Rethinking Data Use in Large Language Models

Sewon Min[1*], Hannaneh Hajishirzi[2], and Luke Zettlemoyer[3]

[1]Paul G. Allen School of Computer Science & Engineering, University of Washington
sewon@cs.washington.edu
[2]Paul G. Allen School of Computer Science & Engineering, University of Washington
hannaneh@cs.washington.edu
[3]Paul G. Allen School of Computer Science & Engineering, University of Washington
lsz@cs.washington.edu

Large language models (LMs) such as ChatGPT have revolutionized natural language processing and artificial intelligence more broadly. In this work, we discuss our research on understanding and advancing these models, centered around how they use the very large text corpora they are trained on. First, we describe our efforts to understand how these models learn to perform new tasks after training, demonstrating that their so-called in-context learning capabilities are almost entirely determined by what they learn from the training data. Next, we introduce a new class of LMs—nonparametric LMs—that repurpose this training data as a data store from which they retrieve information for improved accuracy and updatability. We discuss our work establishing the foundations of such models, including one of the first broadly used neural retrieval models and an approach that simplifies a traditional, two-stage pipeline into one. We also discuss how nonparametric models open up new avenues for responsible data use, e.g., by segregating permissive and copyrighted text and using them differently. Finally, we envision the next generation of LMs we should build, focusing on efficient scaling, improved factuality, and decentralization.

1. Introduction

Developing general-purpose NLP systems has long been a central goal of AI research (Turing 1980; Winograd 1971; Lockman and Klappholz 1983; McCarthy et al. 2006). Such models must address varied tasks, such as

- Classifying the sentiment of movie reviews (sentiment analysis)
- Answering user questions about the world (question answering [QA])

* Corresponding author.

Action Editor: Wei Lu. Submission received: 18 September 2025; revised version received: 21 September 2025; accepted for publication: 23 September 2025.

https://doi.org/10.1162/COLI.a.573

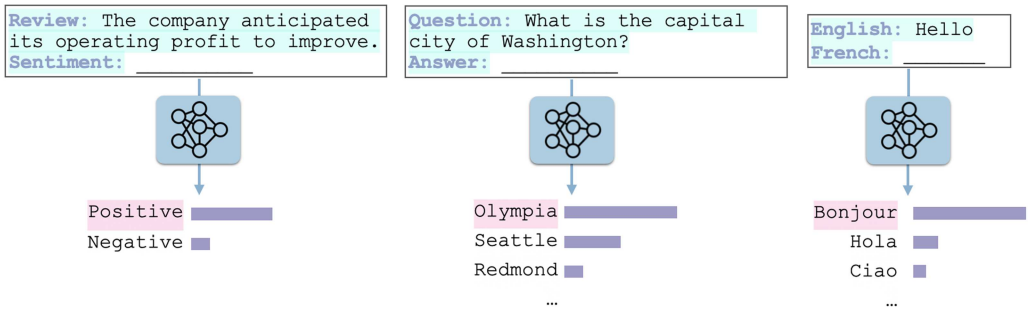

Figure 1
LMs perform a range of NLP tasks—e.g., sentiment analysis, question answering, and machine translation—by casting each task as sentence completion.

- Translating documents from one language to another (machine translation)
- Writing summaries of a hundred-page book (text summarization)
- Assisting with email composition

Recent progress comes from large language models (LMs), trained on massive text corpora with self-supervised objectives such as next-word prediction. This paradigm eliminates reliance on labeled data, enabling training at web scale and yielding models with 100B+ parameters trained on trillions of tokens (Brown et al. 2020a; Hoffmann et al. 2022; Chowdhery et al. 2022). During inference, tasks are reframed as text completion (Figure 1). Earlier models required fine-tuning on task-specific data (Peters et al. 2018; Devlin et al. 2019), but the field has shifted to **prompting**—solving unseen tasks at test time without parameter updates (Brown et al. 2020a). Prompting provides a natural interface for non-experts and enhances generalization. Since 2022, LMs have been widely deployed in production systems such as ChatGPT, Claude, and Gemini.

The central theme of this article addresses ways to better *understand* how these models work and ways to *rethink* how the next generation of models should use data at scale.

Understanding Current Language Models (§3). We focus on first understanding how LMs perform tasks, particularly via in-context learning (ICL). In ICL, labeled examples $(x_1, y_1), \ldots, (x_k, y_k)$ are concatenated with a new input x and passed to the LM for prediction (Figure 2a). ICL was thought to enable models to acquire new abilities on the fly, yet our work shows comparable performance even when labels are random

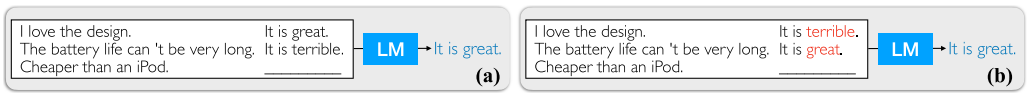

Figure 2
(a) In-context learning (ICL) with $k = 2$, with correctly paired labels. (b) ICL with random labels. Our work shows that ICL performs the task equally well (Min et al. 2022c; Wang et al. 2023a; Lyu et al. 2023).

or incorrect (Figure 2b) (Min et al. 2022c; Wang et al. 2023a; Lyu et al. 2023). This suggests that LMs rely on patterns embedded in pretraining data rather than learning from demonstrations at test time. By challenging a widely held belief about ICL, our findings have inspired extensive follow-up work revealing unexpected LM behaviors and motivating improved designs (Madaan and Yazdanbakhsh 2022; Wei et al. 2023b; Halawi, Denain, and Steinhardt 2023; Jang, Ye, and Seo 2022; Wies, Levine, and Shashua 2023; Lampinen et al. 2022; Pan et al. 2023; Kim et al. 2022; Schaeffer et al. 2023; Zhang et al. 2023b; She et al. 2023; Turpin et al. 2023). Using our analysis, we improve LMs by more effectively activating training patterns (Khashabi et al. 2020; Min et al. 2022a, b; Lyu et al. 2023). We were among the first to train LMs with a multi-task objective over diverse tasks (e.g., sentiment analysis, topic classification, question answering) and then evaluate them on unseen tasks (e.g., hate speech detection) (Khashabi et al. 2020; Min et al. 2022b). In particular, we condition models on k examples during training, enabling them at inference to leverage *new* task-specific examples (Min et al. 2022b). This improves their ability to activate relevant patterns and can be combined with auxiliary information, such as natural language task descriptions. This approach is now a standard component of state-of-the-art LM alignment pipelines (Chung et al. 2022; Ivison et al. 2023).

Nonparametric Language Models (§4). Our research established the foundation for *nonparametric LMs*, a new class of LMs that include not only learned parameters but also a datastore—a massive collection of raw text documents. At inference, these models retrieve relevant text and reason with it, rather than relying solely on memorized training data. This design improves both performance and flexibility, since the datastore can be updated (e.g., with new information) without retraining.

Our initial work developed *retrieval-augmented LMs* (Figure 3a), which operate in two stages: (1) retrieving text from the datastore and (2) providing it to the LM as input. Although intuitive—similar to how humans consult search engines—training models to reason over large datastores is challenging. We addressed these issues by introducing some of the first retrieval-augmented models (Karpukhin et al. 2020; Min et al. 2019b, 2021a; Shi et al. 2024a, b). A key milestone was Dense Passage Retrieval (Karpukhin et al. 2020), which advanced neural retrieval over lexical methods via a *contrastive* objective that distinguishes relevant from distracting text. We further improved retrieval augmentation by conditioning LMs on sets of retrieved documents, enhancing factuality and handling of rare concepts—even outperforming commercial systems like GPT-3 (Shi et al. 2024b). These models are also easily updated and reduce parameter counts (Min et al. 2021a), as we showed in Min et al. (2021a). Despite their success, retrieval-augmented LMs face scaling limits: The design of the architecture makes it difficult to

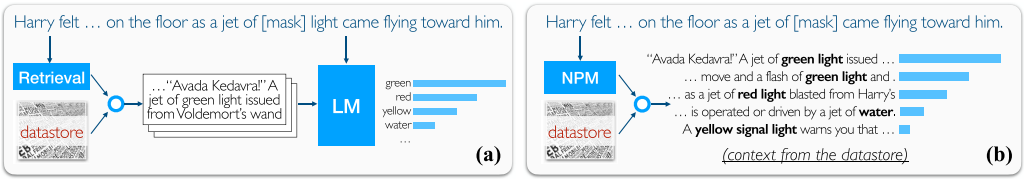

Figure 3
(a) A retrieval-augmented LM (Karpukhin et al. 2020; Min et al. 2021a; Shi et al. 2024a, b) and (b) an LM with a nonparametric softmax (Min et al. 2023b).

scale the amount of text retrieved from the datastore each time. Also, as we showed (Min et al. 2024), performance of these models do not scale as effectively with the datastore size as desired.

To overcome this, we propose a new design: training LMs with a *nonparametric softmax* (Figure 3b). Instead of producing probabilities over a fixed vocabulary, the model assigns scores directly to phrases in the datastore (e.g., highlighting ‘‘`green light`’’ in a Harry Potter passage). This allows broader use of the datastore and better handling of rare or unseen tokens. We introduced NPM (Min et al. 2023b), one of the first such models. Scaling NPM required novel techniques—efficient batching and an objective approximating corpus-wide distributions—to mitigate the cost of scoring all phrases per iteration. Empirically, NPM outperforms alternatives across tasks, is especially effective with rare entities and facts, and can be seamlessly expanded by updating the datastore. Crucially, it scales and generalizes more effectively than retrieval-augmented LMs (Min et al. 2024).

Responsible Language Models (§5). Nonparametric LMs enable us to revisit the boundaries of LM design and address qualitatively new challenges, such as responsible data use. Conventional LMs are trained on all available web data, but this raises issues of copyright, attribution, and the Right to be Forgotten, since data cannot be removed once training is complete.

We propose an alternative: Train models solely on permissively licensed data while placing copyrighted data in a datastore accessed only at inference (Min et al. 2024). This design matches the performance of models trained on full web corpora while improving legal compliance. It (1) credits data creators by enabling attribution with each prediction, and (2) supports data removal requests, since the datastore can be updated without retraining.

2. Terminologies & Background

2.1 Language Models

A LM takes a sequence of n tokens $x_1, \ldots, x_n$ and outputs $P(x_{n+1}|x_1, \ldots, x_n)$, the probability of the next token. Early examples include n-gram models (Katz 1987; Jurafsky and Martin 2000; Brants et al. 2007; Franz and Brants 2006; Aiden and Michel 2011; Lin et al. 2012), while neural LMs were pioneered by Bengio, Ducharme, and Vincent (2000).

In this article, “LMs” (often called pre-trained LMs) refer to Transformer-based networks (Vaswani et al. 2017) trained on large text corpora without explicit supervision. These models are also often widely referred to as “large language models (LLMs).”

A masked LM predicts a missing token given its context (Devlin et al. 2019). By contrast, next-token predictors are sometimes called “causal” or “generative” LMs. Unless otherwise noted, “LM” denotes a model trained only on unlabeled data; supervised fine-tuning or meta-training will be explicitly indicated.

