Meta-training with Demonstration Retrieval for Efficient Few-shot Learning

Aaron Mueller[1*] **Kanika Narang**[2] **Lambert Mathias**[2]
Qifan Wang[2] **Hamed Firooz**[2]
[1] Johns Hopkins University, Baltimore, MD
[2] Meta AI, Menlo Park, CA
amueller@jhu.edu, {kanika13,mathiasl,wqfcr,mhfirooz}@meta.com

Abstract

Large language models show impressive results on few-shot NLP tasks. However, these models are memory and computation-intensive. Meta-training allows one to leverage smaller models for few-shot generalization in a domain-general and task-agnostic manner (Min et al., 2022a; Wei et al., 2022; Chen et al., 2022); however, these methods alone results in models that may not have sufficient parameterization or knowledge to adapt quickly to a large variety of tasks. To overcome this issue, we propose meta-training *with demonstration retrieval*, where we use a dense passage retriever to retrieve semantically similar labeled demonstrations to each example for more varied supervision. By separating external knowledge from model parameters, we can use meta-training to train parameter-efficient models that generalize well on a larger variety of tasks. We construct a meta-training set from UnifiedQA and CrossFit, and propose a demonstration bank based on UnifiedQA tasks. To our knowledge, our work is the first to combine retrieval with meta-training, to use DPR models to retrieve demonstrations, and to leverage demonstrations from many tasks simultaneously, rather than randomly sampling demonstrations from the training set of the target task. Our approach outperforms a variety of targeted parameter-efficient and retrieval-augmented few-shot methods on QA, NLI, and text classification tasks (including SQuAD, QNLI, and TREC). Our approach can be meta-trained and fine-tuned quickly on a single GPU.

1 Introduction

Large language models (LLMs) have become increasingly popular due to their impressive few-shot performance on many NLP tasks and domains (Brown et al., 2020; Chowdhery et al., 2022). This has resulted in many few-shot learning methods based on LLMs that require ever-larger GPUs and increasing computation. Methods requiring no parameter updates such as in-context learning (Brown et al., 2020) and parameter-efficient methods like Adapters (Houlsby et al., 2019) partially mitigate these downsides, but ultimately, larger computation budgets are increasingly necessary to achieve state-of-the-art few-shot performance—even to simply load models and perform inference.

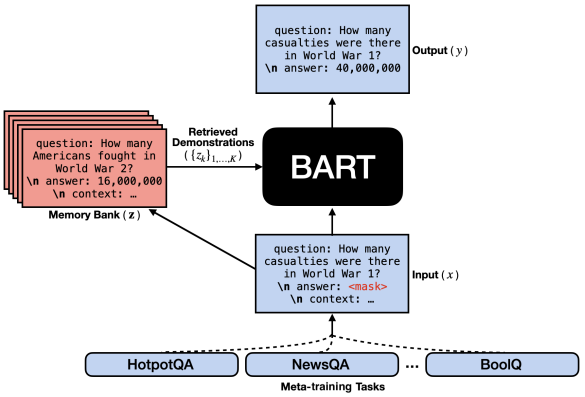

Figure 1: Our approach. Given an input x from one of many possible QA tasks, we use a dense passage retriever to retrieve K semantically similar demonstrations $Z = \{z_k\}_{1,\dots,K}$ from a memory bank $\mathbf{z}$ composed of labeled examples. We meta-train BART, supervising it to generate the (question and) answer y given x and Z across a diverse collection of QA tasks.

Meta-learning (Vilalta and Drissi, 2002; Finn et al., 2017) and meta-training (Min et al., 2022a) are methods that make smaller language models capable of quicker and more robust few-shot performance across multiple tasks and domains. However, smaller models may not be able to store enough knowledge for effective generalization in many domains and tasks simultaneously. Retrieval is one way to overcome this: by separating parametric knowledge in the language model from external knowledge (stored as retrievable text), one can leverage much more information than could be stored in the parameters of a language model. For example, retrieval-augmented generation (RAG;

[*] Work done as an intern at Meta.

Findings of the Association for Computational Linguistics: ACL 2023, pages 6049–6064
July 9-14, 2023

Lewis et al., 2020) and retrieval-enhanced transformers (RETRO; Borgeaud et al., 2022) retrieve natural language passages to improve performance on knowledge-intensive NLP tasks, although they do not perform meta-learning or meta-training and only evaluate on high-resource knowledge-intensive tasks.

We thus propose **meta-training with demonstration retrieval** as a more parameter-efficient way to leverage demonstrations for few-shot learning. We retrieve semantically similar labeled demonstrations for each training and test example during meta-training *and* fine-tuning. On a relatively small sequence-to-sequence model ($\text{BART}_{\text{large}}$, 440M parameters), we show our proposed approach is capable of generalizing quickly and well on a variety of downstream tasks (Table 1). Inspired by retrieval-augmented generation (RAG) models (Lewis et al., 2020), we use a dense passage retriever (DPR; Karpukhin et al., 2020) to retrieve demonstrations instead of Wikipedia passages. We retrieve semantically similar demonstrations from a large and diverse bank (§3.3) that is compiled from many existing question answering tasks (App. A), rather than randomly sampling demonstrations from the training set of the target task like most contemporary work (Min et al., 2022a; Brown et al., 2020; Gao et al., 2021).

Our experiments show that our method (§3) outperforms tailored efficient few-shot baselines and other retrieval-augmented models on various tasks, including natural language inference (NLI), paraphrase detection, and extractive question answering (§5). To our knowledge, our work is the first to combine retrieval with meta-training (or multi-task training more broadly), to use DPR models to retrieve demonstrations, and to leverage demonstrations from many tasks simultaneously, rather than retrieving random or k-nearest demonstrations from the training set of the target task.

