Learning to Filter Context for Retrieval-Augmented Generation

Anonymous ACL submission

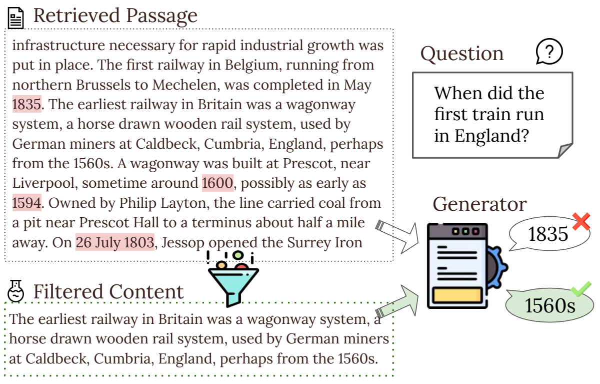
Abstract

On-the-fly retrieval of relevant knowledge has proven an essential element of reliable systems for tasks such as open-domain question answering and fact verification. However, because retrieval systems are not perfect, generation models are required to generate outputs given partially or entirely irrelevant passages. This can cause over- or under-reliance on context, and result in problems in the generated output such as hallucinations. To alleviate these problems, we propose FILCO, a method that improves the quality of the context provided to the generator by (1) identifying useful context based on lexical and information-theoretic approaches, and (2) training context filtering models that can filter retrieved contexts at test time. We experiment on six knowledge-intensive tasks with FLAN-T5 and LLAMA2, and demonstrate that our method outperforms existing approaches on extractive question answering (QA), complex multi-hop and long-form QA, fact verification, and dialog generation tasks.[1]

1 Introduction

Retrieval augmented approaches to generation have proven effective for many knowledge-intensive language tasks such as open-domain question answering and fact verification, producing more faithful (Khandelwal et al., 2020; Lewis et al., 2020; Shuster et al., 2021; Komeili et al., 2022), interpretable (Guu et al., 2020), and generalizable (Khandelwal et al., 2021) outputs. While the de facto approach is to provide the top retrieved passages to the generator indiscriminately, imperfect retrieval systems often return irrelevant or distracting content. Generation models are then trained to produce canonical outputs with the guidance of partially or entirely irrelevant passages, and thus are prone to hallucination or spurious memorization.

[1] https://anonymous.4open.science/r/filco



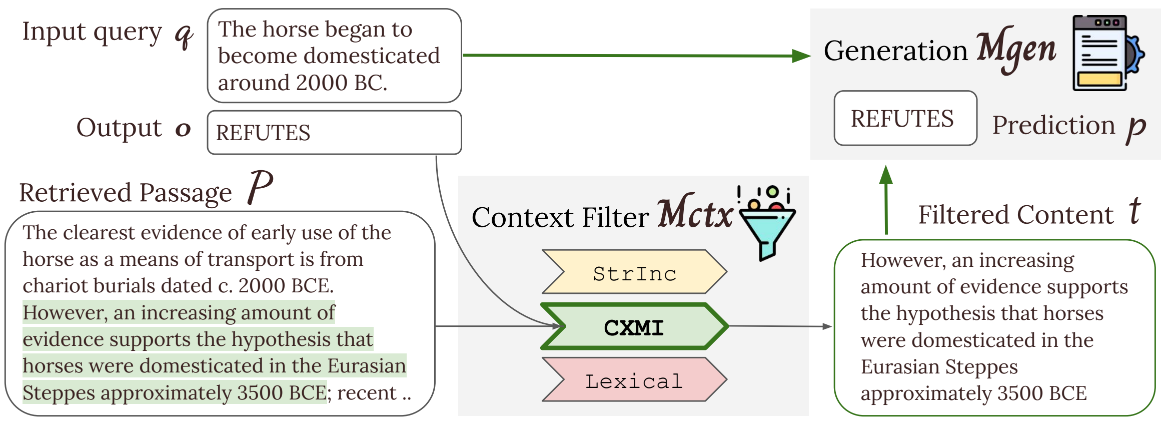
infrastructure necessary for rapid industrial growth was put in place. The first railway in Belgium, running from northern Brussels to Mechelen, was completed in May 1835. The earliest railway in Britain was a wagonway system, a horse drawn wooden rail system, used by German miners at Caldbeck, Cumbria, England, perhaps from the 1560s. A wagonway was built at Prescot, near Liverpool, sometime around 1600, possibly as early as 1594. Owned by Philip Layton, the line carried coal from a pit near Prescot Hall to a terminus about half a mile away. On 26 July 1803, Jessop opened the Surrey Iron

The earliest railway in Britain was a wagonway system, a horse drawn wooden rail system, used by German miners at Caldbeck, Cumbria, England, perhaps from the 1560s.

Figure 1: FILCO filters out irrelevant content (marked in red) and leaves precisely supporting content, making it easier for the generator to predict the correct answer.

Ideally, a model should be grounded on only content that supports the correct output. However, this ideal grounding is hard to achieve with an imperfect retrieval system alone. On one hand, positive passages (i.e., passages that support the output) sometimes contain distracting content. For example in Figure 1, while the passage containing the actual supporting content is successfully retrieved, the model still fails to pay sufficient attention to the supporting content, and is distracted by surrounding sentences that share similar topics (Shi et al., 2023). On the other hand, models learn to over-utilize negative passages, extracting spans from an irrelevant passage, which will usually be incorrect. This can degrade accuracy, as evidenced by the fact that training with higher-quality context often leads to better performance (Dou et al., 2021).

