeding the partial decoding `"The Sun Also Rises" is a novel by this author of "A` will result in BART completing the generation with `"The Sun Also Rises" is a novel by this author of "A Farewell to Arms`. This example shows how the parametric and non-parametric memories *work together*—the non-parametric component helps to guide the generation in a particular direction, drawing out specific knowledge stored in the parametric memory.

4.4 Fact Verification

Table 2 shows our results on the FEVER 3-way and 2-way classification task. For 3-way classification, RAG achieves accuracies that are within 4.3% of state-of-the-art models, which are complex pipeline systems with domain-specific architectures and substantial engineering, trained using intermediate supervision, which RAG does not require.

For 2-way classification, we compare against the model from [51], which trains RoBERTa [31] to classify the claim as true or false given the gold evidence sentence. RAG achieves an accuracy within 2.7% of this model, despite being supplied with only the claim and retrieving its own evidence.

We also analyze whether the documents retrieved by RAG correspond to the documents annotated as gold evidence in FEVER. We analyze the overlap in Wikipedia articles between the top k documents retrieved by RAG and the gold, annotated evidence documents. We find that the top article retrieved by RAG is a gold document for the claim in 71% of cases, and a gold article is present in the top 10 retrieved articles in 90% of cases.

4.5 Ablations

To gain a better understanding of what factors affect RAG's performance, we perform a number of ablation experiments for our tasks on their respective development sets.

Model	NQ	TQA	WQ	CT	Jeopardy-QGen		MSMarco		FVR-3	FVR-2
		Exact Match			B-1	QB-1	R-L	B-1	Label Accuracy	
RAG-Token-BM25	29.7	41.5	32.1	33.1	17.5	22.3	55.5	48.4	**75.1**	**91.6**
RAG-Seq-BM25	31.8	44.1	36.6	33.8	11.1	19.5	56.5	46.9		
RAG-Token-Frozen	37.8	50.1	37.1	51.1	16.7	21.7	55.9	49.4	72.9	89.4
RAG-Seq-Frozen	41.2	52.1	41.8	52.6	11.8	19.6	56.7	47.3		
RAG-Token	43.5	54.8	**46.5**	51.9	**17.9**	**22.6**	56.2	**49.4**	74.5	90.6
RAG-Seq	**44.0**	**55.8**	44.9	**53.4**	15.3	21.5	**57.2**	47.5		

Table 5: Ablations on the development set. As FEVER is a classification dataset, RAG-Token and RAG-Sequence are equivalent.
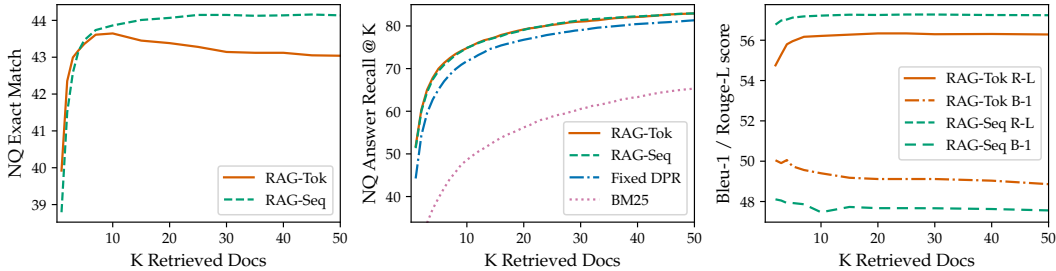

Figure 3: Left: NQ performance as more documents are retrieved. Center: Fraction of answers in NQ where the answer occurs somewhere in the top K documents. Right: MS-MARCO Bleu-1 and Rouge-L as more documents are retrieved.