Our code is available on GitHub.[1]

2 Related Work

Meta-learning (Vilalta and Drissi, 2002; Finn et al., 2017) is a class of methods that supervise a model on *how to learn*; the goal is to leverage a collection of meta-training tasks to learn a better learning algorithm that generalizes to held-out tasks. Inspired by meta-learning, some recent studies have attempted to induce specific abilities in language models in a task- and domain-agnostic manner via **meta-training**; this entails directly supervising a model on labeled examples from various tasks (sometimes using some controlled format or template (Chen et al., 2022; Wei et al., 2022)) to directly induce specific abilities or better inductive biases that improve generalization. Meta-training is typically accomplished via some form of controlled multi-task learning, as in Min et al. (2022a). Many studies have explored multi-task and multi-domain learning (Khashabi et al., 2020; Zhong et al., 2021; Aghajanyan et al., 2021; Ye et al., 2021; Wei et al., 2022), but these studies often leverage tasks that improve a model's abilities for some specific (set of) downstream tasks. In meta-training, we aim to directly improve the learning algorithm via controlled supervision, which should improve out-of-distribution generalization by teaching a model some helpful ability—such as in-context learning—that can result in gains on various downstream tasks (Min et al., 2022a). We focus on meta-training with examples from QA datasets.

Few-shot learning is a common setting in which a model is supervised on only a few labeled examples. Many methods for improving few-shot performance are based on scaling model and data size (Brown et al., 2020; Chowdhery et al., 2022). Our goal is to improve few-shot performance across tasks in a computation- and memory-efficient manner, so we focus on smaller models that can be trained efficiently on a single GPU. Some parameter-efficient few-shot methods have been proposed, including cloze-style prompting (Schick and Schütze, 2021b), fine-tuning with manually tuned (Schick and Schütze, 2021a) and automatically tuned prompts and demonstrations (Gao et al., 2021), and meta-learning (Yu et al., 2018; Bansal et al., 2020; Bao et al., 2020). One advantage of our approach is that it does not require significant prompt tuning: rather, we standardize all of our tasks into a single format, similar to Chada and Natarajan (2021). This saves human time and computational resources.

Crucially, these approaches compare probabilities of single tokens or small pre-selected label sets; thus, they cannot be used for open-domain tasks like question answering. Some work has proposed *generative* few-shot methods for open-domain tasks: this includes reformatting the input

[1] https://github.com/facebookresearch/metatrained-demRAG

data to match a model's pre-training format (Chada and Natarajan, 2021), pre-training models to select relevant spans from context passages (Ram et al., 2021), and running a secondary pre-training step on labeled classification data (Mueller et al., 2022). Our model should be effective on many tasks, even when the label space is large and differs across examples; thus, our method is based on a *generative* sequence-to-sequence model.

In-context learning (ICL; Brown et al., 2020) is increasingly used in few-shot methods; here, labeled *demonstrations* are concatenated to the same context as a test example to teach a model how to perform a task without additional gradient updates. Studies have analyzed what kinds of demonstrations are most effective (Liu et al., 2022), as well as what makes demonstrations effective (Min et al., 2022b; Xie et al., 2022). Our demonstration retrieval approach is most similar to Liu et al. (2022), who encode demonstrations and test examples into a sentence embedding space and retrieve the k-nearest demonstrations. Our method differs in multiple ways: we use dense passage retrievers instead of sentence embeddings; we use demonstrations from many training sets instead of the training set of the target task; and we perform gradient updates with demonstrations, which is more feasible on our relatively small $\text{BART}_{\text{large}}$-based model.

Wei et al. (2022) find that very large LMs (>68B parameters) are required for ICL to be effective, but Min et al. (2022a) find that meta-training can be used to make a much smaller model ($\text{GPT2}_{\text{large}}$, 774M parameters) capable of leveraging demonstrations. Here, we make $\text{BART}_{\text{large}}$ (440M parameters) better at leveraging demonstrations through meta-training with demonstrations, like Min et al. (2022a); however, their method is designed for zero-shot generalization, and it selects from a constrained set of pre-defined labels. Our method is designed for *few-shot* settings and can be applied to open-domain tasks.

Retrieval-augmented generation models consist of two components: *generators* and *retrievers*. The generator is typically a decoder-only LM (Guu et al., 2020) or sequence-to-sequence (seq2seq) model (Lewis et al., 2020; Izacard and Grave, 2021); we use seq2seq models. The retriever is most often a dense passage retrieval (DPR; Karpukhin et al., 2020) model based on $\text{BERT}_{\text{base}}$. RAG models are typically evaluated on knowledge-intensive tasks like abstractive QA and fact verification. Thus, the memory bank typically consists of Wikipedia passages, which augments the model with additional factual knowledge separate from the generator's parameters. Izacard et al. (2022) adapts this architecture for few-shot knowledge-intensive tasks using a very large generator ($\text{T5}_{\text{X(X)L}}$) and a Contriever-based (Izacard et al., 2021) retriever. However, we are interested in more general-purpose methods, as well as more parameter- and memory-efficient methods that train or fine-tune quickly on a single GPU. Thus, we propose a task-agnostic and domain-general method to improve smaller generative models for few-shot settings: specifically, a retrieval-augmented meta-training step and a memory bank of labeled QA demonstrations instead of Wikipedia passages.

3 Method

3.1 Retrieval-augmented Generation

As we wish to retrieve similar labeled examples for every input, our architecture takes inspiration from retrieval-augmented generation (RAG) models (Lewis et al., 2020), which consist of a pre-trained sequence-to-sequence component (we use $\text{BART}_{\text{large}}$) and a pre-trained dense passage retriever (DPR) component. Given an input x, the DPR component retrieves the K most semantically similar memory entries $\{z_k\}_{1,\ldots,K}$ from the memory bank $\mathbf{z}$. Retrieval is performed using a BERT-based input encoder E_I on x and BERT-based demonstration encoder E_D on $\mathbf{z}$ to encode both into a vector space, and then running maximum inner product search:[2]

$$\{z_k\}_{1,\ldots,K} = \underset{z \in \mathbf{z}}{\text{top-}K} \left\{ E_I(x)^\top E_D(z)) \right\} \quad (1)$$

The DPR component also returns the inner products themselves as document scores $p_\eta(z_k|x)$.