2.2 Data, Training, and Inference

Training denotes methods involving gradient updates:

- **Pre-training**: training on unlabeled text with next-token or masked LM objectives. Early word embedding models (word2vec [Mikolov et al.

2013], GloVe [Pennington, Socher, and Manning 2014]) were a form of pre-training with relatively shallow neural networks. ELMo (Peters et al. 2018) marked a breakthrough by making the neural model itself the artifact of pre-training, rather than fixed representations; its model takes an arbitrary word along with its surrounding context and produces contextualized learned representations. However, it required separate neural architectures for individual tasks. The modern paradigm of pre-training was established by GPT (Radford et al. 2018) and BERT (Devlin et al. 2019); they pre-train an entire deep model and reuse the same architecture and most parameters across tasks.

- **Fine-tuning**: supervised training of a pre-trained model on a labeled dataset for a specific task (Radford et al. 2018; Devlin et al. 2019). This follows a traditional supervised learning setup, but it differs from earlier methods in that it is performed on a pre-trained neural network.
- **Meta-training**: task-agnostic training on a collection of downstream tasks, evaluated on unseen tasks. Instruction tuning (Mishra et al. 2022; Sanh et al. 2022; Wei et al. 2022a) is a common instance.

In the context of this article, "training" refers to pre-training unless specified. "Training" and "learning" are distinct: training requires gradient updates, while learning (e.g., in-context learning) may not.

Data refers to text. It may denote a dataset of labeled examples (split into train/test sets), or unlabeled text such as web corpora or Wikipedia. The intended meaning will be clear from context.

Inference refers to methods that do not involve gradient updates, typically used to make predictions.

In-Context Learning (ICL) is an inference method that uses a training set but does not involve any gradient updates. Given k examples from the training data, $(x_1, y_1), \cdots, (x_k, y_k)$, ICL concatenates $(x_1, y_1), \cdots, (x_k, y_k)$ with a test input x, and returns $\text{argmax}_{c \in \mathcal{C}} P(c|x_1, y_1, \cdots, x_k, y_k, x)$ as a prediction, where $\mathcal{C}$ is a set of candidate labels. ICL was first proposed by Brown et al. (2020a). Despite its name, no training occurs. ICL is a form of few-shot learning.

Zero-shot inference refers to a method that makes predictions without using any training data, typically via a template. The modern context of zero-shot inference was first discussed by Radford et al. (2019) and is often referred to as "zero-shot prompting."

Few-shot learning is a method that uses training data—typically small, between 1 and 1,000 examples. To be an effective few-shot learning method, its performance should exceed that of zero-shot inference. Few-shot learning has traditionally been achieved through fine-tuning. Since 2020, ICL has emerged as an alternative.

2.3 Task

A task or dataset is a set of (x, y) pairs defining a problem (e.g., SQuAD [Rajpurkar et al. 2016]). While tasks and datasets are sometimes distinguished (task = problem, dataset = examples), in this article they are used interchangeably, based on the idea that each dataset defines a unique task (e.g., SQuAD vs. TriviaQA vs. NQ).

Template is a function that maps a test input x into a short natural language text, so that the label can be predicted as the next token. An **instruction** is a more detailed

Table 1
Examples for the template and the instruction for the Natural Language Inference (NLI) task. Examples are taken from Wei et al. (2022a).

Template	Instruction
Sentence 1: At my age you will probably have learnt one lesson. Sentence 2: It's not certain how many lessons you'll learn by your thirties. Label: {Entailment, Neutral, Contradiction}	Read the following and determine if the hypothesis can be inferred from the premise: Premise: At my age you will probably have learnt one lesson. Hypothesis: It's not certain how many lessons you'll learn by your thirties. Answer: {Yes, It is not possible to tell, No}

template with natural language guidance (Mishra et al. 2022). Templates add minimal formatting, while instructions add full sentences (Table 1).

The distinction between templates and instructions became more important as LMs advanced to more complex tasks requiring longer and more detailed guidelines. At the time of this research, most NLP tasks did not demand such instructions, so the boundary between templates and instructions was often blurred. In this article, a function that merely adds a few words or punctuation (e.g., ":") is considered a template, while one that adds full sentences to explain the task is considered an instruction (see Table 1).

2.4 Nonparametric Language Models & Retrieval

Nonparametric LMs refer to a new class of LMs that combine learned parameters with a datastore of raw text used at inference. During inference, these models can identify relevant text from the datastore and reason with it, unlike conventional models that must remember every relevant detail from the training set. An n-gram language is a simple example (Katz 1987; Jurafsky and Martin 2000; Brants et al. 2007; Franz and Brants 2006; Aiden and Michel 2011; Lin et al. 2012), because it keeps the n-gram table inferred from the data. These models are called *nonparametric* because, unlike parametric models with fixed capacity, their effective complexity scales with data size (Siegel 1957; Hollander, Wolfe, and Chicken 2013; Freeman, Jones, and Pasztor 2002). As noted by Freeman, Jones, and Pasztor (2002), the term nonparametric does not imply an absence of parameters but rather indicates that the number and nature of the *effective* parameters are flexible and depend on the data.

Retrieval maps a query q to a subset of k documents from a large corpus $\mathcal{D}$. Typically, $N \geq 10^6$ and $k \leq 100$. Retrieval is based on some notion of *relevance*, which depends on the task and retrieval model. In information retrieval, relevance is defined as whether $d \in \mathcal{D}$ contains the information that the author of the query q is likely seeking. Traditionally, retrieval has been treated as a standalone task (Voorhees 1999; Baeza-Yates, Ribeiro-Neto et al. 1999; Kobayashi and Takeda 2000; Manning, Raghavan, and Schütze 2008). In the context of this article, retrieval is typically used *for* an NLP task, such as finding a short answer to a query.

Retrieval augmentation is an instance of a nonparametric LM that integrates retrieved results R into the LM input, aiming to maximize $P(r|q, R)$. It is also often called retrieval-augmented generation, the term coined by Lewis et al. (2020b). The concept of retrieval augmentation has existed for a long time (Teh 2006; Zhang et al. 2006; Ceri et al. 2013; Zhao and Cho 2018; Chen et al. 2017; Gu et al. 2018; Song et al. 2018), but

Tian et al. (2019) is among the first to provide a definition that aligns with its modern use.

Retrieval can be broadly categorized into two classes:

- **Sparse retrieval**: lexical methods such as TF-IDF and BM25 (Robertson, Zaragoza et al. 2009), assigning high similarity scores to document pairs with substantial n-gram overlap. It is termed "sparse retrieval" because it can be seen as representing each document into a sparse, high-dimensional space.
- **Dense retrieval**: neural methods mapping texts into dense embeddings, with similarity measured via inner product or cosine (Deerwester et al. 1990; Yih et al. 2011; Huang et al. 2013; Gillick et al. 2019; Lee, Chang, and Toutanova 2019). Dense methods capture semantic similarity beyond token overlap, though until 2020 they often underperformed sparse methods.

This section has covered pre-2020 background; more work is discussed in §4.1.

3. Understanding Current Language Models

Large language models (LMs) have been shown to be able to do **in-context learning** (Brown et al. 2020a), where they learn a new task simply by conditioning on a few training examples and predicting which tokens best complete a test input. This type of learning is attractive because the model learns a new task through inference alone, without any parameter updates. However, it remains less effective than supervised fine-tuning, shows high variance (Zhao et al. 2021; Perez, Kiela, and Cho 2021), and often requires carefully engineered templates. Moreover, its mechanisms are still poorly understood.

In the first half of the section, we improve ICL with MetaICL: **Meta**-training for **I**n-**C**ontext **L**earning (§3.2). MetaICL meta-trains a pre-trained LM on a large set of tasks to learn how to in-context learn, and is evaluated on strictly new unseen tasks. Each meta-training instance mirrors test-time ICL: k labeled examples and a query, where the model predicts the final label via cross-entropy loss. Fine-tuning on this setup directly improves ICL, as the model learns to recover task semantics from examples, as must be done at test time. Unlike prior multi-task approaches for zero-shot transfer (Khashabi et al. 2020; Zhong et al. 2021; Mishra et al. 2022; Wei et al. 2022a; Sanh et al. 2022), MetaICL learns new tasks from k examples without reducing them to a common format or requiring task-specific templates. Experiments show that MetaICL consistently outperforms (1) LM ICL without meta-training (Brown et al. 2020a; Zhao et al. 2021; Holtzman et al. 2021; Min et al. 2022a) and (2) multi-task learning with zero-shot transfer (Zhong et al. 2021; Wei et al. 2022a; Sanh et al. 2022).

In the second part, we challenge the assumption that correct input-label pairs are essential for ICL (§3.6). Replacing the labels in demonstrations with random labels barely hurts performance in a range of classification and multi-choice tasks (Figure 6), a result consistent over 12 different models including the GPT-3 family (Radford et al. 2019; Min et al. 2022b; Wang and Komatsuzaki 2021; Artetxe et al. 2021; Brown et al. 2020a). This suggests, counter-intuitively, that the model *does not* rely on the input-label mapping in the demonstrations to perform the task.

Our ablations show that ICL depends instead on (1) the label space and input distribution specified by the demonstrations, and (2) the overall format (e.g., random English words as labels outperform no labels). Moreover, meta-training amplifies these effects—the models almost exclusively exploit simpler aspects of the demonstrations like the format rather than the input-label mapping.

These findings indicate that ICL exploits latent patterns from pre-training data rather than learning tasks on the fly. This motivated our exploration of models that explicitly retrieve data instead of memorizing it: nonparametric LMs (Section 4).

3.1 Background

3.1.1 In-context Learning. When pre-trained LMs were first introduced, fine-tuning was the standard method for adapting pre-trained LMs to new tasks (Devlin et al. 2019). As models scaled to tens of billions of parameters, fine-tuning became very expensive. Brown et al. (2020a) proposed ICL as an alternative: The model learns a new task at inference by conditioning on a concatenation of training examples, without gradient updates (Figure 2).

ICL has been the focus of significant study since its introduction. Prior work proposes better ICL formulations (Zhao et al. 2021; Holtzman et al. 2021; Min et al. 2022a), improved example selection (Liu et al. 2021; Lu et al. 2021; Rubin, Herzig, and Berant 2021), and learning to follow instructions as a variant of ICL (Efrat and Levy 2020; Wei et al. 2022a; Sanh et al. 2022). However, ICL was also reported to achieve poor performance when the target task is very different from language modeling in nature or with smaller LMs; it often exhibits high variance and poor worst-case accuracy (Zhao et al. 2021; Perez, Kiela, and Cho 2021; Lu et al. 2021).