Some works have attempted to optimize the provided content on the passage level, by reranking more relevant passages to the top of the retrieved list (Wang et al., 2018; Nogueira and Cho, 2020; Mao et al., 2021), selecting only evidential passages to include (Asai et al., 2022), or only retrieving passages when generation models need assistance (Mallen et al., 2023; Jiang et al., 2023). Choi et al. (2021) proposed to decontextualize sentences by integrating surrounding context, but require sub-



Input query q
The horse began to become domesticated around 2000 BC.
Output o
REFUTES
Retrieved Passage P
The clearest evidence of early use of the horse as a means of transport is from chariot burials dated c. 2000 BCE. However, an increasing amount of evidence supports the hypothesis that horses were domesticated in the Eurasian Steppes approximately 3500 BCE; recent ..
Context Filter M_{ctx}
StrInc
CXMI
Lexical
Generation M_{gen}
REFUTES
Prediction p
Filtered Content t
However, an increasing amount of evidence supports the hypothesis that horses were domesticated in the Eurasian Steppes approximately 3500 BCE

Figure 2: The FILCO pipeline: (i) filtering retrieved passages, (ii) generation with filtered context.

stantial human annotation effort and still can suffer from distracting content, even in positive passages.

In this paper, we propose FILCO (§2), a method that learns to FILter COntext retrieved in a fine-grained sentence-wise manner by training on content selected via three measures: (i) STRINC: whether passages contain the generation output, (ii) LEXICAL overlap: how much unigram overlap the content and output has, and (iii) Conditional cross-mutual information (CXMI): how much more likely the generator is to generate the output when the content is provided.

We experiment on six knowledge-intensive language datasets from three tasks (§3). (i) question answering: including NaturalQuestions (NQ) and Trivia QA (TQA), as well as more complex multi-hop HotpotQA and long-form ELI5, (ii) fact verification: Fact Extraction and VERificaton (FEVER), and (iii) knowledge-grounded dialog generation: the Wizard of Wikipedia (WoW) dataset.

Using FLAN-T5 and LLAMA2 models, our method outperforms both baseline methods, i.e., full-context augmentation and passage-wise filtering, on all six datasets. FILCO also greatly reduces the prompt length by $44 - 64\%$ across tasks, with corresponding efficiency benefits during generation. We further split examples retrieved with positive and negative passages, and show that FILCO effectively improves generation in both scenarios (§4). Moreover, we extend experiments to the more complex multi-passage settings, where FILCO maintains its advantage over baseline methods.

Comparing filtering methods on each task, we observe that STRINC, LEXICAL and CXMI-based filtering were best for extractive QA, dialog generation, and more complex tasks, respectively (§5).

2 Generation with Filtered Contexts

In this section, we first outline notation (§2.1), then introduce three oracle filtering measures (§2.2). Next, we describe how to train context filtering models with oracle filtered context (§2.3) and learn to generate with filtered contexts (§2.4).

2.1 Problem Statement

In retrieval-augmented generation, we are given an input query q and annotated output o from an example $e = \{q, o\}$, and want to improve the output of a generative model M_{gen}. We assume a set of retrieved passages $P = \{p_i\}, i \in K$, each consisting of n_i text spans $p_i = [t_i^1, \cdots, t_i^{n_i}]$. We can provide the model with *one or more* selected text spans $T = \{t_i^j\}$ when generating output o, namely $M_{gen}(o \mid q, T)$. In traditional retrieval-based methods, however, *all text spans* in the top-K passages $\{t_i^j\}, \forall j \in n_i, \forall i \in K$ are provided to the model. In experiments, we split passages into sentences using the spaCy tokenizer[2] as candidate text spans. Later in §4, we will show that sentence-wise splitting performs the best among other granularities.

2.2 Obtaining Oracle Contexts

In this section, we propose methods that select oracle text spans that can be used to train a context filtering model. We select spans using a filtering function $F(\cdot)$, denoted as $F(T|e, P)$, where text spans in $T = \{t_i^j\}$ are selected by the underlying score function $f(\cdot)$ according to individual filtering methods. We select a single best span $T = t_i^j$, $(i, j) = \arg\max_{i,j} f(t_i^j, e)$ when performing oracle filtering, as it outperforms multi-span filtering in our preliminary studies.

[2] https://spacy.io/api/tokenizer

We now introduce three approaches to filter potentially useful content from retrieved passages.

String Inclusion The STRINC measure makes a binary decision $f_{inc}(t, o) \in \{0, 1\}$ on whether text span t lexically contains the output o. We enumerate the ranked passages retrieved $\{p_1, p_2, \cdots\}$ and select the first text span that contains the output $f_{inc}(t, o) = 1$. STRINC is effective when the supporting document p_{gold} contains the exact output o. However, f_{inc} may fail to distinguish supporting passages from spurious ones, that accidentally contain the output but do not answer the question. Applying f_{inc} to abstractive tasks may result in selecting zero spans since no exact matches exist.

Lexical Overlap We next introduce the more flexible LEXICAL measure $f_{uf1} \in [0, 1]$ that calculates the unigram overlap between the example e and the candidate text span t. Intuitively speaking, higher lexical overlap indicates greater topic similarity, hence higher utility at generation time.