The input and retrieved entries are then passed to a pre-trained sequence-to-sequence model, $\text{BART}_{\text{large}}$, for autoregressive generation. At each timestep, we marginalize over the retrieved demonstrations by creating K separate input contexts, consisting of the input x and one retrieved entry z_k. We then sum over BART's token probabilities p_θ given each context, weighted by z_k's document

[2]Maximum inner product search can be approximately solved in sub-linear time (Johnson et al., 2021). We use the `faiss` library for this: https://github.com/facebookresearch/faiss

Category	Dataset	Type	#Train	#Test	L	$\lvert\mathcal{Y}\rvert$
Extractive QA	SQuAD (Rajpurkar et al., 2016)	Open QA	86,588	10,507	10 / 120	-
	BioASQ (Tsatsaronis et al., 2015)	Open QA	24,559	1,504	10 / 200	-
	QASC (Khot et al., 2020)	Multi-choice QA	8,134	926	8 / 18	-
Knowledge-intensive QA	TriviaQA (Joshi et al., 2017)	Open QA	61,688	7,785	13 / 677	-
	TextbookQA (Kembhavi et al., 2017)	Open QA	15,154	1,503	10 / 581	-
Classification	TREC (Voorhees and Tice, 2000)	Question class.	5,452	500	10	6
	MRPC (Dolan and Brockett, 2005)	Paraphrase class.	3,668	408	22 / 21	2
	MNLI (Williams et al., 2018)	NLI	392,702	9,815	22 / 11	3
	MNLI-mm (*ibid.*)	NLI	392,702	9,832	22 / 11	3
	QNLI (Wang et al., 2018)	NLI	104,743	5,463	11 / 30	3

Table 1: Evaluation sets used in this study. L: mean # of words in question/context or input sentence(s). For more straightforward comparison to prior few-shot question answering and classification methods, we use Ram et al. (2021)'s few-shot splits of SQuAD and BioASQ derived from MRQA, as well as Gao et al. (2021)'s splits of TREC, MRPC, MNLI(-mm), and QNLI. We generate our own few-shot splits of QASC using 5 random seeds for each split size.

score:[3]

$$p(y|x) \approx \prod_{i}^{N} \sum_{k}^{K} p_\eta(z_k|x) p_\theta(y|x, z_k, y_{1:i-1}) \tag{2}$$

3.2 Meta-training

To adapt a sequence-to-sequence model for general-purpose demonstration retrieval and answer generation, we perform a meta-training step by supervising the model with demonstrations on a collection[4] of 18 QA tasks (Table 7). We update the parameters of the BART component of our model during meta-training by supervising BART (using its normal cross-entropy loss) to generate the question and its answer given the question and a set of retrieved demonstrations. We use QA tasks due to the semantic diversity of inputs and labels; compare to text classification tasks, where the label space is much smaller and labels are often less informative.

We modify and use the QA meta-training task collections from (Min et al., 2022a).This consists of various extractive, multiple-choice, and/or abstractive QA tasks from CROSSFIT *and* a subsample of UNIFIEDQA (Khashabi et al., 2020, 2022), including NaturalQuestions, MCTest, BIOMRC, *inter alia*. We modify the meta-training collections by (1) removing our evaluation sets if they are present,[5] and (2) standardizing the format of each task. Our final meta-training collection contains 32 tasks, which we subsample to 18 tasks based on semantic similarity to our evaluation tasks; see Appendix A for a full list of tasks and details on our semantic subsampling procedure, and §5.2 for a description of the downstream effect of semantic subsampling.

Following Chada and Natarajan (2021), we standardize each input in the meta-training data to a "`question:... \n answer: [MASK] \n context:...`" format. Then, the output sequence consists of both the question *and* answer sequences,[6] which aligns with BART's pre-training objective of reconstructing the entire input sequence (not just masked spans). Like Chada and Natarajan (2021), we find that aligning the input/output format with BART's pre-training objective makes a positive difference for downstream performance. For QASC, which is a multiple-choice QA task, we put all of the answer options in the `context` field before the two context sentences and generate the full answer string. This outperformed all other formats we tried by a significant margin.[7]

For classification tasks, we use the same question/answer/context format. For our single-sentence classification task (TREC), we place the input in the `question` field, and present all of the possible labels in the `context` field using a similar format as for QASC. For sentence-pair classifica-

[3]This is similar to the RAG-Token approach in Lewis et al. (2020). The number of demonstrations we can use is *not* limited by the context length since we marginalize over each demonstration in its own separate context.

[4]Throughout this study, we use "task" to refer to a single dataset like SQuAD or NaturalQuestions, and "collection" to refer to the dataset obtained by concatenating a set of tasks.

[5]We also remove any examples where the question has a Jaccard similarity > 0.9 with *any* training or test question in our evaluation tasks, *and* where the answers are the same; only 4 such examples existed in our data.

[6]We only compute F_1 on the answer sequences.

[7]We tried placing the answer options in the `question` field, not including the answer options at all, and only generating the letter label instead of the full answer string. See Appendix B for examples and scores.

tion tasks (MRPC, MNLI(-mm), QNLI), we place the first sentence or hypothesis in the `question` field and place the second sentence or premise in the `context` field. As with QA tasks, we generate both the question and answer fields in the target sequence, but only evaluate F_1 on answer sequences.

3.3 Demonstration Memory

For the demonstration memory bank, we use training sets from UnifiedQA, excluding our evaluation tasks; the memory contains examples from 16 tasks. UnifiedQA has approximately 40% overlap with the QA meta-training collection, and no overlap with the non-QA collection. See Table 8 in Appendix A for a full list of tasks in our demonstration memory bank.

We format each demonstration in the memory bank in the same question/answer/context format as described above, except that demonstrations have the ground-truth label after the `answer:` header instead of a `[MASK]` token. Note that memory entries consist of a text passage (the demonstration) *and* a title; for the title, we simply use the answer to the question.

4 Experimental Setup

We evaluate on a variety of QA and classification tasks (Table 1). We select open-domain QA tasks from the MRQA shared task (Fisch et al., 2019) to reflect a variety of extractive QA formats, including a standard QA benchmark (SQuAD), a domain-specific challenging benchmark (BioASQ), and two knowledge-intensive QA benchmarks (TriviaQA and TextbookQA).[8] Our few-shot QA splits of size {16, 32, 64, 128} for these tasks are from Ram et al. (2021), which are themselves derived from MRQA (Fisch et al., 2019). We also generate few-shot splits for QASC, which is a multiple-choice QA task; we evaluate on QASC to determine whether our model is also effective in dealing with much shorter contexts, and to ensure that it is not overfitting to more typical MRQA-style extractive tasks.