3.1.2 Meta-training via Multi-task Learning. MetaICL (§3.2) is inspired by a large body of work in meta-learning (Vilalta and Drissi 2002; Finn, Abbeel, and Levine 2017) and multi-task learning (Evgeniou and Pontil 2004; Ruder 2017). Prior work has shown that multi-task learning on a large collection of tasks leads to better performance on a new task, either when tested zero-shot (Khashabi et al. 2020; Zhong et al. 2021; Mishra et al. 2022; Wei et al. 2022a) or when further fine-tuned (Aghajanyan et al. 2021; Ye, Lin, and Ren 2021). However, zero-shot methods are typically restricted to tasks sharing the same format (e.g., QA) (Khashabi et al. 2020; Zhong et al. 2021) or depend on brittle task-specific templates (Mishra et al. 2022; Wei et al. 2022a; Sanh et al. 2022).

In §3.2, we propose a meta-training method that improves in-context learning by learning task semantics from demonstrations alone, eliminating manual templates. It significantly outperforms zero-shot transfer methods, and is complementary to natural language instructions (Sanh et al. 2022; Wei et al. 2022a). While Wei et al. (2022a) see benefits of meta-training only with 68B or more parameters, our experiments demonstrate improvements with a much smaller model (770M). Concurrent work by Chen et al. (2022b) also explores meta-training for ICL, but our approach uses more diverse tasks, avoids templates, and evaluates at larger scale with stronger baselines.

3.1.3 Understanding In-context Learning. Less work has examined why ICL works. Xie et al. (2022) formalize it as Bayesian inference over latent concepts, while Razeghi et al. (2022) link performance to pre-training term frequencies. Our empirical analysis (§3.6) is, to our knowledge, the first to systematically test this: we find that the ground truth input-label mapping in demonstrations contributes little, and instead quantify the effects of finer-grained factors.

Table 2
Overview of MetaICL (§3.2). MetaICL uses the same ICL setup at both meta-training and inference. At meta-training time, $k+1$ examples for a task is sampled, where the last example acts as the test example and the remaining k examples act as the training examples. Inference is the same as typical ICL where k labeled examples are used to make a prediction for a test input.

Task	**Meta-training**: C *meta-training* tasks **Inference**: An unseen *target* task
Data given	**Meta-training**: Training examples $\mathcal{T}_i = \{(x_j^i, y_j^i)\}_{j=1}^{N_i}$, $\forall i \in [1, C]$ $(N_i \gg k)$ **Inference**: Training examples $(x_1, y_1), \cdots, (x_k, y_k)$ & Test input x
Objective	**Meta-training**: For each iteration, 1. Sample task $i \in [1, C]$ 2. Sample $k+1$ examples from $\mathcal{T}_i$: $(x_1, y_1), \cdots, (x_{k+1}, y_{k+1})$ 3. Maximize $P(y_{k+1}\|x_1, y_1, \cdots, x_k, y_k, x_{k+1})$ **Inference**: $\text{argmax}_{c \in \mathcal{C}} P(c\|x_1, y_1, \cdots, x_k, y_k, x)$

3.2 MetaICL: Meta-training for In-Context Learning

We introduce MetaICL (**Meta**-training for **I**n-**C**ontext **L**earning), a new meta-training framework for few-shot learning where a pre-trained language model is tuned to do in-context learning on a large set of training tasks.

The key idea of MetaICL is to use a multi-task learning scheme over a large collection of meta-training tasks, in order for the model to learn how to condition on a small set of training examples, recover the *semantics* of a task, and predict the output based on it (Table 2). Following previous literature (Brown et al. 2020a), the training examples are concatenated and provided as an single input to the model, which is feasible for k-shot learning (e.g., $k = 16$). At test time, the model is evaluated on an unseen target task that comes with k training examples, and inference directly follows the same data format as in meta-training.

3.2.1 Meta-training. The model is trained on a collection of tasks which we call meta-training tasks. For every iteration, one meta-training task is sampled, and $k+1$ training examples $(x_1, y_1), \cdots, (x_{k+1}, y_{k+1})$ are sampled from the training examples of the chosen task. We then supervise the model by feeding the concatenation of $x_1, y_1, \cdots, x_k, y_k, x_{k+1}$ to the model as an input and train the model to generate y_{k+1} using a negative log likelihood objective. This simulates in-context learning where the first k examples serve as training examples and the last $(k+1)$-th example is regarded as the test example.

3.2.2 Inference. For a new target task, the model is given k training examples $(x_1, y_1), \cdots, (x_k, y_k)$ as well as a test input x. It is also given a set of candidates $\mathcal{C}$ that is either a set of labels (in classification) or answer options (in question answering). As in meta-training, the model takes a concatenation of $x_1, y_1, \cdots, x_k, y_k, x$ as the input, and computes the conditional probability of each label $c_i \in \mathcal{C}$. The label with the maximum conditional probability is returned as a prediction.

3.2.3 Channel MetaICL. We introduce a noisy channel variant of MetaICL called Channel MetaICL, following Min et al. (2022a). At meta-training time, the model is given a concatenation of $y_1, x_1, \cdots, y_k, x_k, y_{k+1}$ and is trained to generate x_{k+1}. At inference, the model computes $\arg\max_{c \in \mathcal{C}} P(x|y_1, x_1, \cdots, y_k, x_k, c)$.

3.3 MetaICL: Experimental Setup

3.3.1 Datasets. We use 142 English tasks spanning text classification, QA, Natural Language Inference [NLI], and paraphrase detection. Experiments cover seven settings, with no overlap between meta-training and target tasks. In total, 52 unique target tasks are included, substantially more than prior work (Khashabi et al. 2020; Zhong et al. 2021; Mishra et al. 2022; Wei et al. 2022a; Sanh et al. 2022). Each target task is either classification or multi-choice with a candidate label set $\mathcal{C}$.

- **HR→LR** (High→Low resource): Meta-training uses datasets with ≥10k examples; target tasks are smaller datasets, reflecting realistic few-shot learning.
- **X→X (X = {CLS, QA})**: Meta-training and target tasks share the same format but differ in task identity.
- **Non-X→X (X = {CLS, QA, NLI, Paraphrase})**: Meta-training and target tasks differ in both format and required capabilities, presenting the most challenging generalization.

Each setting has a subset of target tasks with no domain overlap with any meta-training tasks (e.g., finance, poem, climate, or medical). We report both on all target tasks or on target tasks with no domain overlap only. See the original paper (Min et al. 2022b) for full details of the setting, statistics, and datasets.

3.3.2 Baselines. We compare MetaICL and Channel MetaICL against several baselines.

- **0-shot**: Zero-shot inference.
- **In-context**: ICL with k training examples, following Brown et al. (2020a).
- **PMI 0-shot, PMI In-context**: PMI method from Holtzman et al. (2021) and Zhao et al. (2021) with 0-shot or ICL settings.
- **Channel 0-shot, Channel In-context**: Noisy channel method from Min et al. (2022a) with 0-shot or ICL settings.
- **Multi-task 0-shot**: Multi-task training without ICL, i.e., maximize $P(y|x)$ without k other training examples, and then use zero-shot transfer on a target task. This is equivalent to MetaICL with $k = 0$. This is an approach from previous work (Khashabi et al. 2020; Zhong et al. 2021; Wei et al. 2022a).
- **Channel Multi-task 0-shot**: Channel variant of Multi-task 0-shot.
- **Fine-tune**: Standard fine-tuning on an individual target task (not directly comparable to other methods as parameter updates are required for every target task).
- **Fine-tune w/ meta-train**: Pre-train with meta-training, then fine-tune on the target task (also not directly comparable for the same reason).

3.3.3 Evaluation. We use Macro-F1 and Accuracy for classification tasks and non-classification tasks, respectively. Each target task uses $k = 16$ training examples, sampled uniformly at random. We relax the assumption of perfect balance between labels on k training examples, following Min et al. (2022a). To mitigate ICL's high variance (Zhao et al. 2021; Perez, Kiela, and Cho 2021; Lu et al. 2021), we evaluate on 5 random seeds, reporting both mean and worst-case performance per task, then macro-averaged across tasks.

3.3.4 Model Details. Our base LM is GPT-2 Large (770M parameters) (Radford et al. 2019). For raw LM baselines, we also include GPT-J (6B) (Wang and Komatsuzaki 2021), the largest public causal LM at the time.

Unlike prior work relying on human-authored templates (Zhong et al. 2021; Mishra et al. 2022; Wei et al. 2022a; Chen et al. 2022b), which require significant effort (e.g., 136 templates for 136 tasks) and yield unstable performance (Mishra et al. 2021), we eliminate templates. Inputs are concatenated directly with label words from the dataset.[1]

Implementation uses PyTorch (Paszke et al. 2019) and Transformers (Wolf et al. 2020). Meta-training uses up to 16,384 examples per task, batch size 8, learning rate 1×10^{-5}, and sequence length 1024 (256 for multi-task 0-shot). Models train for 30k steps.[2] To reduce memory, we use 8-bit Adam (Dettmers et al. 2022; Kingma and Ba 2015) and mixed precision (Micikevicius et al. 2017). Training takes 4.5 hours on eight 32GB GPUs, compared with 270 hours on a 512GB TPU in Sanh et al. (2022).

3.4 MetaICL: Results

Table 3 reports results with GPT-2 Large (770M parameters) (Radford et al. 2019). The top half shows all target tasks; the bottom half, unseen-domain tasks only.

3.4.1 Our Baselines Are Strong. Among raw LMs without meta-training (the first six rows), channel in-context baselines are the most competitive, consistent with Min et al. (2022a). Multi-task 0-shot generally fails to outperform the best raw LM, despite supervision on many tasks, differing from prior findings (Wei et al. 2022a; Sanh et al. 2022). This is likely because (1) our models are much smaller than theirs (770M vs. 11B–137B),[3] and (2) our baselines, such as PMI and channel, are stronger.

3.4.2 MetaICL Outperforms Baselines. MetaICL and Channel MetaICL consistently outperform strong baselines. Channel MetaICL achieves the best performance in 6 out of 7 settings. Gains are largest in the HR→LR, non-NLI→NLI, and non-Para→Para settings (6–15% absolute), showing robustness to both low-resource tasks and distribution shifts. This demonstrates that MetaICL can infer the semantics of new tasks in context even when there are no closely related training tasks.

While MetaICL significantly outperforms baselines in most settings, its gain over Multi-task 0-shot is smaller in the QA→QA setting, likely because the meta-training and target tasks are relatively similar, benefiting Multi-task 0-shot.

1 We tested prior templates and found they did not reliably improve few-shot results, though they sometimes helped zero-shot performance.
2 Longer training did not improve performance.
3 Wei et al. (2022a) found gains only above 68B parameters.

Table 3
Main results, using GPT-2 Large. Two numbers indicate the average and the worst-case performance over different seeds used for k target training examples. **Bold** indicates the best result except results from fine-tuned models (not comparable).