We select sentences t using different parts of the example e for tasks of different types. We measure the F_1 score $f_{uf1}(t, o) \in [0, 1]$ between t and output o for tasks having responses grounded on provided knowledge, i.e., QA and dialog generation. We measure t using query q for fact verification as $f_{uf1}(t, q)$ since o is a one-word binary label. We select the span t_i^j with the highest similarity to example e and above a pre-defined threshold $\lambda = 0.5$,[3] where $(i, j) = \arg\max_{i,j}(f_{uf1}(t_i^j, e))$, and $i, j \in \{i, j \mid f_{uf1}(t_i^j, e) > \lambda\}$. Nonetheless, for tasks having queries that may be factually incorrect (e.g., fact verification), spans of high lexical overlap to an erroneous claim may reinforce the misinformation and lead to erroneous generations.

Conditional Cross-Mutual Information (CXMI) We adopt a measure f_{cxmi} from the conditional cross-mutual information (CXMI) score in contextual machine translation (Fernandes et al., 2021).

Given a pair of input sequences with and without context augmentation, $t \oplus q$ and q, we measure the probability difference in model M_{gen} generating the expected output o, the process being denoted as $f_{cxmi}(t, e) = \frac{M_{gen}(o|t \oplus q)}{M_{gen}(o|q)} \in \mathbb{R}$, as illustrated in Figure 3. We select the text span t_i^j having the highest CXMI score above a pre-defined threshold $\lambda = 0.0$,[4] where $(i, j) = \arg\max_{i,j}(f_{cxmi}(t_i^j, e))$, and $i, j \in \{i, j \mid f_{cxmi}(t_i^j, e) > \lambda\}$. f_{cxmi} can overcome the lexical barrier and is applicable to all tasks, albeit at the cost of more computation.

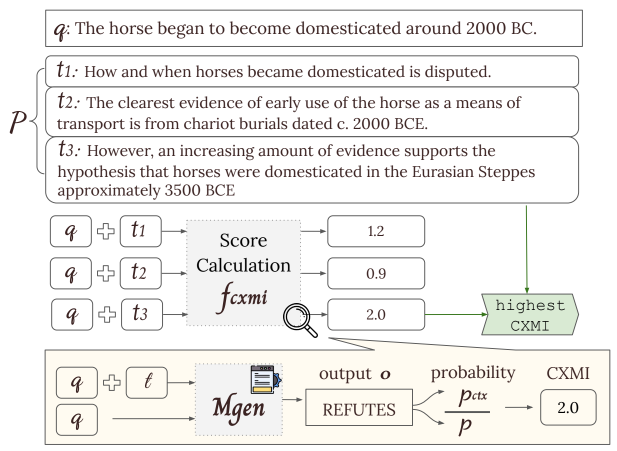

q: The horse began to become domesticated around 2000 BC.
P
t1: How and when horses became domesticated is disputed.
t2: The clearest evidence of early use of the horse as a means of transport is from chariot burials dated c. 2000 BCE.
t3: However, an increasing amount of evidence supports the hypothesis that horses were domesticated in the Eurasian Steppes approximately 3500 BCE
q + t1
q + t2
q + t3
Score Calculation
fcxmi
1.2
0.9
2.0
highest CXMI
q + t
q
Mgen
output o
REFUTES
probability
pctx
p
CXMI
2.0

Figure 3: An example illustration of context filtering with the CXMI measure.

2.3 Learning to Filter Contexts

While the previous section described how to identify useful contexts at training time when the gold-standard answer is known, we also need methods that can apply at test time when the answer is unknown. To this end, we train the context filtering models, M_{ctx}, using context filtered with the three measures in §2.2. To create training data for M_{ctx}, for each training example with query q, we concatenate the retrieved passages P to query q as input, then we apply the filter method f to obtain filtered context t_{silver} as output. We use $silver$ instead of $oracle$ to represent the non-perfect filtering result due to unknown gold labels for non-extractive tasks. As shown in Figure 2, we train M_{ctx} by feeding in query q and retrieved passages P, and ask it to generate filtered context t_{silver}, formalized as $M_{ctx}(t_{silver} \mid q \oplus P)$.

At test time, given the retrieved passages P for a test query q, we leverage M_{ctx} to predict filtered context t_{pred}, as $t_{pred} = M_{ctx}(q \oplus P)$. t_{pred} is subsequently provided to the generation model M_{gen} together with the query q, to predict the output.

2.4 Generation With Filtered Contexts

We similarly use t_{silver} filtered context for training and model predicted context t_{pred} for inference.

[3]We compare different thresholds (0.1, 0.3, 0.5, 0.7, 0.9) in preliminary studies, 0.5 gives the best generation results.

[4]1.0 naturally distinguishes context that adds to or reduces output probability. We compare other values in preliminary studies (0.5, 2.0), where 1.0 gives the best results.

For each training example (q, o), we prepend the silver filtered context t_{silver} to the query q, and get the model input $q \oplus t_{silver}$. We feed this input into the generation model M_{gen} and train it to output the gold output o, formalized as $M_{gen}(o \mid t_{silver} \oplus q)$.

At inference time, we provide the context t_{pred} filtered by model M_{ctx} for generation, denoted as $M_{gen}(o \mid t_{pred} \oplus q) = M_{gen}(o \mid M_{ctx}(q, P) \oplus q)$.