Our few-shot classification task splits are from Gao et al. (2021). We evaluate on sentence pair classification tasks which are not contained in our meta-training or demonstration tasks; sentence pair classification tasks like natural language inference (NLI) and paraphrase classification can be easily reformatted to our question/answer/context format. We also evaluate on TREC, which is a single-sentence text classification task where the model must guess the *category* of the answer to a question (e.g., human, location, number), rather than the answer itself.

For each task and few-shot split size, we average scores across 5 random few-shot samples.

4.1 Baselines

We compare against strong efficient few-shot methods, as well as similar models that will tell us *why* our method performs better. Note that our approach is *generative*, unlike iPET and LM-BFF; thus, it is usable on a wider variety of tasks.

FewshotQA (Chada and Natarajan, 2021). A few-shot question answering method. We compare to the FewshotBARTL model, which is based on $\text{BART}_{\text{large}}$ like our model and is the best-performing variant. We use the same few-shot splits such that we can directly compare to the numbers reported in that paper. We also try meta-training this non-retrieval-augmented model, which is essentially our method without retrieval; we call this baseline FewshotQA-m.

Splinter (Ram et al., 2021). A few-shot question answering model pre-trained to select salient spans from context passages.

RAG (Lewis et al., 2020). The original RAG-Token model with a memory of Wikipedia passages. We use the released model fine-tuned on NaturalQuestions (NQ), as this was the best-performing RAG model on our tasks. To see whether our demonstration memory is more effective than Wikipedia passages when meta-training, we also try meta-training the RAG model with its Wikipedia memory; we call this baseline RAG-m.

iPET (Schick and Schütze, 2021b). A manual prompt-tuning approach that induces better few-shot performance than GPT3 with much smaller LMs. We tune the best-performing $\text{ALBERT}_{\text{xxl}}$ (Lan et al., 2020) model on our tasks.

LM-BFF (Gao et al., 2021). An automatic prompt-tuning approach based on $\text{RoBERTa}_{\text{large}}$ (Liu et al., 2019). It requires no unlabeled text data to work well, unlike iPET. This model and iPET compare token probabilities to perform classifica-

[8] While "knowledge-intensive" does not have a standard definition or straightforward measurement, the length of the contexts may act as a proxy for how knowledge-intensive a question answering task is. Contexts for our knowledge-intensive tasks are much longer, and thus require a model to synthesize much more information and/or retrieve information that is more relevant to the inputs to semantically prime the model for question-specific information.

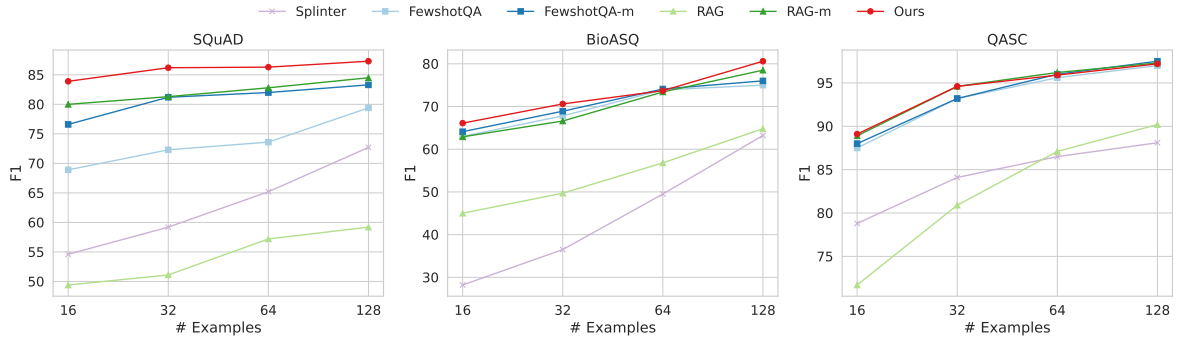

Figure 2: F_1 scores at each few-shot split size for extractive and multiple-choice question answering evaluation tasks. Scores are averaged across 5 random few-shot samples. Our model outperforms or maintains similar performance to the strongest baselines on each task and split size. Performance gains on SQuAD are especially large—up to 3.9 F_1 (4.9% improvement). FewshotQA and Splinter scores are from Chada and Natarajan (2021).

tion, so we cannot use them for open-domain tasks like question answering. Thus, we only compare to these models on classification.

4.2 Hyperparameters

For meta-training, we use hyperparameters from Min et al. (2022a) where possible: init. LR 1×10^{-5}, effective batch size 8,[9] training for a maximum of 30,000 steps. We checkpoint every 2,000 steps and select the checkpoint with the lowest mean loss on our 16-shot QA training sets. Meta-training finishes in ≈14 hours on 1 A100 GPU (40GB).[10]

For fine-tuning, we use hyperparameters from Chada and Natarajan (2021) where possible: init. LR 2×10^{-5}, batch size 4, fine-tuning for a maximum of 1,000 steps or 35 epochs (whichever is larger). We checkpoint every 2 epochs and select the checkpoint with the highest exact match on the training set. Fine-tuning finishes in 30–60 minutes on 1 A100 GPU (40GB).

For each meta-training and fine-tuning input, we retrieve 5 demonstrations from the memory.[11]

5 Results

Our model's F_1 scores for extractive question answering (Figure 2) are higher than models of similar parameterizations, including similar models that have been meta-trained using the same training data. Our model also outperforms strong classification approaches on TREC, MNLI, and QNLI (Table 2). Thus, **meta-training with semantically similar demonstrations induces a more general-purpose system that can perform well across a variety of low-resource downstream tasks.**