Method	HR→LR	Class →Class	non-Class →Class	QA →QA	non-QA →QA	non-NLI →NLI	non-Para →Para
			All target tasks				
0-shot	34.8	34.2	34.2	40.2	40.2	25.5	34.2
PMI 0-shot	35.1	33.8	33.8	40.2	40.2	27.9	39.2
Channel 0-shot	36.5	37.3	37.3	38.7	38.7	33.9	39.5
In-context	38.2/35.3	37.4/33.9	37.4/33.9	40.1/38.7	40.1/38.7	34.0/28.3	33.7/33.1
PMI In-context	39.2/33.7	38.8/30.0	38.8/30.0	40.3/38.8	40.3/38.8	33.0/28.0	38.6/33.4
Channel In-context	43.1/38.5	46.3/40.3	46.3/40.3	40.8/38.1	40.8/38.1	39.9/34.8	45.4/40.9
Multi-task 0-shot	35.6	37.3	36.8	45.7	36.0	40.7	30.6
Channel Multi-task 0-shot	38.8	40.9	42.2	42.1	36.4	36.8	35.1
MetaICL	43.3/41.7	43.4/39.9	38.1/31.8	**46.0**/44.8	38.5/36.8	49.0/44.8	33.1/33.1
Channel MetaICL	**49.1**/46.8	**50.7**/48.0	**50.6**/48.1	44.9/43.5	**41.9**/40.5	**54.6**/51.9	**52.2**/50.3
Fine-tune	46.4/40.0	50.7/44.0	50.7/44.0	41.8/39.1	41.8/39.1	44.3/32.8	54.7/48.9
Fine-tune w/ meta-train	52.0/47.9	53.5/48.5	51.2/44.9	46.7/44.5	41.8/39.5	57.0/44.6	53.7/46.9
			Target tasks in unseen domains				
0-shot	32.6	32.6	32.6	45.9	45.9	33.4	38.3
PMI 0-shot	28.9	28.9	28.9	44.4	44.4	33.4	32.9
Channel 0-shot	29.1	29.1	29.1	41.6	41.6	33.1	32.6
In-context	30.6/27.5	30.6/27.5	30.6/27.5	45.6/44.7	45.6/44.7	52.0/41.3	35.8/34.1
PMI In-context	34.9/27.7	34.9/27.7	34.9/27.7	45.4/44.7	45.4/44.7	47.8/35.2	38.5/33.3
Channel In-context	39.6/33.6	39.6/33.6	39.6/33.6	44.7/40.6	44.7/40.6	40.4/35.7	44.1/36.8
Multi-task 0-shot	35.4	28.0	28.6	**71.2**	40.3	33.5	35.0
Channel Multi-task 0-shot	36.3	31.1	34.3	54.4	39.4	50.8	34.1
MetaICL	35.3/32.7	32.3/29.3	28.1/25.1	69.9/68.1	**48.3**/47.2	**80.1**/77.2	34.0/34.0
Channel MetaICL	**47.7**/44.7	**41.9**/37.8	**48.0**/45.2	57.9/56.6	47.2/45.0	62.0/57.3	**51.0**/49.9
Fine-tune	44.9/37.6	44.9/37.6	44.9/37.6	43.6/39.1	43.6/39.1	56.3/33.4	56.6/51.6
Fine-tune w/ meta-train	53.3/43.2	53.2/43.7	46.1/36.9	67.9/66.2	44.5/42.8	71.8/58.2	65.6/61.4

3.4.3 Gains Are Larger on Unseen Domains. Improvements over Multi-task 0-shot are even larger on unseen-domain domains. In particular, Multi-task 0-shot is generally less competitive compared with raw LM baselines, likely because they require more challenging generalization. MetaICL suffers less from this problem and is consistently better or comparable to raw LM baselines across all settings.

3.4.4 Comparison to Fine-tuning. MetaICL matches or exceeds fine-tuning without meta-training. However, fine-tuning with meta-training achieves the best results overall, because meta-training helps in supervised learning as it does in ICL.

3.4.5 Comparison to GPT-J. In Table 4, we compare GPT-2 Large based models with raw LM baselines based on GPT-J which consists of 6B parameters. MetaICL, despite being 8× smaller, outperforms or matches GPT-J baselines.

3.4.6 Varying the Number of Training Examples. Varying the number of training examples ($k = 0, 4, 8, 16, 32$; Figure 4) shows that performance generally improves with k and saturates near 16. Channel MetaICL consistently outperforms raw ICL across all k. Additionally, performance tends to saturate when k is closer to 16, which we discuss in more detail in §3.6.2.

Table 4
Comparison between raw LM in-context learning (based on GPT-2 Large and GPT-J) and MetaICL (based on GPT-2 Large). GPT-2 Large used unless otherwise specified. Two numbers indicate the average and the worst-case performance over different seeds used for k target training examples. For raw LM baselines, Channel In-context is reported because it is the best raw LM baseline overall across the settings.

Method	HR→LR	Class →Class	non-Class →Class	QA →QA	non-QA →QA	non-NLI →NLI	non-Para →Para
			All target tasks				
Channel In-context	43.1/38.5	46.3/40.3	46.3/40.3	40.8/38.1	40.8/38.1	39.9/34.8	45.4/40.9
MetaICL	43.3/41.7	43.4/39.9	38.1/31.8	46.0/44.8	38.5/36.8	49.0/44.8	33.1/33.1
Channel MetaICL	**49.1**/46.8	50.7/48.0	50.6/48.1	44.9/43.5	42.1/40.8	**54.6**/51.9	**52.2**/50.3
GPT-J Channel In-context	48.6/44.4	**51.5**/47.0	**51.5**/47.0	**47.0**/45.2	**47.0**/45.2	47.2/41.7	51.0/47.5
			Target tasks in unseen domains				
Channel In-context	39.6/33.6	39.6/33.6	39.6/33.6	44.7/40.6	44.7/40.6	40.4/35.7	44.1/36.8
MetaICL	35.3/32.7	32.3/29.3	28.1/25.1	**69.9**/68.1	48.3/47.2	**80.1**/77.2	34.0/34.0
Channel MetaICL	**47.7**/44.7	41.9/37.8	**48.0**/45.2	57.9/56.6	47.2/45.0	62.0/57.3	51.0/49.9
GPT-J Channel In-context	42.8/38.4	**42.8**/38.4	42.8/38.4	55.7/54.4	**55.7**/54.4	51.1/40.4	**52.0**/46.5

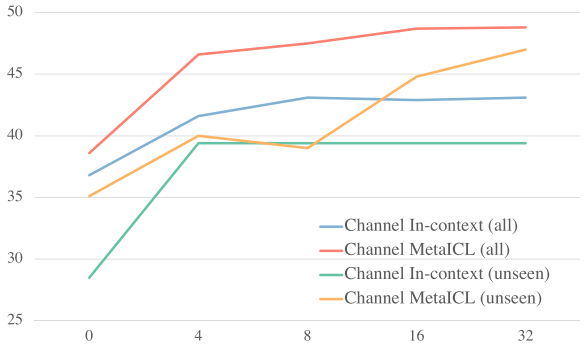

Figure 4
Ablation on the number of training examples (k) in HR→LR. $k = 0$ is equivalent to the zero-shot methods.

3.4.7 Number of Meta-training Tasks. We study the effect of meta-training task count by subsampling 7, 15, 30 tasks (out of 61 in HR→LR), each with ten different random seeds to see the impact of the choice of meta-training tasks.

As shown in Figure 5, performance generally increases with more tasks, consistent with Mishra et al. (2022) and Wei et al. (2022a). Channel MetaICL outperforms alternatives across all settings. However, variance across seeds is non-negligible (bottom of Figure 5), indicating that the choice of meta-training tasks strongly affects performance.

3.4.8 Diversity in Meta-training Tasks. We further test whether task diversity matters. From the 61 HR→LR tasks, we subsample two sets of 13: one diverse (QA, NLI, relation extraction, sentiment, topic classification, hate speech detection, etc.) and one narrow (sentiment, topic classification, hate speech only). See Min et al. (2022b) for the full dataset lists. We compare multi-task zero-shot transfer and MetaICL under both settings.

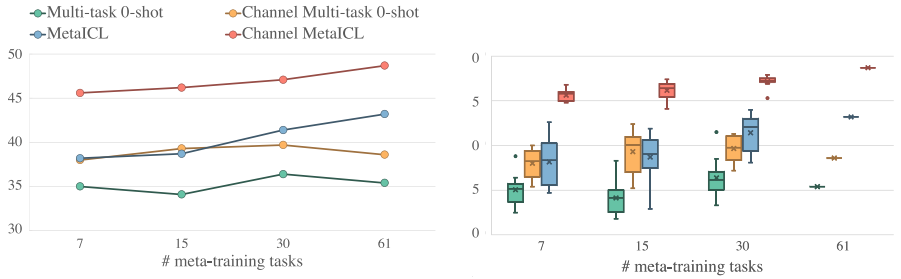

Figure 5
Ablation on the number of meta-training tasks ($\{7, 15, 30, 61\}$). The graph of the average (left) and the box chart (right) over different meta-training sets using 10 different random seeds (except for 61).

Table 5
Ablation on the diversity of meta-training tasks in the HR→LR setting. For both settings, the number of meta-training tasks is 13, and the number of target tasks is 26 as in the original HR→LR setting.

Method	Diverse	No Diverse
0-shot	34.9	
PMI 0-shot	34.8	
Channel 0-shot	36.8	
In-context	38.2/35.4	
PMI In-context	38.9/33.3	
Channel In-context	42.9/38.5	
Multi-task 0-shot	35.2	29.9
Channel Multi-task 0-shot	41.6	38.3
MetaICL	45.6/43.4	38.8/35.4
Channel MetaICL	**47.2**/44.7	45.3/42.6

Table 5 shows that MetaICL with a diverse set outperforms MetaICL with a non-diverse set by a substantial margin, indicating that diversity among meta-training tasks is one of substantial factors.

We also conducted ablations that provide more insights on the choice of meta-training tasks, such as (1) high-quality data with diverse domains tend to help (e.g., GLUE family [Wang et al. 2018a]) and (2) adversarially collected data tends to be unhelpful. However, more systematic studies on how to choose the best meta-training tasks and how they relate to particular target tasks should be done, which we leave for future work. We refer readers to Appendix C.3 of the original publication (Min et al. 2022b) for more details.

3.4.9 Are Instructions Necessary?. Most recent work has used human-written natural instructions for zero- or few-shot learning (Mishra et al. 2022; Wei et al. 2022a; Sanh et al. 2022). While we avoid instructions to reduce manual effort and variance, we ask: *are they still useful with MetaICL?* Conditioning on k examples may make instructions redundant, but they could also provide complementary information.

We test this using 32 meta-training tasks and 12 target tasks from HR→LR with instructions available from Sanh et al. (2022). We consider two variants: (a) one

Table 6
Ablation on *natural instructions*. w/ Instruct uses instructions from Sanh et al. (2022). # instruct/task is the average number of instructions per task. MT 0-shot is Multi-task 0-shot. Both settings have the same meta-training and target tasks.

Method	w/o Instruct	w/ Instruct	
# instruct/task	0	1	8.3
0-shot	33.3	34.2	
PMI 0-shot	34.6	27.8	
Channel 0-shot	32.5	30.6	
In-context	34.5/31.5	45.2/42.3	
PMI In-context	37.7/32.7	41.9/37.6	
Channel In-context	39.0/35.4	39.6/35.3	
MT 0-shot	35.7	32.6	37.1
Channel MT 0-shot	36.7	30.6	36.0
MetaICL	40.4/37.7	42.6/41.0	43.2/41.0
Channel MetaICL	42.2/40.0	45.3/43.9	**46.9**/44.2

instruction per task, and (b) all available instructions (267 total; 8.3 per task), which Sanh et al. (2022) found more effective.