Using more documents Models are trained with either 5 or 10 retrieved latent documents, and we do not observe significant differences in performance between them. We also have the flexibility to adjust the number of retrieved documents at test time, which does affect performance. Figure 3 (left) shows that retrieving more documents at test time monotonically improves Open-domain QA results for RAG-Sequence, but performance peaks for RAG-Token at 10 retrieved documents. Figure 3 (right) shows that retrieving more documents leads to higher Rouge-L for RAG-Token at the expense of Bleu-1, but the effect is less pronounced for RAG-Sequence.

Retrieval A key feature of RAG is the ability to learn to retrieve relevant information for the task at hand. To assess the effectiveness of the retrieval mechanism, we run ablations on RAG where we prevent gradients from propagating into the retriever. Table 5 shows the results across all tasks. In each case, learned retrieval improves results, with the largest improvements in question answering. Figure 3 (center) shows that the learned retriever shows a higher recall for gold documents compared to the fixed retriever. The improvements on TriviaQA and Natural Questions are notable, as we initialize the retriever from DPR, which is trained with strong, document-level supervision to perform well on these tasks. We also compare RAG's dense embedding-based retrieval mechanism to a word overlap-based BM25 retriever [47]. Here, we replace RAG's differentiable retriever with a fixed BM25 system. We use the BM25 retrieval scores as logits when calculating $p_\eta(z_i|x)$. Table 5 and Figure 3 show the results. For FEVER, we find that BM25 performs best, perhaps since FEVER claims are heavily entity-centric and thus well-suited for word overlap-based retrieval. On all other tasks, we find the differentiable retrieval to be helpful, especially question answering, where it is crucial.

4.6 Generation Diversity

Section 4.3 established that RAG models generate are more factual and specific than BART for Jeopardy question generation. Similar to Li et al. [29], Vijayakumar et al. [53] and Massarelli et al. [35], we also investigate the diversity of generations by calculating the ratio of distinct ngrams to total ngrams generated by different models. Table 6 shows that RAG-Sequence generations are more

Dataset	Gold	BART	RAG-Token	RAG-Sequence
MSMARCO	89.6%	70.7%	77.8%	83.5%
Jeopardy Generation	90.0%	32.4%	46.8 %	53.8%

Table 6: Ratio of distinct tri-grams to total tri-grams in the development set generations for MS-MARCO and Jeopardy Question Generation.

diverse than RAG-Token generations, and both generate significantly more diverse outputs than BART without requiring any diversity-promoting decoding strategy.

4.7 Hot-swapping indices

An advantage of non-parametric knowledge models such as RAG is that the knowledge base can be easily updated at test time. Parametric-only models such as T5 or BART require additional training to update their behavior as facts about the world change. As a demonstration, we build an index using the DrQA Wikipedia dump [5], (dated December 21st, 2016) and compare generations from RAG using this index to the newer index used in our main results (December 20th, 2018). We prepared a list of 82 heads of states who had changed between these dates and used a template "Who is {position}?" (e.g., "Who is the prime minister of the UK?") to query our Natural Questions -finetuned RAG model with each index. RAG achieved an accuracy of 70% using the 2016 index for 2016 world leaders and an accuracy of 68% using the 2018 index for the 2018 world leaders. Only 21% of the model's predictions were the same using the two indices, and accuracy using mismatched indices is very low (12% using the 2018 index for 2016 leaders and 4% using the 2016 index for 2018 leaders). Our result shows that we can effectively update RAG's behavior with new world knowledge by simply replacing its non-parametric memory.

5 Related Work

Single-Task Retrieval Prior work has shown that retrieval improves performance across a variety of NLP tasks when considered in isolation. Such tasks include open-domain question answering [5, 25], fact checking [50], fact completion [42], long-form question answering [12], Wikipedia article generation [32], dialogue [36, 59, 9, 13], translation [16], and language modeling [17, 23]. Our work unifies previous successes in incorporating retrieval into individual tasks, showing that a single retrieval-based architecture is capable of achieving strong performance across several tasks.