	TREC	MNLI	MNLI-mm	QNLI	MRPC	*Avg.*
Majority	18.8	32.7	33.3	49.5	81.2	*43.1*
RoBERTa	*88.8$_{2.1}$	*45.8$_{6.4}$	*47.8$_{6.8}$	*60.2$_{6.5}$	76.6$_{2.5}$	*63.8*
iPET	*85.0$_{4.1}$	71.2$_{1.7}$	71.8$_{2.6}$	*70.3$_{6.2}$	70.4$_{4.7}$	*73.7*
LM-BFF	*89.4$_{1.7}$	70.7$_{1.3}$	***72.0**$_{1.2}$	*69.2$_{1.9}$	***78.1**$_{3.4}$	*75.9*
FewshotQA	91.0$_{2.0}$	*47.9$_{6.3}$	*46.1$_{5.9}$	*61.0$_{6.4}$	*67.6$_{4.8}$	*62.7*
FewshotQA-m	**92.4**$_{1.4}$	*50.1$_{1.0}$	*50.6$_{2.5}$	*71.8$_{2.1}$	74.0$_{3.7}$	*67.8*
RAG	*81.1$_{2.0}$	*62.4$_{0.9}$	*61.8$_{1.2}$	*74.9$_{1.5}$	70.2$_{3.3}$	*70.1*
RAG-m	*87.8$_{1.7}$	*70.0$_{1.4}$	69.1$_{1.4}$	*83.2$_{1.5}$	74.9$_{2.8}$	*77.0*
Ours	91.7$_{1.3}$	**72.9**$_{1.7}$	69.6$_{1.4}$	**84.4**$_{1.8}$	73.4$_{2.5}$	***78.4***

Table 2: Accuracies on classification tasks, averaged across 5 random few-shot samples (std. dev. in subscript). All datasets are well-balanced except MRPC; thus, we report accuracies for all tasks except MRPC, where we report macro-F_1. LM-BFF and RoBERTa scores are from Gao et al. (2021). * indicates that $p < .05$ in a t-test between our model's score and the marked score.

Contrast this with RAG, which often performs *worst* out of each model we test across tasks. Thus, the architecture itself is not inherently strong in few-shot settings, suggesting that meta-training makes a significant contribution to increased performance. This is also supported by the increased performance we observe with FewshotQA and RAG after meta-training, though note that meta-training does not help FewshotQA to the same extent it helps retrieval-augmented models. Also note that FewshotQA does not perform well on classification tasks, whereas our method achieves performance exceeding or close to the strongest baselines. This means that the combination of meta-training and retrieval enables a more general-purpose model than

[9] We use gradient accumulation to get this effective batch size on a single GPU.

[10] Our model could also be trained and tuned on a cheaper 32GB GPU (e.g., a V100) in similar time.

[11] Higher values result in better performance (§5.2), but this saturates at 5–10 retrieved demonstrations, and retrieving more demonstrations slows down training.

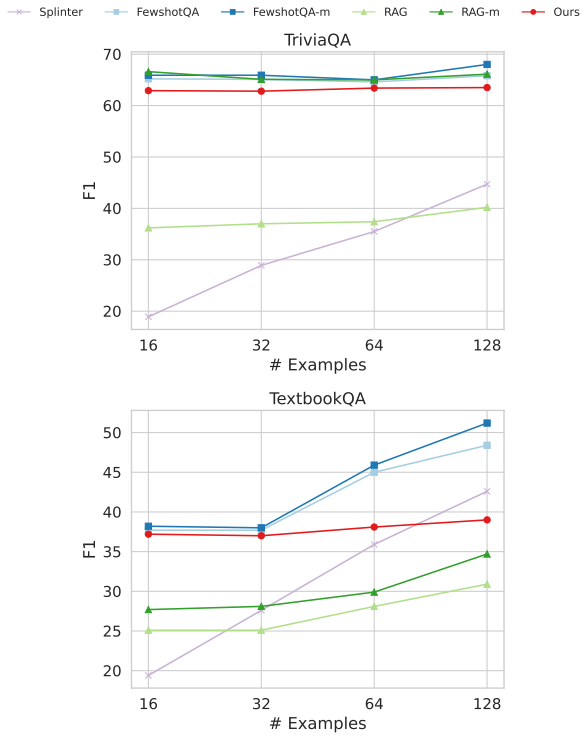

Figure 3: F_1 scores for each few-shot split size for knowledge-intensive question answering tasks. Our model is outperformed by a strong few-shot QA baseline, though meta-training still greatly improves performance.

either of these components separately.

With meta-training, RAG-m obtains performance much closer to our model. This tells us that **meta-training is responsible for much of the performance gains we observe**, though the demonstration memory bank also improves performance to a lesser extent. On MRPC, RAG-m outperforms our model, indicating that there exist some non-knowledge-intensive tasks where Wikipedia passages are more helpful than QA demonstrations.

5.1 Knowledge-intensive QA

We also evaluate on few-shot knowledge-intensive QA tasks (Figure 3): here, TriviaQA and TextbookQA, using the few-shot splits from the MRQA shared task. While these are also technically extractive QA tasks, their contexts have an average length of 677 and 581 words, respectively, meaning that BART will likely struggle more to synthesize all of the information in these tasks (even with retrieval). We find that FewshotQA outperforms our method on both of these tasks, and that even Splinter outperforms our method at larger split sizes for TextbookQA. This means that demonstration retrieval

Model	SQuAD	BioASQ	QASC	TriviaQA	TbQA
FewshotQA	68.9	63.0	82.6	65.2	37.7
FewshotQA-m	76.6	63.4	85.9	65.9	**38.2**
RAG-m	80.0	62.9	88.9	**66.6**	27.7
Ours	**83.9**	**64.7**	**89.2**	62.9	37.2
Ours (*oracle*)	93.5	94.2	99.1	80.7	83.2

Table 3: F_1 scores on QA tasks for our strongest baselines, our approach, and our approach where the memory has been replaced with labeled test examples (*oracle*). The oracle approach establishes an approximate upper bound for our model. Large gaps between our approach and the oracle indicate room for improvement in what constitutes our memory bank.

may be actively harmful for these tasks. Thus, our meta-training method is optimizing RAG architectures for non-knowledge-intensive tasks, but not for knowledge-intensive tasks. Wikipedia passages are more effective than demonstrations in the memory bank for TriviaQA as well, as indicated by RAG-m outperforming our approach.

However, meta-training with or without the memory bank still induces far better performance than the base RAG model, which performs worse than all baselines except Splinter. Thus, our method is still improving over RAG, making this model more versatile and better able to handle such tasks even if it is not the optimal approach.

5.2 Ablations

Here, we perform further analyses to understand the contribution of individual model components and (meta-)training decisions.

Memory bank. We find that performance is generally higher for question answering and classification when retrieving demonstrations instead of Wikipedia passages, as in Figure 2 and Table 2. This raises two questions: how much could the memory bank impact downstream performance in the best-case scenario? Relatedly, what is the upper bound on performance for our model given the best possible demonstration memory bank?