Table 6 shows the results. Consistent with prior work, Multi-task 0-shot outperforms raw LM 0-shot. However, MetaICL without instructions already surpasses Multi-task 0-shot with instructions, and MetaICL improves further when instructions are added. Increasing instructions per task boosts MetaICL much more than Multi-task 0-shot. In short: (1) learning to in-context learn (MetaICL) outperforms instruction-based learning; (2) instructions and MetaICL are complementary; and (3) MetaICL gains more from instructions than Multi-task 0-shot. Notably, Channel MetaICL trained on tasks with instructions performs worse than the version without (46.9 vs. 49.1; Tables 6, 3), likely because only 32 of 61 tasks had instructions.

It is worth noting that direct comparison with Wei et al. (2022a) and Sanh et al. (2022) is difficult due to differences in model size and task splits.

3.5 MetaICL: Summary & Limitations

We introduced MetaICL, a few-shot learning method that meta-trains LMs to perform ICL: conditioning on examples to infer tasks and make predictions. Experiments on 142 tasks (52 targets, across seven settings) show that MetaICL consistently outperforms strong baselines such as default ICL and multi-task zero-shot transfer, and matches or surpasses models 8× larger. We identify key factors for success, including the number and diversity of meta-training tasks, and show that while MetaICL outperforms instruction-based methods, combining the two yields the best results.

Our study has several limitations. First, ICL requires long contexts at both training and inference, making it less efficient than non-ICL baselines. Second, we experiment only with a 770M-parameter LM; scaling to larger models remains open. Third, we focus on classification and multi-choice tasks, leaving free-form generation for future work. Promising directions include enhancing MetaICL to outperform supervised models, identifying which meta-training tasks are most helpful for a broader range of target tasks, and how to better combine human-written instructions and MetaICL.

3.6 Rethinking ICL

Despite ICL consistently outperforming zero-shot inference on a wide range of tasks, there is little understanding of *how* it works and *which* aspects of the demonstrations contribute to end task performance. In this section, we show that ground truth demonstrations are in fact not required—randomly replacing labels in the demonstrations barely hurts performance on a range of classification and multi-choice tasks, consistently over 12 different models including GPT-3. Instead, we find that other aspects of the demonstrations are the key drivers of end task performance, including the fact that they provide a few examples of (1) the label space, (2) the distribution of the input text, and (3) the overall format of the sequence. Our analysis provides a new way of understanding how and why in-context learning works, while opening up new questions about how much can be learned from large language models through inference alone.

3.6.1 Method. To see the impact of correctly-paired inputs and labels in the demonstrations—which we call the ground truth input-label mapping—we compare the following three methods.[4]

No demonstrations is a typical zero-shot method that does not use any labeled data. A prediction is made via $\text{argmax}_{y \in \mathcal{C}} P(y|x)$, where x is the test input and $\mathcal{C}$ is a small discrete set of possible labels.
Demonstrations w/ gold labels is used in a typical in-context learning method with k labeled examples $(x_1, y_1)...(x_k, y_k)$. A concatenation of k input-label pairs is used to make a prediction via $\arg\max_{y \in \mathcal{C}} P(y|x_1, y_1...x_k, y_k, x)$.
Demonstrations w/ random labels is formed with random labels, instead of gold labels from the labeled data. Each x_i $(1 \leq i \leq k)$ is paired with $\tilde{y}_i$ that is randomly sampled at uniform from $\mathcal{C}$. A concatenation of $(x_1, \tilde{y}_1)...(x_k, \tilde{y}_k)$ is then used to make a prediction via $\text{argmax}_{y \in \mathcal{C}} P(y|x_1, \tilde{y}_1...x_k, \tilde{y}_k, x)$.

We used the following experimental setup.

Models. We experiment with 12 models in total. We include 6 language models, all of which are decoder-only, dense LMs. We use each LM with two inference methods, direct and channel, following Min et al. (2022a). The sizes of LMs vary from 774M to 175B. We include the largest dense LM (GPT-3) and the largest publicly released dense LM (fairseq 13B) at the time of conducting experiments. We also include MetaICL, our model in §3.2 that is initialized from GPT-2 Large and then meta-trained on a collection of supervised datasets with an ICL objective. We ensure that our evaluation datasets do not overlap with those used at meta-training time.

Evaluation Data. We evaluate on 26 datasets, including sentiment analysis, paraphrase detection, natural language inference, hate speech detection, question answering, and sentence completion. They are exactly the same as the evaluation datasets used in the HR→LR setting in §3.2.[5] All datasets are classification and multi-choice tasks.

4 Without loss of generality, all methods are described based on the direct method, but can be trivially converted to the channel method by flipping x and y.
5 For convenience, we use "labels" to refer to the output for the task, though our datasets include non-classification tasks.

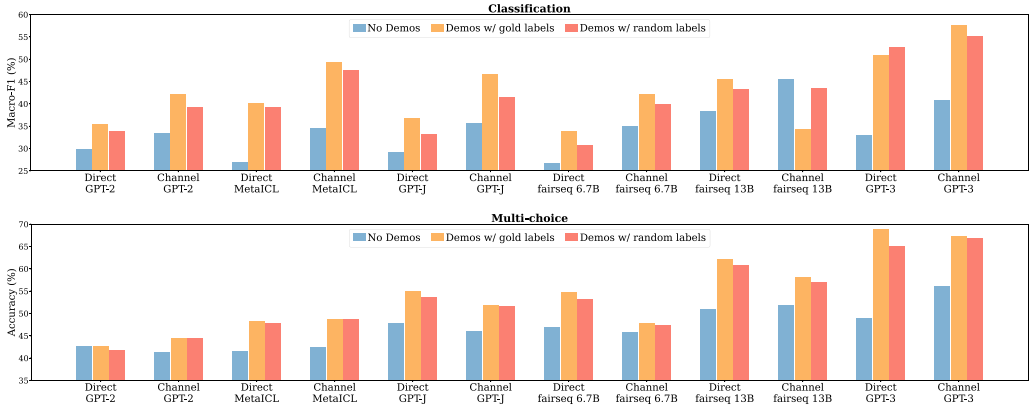

Figure 6
Results with no demonstrations, gold-label demonstrations, and random-label demonstrations on classification (top) and multi-choice (bottom). The first eight models are tested on 16 classification and 10 multi-choice datasets; the last four on 3 classification and 3 multi-choice datasets. Performance with random labels is nearly identical to that with gold labels.

We use these datasets because they (1) are true low-resource datasets with less than 10K training examples, (2) include well-studied benchmarks from GLUE (Wang et al. 2018a) and SuperGLUE (Wang et al. 2019a), and (3) cover diverse domains including science, social media, finance, and more.

Other Details. We use $k = 16$ examples as demonstrations by default for all experiments in the paper, unless otherwise specified. Examples are sampled uniformly from the training data. We choose a set of k training examples using 5 different random seeds and run experiments 5 times.[6] We report Macro-F1[7] for classification tasks and Accuracy for multi-choice tasks. We compute per-dataset average over seeds, and then report macro-average over datasets. We use the minimal templates in forming an input sequence from an example. All experiments are reproducible from `github.com/Alrope123/rethinking-demonstrations`.

3.6.2 Ground Truth Matters Little. Results are reported in Figure 6. First, using the demonstrations with gold labels significantly improves the performance over no demonstrations,[8] as has been consistently found in much prior work (Brown et al. 2020a; Zhao et al. 2021; Liu et al. 2021). We then find that *replacing gold labels with random labels only marginally hurts performance*. The trend is consistent over nearly all models: Models see performance drop in the range of 0–5% absolute. There is less impact in replacing labels in multi-choice tasks (1.7% on average) than in classification tasks (2.6% absolute).

This result indicates that the ground truth input-label pairs are not necessary to achieve performance gains. This is counter-intuitive, given that correctly paired training

6 For fairseq 13B and GPT-3, due to limited resources, we experiment with a subset of 6 datasets—MRPC, RTE, Tweet_eval-hate, OpenbookQA, CommonsenseQA, COPA—and 3 random seeds.

7 Known to be better for imbalanced classes.

8 There are some exceptions, e.g., in the classification tasks, Direct GPT-2, Direct GPT-J, and Direct fairseq 6.7B models are not significantly better than random guessing on many datasets. Channel fairseq 13B has significantly better no-demonstrations performance compared to demonstrations with gold labels. We thus discuss the results from these models less significantly for the rest of analysis.

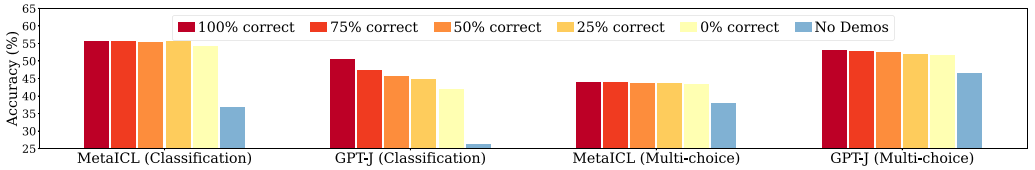

Figure 7
Results with varying number of correct labels in the demonstrations. Channel and Direct used for classification and multi-choice, respectively. Performance with no demonstrations (blue) is reported as a reference.

data is critical in typical supervised training—it informs the model of the expected input-label *correspondence* required to perform the downstream task. Nonetheless, the models *do* achieve non-trivial performance on the downstream tasks. This strongly suggests that the models are capable of recovering the expected input-label correspondence for the task; however, it is *not* directly from the pairings in the demonstrations.

It is also worth noting that there is particularly little performance drop in MetaICL: 0.1–0.9% absolute. This suggests that meta-training with an explicit ICL objective actually encourages the model to essentially ignore the input-label mapping and exploit other components of the demonstrations (more discussion in §3.7).

In the Appendix of the original publication (Min et al. 2022c), we provide additional results showing that (1) selecting random labels from a true distribution of labels (instead of a uniform distribution) reduces the gap even further, and (2) the trends may depend on the dataset, although the overall trend is consistent over most datasets.

Does the Number of Correct Labels Matter?. We conduct an ablation study[9] by varying the number of correct labels in the demonstrations. We evaluate "Demonstrations w/ a% correct labels" ($0 \leq a \leq 100$), which consist of $k \times a/100$ correct pairs and $k \times (1 - a/100)$ incorrect pairs. $a = 100$ is equivalent to typical ICL using gold labels.

Figure 7 shows that performance is fairly insensitive to the number of correct labels. In fact, always using incorrect labels significantly outperforms no-demonstrations, e.g., preserving 92%, 100%, and 97% of improvements ICL with MetaICL in classification, MetaICL in multi-choice, and GPT-J in multi-choice, respectively. In contrast, GPT-J in classification sees relatively significant performance drop with more incorrect labels, e.g., nearly 10% drop in performance when always using incorrect labels. Still, always using incorrect labels is significantly better than no demonstrations.