General-Purpose Architectures for NLP Prior work on general-purpose architectures for NLP tasks has shown great success without the use of retrieval. A single, pre-trained language model has been shown to achieve strong performance on various classification tasks in the GLUE benchmarks [54, 55] after fine-tuning [43, 8]. GPT-2 [44] later showed that a single, left-to-right, pre-trained language model could achieve strong performance across both discriminative and generative tasks. For further improvement, BART [28] and T5 [45, 46] propose a single, pre-trained encoder-decoder model that leverages bi-directional attention to achieve stronger performance on discriminative and generative tasks. Our work aims to expand the space of possible tasks with a single, unified architecture, by learning a retrieval module to augment pre-trained, generative language models.

Learned Retrieval There is significant work on learning to retrieve documents in information retrieval, more recently with pre-trained, neural language models [39, 22] similar to ours. Some work optimizes the retrieval module to aid in a specific, downstream task such as question answering, using search [40], reinforcement learning [6, 57, 56], or a latent variable approach [27, 18] as in our work. These successes leverage different retrieval-based architectures and optimization techniques to achieve strong performance on a single task, while we show that a single retrieval-based architecture can be fine-tuned for strong performance on a variety of tasks.

Memory-based Architectures Our document index can be seen as a large external memory for neural networks to attend to, analogous to memory networks [58, 48]. Concurrent work [14] learns to retrieve a trained embedding for each entity in the input, rather than to retrieve raw text as in our

work. Other work improves the ability of dialog models to generate factual text by attending over fact embeddings [9, 13] or, closer to our work, over retrieved text directly [15]. A key feature of our memory is that it is comprised of raw text rather distributed representations, which makes the memory both (i) human-readable, lending a form of interpretability to our model, and (ii) human-writable, enabling us to dynamically update the model's memory by editing the document index.

6 Discussion

In this work, we presented hybrid generation models with access to parametric and non-parametric retrieval-based external memory, in the form of Wikipedia. We showed that our RAG models obtain state-of-the-art performance on open domain question answering. We found that people prefer RAG's generation over purely parametric BART and find RAG more factual, and we conducted a detailed investigation of the learned retrieval component, validating its effectiveness. We also showed that the model's grounding in external data leads it to generate more diverse, and illustrated by how the retrieval index can be hot-swapped on the fly without having to retrain the model. In future work, it would be interesting to investigate if the two components can be jointly pre-trained from scratch, either on a denoising objective similar to BART, or through some other objective. Our work opens new research directions on how parametric and non-parametric memories interact and how to most effectively combine the different components, showing promise in being applied to a wide variety of NLP tasks.

References

[1] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv:1611.09268 [cs]*, November 2016. URL `http://arxiv.org/abs/1611.09268`. arXiv: 1611.09268.

[2] Petr Baudiš and Jan Šedivỳ. Modeling of the question answering task in the yodaqa system. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 222–228. Springer, 2015. URL `https://link.springer.com/chapter/10.1007%2F978-3-319-24027-5_20`.

[3] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 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, October 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D13-1160`.

[4] Bin Bi, Chenliang Li, Chen Wu, Ming Yan, and Wei Wang. Palm: Pre-training an autoencoding&autoregressive language model for context-conditioned generation. *ArXiv*, abs/2004.07159, 2020. URL `https://arxiv.org/abs/2004.07159`.

[5] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 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, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1171. URL `https://www.aclweb.org/anthology/P17-1171`.

[6] Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste, and Jonathan Berant. Coarse-to-fine question answering for long documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 209–220, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1020. URL `https://www.aclweb.org/anthology/P17-1020`.

[7] Christopher Clark and Matt Gardner. Simple and Effective Multi-Paragraph Reading Comprehension. *arXiv:1710.10723 [cs]*, October 2017. URL `http://arxiv.org/abs/1710.10723`. arXiv: 1710.10723.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 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 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://www.aclweb.org/anthology/N19-1423`.