To obtain an estimate, we create an *oracle* memory consisting of labeled test examples from our evaluation data. We find that scores significantly improve over our method and others in this setting, indicating that **this architecture has significant potential to achieve further gains if the memory bank is improved**.

Number of retrieved demonstrations. Is retrieving more demonstrations always better? We compare performance when retrieving

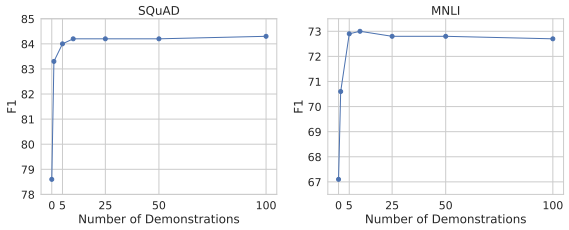

Figure 4: F_1 scores for an extractive QA (SQuAD) and sentence pair classification (MNLI) task by the number of retrieved demonstrations ($\{0, 1, 5, 10, 25, 50, 100\}$) during fine-tuning. Scores generally increase with the number of retrieved demonstrations, though performance saturates early at 5–10 demonstrations.

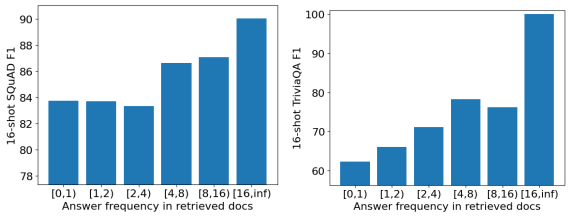

Figure 5: F_1 scores for non-knowledge-intensive (SQuAD, left) and knowledge-intensive (TriviaQA, right) QA tasks by the frequency of the true answer string in the retrieved demonstrations. While not monotonic, there is a clear correlation between these variables, indicating that lexical features may be responsible for much of retrieval's contributions to performance.

$K = \{0, 1, 5, 10, 25, 50\}$ demonstrations during fine-tuning and evaluation on non-knowledge-intensive QA (SQuAD) and sentence-pair classification (MNLI). Our results (Figure 4) show that F_1 scores begin to saturate at 5–10 demonstrations for both tasks. However, using more demonstrations generally does not harm performance; the model is able to handle less helpful demonstrations without performance decreasing significantly.

Why is retrieval helpful? Is the model abstracting semantic content from the retrieved demonstrations for improved performance, or is it simply learning to copy token sequences from the retrieved demonstrations? As an initial test, we can correlate the frequency of the ground-truth answer sequence in the retrieved documents with F_1 scores on our QA tasks. Our results (Figure 5) suggest that the model is indeed learning to retrieve certain text strings from the demonstrations. This provides one possible path forward for improving the memory bank: higher semantic overlap with one's evaluation task increases the likelihood of these overlaps, so future work could focus on collecting (or perhaps generating) more semantically similar demonstrations that feature more lexical overlaps.

Retriever	SQuAD	BioASQ	QASC	TriviaQA	TbQA
Random	*1.8*	*1.5*	*1.2*	*1.8*	*2.3*
DPR (Wiki)	11.5	1.8	15.7	4.9	**24.3**
DPR (PAQ)	**16.9**	1.5	26.1	**29.3**	24.0
Contriever	14.1	**7.3**	**28.0**	27.9	**24.3**

Table 4: The proportion of test examples for which each retriever retrieves at least 1 demonstration containing the ground-truth answer as a substring. DPR (PAQ) and Contriever appear to be better at retrieving more relevant demonstrations on average, though this does not necessarily lead to higher downstream performance (Table 5).

Retriever	SQuAD	BioASQ	QASC	TriviaQA	TbQA
Random	*74.3*	*61.8*	*88.7*	*56.5*	*29.6*
DPR (Wiki)	**83.9**	**64.7**	**89.2**	**62.9**	**37.2**
DPR (PAQ)	78.8	63.5	86.8	57.6	33.5
Contriever	81.1	62.5	88.7	58.9	32.4

Table 5: F_1 scores on 16-shot extractive QA tasks across retrievers. We fine-tune with different retrievers given the same (best) meta-trained model. Despite DPR (Wiki)'s lower retriever scores (Table 4), its downstream performance is the best among the retrievers we try.

However, this does not explain how retrieval improves performance on classification tasks, where the label space is small and labels are less informative. For NLI, the label space includes "entailment"/"neutral"/"contradiction", which we would not expect to see often in our demonstrations and which do not carry significant semantic content. Yet retrieval-augmented models outperform FewshotQA by a large margin on MNLI(-mm), so what is helping our model? There could exist some QA demonstrations which semantically prime our model toward correct completions, though sentence embedding similarity may not capture this helpfulness. Future work could ablate over specific features in the demonstrations.

What type of retriever is best? For our experiments thus far, we have used the DPR component of the RAG-Token (NQ) model, which is pre-trained on Wikipedia and fine-tuned on NaturalQuestions. Is this an optimal starting point, or would some other retreiver be better? We compare against a DPR model pre-trained on the Probably-Asked Questions (PAQ; Lewis et al., 2021) dataset, as well as the Contriever model (Izacard et al., 2021). Contrievers are unsupervised, whereas DPR models receive explicit supervision during pre-

Memory	SQuAD	BioASQ	QASC	TriviaQA	TbQA
All tasks	83.5	63.2	**89.2**	61.4	36.8
Semantically similar tasks	**83.9**	**64.7**	**89.2**	**62.9**	**37.2**

Table 6: 16-shot F_1 scores on QA tasks after meta-training on either all QA tasks from MetaICL's QA meta-training collection, or QA tasks subsampled by semantic similarity to our evaluation tasks. A full list of meta-training tasks can be found in Appendix A.

training. DPR tends to perform better when the downstream task is similar to the pre-training or fine-tuning data; however, in our case, demonstration retrieval is dissimilar from Wikipedia passage retrieval, and Contriever may handle larger train-test shifts better (Izacard et al., 2021).

We evaluate both the relevance of the retrieved demonstrations (Table 4) and downstream F_1 (Table 5) on our QA tasks. We find that DPR (PAQ) and Contriever are both better at retrieving similar demonstrations, as measured by the frequency with which they retrieve examples that contain the answer. For BioASQ, only Contriever retrieves more relevant demonstrations than a random retriever.