Is the Result Consistent with Varying k?. We study the impact of the number of input-label pairs (k) in the demonstrations. Results are reported in Figure 8. First, using the demonstrations significantly outperforms the no demonstrations method even with small k ($k = 4$), and performance drop from using gold labels to using random labels is consistently small across varying k, in the range of 0.8–1.6%.[10] Interestingly, model performance does not increase much as k increases when $k \geq 8$, both with gold labels and with random labels. This is in contrast with typical supervised training where

9 For ablation studies, we experiment with 5 classification and 4 multi-choice datasets. Classification includes: MRPC, RTE, Tweet_eval-hate, SICK, poem-sentiment; Multi-choice includes OpenbookQA, CommonsenseQA, COPA, and ARC.

10 With an exception of 4.4% in classification with $k = 4$, likely due to a high variance.

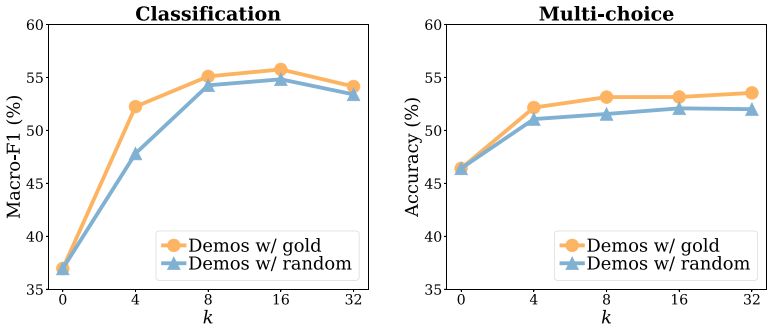

Figure 8
Ablations on varying numbers of examples in the demonstrations (k). Models that are the best under 13B in each task category used.

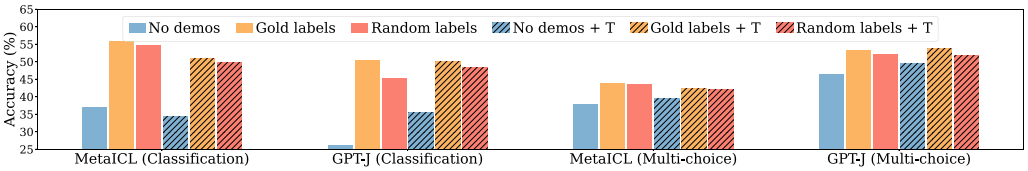

Figure 9
Results with minimal templates and manual templates. +T indicates that manual templates are used.

model performance rapidly increases as k increases, especially when k is small. We hypothesize that larger labeled data is beneficial mainly for supervising the input-label correspondence, and other components of the data like the example inputs, example labels, and the data format are easier to recover from the small data, which is potentially a reason for minimal performance gains from larger k (more discussion in Section 3.7).

Is the Result Consistent with Better Templates?. While we use minimal templates by default, we also explore data-specific templates. Figure 9 shows that the trend—replacing gold labels with random labels barely hurting performance—holds with manual templates. It is worth noting that using manual templates does not always outperform using minimal templates.

3.7 Rethinking ICL: Why *Does* ICL Work?

To further examine what other aspects of the demonstrations lead to good performance of ICL, we identify four aspects of the demonstrations $(x_1, y_1)...(x_k, y_k)$ that potentially provide learning signals (Figure 10): (1) **The input-label mapping**, i.e., whether each input x_i is paired with a correct label y_i. (2) **The distribution of the input text**, i.e., the underlying distribution that $x_1...x_k$ are from. (3) **The label space**, i.e., the space covered by $y_1...y_k$. (4) **The format**—specifically, the use of input-label pairing as the format.

We design a series of variants of the demonstrations that quantify the impact of each aspect in isolation, and then additionally discuss the trend of the models meta-trained with an ICL objective. For all experiments, models are evaluated on five classification

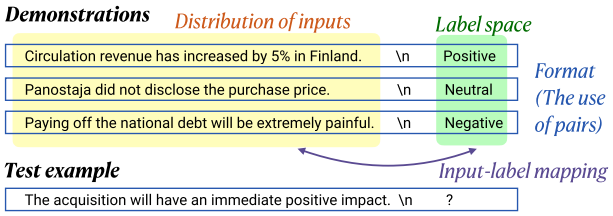

Figure 10
Four different aspects of demonstrations: the input-label mapping, the distribution of the input text, the label space, and the use of input-label pairing as the format.

and four multi-choice datasets as in Section 3.6.2. See the Appendix of the original publication (Min et al. 2022c) for implementation details.

3.7.1 Impact of the Distribution of the Input Text. We experiment with *OOD demonstrations*, which include out-of-distribution (OOD) text instead of the inputs from unlabeled training data. Specifically, a set of k sentences $\{x_{i,\text{rand}}\}_{i=1}^{k}$ are randomly sampled from an external corpus, and replace $x_1 ... x_k$ in the demonstrations. This variant assesses the impact of the distribution of the input text, while keeping the label space and the format of the demonstrations.

Figure 11 shows that using out-of-distribution inputs instead of the inputs from the training data significantly drops the performance when Channel MetaICL, Direct GPT-J, or Channel GPT-J are used, both in classification and multi-choice, by 3–16% in absolute. In the case of Direct GPT-J in multi-choice, it is even significantly worse than no demonstrations. Direct MetaICL is an exception, which we think is the effect of meta-training (discussion in a later paragraph). This suggests that in-distribution inputs in the demonstrations substantially contribute to performance gains. This is likely because conditioning on the in-distribution text makes the task closer to language modeling, since the LM always conditioned on the in-distribution text during training.

3.7.2 Impact of the Label Space. We also experiment with *demonstrations w/ random English words* that use random English words as labels for all k pairs. Specifically, we sample a

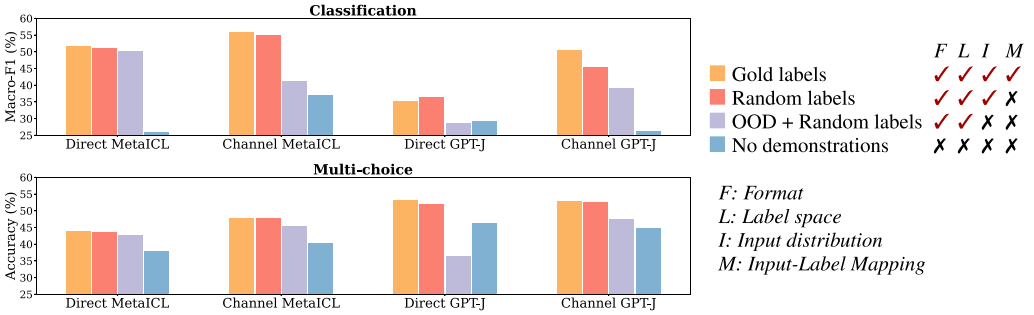

Figure 11
Impact of the distribution of the inputs. Evaluated in classification (top) and multi-choice (bottom). The impact of the distribution of the input text can be measured by comparing ■ and ■. The gap is substantial, with an exception in Direct MetaICL.

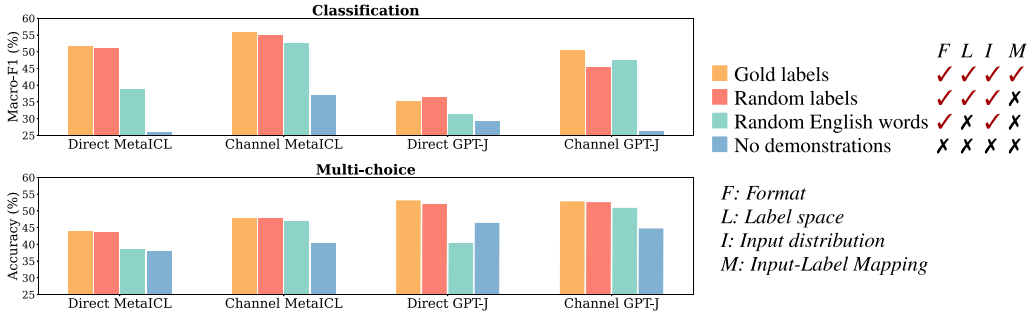

Figure 12
Impact of the label space. Evaluated in classification (top) and multi-choice (bottom). The impact of the label space can be measured by comparing ■ and ■. The gap is significant in the direct models but not in the channel models.

random subset of English words $\mathcal{C}_{\text{rand}}$ where $|\mathcal{C}_{\text{rand}}| = |\mathcal{C}|$, and randomly pair $\tilde{y}_i \in \mathcal{C}_{\text{rand}}$ with x_i. This variant assesses the impact of the label space, while keeping the distribution of the input text and the format of the demonstrations.

Based on Figure 12, direct models and channel models exhibit different patterns. With direct models, the performance gap between using random labels within the label space and using random English words is significant, ranging between 5% and 16% absolute. This indicates that conditioning on the label space significantly contributes to performance gains. This is true even for multi-choice tasks where there is no fixed set of labels—we hypothesize that multi-choice tasks still do have a particular distribution of the choices (e.g., objects like "Bolts" or "Screws" in the OpenBookQA dataset) that the model uses. On the other hand, removing the output space does not lead to significant drop in the channel models: there is 0–2% drop in absolute, or sometimes even an increase. We hypothesize that this is because the channel models only condition on the labels, and thus are not benefiting from knowing the label space. This is in contrast to direct models which must *generate* the correct labels.

3.7.3 Impact of Input-label Pairing. While previous experiments examined variants preserving demonstrations format, here we test variants that alter it by removing input-label pairings. Specifically, we evaluate *demonstrations with no labels* (LM conditioned on $x_1 \ldots x_k$ only) and *demonstrations with labels only* (conditioned on $y_1 \ldots y_k$ only). These are no-format counterparts to "random English words" and "OOD inputs" variants.

Figure 13 shows that removing the format is close to or worse than no demonstrations, indicating the importance of the format. This is likely because conditioning on a sequence of input-label pairs triggers the model to mimic the structure and complete the test input accordingly. Notably, preserving format retains much of the ICL gains even with irrelevant content. For example, Direct MetaICL preserves 95% (classification) and 82% (multi-choice) of gold-label improvements using random sentences paired with label sets (■). Channel models retain 82–87% (classification) and 75–86% (multi-choice) by pairing unlabeled inputs with random words (■). In all cases, discarding format (e.g., inputs without labels, or labels without inputs) performs far worse, highlighting that *format is essential*.