In comparison to appending all retrieved text spans $P \oplus q$, including only selected text can effectively reduce the computational cost by $\frac{|P|}{|t|}$ at both training and inference time.

3 Knowledge-Intensive Language Tasks

We experiment on six knowledge-intensive language tasks that necessitate retrieval augmentation for generation (§3.1), where a limited portion of examples are supported by retrieved passages (§3.2).

3.1 Tasks and Datasets

We use six datasets that contain outputs supported by Wikipedia articles, as listed in Table 1.

Open-Domain Question Answering We adopt NaturalQuestions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (TQA) (Joshi et al., 2017) to experiment with the open-domain QA task.

Each example in NQ has a question q and annotated short answers o. We experiment with the processed version (Lee et al., 2019) that includes all examples having short answers of no more than five tokens. In the TQA dataset, each example has a question q and answers o, which are extracted spans from supporting Wikipedia articles P. Following Lewis et al. (2020), we use the Exact Match (EM) metric to evaluate model predictions.

Multi-Hop Question Answering We also adopt more complex QA scenarios, the first of which is multi-hop QA, where answering each question q requires reasoning over a chain of passages P. For this task, we use the HotpotQA dataset (Yang et al., 2018). Because the answers do not always exist in the supporting documents, this dataset belongs to abstractive generation, in contrary to the extractive answers in NQ and TQA. Following Yang et al. (2018) and accommodating its abstractive nature, we use unigram F_1 for evaluation.

Long-Form Question Answering Another complex QA task is generating long, abstract answers given the question, i.e., long-form QA. For this we use the ELI5 (Fan et al., 2019) dataset, which requires elaborate, in-depth answers to open-ended questions. The dataset is derived from the Reddit forum "Explain Like I'm Five" and features diverse questions with multi-sentence answers. We experiment with the *generative short* setting, and evaluate model predictions using unigram F_1.

Fact Verification We use the Fact Extraction and VERification (FEVER) dataset (Thorne et al., 2018) aggregated by the KILT benchmark (Petroni et al., 2021). It contains claims q generated by rephrasing sentences in Wikipedia articles. A claim has the label o = "SUPPORTS" if it preserves the fact in the Wikipedia reference, otherwise is labeled as "REFUTES" due to the factual contradiction. Following the original baseline (Thorne et al., 2018), we use accuracy for evaluation.

Knowledge-Grounded Dialog Generation We adopt the Wizard of Wikipedia (WoW) dataset (Dinan et al., 2019) from KILT, which aims to generate the next dialog by grounding on Wikipedia articles. In each example, the input q is the conversation history involving multiple utterance turns, and the next-turn response is the output o. We evaluate with unigram F_1 following Petroni et al. (2021).

Dataset	# Examples (thousands) train	dev	test	Evaluation Metric
NQ	79.2	8.7	3.6	EM
TQA	78.8	8.8	11.3	EM
HOTPOTQA	88.9	5.6	5.6	F_1
ELI5	273.0	1.5	0.6	F_1
FEVER	105.0	10.4	10.1	Accuracy
WOW	63.7	3.1	2.9	F_1

Table 1: Statistics and evaluation metric for six tasks.

Table 1 lists the dataset statistics. Because test sets are not available for datasets adopted from the KILT benchmark (i.e., HotpotQA, ELI5, FEVER, WoW), we report the results on development sets.

3.2 Wikipedia Passage Retrieval

To better understand the quality of passages provided in the generation stage, we evaluate the quality of retrieval passages.

To retrieve Wikipedia passages for all examples, we use the adversarial Dense Passage Retriever (DPR) (Karpukhin et al., 2020)[5] to retrieve the top 5 passages from all Wikipedia passages.

[5]https://github.com/facebookresearch/DPR#new-march-2021-retrieval-model

A Mixture of Positive and Negative Passages We evaluate the *recall* of the top 1 and top 5 retrieved passages in Table 2 (left). For the extractive NQ and TQA tasks, we measure if any of the passages contain the answer strings. For the other four tasks where outputs are not spans in supporting documents, we calculate if any of the passages come from the provenance articles annotated in KILT.

Notably, for all six datasets, top-1 passages only support the canonical output half or less of the time. Although involving more passages increases the coverage of supporting documents, it often brings along linearly (Izacard and Grave, 2021) or quadratically increased computation.

Dataset	**Recall** (pos. + neg.)		**Precision** (pos.)	
	1	**5**	**1**	**5**
NQ	50.1	74.1	2.5	2.7
TQA	61.2	77.8	4.5	4.8
HotpotQA	16.7	27.3	2.1	0.4
ELI5	13.1	25.7	97.7	55.1
FEVER	57.0	75.9	1.3	1.4
WoW	34.9	54.8	16.4	17.7

Table 2: Recall of the top 1 and top 5 DPR-retrieved passages, and precision on positive passages.

Noise in Positive Passagess To measure the ratio of precisely supporting context in retrieved positive passages, we further calculate their unigram *precision* to the gold output, as shown in Table 2 (right). In general, the precision is pretty low: scoring less than 20% for WoW, and less than 5% for NQ, TQA, HotpotQA, and FEVER. ELI5 has exceptionally high top-1 precision, because its output often aggregates large text chunks from multiple passages. Yet still, the precision drops by over 40% when adding 4 more passages. These numbers indicate the potential existence of noisy content, which could distract the model and deteriorate its generation.