[9] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=r1l73iRqKm`.

[10] Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Guney, Volkan Cirik, and Kyunghyun Cho. SearchQA: A New Q&A Dataset Augmented with Context from a Search Engine. *arXiv:1704.05179 [cs]*, April 2017. URL `http://arxiv.org/abs/1704.05179`. arXiv: 1704.05179.

[11] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1082. URL `https://www.aclweb.org/anthology/P18-1082`.

[12] Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1346. URL `https://www.aclweb.org/anthology/P19-1346`.

[13] Angela Fan, Claire Gardent, Chloe Braud, and Antoine Bordes. Augmenting transformers with KNN-based composite memory, 2020. URL `https://openreview.net/forum?id=H1gx1CNKPH`.

[14] Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. Entities as experts: Sparse memory access with entity supervision. *ArXiv*, abs/2004.07202, 2020. URL `https://arxiv.org/abs/2004.07202`.

[15] Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen tau Yih, and Michel Galley. A knowledge-grounded neural conversation model. In *AAAI Conference on Artificial Intelligence*, 2018. URL `https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16710`.

[16] Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. Search engine guided neural machine translation. In *AAAI Conference on Artificial Intelligence*, 2018. URL `https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17282`.

[17] Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450, 2018. doi: 10.1162/tacl_a_00030. URL `https://www.aclweb.org/anthology/Q18-1031`.

[18] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909, 2020. URL `https://arxiv.org/abs/2002.08909`.

[19] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017. URL `https://arxiv.org/abs/1702.08734`.

[20] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 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, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://www.aclweb.org/anthology/P17-1147.

[21] Armand Joulin and Tomas Mikolov. Inferring algorithmic patterns with stack-augmented recurrent nets. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 190–198, Cambridge, MA, USA, 2015. MIT Press. URL https://papers.nips.cc/paper/5857-inferring-algorithmic-patterns-with-stack-augmented-recurrent-nets.

[22] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020. URL https://arxiv.org/abs/2004.04906.

[23] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HklBjCEKvH.

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[25] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: a Benchmark for Question Answering Research. *Transactions of the Association of Computational Linguistics*, 2019. URL https://tomkwiat.users.x20web.corp.google.com/papers/natural-questions/main-1455-kwiatkowski.pdf.

[26] Guillaume Lample, Alexandre Sablayrolles, Marc' Aurelio Ranzato, Ludovic Denoyer, and Herve Jegou. Large memory layers with product keys. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8548–8559. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/9061-large-memory-layers-with-product-keys.pdf.

[27] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 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, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1612. URL https://www.aclweb.org/anthology/P19-1612.

[28] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019. URL https://arxiv.org/abs/1910.13461.

[29] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1014. URL https://www.aclweb.org/anthology/N16-1014.

[30] Margaret Li, Jason Weston, and Stephen Roller. Acute-eval: Improved dialogue evaluation with optimized questions and multi-turn comparisons. *ArXiv*, abs/1909.03087, 2019. URL https://arxiv.org/abs/1909.03087.

[31] Hairong Liu, Mingbo Ma, Liang Huang, Hao Xiong, and Zhongjun He. Robust neural machine translation with joint textual and phonetic embedding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3044–3049, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1291. URL https://www.aclweb.org/anthology/P19-1291.

[32] Peter J. Liu*, Mohammad Saleh*, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Hyg0vbWC-`.

[33] Yury A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:824–836, 2016. URL `https://arxiv.org/abs/1603.09320`.

[34] Gary Marcus. The next decade in ai: four steps towards robust artificial intelligence. *arXiv preprint arXiv:2002.06177*, 2020. URL `https://arxiv.org/abs/2002.06177`.

[35] Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktäschel, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. How decoding strategies affect the verifiability of generated text. *arXiv preprint arXiv:1911.03587*, 2019. URL `https://arxiv.org/abs/1911.03587`.