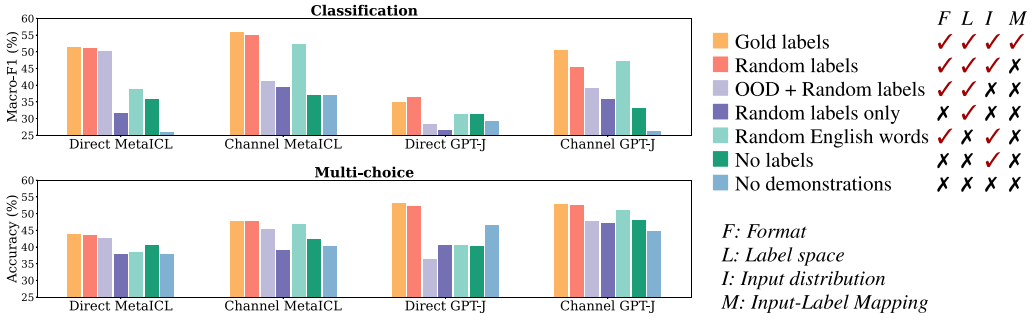

Figure 13
Impact of the format, i.e., the use of the input-label pairs. Evaluated in classification (top) and multi-choice (bottom). Variants of demonstrations without keeping the format (■ and ■) are overall not better than no demonstrations (■). Keeping the format is especially significant when it is possible to achieve substantial gains with the label space but without the inputs (■ vs. ■ in Direct MetaICL), or with the input distribution but without the labels (■ vs. ■ in Channel MetaICL and Channel GPT-J).

3.7.4 Impact of Meta-training. Different from other models, MetaICL is trained with an ICL objective (§3.2). We aim to better understand the role of this meta-training in relation with our findings by closely examining the result of MetaICL. In particular, the patterns we see so far are significantly more evident with MetaICL than with other models. For instance, the ground truth input-label mapping matters even less, and keeping the format of the demonstrations matters even more. There is nearly zero influence of the input-label mapping and the input distribution in Direct MetaICL, and the input-label mapping and the output space in Channel MetaICL.

We hypothesize that *meta-training encourages the model to exclusively exploit simpler aspects of the demonstrations and to ignore others*. This is based on our intuition that (1) the input-label mapping is likely harder to exploit, (2) the format is likely easier to exploit, and (3) the space of the text that the model is trained to generate is likely easier to exploit than the space of the text that the model conditions on.[11]

3.8 Rethinking ICL: Summary, Discussion, & Limitations

In summary, we study the role of the demonstrations with respect to the success of in-context learning. We find that the ground truth input-label mapping in the demonstrations matters significantly less than one might think—replacing gold labels with random labels in the demonstrations only marginally lowers the performance. We then identify a series of aspects in the demonstrations and examine which aspect actually contributes to performance gains. Results reveal that (1) gains are mainly coming from *independent* specification of the input space and the label space, (2) the models can still retain up to 95% of performance gains by using either the inputs only or the label set only if the right format is used, and (3) meta-training with an in-context learning objective magnifies these trends. Together, our findings lead to a set of broader indications about in-context learning, as well as avenues for future work.

11 That is, the direct model exploits the label space better than the input distribution, and the channel model exploits the input distribution better than the label space.

3.8.1 Does the Model learn at Test Time?. If we take a strict definition of learning (capturing the input-label correspondence given in the training data), then our findings suggest that LMs do not learn new tasks at test time. Our analysis shows that the model may ignore the task defined by the demonstrations and instead use priors from pre-training.

However, *learning* a new task can be interpreted more broadly: It may include adapting to specific input and label distributions and the format suggested by the demonstrations, and ultimately getting to make a prediction more accurately. With this definition of learning, the model *does* learn the task from the demonstrations. Our experiments indicate that the model *does* make use of aspects of the demonstrations and achieve performance gains.

3.8.2 Capacity of LMs. The model performs a downstream task without relying on the input-label correspondence from the demonstrations. This suggests that the model has learned the (implicit notion of) input-label correspondence from the language modeling objective alone, e.g., associating a positive review with the word "positive". This is in line with Reynolds and McDonell (2021) who claim that the demonstrations are for *task location* and the intrinsic ability to perform the task is obtained at pre-training time.[12]

On one hand, this suggests that the language modeling objective has led to great zero-shot *capacity*, even if it is not always evident from the naive zero-shot *accuracy*. On the other hand, this suggests that in-context learning may not work on a task whose input-label correspondence is not already captured in the LM. This leads to the research question of how to make progress in NLP problems that in-context learning does not solve: whether we need a better way of extracting the input-label mappings that are already stored in the LM, a better variant of the LM objective that learns a wider range of task semantics, or explicit supervision through fine-tuning on the labeled data.

3.8.3 Connection to Instruction-following Models. Prior work has found it promising to train the model that reads the natural language description of the task (called instructions) and performs a new task at inference (Mishra et al. 2022; Efrat and Levy 2020; Wei et al. 2022a; Sanh et al. 2022). We think the demonstrations and instructions largely have the same role as LMs, and hypothesize that our findings hold for instruction-following models: the instructions prompt the model to recover the capacity it already has, but do not supervise the model to learn novel task semantics. This has been partially verified by Webson and Pavlick (2022), who showed that model performance does not degrade much with irrelevant or misleading instructions. We leave more analysis on instruction-following models for future work.

3.8.4 Significantly Improved Zero-shot Performance. One of our key findings is that it is possible to achieve nearly k-shot performance without using any labeled data, by simply pairing each unlabeled input with a random label and using it as the demonstrations. This means our zero-shot baseline level is significantly higher than previously thought.[13] Future work can further improve the zero-shot performance with relaxed assumptions in access to the unlabeled training data.

12 However, while Reynolds and McDonell (2021) claim that the demonstrations are thus unnecessary, we think using the demonstrations is actually the most unambiguous and the easiest way to prompt the model to perform a task.

13 We take the perspective that using the unlabeled training data is permitted (Kodirov et al. 2015; Wang et al. 2019b; Schick and Schütze 2021).

3.8.5 Limitations. This section focuses on the tasks from established NLP benchmarks that have *real* natural language inputs. Synthetic tasks with more limited inputs may actually use the ground truth labels more, as observed by Rong (2021).

We report macro-level analysis by examining the average performance over multiple NLP datasets, but different datasets may behave differently. See the Appendix of the original publication (Min et al. 2022c) for the relevant discussion, including findings that there are larger gaps between using the ground truth labels and using the random labels in some dataset-model pairs.

Our experiments are limited to classification and multi-choice tasks. We hypothesize that ground truth output may not be necessary for in-context learning in the open-set tasks such as generation, but leave this to future work. Extending our experiments to such tasks is not trivial, because it requires a variation of the output which has incorrect input-output correspondence while keeping the correct output distribution (which is important based on our analysis in § 3.7).

3.9 Discussion of Subsequent Work

Our work in this section has led to a significant body of follow-up research, including LMs that adopted meta-training (Chung et al. 2022; Wang et al. 2022; Ye et al. 2023b; Gu et al. 2023; Ye et al. 2023a; Ivison et al. 2023), work discovering unexpected behaviors of LMs (Madaan and Yazdanbakhsh 2022; Wei et al. 2023b; Burns et al. 2022; Halawi, Denain, and Steinhardt 2023; Jang, Ye, and Seo 2022; Wies, Levine, and Shashua 2023; Lampinen et al. 2022; Pan et al. 2023; Kim et al. 2022; Schaeffer et al. 2023; Zhang et al. 2023b; She et al. 2023; Turpin et al. 2023), and work that used our analysis to inspire the development of better models (Lyu et al. 2023; Wei et al. 2023a). In this section, we discuss closely relevant work at the time of writing this article.

3.9.1 Work That Adopted Meta-training. Since the introduction of MetaICL and other concurrent work in training the LM with in-context learning and natural instructions (Chen et al. 2021; Sanh et al. 2022; Wei et al. 2022a), this recipe has become a standard practice in state-of-the-art LM alignment pipelines. MetaICL is incorporated into FLAN-PaLM, one of Google's leading models whose training recipe is publicly available (Chung et al. 2022), and Tulu v2, one of the best, fully open-sourced instruction-tuned models (Ivison et al. 2023). Arguably, the significance of the ability to perform in-context learning may have diminished in recent models. This is partially because the use of natural instructions alone is often enough to achieve good performance with a sufficiently large base model. Additionally, it is generally easier for users to provide natural instructions than to write a handful of labeled examples. Nevertheless, for complex tasks requiring a chain-of-thought prompting—a variant of in-context learning (Wei et al. 2022b)—further training with in-context examples remains critical.

There have also been counterexamples: For instance, Iyer et al. (2022) found that incorporating MetaICL can actually decrease performance in generation tasks, and its improvements are largely limited to classification and multiple-choice tasks. They suggested that MetaICL may cause the model to become overfitted to the format used in in-context examples. We consider this hypothesis likely to be true; in fact, it aligns with our findings in §3.7. We also posit that differences in results are likely due to variations in settings, such as the choice of base models, meta-training datasets, and evaluation datasets, and these need further investigation.

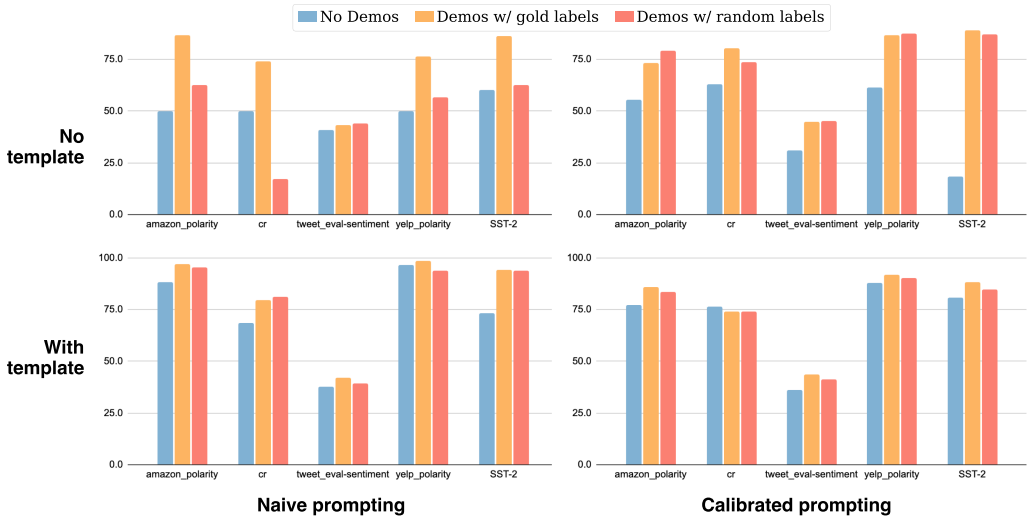

Figure 14
Results with various configurations following Kim et al. (2022): with and without a template, and with and without calibration. Using random labels significantly degrades performance with no template and naive prompting (top left) but not with other configurations.

3.9.2 How Are Findings in Section 3.6 Applied to the Current State-of-the-art LMs?. Several subsequent studies examined the extent to which our findings generalize to larger, state-of-the-art LMs.

Kim et al. (2022) studied the impact of using random or negated labels in in-context learning through various conditions; they claimed that using random labels can perform poorly depending on the choice of datasets, prompting methods (e.g., whether to apply calibration), and the template used for prompting. We find that: (1) although certain combinations of settings (e.g., not using calibration and templates in prompting; see Figure 14) lead to significant degradation in performance with random labels, *in most settings, random labels have minimal impact on performance*; and (2) because the setting for the best in-context learning performance sees minimal impact from using random labels, *our key claim that correct labels are not essential for in-context learning remains true*. For further discussion, we refer readers to slides 44 to 51 from our presentation at the EMNLP 2023 Workshop.