In the next section, we attempt to filter the sufficient and precisely necessary context, as described in §2, to achieve more efficient generation.

4 Experiments and Analysis

We first introduce the experimental setup (§4.1) and baseline approaches for comparison (§4.2). Then, we evaluate models on both end generation (§4.3) and context filtering (§4.5).

4.1 Experimental Setup

We use Flan-T5 (Chung et al., 2022) and Llama 2 (Touvron et al., 2023) as the backbone model architectures, because of their potential superior performance among open-source models. We fine-tune both models for (i) the context filtering task as M_{ctx}, and (ii) the end generation task as M_{gen}.

Flan-T5 Flan-T5 is a family of instruction-tuned encoder-decoder models for seq2seq generation tasks, which makes it suitable for our retrieval-augmented generation setting. Due to constraints in computational resources, we use the XL version with $3B$ parameters. We load model checkpoints from and implement training using HuggingFace Transformers (Wolf et al., 2020).

Llama 2 Llama 2 represents a collection of foundation model ranging from $7B$ to $70B$ parameters, particularly optimized for dialog uses cases, but also achieve good performance on many other tasks. We train the $7B$ model version with LoRA (Hu et al., 2022) using the xTuring platform.[6]

Implementation Details For both models, we allow a maximum of 1024 input tokens at training and inference. We allow M_{ctx} to generate at most 512 tokens and M_{gen} to 128 tokens. We use greedy decoding for generating both filtered context and end-task output. Unless otherwise specified, we train M_{ctx} and M_{gen} models for 3 epochs, using a learning rate of $5\text{e}{-5}$ and batch size of 32.

4.2 Experiment Methods

We describe two baselines Full and Psg, our main approach FilCo, and the Silver setting.

Baseline 1: Augmenting Full Passages The most common method for retrieval-augmented generation is to concatenate all passages into the input. We denote this method as Full and use it as our first baseline. To conduct a fair comparison with sufficient training in a full-context genreation style, we fine-tune Flan-T5 and Llama 2 to generate output using the full content of the top-1 passage under the same experiment setting as in §4.1.

Baseline 2: Passage-Wise Filtering An alternative method inspired by Asai et al. (2022) is to filter context on a passage level. Specifically, for the top-1 retrieved passage for each example, the model decides whether to include the entire piece of the passage in the input. In comparison, our method operates in a finer granularity (i.e., the sentence level). We denote this the Psg baseline and show the empirical advantage of our method.

[6] https://github.com/stochasticai/xTuring

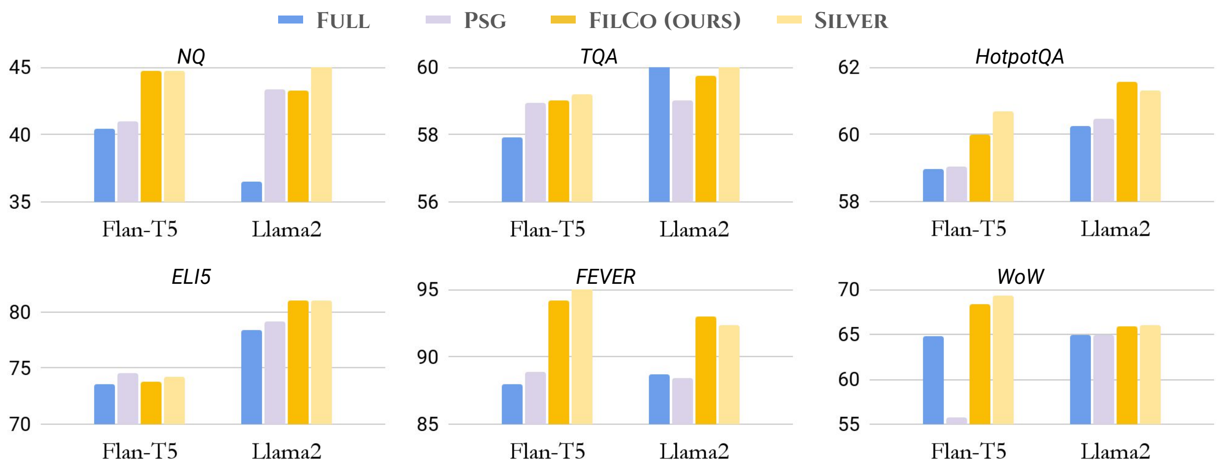


Figure 4: Generation performance when passages are filtered with different approaches.

Main Approach: Augmenting Filtered Context As described in §2, we train M_{ctx} to filter the top-1 retrieved passage p_1 to t_{silver}, and M_{gen} to generate output o with t_{silver}. To create t_{silver}, we use the STRINC measure for NQ and TQA, LEXICAL for WoW, and CXMI for FEVER, HotpotQA, and ELI5. These measures are shown to be optimal based on further analysis in §5.

At test time, we provide model-filtered context t_{pred} to M_{gen}, and denote the results as FILCO. To show the prospective performance upper-bound, we also evaluate M_{gen} generations by providing silver-filtered context t_{silver}, as the SILVER setting.