[36] Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M. Khapra. Towards exploiting background knowledge for building conversation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2322–2332, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1255. URL `https://www.aclweb.org/anthology/D18-1255`.

[37] Preksha Nema and Mitesh M. Khapra. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1429. URL `https://www.aclweb.org/anthology/D18-1429`.

[38] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. In Tarek Richard Besold, Antoine Bordes, Artur S. d'Avila Garcez, and Greg Wayne, editors, *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016. URL `http://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf`.

[39] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019. URL `https://arxiv.org/abs/1901.04085`.

[40] Ethan Perez, Siddharth Karamcheti, Rob Fergus, Jason Weston, Douwe Kiela, and Kyunghyun Cho. Finding generalizable evidence by learning to convince q&a models. 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 2402–2411, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1244. URL `https://www.aclweb.org/anthology/D19-1244`.

[41] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? 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 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250. URL `https://www.aclweb.org/anthology/D19-1250`.

[42] Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*, 2020. URL `https://openreview.net/forum?id=025X0zPfn`.

[43] Alec Radford. Improving Language Understanding by Generative Pre-Training, 2018. URL `https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf`.

[44] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019. URL `https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`.

[45] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019. URL `https://arxiv.org/abs/1910.10683`.

[46] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? *arXiv e-prints*, 2020. URL `https://arxiv.org/abs/2002.08910`.

[47] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, April 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL `https://doi.org/10.1561/1500000019`.

[48] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5846-end-to-end-memory-networks.pdf`.

[49] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-To-End Memory Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5846-end-to-end-memory-networks.pdf`.

[50] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 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, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1074. URL `https://www.aclweb.org/anthology/N18-1074`.

[51] James H. Thorne and Andreas Vlachos. Avoiding catastrophic forgetting in mitigating model biases in sentence-pair classification with elastic weight consolidation. *ArXiv*, abs/2004.14366, 2020. URL `https://arxiv.org/abs/2004.14366`.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

[53] Ashwin Vijayakumar, Michael Cogswell, Ramprasaath Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search for improved description of complex scenes. 2018. URL `https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17329`.

[54] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL `https://www.aclweb.org/anthology/W18-5446`.

[55] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3261–3275. Curran Associates, Inc., 2019. URL `https://arxiv.org/abs/1905.00537`.

[56] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. R^3: Reinforced ranker-reader for open-domain question answering. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5981–5988. AAAI Press, 2018. URL `https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16712`.

[57] Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. Evidence aggregation for answer re-ranking in open-domain question answering. In *ICLR*, 2018. URL `https://openreview.net/forum?id=rJl3yM-Ab`.

[58] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1410.3916`.

[59] Jason Weston, Emily Dinan, and Alexander Miller. Retrieve and refine: Improved sequence generation models for dialogue. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, pages 87–92, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5713. URL `https://www.aclweb.org/anthology/W18-5713`.

[60] Shiyue Zhang and Mohit Bansal. Addressing semantic drift in question generation for semi-supervised question answering. 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 2495–2509, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1253. URL `https://www.aclweb.org/anthology/D19-1253`.

[61] Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. Reasoning over semantic-level graph for fact checking. *ArXiv*, abs/1909.03745, 2019. URL `https://arxiv.org/abs/1909.03745`.

Figure 4: Annotation interface for human evaluation of factuality. A pop-out for detailed instructions and a worked example appear when clicking "view tool guide".

A   Human evaluation

Figure 4 shows the user interface for human evaluation. To avoid any biases for screen position, which model corresponded to sentence A and sentence B was randomly selected for each example. Annotators were encouraged to research the topic using the internet, and were given detailed instructions and worked examples in a full instructions tab. We included some gold sentences in order to assess the accuracy of the annotators. Two annotators did not perform well on these examples and their annotations were removed from the results.