However, retrieving more relevant demonstrations does not translate into increased downstream performance: DPR (Wiki) consistently outperforms the others. Why? Through qualitative analysis, we find that DPR (Wiki) retrieves more semantically diverse demonstrations, whereas DPR (PAQ) and Contriever retrieve demonstrations that are technically more similar to the test example, but also less diverse *across* test examples.Thus, there should be a balance between diversity and relevance: completely random retrieval is not effective (as indicated by our random retrieval baseline scoring worst), but neither is the more constrained demonstration set we retrieve using an arguably more optimal retriever.

Meta-training data. Is meta-training helpful because of the variety of tasks included in our setup (the *more is better* hypothesis), or would it be better to select meta-training data in a more principled way (the *similar datasets are better* hypothesis)? We compare downstream performance when meta-training on all QA tasks from MetaICL versus the top tasks by mean instance-level semantic similarity to our evaluation tasks (Table 6). To compute semantic similarity, we use the `stsb-roberta-base-v2` model from SentenceTransformers (Reimers and Gurevych, 2019) and compute the mean pairwise cosine similarity between the 16-shot training examples in our evaluation tasks and all examples in a meta-training task. We then select the top tasks by similarity until we have over 240,000 examples (enough for 30,000 training steps using batch size 8). See Appendix A for a list of meta-training tasks before and after subsampling.

We find that **selecting meta-training data based on semantic similarity to our evaluation tasks is helpful for both our QA *and* non-QA tasks**: F_1 increases across tasks when only meta-training on the most similar data. This contrasts with the findings of Min et al. (2022a), who find that more meta-training tasks is generally better.

6 Conclusions

We have proposed a meta-training method (§3.2) that retrieves (§3.1) semantically similar demonstrations from a diverse demonstration bank (§3.3). Our method achieves higher performance on average across many tasks than other strong parameter-efficient few-shot baselines (§5). In future work, one could explore a mixture of demonstration retrieval and passage retrieval for improved performance on a wider variety of tasks—including knowledge-intensive tasks.

Limitations

Our method requires access to a large set of labeled examples for the memory bank—ideally with some relevance to the evaluation tasks. This limits the languages and tasks that are optimal for this method: there does not exist a large variety of training examples for low-resource language varieties, nor for certain much more specific tasks—as in, for example, industry applications with domain-specific customer data. And while multilingual models could leverage cross-lingual transfer, it is unclear how well this model would generalize into low-resource languages when (for example) using multilingual BART.

When using the full demonstration memory, meta-training does not run on a 16GB GPU using our current implementation. While this does exclude more common GPUs, our approach could still run quickly on a 32GB GPU in a few hours, thus costing far less than pre-training a language model of comparable few-shot performance from scratch.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5799–5811, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534, Online. Association for Computational Linguistics.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Rakesh Chada and Pradeep Natarajan. 2021. FewshotQA: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6081–6090, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2022. Meta-learning via language model in-context tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 719–730, Dublin, Ireland. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *CoRR*, abs/2204.02311.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering, MRQA@EMNLP 2019, Hong Kong, China, November 4, 2019*, pages 1–13. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. 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 3816–3830, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 874–880. Association for Computational Linguistics.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot learning with retrieval augmented language models. *CoRR*, abs/2208.03299.

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

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

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

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5376–5384.

Daniel Khashabi, Yeganeh Kordi, and Hannaneh Hajishirzi. 2022. Unifiedqa-v2: Stronger generalization via broader cross-format training. *CoRR*, abs/2202.12359.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1896–1907. Association for Computational Linguistics.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. Qasc: A dataset for question answering via sentence composition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8082–8090.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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. Curran Associates, Inc.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

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

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022a. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, Seattle, United States. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022b. Rethinking the role of demonstrations: What makes in-context learning work? *CoRR*, abs/2202.12837.

Aaron Mueller, Jason Krone, Salvatore Romeo, Saab Mansour, Elman Mansimov, Yi Zhang, and Dan Roth. 2022. Label semantic aware pre-training for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8318–8334, Dublin, Ireland. Association for Computational Linguistics.

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

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. 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 3066–3079, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artières, Axel-Cyrille Ngonga Ngomo, Norman Heino, Éric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinform.*, 16:138:1–138:28.

Ricardo Vilalta and Youssef Drissi. 2002. A perspective view and survey of meta-learning. *Artif. Intell. Rev.*, 18(2):77–95.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, page 200–207, New York, NY, USA. Association for Computing Machinery.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. 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. Association for Computational Linguistics.

Jason Wei, Maarten Paul Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew Mingbo Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. CrossFit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. 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 1206–1215, New Orleans, Louisiana. Association for Computational Linguistics.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2856–2878, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Tasks

Meta-training. Our meta-training data is from MetaICL's (Min et al., 2022a) meta-training sets. Specifically, we use the QA task collection from the paper, which is a mixture of CROSSFIT and UNIFIEDQA tasks as shown in Table 7. We exclude any task on which we evaluate. As in MetaICL, we subsample 16,384 examples per task such that no individual task is overrepresented during meta-training. Some tasks are sampled twice due to the inclusion of both CROSSFIT and UNIFIEDQA versions of some tasks, as in Min et al. (2022a).

All meta-training tasks: biomrc, boolq, freebase_qa, hotpot_qa, kilt_hotpotqa, kilt_nq, kilt_trex, kilt_zsre, lama-conceptnet, lama-google_re, lama-trex, mc_taco, numer_sense, quoref, ropes, search_qa, superglue-multirc, superglue-record, tweet_qa, web_questions, unifiedqa:boolq, unifiedqa:commonsenseqa, unifiedqa:drop, unifiedqa:narrativeqa, unifiedqa:natural_questions_with_dpr_para, unifiedqa:newsqa, unifiedqa:physical_iqa, unifiedqa:quoref, unifiedqa:race_string, unifiedqa:ropes, unifiedqa:social_iqa, unifiedqa:winogrande_xl
Subsampled by similarity: biomrc, boolq, freebase_qa, hotpot_qa, lama-google_re, quoref, ropes, superglue-multirc, superglue-record, unifiedqa:boolq, unifiedqa:commonsenseqa, unifiedqa:drop, unifiedqa:narrativeqa, unifiedqa:natural_questions_with_dpr_para, unifiedqa:newsqa, unifiedqa:quoref, unifiedqa:race_string, unifiedqa:ropes

Table 7: Tasks used in our meta-training data. We subsample 16,384 examples per task to ensure balanced supervision during meta-training. All tasks are from CROSSFIT unless prefixed with "unifiedqa:".