4.3 Generation Performance

Results using four methods and two models are shown in Figure 4. In general, applying context filtering beforehand *significantly improves* the results on *all datasets* than FULL and PSG. Compared to providing M_{gen} with SILVER filtered contexts, using M_{ctx} filtered context (FILCO) achieves *comparable performance* on all six tasks.

For *extractive QA tasks*, FILCO achieves +4.3 and +8.6 EM increase in NQ with FLAN-T5 and LLAMA2, +1.1 and +0.2 EM in TQA. For *complex QA tasks*, FILCO brings +1.0 and +1.3 F_1 increase in HotpotQA with FLAN-T5 and LLAMA2, and +0.6, +2.6 EM increase in ELI5. The overall improvement is less significant than extractive tasks, presumably due to the increased difficulty. For *abstractive generation tasks*, FILCO brings about even larger gains: +6.2 and +4.3 accuracy increase for FEVER with FLAN-T5 and LLAMA2, and +3.5, +1.1 F_1 increase for WoW. As could be partially conjectured from the low precision in Table 2, filtering irrelevant content helps the model focus on the concerned knowledge.

4.4 Providing Positive and Negative Passages

We decompose datasets into examples with positive and negative top-1 retrieved passages, to examine FILCO effectiveness under both scenarios.

As shown in Figure 6, applying FILCO effectively improves the utility of both positive and negative passages retrieved, and hence yields better generations, particularly for abstractive tasks such as FEVER and WoW. Aligning with our hypothesis, the generation model produces better outputs when we remove (i) distracting content in positive passages, and (ii) negative passages.

4.5 Evaluating Filtered Contexts

We evaluate filtering outputs from two aspects: reduced input length and increased answer precision.

Shorter Inputs In Figure 5, we measure the average number of tokens in model inputs after filtering the retrieved passages using different methods. We do not filter context in the FULL setting, filter by passage in PSG, and filter on the sentence level with FILCO. Model inputs contain the example query and (filtered) context.[7] FILCO effectively reduces input length by $44-64\%$.

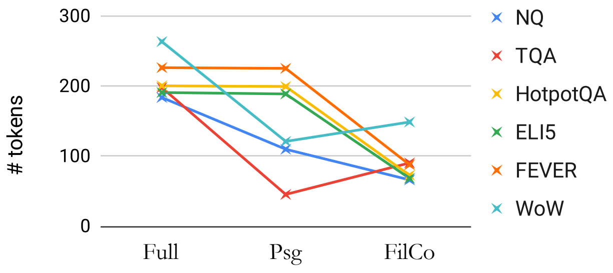


Figure 5: Number of input tokens after filtering retrieved contexts with different strategies.

[7]We tokenize all text sequences with the LlamaTokenizer.

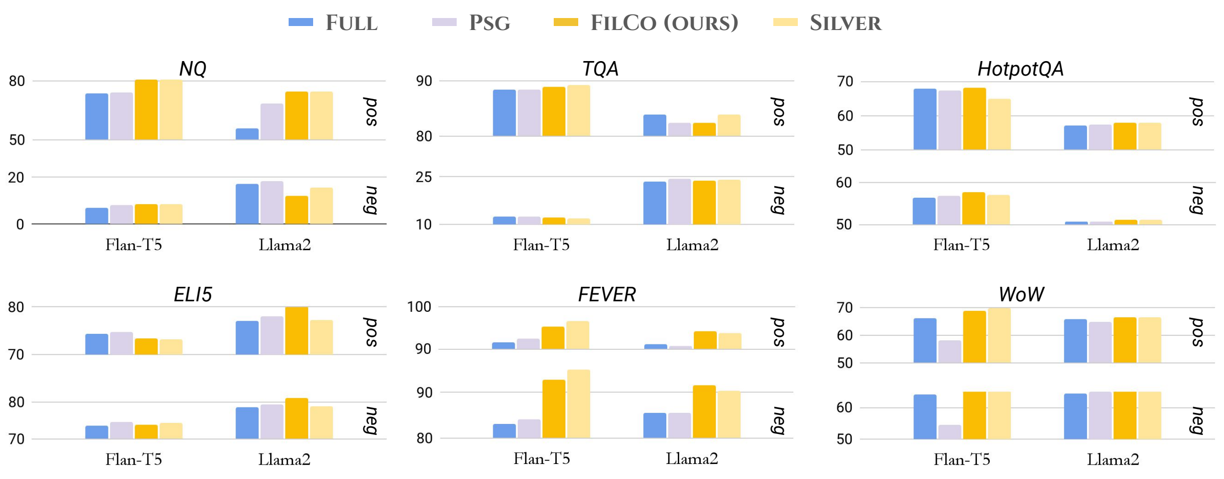


Figure 6: Improvement on examples retrieved with positive (top) and negative passages (bottom), respectively.

Higher Precision To measure the amount of unhelpful content in input contexts, we calculate the unigram precision of gold outputs to input contexts.

As in Table 3, filtered context achieves much higher output precision for all tasks. Moreover, model-filtered contexts (FilCo) are largely comparable to Silver, and sometimes even better, such as +3.8 points in TQA. For other tasks, the small gaps between them minimally affect the end generation, as already shown in Figure 4.

However, Psg filtering often leads to precisions lower than Full. The coarse granularity of filtering may be one reason for its precision loss.