B   Further details on Open-Domain QA

For open-domain QA, multiple answer annotations are often available for a given question. These answer annotations are exploited by extractive models during training as typically all the answer annotations are used to find matches within documents when preparing training data. For RAG, we also make use of multiple annotation examples for Natural Questions and WebQuestions by training the model with each (q, a) pair separately, leading to a small increase in accuracy. For TriviaQA, there are often many valid answers to a given question, some of which are not suitable training targets, such as emoji or spelling variants. For TriviaQA, we filter out answer candidates if they do not occur in top 1000 documents for the query.

TriviaQA Evaluation setups   The open-domain QA community customarily uses public development datasets as test datasets, as test data for QA datasets is often restricted and dedicated to reading compehension purposes. We report our results using the datasets splits used in DPR [22], which are consistent with common practice in Open-domain QA. For TriviaQA, this test dataset is the public TriviaQA Web Development split. Roberts et al. [46] used the TriviaQA official Wikipedia test set instead. Févry et al. [14] follow this convention in order to compare with Roberts et al. [46] (See appendix of [14]). We report results on both test sets to enable fair comparison to both approaches. We find that our performance is much higher using the official Wiki test set, rather than the more conventional open-domain test set, which we attribute to the official Wiki test set questions being simpler to answer from Wikipedia.

C   Further details on FEVER

For FEVER classification, we follow the practice from [28], and first re-generate the claim, and then classify using the representation of the final hidden state, before finally marginalizing across documents to obtain the class probabilities. The FEVER task traditionally has two sub-tasks. The first is to classify the claim as either "Supported", "Refuted" or "Not Enough Info", which is the task we explore in the main paper. FEVER's other sub-task involves extracting sentences from Wikipedia

as evidence supporting the classification prediction. As FEVER uses a different Wikipedia dump to us, directly tackling this task is not straightforward. We hope to address this in future work.

D "Null document" Probabilities

We experimented with adding "Null document" mechanism to RAG, similar to REALM [18] in order to model cases where no useful information could be retrieved for a given input. Here, if k documents were retrieved, we would additionally "retrieve" an empty document and predict a logit for the null document, before marginalizing over $k + 1$ predictions. We explored modelling this null document logit by learning (i) a document embedding for the null document, (ii) a static learnt bias term, or (iii) a neural network to predict the logit. We did not find that these improved performance, so in the interests of simplicity, we omit them. For Open MS-MARCO, where useful retrieved documents cannot always be retrieved, we observe that the model learns to always retrieve a particular set of documents for questions that are less likely to benefit from retrieval, suggesting that null document mechanisms may not be necessary for RAG.

E Parameters

Our RAG models contain the trainable parameters for the BERT-base query and document encoder of DPR, with 110M parameters each (although we do not train the document encoder ourselves) and 406M trainable parameters from BART-large, 406M parameters, making a total of 626M trainable parameters. The best performing "closed-book" (parametric only) open-domain QA model is T5-11B with 11 Billion trainable parameters. The T5 model with the closest number of parameters to our models is T5-large (770M parameters), which achieves a score of 28.9 EM on Natural Questions [46], substantially below the 44.5 that RAG-Sequence achieves, indicating that hybrid parametric/non-parametric models require far fewer trainable parameters for strong open-domain QA performance. The non-parametric memory index does not consist of trainable parameters, but does consists of 21M 728 dimensional vectors, consisting of 15.3B values.

F Retrieval Collapse

In preliminary experiments, we observed that for some tasks such as story generation [11], the retrieval component would "collapse" and learn to retrieve the same documents regardless of the input. In these cases, once retrieval had collapsed, the generator would learn to ignore the documents, and the RAG model would perform equivalently to BART. The collapse could be due to a less-explicit requirement for factual knowledge in some tasks, or the longer target sequences, which could result in less informative gradients for the retriever. Perez et al. [40] also found spurious retrieval results when optimizing a retrieval component in order to improve performance on downstream tasks.