We also perform a targeted subsampling procedure, where we select tasks by semantic similarity to our evaluation tasks. For this, we compute the mean pairwise semantic similarity between a meta-training task's examples and one 16-shot split of each of our evaluation tasks, then select meta-training tasks in decreasing order of similarity. Semantic similarity is computed by calculating the cosine similarity of the sentence embeddings from the `stsb-roberta-base-v2` model in Sentence-Transformers (Reimers and Gurevych, 2019).

Demonstrations. Our demonstrations are from the UNIFIEDQA collection, which includes extractive, abstractive, and multiple-choice QA tasks as shown in Table 8. We exclude any task on which we evaluate.

Note that there is some overlap between the demonstration set and the meta-training set, though the demonstrations contain the correct answer whereas the meta-training examples do not.

Demonstration task bank: unifiedqa:ai2_science_middle, unifiedqa:boolq, unifiedqa:commonsenseqa, unifiedqa:drop, unifiedqa:mctest, unifiedqa:narrativeqa, unifiedqa:natural_questions_with_dpr_para, unifiedqa:newsqa, unifiedqa:openbookqa, unifiedqa:openbookqa_with_ir, unifiedqa:physical_iqa, unifiedqa:quoref, unifiedqa:race_string, unifiedqa:ropes, unifiedqa:social_iqa, unifiedqa:winogrande_xl

Table 8: Tasks used in our demonstration memory bank. Note that there is no subsampling within each task, since the retriever can simply ignore irrelevant demonstrations. All tasks are from UNIFIEDQA.

B Format Tuning for Multiple-choice QA

Chada and Natarajan (2021) observe significant performance gains by simply changing the format of the QA inputs and outputs. We use a format similar to theirs for most QA tasks, but it is not immediately clear how to extend the question/answer/context format to multiple-choice QA, or if including the answer options in the context would be helpful at all. Thus, we try three different formats for QASC and compare performance.

Every example consists of a question q, two context sentences c_1 and c_2, a set of 8 answer options with letter labels $\{a_A, a_B, \ldots, a_H\}$, and a correct answer $a \in \{a_A, \ldots, a_H\}$. We can generate either the full answer string, or the letter label of the answer i, where $i \in \{A, B, \ldots, H\}$. We try putting the answer options in the question or the context, excluding the answer options altogether, generating the answer string a, and generating the answer letter i.

Our results using $\text{BART}_{\text{large}}$ (Table 9) indicate that generating the answer is better than just generating the letter label, that including the options in the context is helpful, and that excluding the options from the context or putting the options in the question is harmful to performance. The performance gap between different formats is *very* large, which aligns with the findings of Chada and Natarajan (2021): using an example format aligned with the model's pre-training format is one of the most important factors contributing to few-shot performance.

Format name	Format	Example	F_1
Options in question, generate letter	question: q? $\{a_A, \ldots, a_H\}$ \n answer: [MASK] \n context: c_1. c_2. ⇒ question: q? \n answer: i	question: What does sunlight do for a plant? (A) during the day (B) Kills it (C) it can be seen (D) Helps it survive (E) Helps it drink water (F) It gets heated up (G) adding heat (H) Makes the color darker \n answer: [MASK] \n context: A plant requires food for survival. All plants require sunlight to make their food. ⇒ question: ... \n answer: D	15.6
Options in question, generate answer	question: q? $\{a_A, \ldots, a_H\}$ \n answer: [MASK] \n context: c_1. c_2. ⇒ question: q? \n answer: a	question: What does sunlight do for a plant? (A) during the day (B) Kills it (C) it can be seen (D) Helps it survive (E) Helps it drink water (F) It gets heated up (G) adding heat (H) Makes the color darker \n answer: [MASK] \n context: A plant requires food for survival. All plants require sunlight to make their food. ⇒ question: ... \n answer: Helps it survive	39.4
Options in context, generate answer	question: q? \n answer: [MASK] \n context: $\{a_A, \ldots, a_H\}$. c_1. c_2. ⇒ question: q? \n answer: a	question: What does sunlight do for a plant? \n answer: [MASK] \n context: (A) during the day (B) Kills it (C) it can be seen (D) Helps it survive (E) Helps it drink water (F) It gets heated up (G) adding heat (H) Makes the color darker. A plant requires food for survival. All plants require sunlight to make their food. ⇒ question: ... \n answer: Helps it survive	**82.6**
No options, generate answer	question: q? \n answer: [MASK] \n context: c_1. c_2. ⇒ question: q? \n answer: a	question: What does sunlight do for a plant? \n answer: [MASK] \n context: A plant requires food for survival. All plants require sunlight to make their food. ⇒ question: ... \n answer: Helps it survive	49.8

Table 9: The formats we try for QASC and 16-shot F_1 scores from $\text{BART}_{\text{large}}$ (no retrieval) after fine-tuning on each format. We find that generating the answer is better than just generating the letter label, that including the options in the context is helpful, and that excluding the options from the context is harmful to performance. "⇒" separates the input from the output sequence, and "\n" indicates a newline.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
No section number. Final section after conclusions.

☑ A2. Did you discuss any potential risks of your work?
Yes, in limitations.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1.

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

B ☑ Did you use or create scientific artifacts?

Yes (Sections 3 and 4). UnifiedQA, CrossFit, MetaICL data collection scripts.

☑ B1. Did you cite the creators of artifacts you used?
Yes (Sections 3 and 4).

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No. Will be releasing data and models upon approval on GitHub under a permissive license.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No. We use existing QA and classification datasets for a similar research purposes as was originally intended.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No. Our data are pre-existing common public datasets and do not contain PII.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No. These datasets and their domains are diverse and better documented and described in their original papers (which we cite).

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Sections 3 and 4.

C ☑ Did you run computational experiments?

Sections 3, 4, 5.

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

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Sections 4 and 5.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Sections 3, 4, 5.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.