Method	Full	Psg	FilCo	Silver
NQ	2.5	1.3	5.1	**7.3**
TQA	4.5	3.0	**8.4**	4.6
HotpotQA	2.6	2.6	10.8	**17.1**
ELI5	92.9	92.5	**98.8**	**98.8**
FEVER	1.2	1.2	**5.1**	4.4
WoW	10.8	35.5	62.9	**71.5**

Table 3: Precision of canonical outputs with respect to contexts filtered with different methods.

4.6 Generation with Multiple Passages

Integrating multiple passages as context input is often helpful. Some tasks such as multi-hop QA may naturally necessitate multiple passages to perform the task. To verify the generality of our method, we further take multiple passages as source context.

We experiment with top-K passages, where $K = 5$, to minimize the loss from length truncation due to model input limits, and conduct fairer experiments than using larger Ks. We use Flan-T5 since it has more consistent behaviors across tasks.

As shown in Table 4, FilCo surpasses Full and Psgs settings by a large margin, $+1.2-14.2$ points in all six tasks. FilCo also outperforms existing performant baselines: RAG (Lewis et al., 2020), FiD (Izacard and Grave, 2021), and evidentiality-guided (Evi.) generation (Asai et al., 2022).

Compared to using top-1 passages only, performance increases on extractive tasks when aggregating multiple top-ranked passages. Interestingly, performance on FEVER and WoW drops by -3.2 and -2.3 points, potentially due to the degraded retrieval quality of lower-ranked passages, as the top-1 passage recall is relatively high.

Context	NQ	TQA	HotpotQA	ELI5	FEVER	WoW
Baseline, top 5						
RAG	44.5	56.8	-	-	88.1	13.8
FiD	48.3	67.2	-	-	89.5	16.9
Evi.	49.8	67.8	-	-	89.8	17.9
FilCo, top 1						
FilCo	44.7	59.0	60.0	73.8	**94.2**	**68.3**
FilCo, top 5						
Full	47.6	67.3	61.5	72.7	88.0	64.8
Psgs	52.9	69.1	62.3	73.7	90.7	64.6
FilCo	**61.8**	**71.1**	**65.0**	**73.9**	91.4	66.0
Silver	62.0	71.1	65.2	73.9	92.2	66.1

Table 4: Generation results when providing top-5 retrieved passages filtered by passages or sentences. RAG, FiD, and Evi. are existing performant methods. We **bold-type** the best results that do not use silver contexts.

5 Comparing Context Filtering Strategies

To justify the context filtering strategies in §4, we compare the measures in §2 (StrInc, Lexical, and CXMI) on individual tasks.

5.1 Results with Different Strategies

Results in Table 5 reveal that different tasks benefit the most from different measures. NQ and TQA favor StrInc, WoW works best with Lexical, while more complex tasks (FEVER, HotpotQA, and ELI5) perform the best using CXMI. Model wise, Flan-T5 and Llama2 align on most tasks, with slight divergence on ELI5.

Measure	StrInc	Lexical	CXMI
Flan-T5			
NQ	**44.7**	30.0	39.9
TQA	**59.2**	39.0	45.3
HotpotQA	59.2	57.4	**60.0**
ELI5	73.6	73.9	**74.2**
FEVER	80.9	86.4	**95.8**
WoW	63.4	**69.3**	66.6
Llama 2			
NQ	**43.3**	35.2	41.8
TQA	**60.7**	57.1	**60.7**
HotpotQA	59.5	61.1	**61.3**
ELI5	78.6	**78.8**	72.8
FEVER	86.6	88.4	**92.3**
WoW	65.5	**66.0**	65.4

Table 5: Results using varied context filtering measures.

5.2 In-Depth Analysis for Different Tasks

Extractive tasks performs the best with an StrInc context filter, which reasonably aligns with their extractive nature. While StrInc strategy falls short on *abstractive tasks* due to empty filtered content, Lexical flexibly measures unigram-level matches and is the most suitable for *dialog generation*.

q The horse began to become domesticated around 2000 BC.
o REFUTES

P Domestication of the horse A number of hypotheses exist on many of the key issues regarding the domestication of the horse. Although horses appeared in Paleolithic cave art as early as 30,000 BCE, these were wild horses and were probably hunted for meat. How and when horses became domesticated is disputed. The clearest evidence of early use of the horse as a means of transport is from chariot burials dated c. 2000 BCE. However, an increasing amount of evidence supports the hypothesis that horses were domesticated in the Eurasian Steppes approximately 3500 BCE; recent discoveries in ...

Figure 7: Filter outputs on a FEVER example. StrInc gets empty context, Lexical selects red misleading content, and CXMI selects green supporting content.

CXMI works the best for more complex tasks, i.e., *multi-hop QA*, *long-form QA*, and *fact verification*. Taking the fact verification task in Figure 7 for example, while "REFUTED" claims often contain noisy contents, such as "2000 BC", lexical measures would falsely pick the misleading content that matches "2000 BC" but is about "evidence of early use" instead of "become domesticated". Augmenting this can reinforce the misleading fact ("2000 BC") and deteriorate model generation. In comparison, selecting only the content supportive of making factual judgment can inform that horses became domesticated around "3500 BCE".

6 Related Work

Augmented Generation Providing additional contexts to generation are effective (Lewis et al., 2020; Guu et al., 2020; Mialon et al., 2023) to many knowledge-intensive tasks (Petroni et al., 2021). Many works explored retrieval at varied granularity: paragraph (Lee et al., 2019; Feldman and El-Yaniv, 2019), phrase (Lee et al., 2021), and even token levels (Khandelwal et al., 2020; Alon et al., 2022). They all revealed a trade-off between retrieval and generation difficulty: it is easier to retrieve longer sequences, but harder to generate from them, presumably because of the noisier content impairing models (Shi et al., 2023). Our method allows arbitrary passage sizes at retrieval time, and alleviates this in-passage distraction via filtering.

Optimizing Retrieval for Augmentation Many works post-process retrieved content to augment the generation. A common way is to rerank passages and provide only the top few under limited input capacity, based on the query-passage similarity (Nogueira and Cho, 2020), reader prediction majority (Mao et al., 2021), and utility for generation (Wang et al., 2018). Asai et al. (2022) removes passages having low evidentiality scores, and (Mallen et al., 2023) skips retrieval when unnecessary. Nonetheless, these methods operate on the coarse passage level, thus still suffering from in-passage distractions. Our method operates at a more fine-grained sentence level, and can capture more subtle variances.

7 Conclusion

We propose a context filtering method, FilCo, to provide precisely supportive content to assist model generations, which effectively removes distracting content in both passages partially supporting and irrelevant to the queries. Applying our method brings an average of 2.8 and 3.0 point increase with Flan-T5 and Llama2, across six knowledge-intensive language datasets from question answering, fact verification, to knowledge-grounded dialog generation. Our work also reveals varied recipes to effectively filter context for different tasks. We hope that FilCo can facilitate more developments toward faithful generations in more scenarios.

Limitations

Our proposed method has been shown effective across various tasks, however, may be in certain data domains, under automatic evaluation metrics, and with sufficient computational resources.

Our approach is domain-agnostic in principle, however, all the datasets we experiment with are built from Wikipedia articles, i.e., the open domain. Tasks of other domains such as news (Trischler et al., 2017), biomedical knowledge (Nentidis et al., 2023), and even fictional stories (Kočiský et al., 2018; Xu et al., 2022), can readily adopt our method and potentially benefit from it. Nonetheless, we encourage readers to verify its effectiveness before directly extrapolating our conclusion to special-domain datasets.

We evaluate model retrieval, filtering, and generation performance using automatic metrics such as Exact Match and Unigram F1, which have become the standard metrics. Beyond lexical-based metrics, we keep open to neural- or human-based evaluations, given the potentially inaccurate automatic measures, especially with increasingly complex tasks (Pugaliya et al., 2019) and models of greater capacities (Kamalloo et al., 2023).

Our method requires training models to (i) filter context, and (ii) generate output, which necessitates certain computational resources, according to the model architecture and size of choice. Nonetheless, our method costs less computation compared to traditional full-passage augmentation. As shown by §5, a generation model with filtered content requires at least 4.7 times less computation, at both training and inference time.

References

Uri Alon, Frank F. Xu, Junxian He, Sudipta Sengupta, Dan Roth, and Graham Neubig. 2022. Neuro-symbolic language modeling with automaton-augmented retrieval. In *ICML 2022 Workshop on Knowledge Retrieval and Language Models*.

Akari Asai, Matt Gardner, and Hannaneh Hajishirzi. 2022. Evidentiality-guided generation for knowledge-intensive NLP tasks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2226–2243, Seattle, United States. Association for Computational Linguistics.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of Wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

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

Patrick Fernandes, Kayo Yin, Graham Neubig, and André F. T. Martins. 2021. Measuring and increasing context usage in context-aware machine translation. 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 6467–6478, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *International Conference on Machine Learning*. JMLR.org.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*.

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.

Ehsan Kamalloo, Nouha Dziri, Charles LA Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. *arXiv preprint arXiv:2305.06984*.

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.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-augmented dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, Dublin, Ireland. 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.

Jinhyuk Lee, Alexander Wettig, and Danqi Chen. 2021. Phrase retrieval learns passage retrieval, too. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3661–3672, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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, 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*, volume 33, pages 9459–9474.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Reader-guided passage reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 344–350, Online. Association for Computational Linguistics.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.

Anastasios Nentidis, Anastasia Krithara, Georgios Paliouras, Eulàlia Farré-Maduell, Salvador Lima-López, and Martin Krallinger. 2023. Bioasq at clef2023: The eleventh edition of the large-scale biomedical semantic indexing and question answering challenge. In *Advances in Information Retrieval*.

Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, pages 2523–2544, Online. Association for Computational Linguistics.

Hemant Pugaliya, James Route, Kaixin Ma, Yixuan Geng, and Eric Nyberg. 2019. Bend but don't break? multi-challenge stress test for QA models. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 125–136, Hong Kong, China. Association for Computational Linguistics.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. *arXiv preprint arXiv:2302.00093*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ying Xu, Dakuo Wang, Mo Yu, Daniel Ritchie, Bingsheng Yao, Tongshuang Wu, Zheng Zhang, Toby Li, Nora Bradford, Branda Sun, Tran Hoang, Yisi Sang, Yufang Hou, Xiaojuan Ma, Diyi Yang, Nanyun Peng, Zhou Yu, and Mark Warschauer. 2022. Fantastic questions and where to find them: FairytaleQA – an authentic dataset for narrative comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 447–460, Dublin, Ireland. 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.