Wei et al. (2023b) argued that large LMs perform in-context learning in a qualitatively different manner, claiming that the capability to in-context learn with negated labels[14] emerges with scale. Although we generally agree that current state-of-the-art LMs perform in-context learning much better, we think that care must be taken in evaluating the impact of compounding factors. For instance, we posit that results in Wei et al. (2023b) reveal that in-context learning with negated labels is near random guessing across all model sizes for models without instruction tuning (see GPT-3 in Figure 15), indicating that the model size is not a contributing factor. This holds true even with instruction tuning up to OpenAI's `text-davinci-001`, although `text-davinci-001` performs well on in-context learning with negated labels. Therefore, though some LLMs

14 For instance, assigning `negative` to a positive review and `positive` to a negative review.

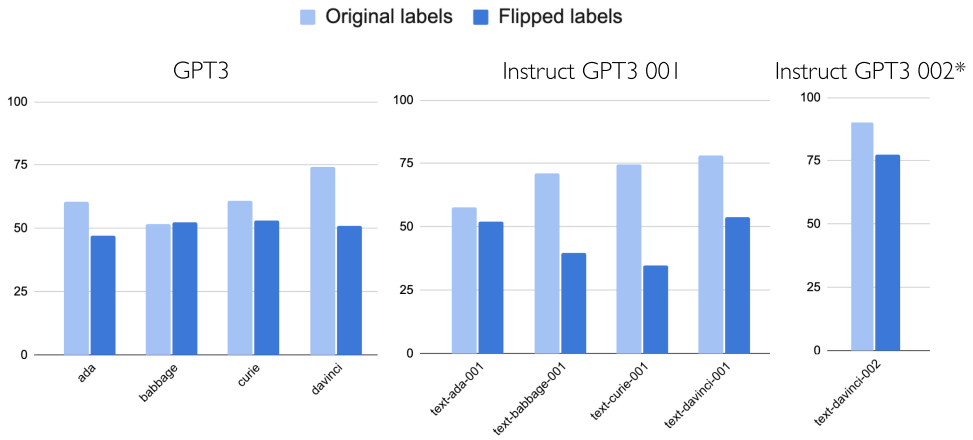

Figure 15
In-context learning with original labels and negated labels. Results are from Wei et al. (2023b); we redrew the figures. *: `text-davinci-002` was released after our paper was posted. All models except for the 002 one achieve near random guessing performance (50%) with flipped labels, suggesting that even large models cannot perform in-context learning when the task defined by in-context examples does not entail with semantics from pre-training.

that are available to us can perform in-context learning with negated labels, we maintain that it is difficult to attribute this solely to model size; it is more likely influenced by some unknown variant of instruction tuning.[15]

3.9.3 Combining Findings from Sections 3.2 and 3.6. Wei et al. (2023a) proposed a new meta-training method that uses $k+1$ labeled examples $(x_1, y_1), \cdots, (x_k, y_k), (x_{k+1}, y_{k+1})$ and maximizes $P(y_{k+1}|x_1, y_1, \cdots, x_k, y_k, x_{k+1})$, as MetaICL does. However, instead of using standard NLP tasks for training, they used synthetic tasks designed to isolate the impact of semantics from the training data, e.g., assigning `negative` to a positive review and `positive` to a negative review, or assigning `foo` to a positive review and `bar` to a negative review. Their experiments demonstrate that models trained on this objective show improved performance in in-context learning with negated labels or semantically unrelated labels.

3.9.4 Extensions to Generation Tasks. Since the initial release of our paper, Madaan and Yazdanbakhsh (2022) conducted a similar analysis with the chain of thought prompting (Wei et al. 2022b), which generates a rationale for performing complex tasks such as arithmetic reasoning. They find that simply using random rationales (e.g., pairing a rationale from a different example) significantly degrades performance, but other types of incorrect rationales (e.g., incorrect equations) do not degrade performance.

15 Wei et al. (2023b) also experimented with in-context learning with semantically unrelated labels, e.g., assigning `foo` to a positive review and `bar` to a negative review. Our findings largely remain the same: Plain language models achieve near random guessing performance over all model sizes, although instruction-tuned models achieve better performance. For further discussion, please refer to slides 55 to 63 from the presentation at the EMNLP 2023 Workshop.

Our own work (Wang et al. 2023a) further studied the impact of using rationales based on flawed reasoning, broken down by different categories of flaws. For instance, we find that maintaining input query relevance and rationale coherence is critical, e.g., not keeping keywords from the input query, or using `32 + 42 = 74` before the introduction of `32` or `42`, significantly degrade performance. Nevertheless, as long as relevance and coherence are maintained, inaccuracies in the underlying reasoning only marginally affect performance. Notably, this approach preserves 81–91% of the accuracy in arithmetic reasoning and 87–98% in multi-hop question answering across a range of state-of-the-art models, including `text-davinci-002` and `text-davinci-003`, as well as the largest versions of PaLM and FLAN-PaLM. For further discussion, see Madaan and Yazdanbakhsh (2022) and Wang et al. (2023a).

4. Nonparametric Language Models

This section describes our work on **nonparametric LMs**, a new class of LMs that include both learned parameters and *a datastore*, i.e., a massive collection of raw text documents. During inference, these models can identify relevant text from the datastore and reason with it, unlike conventional models that must remember every relevant detail from the training set. These models are called nonparametric because their data distribution is not defined by a fixed set of parameters; rather, it is a function of the available data (Siegel 1957; Hollander, Wolfe, and Chicken 2013). With complexity that grows as the data grows, they are differentiated from parametric models, whose complexity is bounded a priori. As noted by Freeman, Jones, and Pasztor (2002), the term nonparametric does not imply an absence of parameters but rather indicates that the number and nature of the *effective* parameters are flexible and depend on the data.

Nonparametric LMs offer several advantages, as described below.

Parameter efficiency. Conventional models require a large number of parameters in order to memorize every relevant detail from the training set (Brown et al. 2020a; Rae et al. 2021; Chowdhery et al. 2022). Nonparametric models remove the need for such extensive memorization since they can look up relevant information from a datastore. Consequently, they outperform heavily parameterized models, e.g., LMs with 30x more parameters (Raffel et al. 2020) in answering factoid questions (Min et al. (2021a); §4.3).

Capturing of long-tail distributions. Standard state-of-the-art models often fail to capture information within long-tail distributions in the training data. Nonparametric models address this issue by retrieving relevant information at test time, significantly improving accuracy in long-tail distribution data, as shown in Mallen et al. (2022), Min et al. (2023b). This is also evident in commercial systems; for instance, our findings in Min et al. (2023a) demonstrate that factual accuracy in writing biographies increases from 58% with ChatGPT to 72% with PerplexityAI (ChatGPT with a search engine that uses a web datastore).

Ease of updating. Conventional models are strictly inflexible and remain stale once training is completed. In contrast, nonparametric LMs are more flexible by design since the datastore can be altered at any time, e.g., to include up-to-date information without additional training (Min et al. 2021a; Izacard et al. 2022b; Min et al. 2023b).

Ease of data removal. Removing the effect of specific data from LMs after training is very challenging. Despite techniques like machine unlearning (Cao and Yang 2015; Bourtoule et al. 2020; Jang et al. 2023b), such models lack guarantees and are difficult to scale. Nonparametric models, in contrast, enable direct removal of data from the datastore, effortlessly eliminating its influence without requiring additional training.

Ease of data attributions. Nonparametric LMs allow inherent data attributions since they are straightforward to trace, i.e., show which pieces of text in a datastore contribute to predictions. This lets users verify the model output and sources of information.

A combination of the last two features also helps **mitigate legal risks in LM training**, which we will discuss further in Section 5.

The concept of integrating an LM with a text datastore is not new and has been extensively studied in prior research (Chen et al. 2017; Gu et al. 2018; Song et al. 2018; Tian et al. 2019). We provide background context in §4.2.3 and discuss key contributions we made for a nonparametric approach within the context of language models.

In §4.2, we discuss retrieval augmentation, which operates in two stages: (1) retrieving text from a datastore and (2) conditioning the LM on the retrieved passages (Figure 3a). While intuitive, similar to humans using a search engine, training models to reason with large datastores is challenging. We introduced some of the first retrieval-augmented models that addressed these issues and spurred extensive follow-up work (Karpukhin et al. 2020; Min et al. 2019b, 2021a; Shi et al. 2024a, b).

A key milestone was Dense Passage Retrieval (Karpukhin et al. 2020), which demonstrated the advantages of neural over lexical retrieval by introducing a contrastive objective to better separate relevant from distracting text. We further improved the second stage by training LMs to condition on multiple retrieved documents, enhancing their utility (Shi et al. 2024a). Overall, retrieval augmentation boosts performance across tasks, improves handling of rare facts, and even outperforms large commercial LMs like GPT-3 (Shi et al. 2024b). Moreover, such models are easily updatable and substantially smaller than conventional alternatives, as we showed in Min et al. (2021a).

While retrieval augmentation has made significant impact, whether it is the optimal approach for using the datastore remains an open question. The design of the architecture makes it difficult to scale the amount of text retrieved from the datastore each time. Also, as we showed (Min et al. 2024), performance of these models do not scale as effectively with the datastore size as desired.

In Section 4.4, we explore an alternative that incorporates the datastore differently—by training an LM with *a nonparametric softmax*. Instead of outputting a categorical distribution over words, this method assigns scores to every word or phrase in the datastore as a nonparametric distribution, e.g., assigning a high score to ‘‘`green light`’’ from a Harry Potter book (Figure 3b). This approach both uses the datastore more effectively by incorporating a larger portion of the data at each step and handles rare concepts better, assigning high scores to unseen tokens or phrases if their surrounding context is relevant. We introduce NPM (Min et al. 2023b), one of the first such models. Training NPM at scale posed several technical challenges: For instance, scoring all phrases in large-scale data for every training iteration is very expensive. To address this, we designed novel techniques such as scalable batching and a training objective that effectively approximates a distribution over the full corpus. Our empirical results demonstrate that NPM outperforms alternatives on a range of tasks, is especially effective in handling rare concepts (such as rare entities and facts), and can grow and be updated by expanding and replacing the datastore. Notably, this approach scales and generalizes better than retrieval-augmented LMs (Min et al. 2024).

4.1 Background

The idea of incorporating the external datastore for end tasks has existed for a long time (Voorhees 1999; Teh 2006; Zhang et al. 2006; Ceri et al. 2013; Zhao and Cho 2018). This

section provides the background focusing on using the raw text datastore for NLP tasks in the context of deep neural models.

4.1.1 Retrieval Augmentation. The most widely recognized form of the nonparametric method is known as retrieval augmentation. This technique is often referred to as retrieval augmented generation, a term coined by Lewis et al. (2020b). While the idea of retrieval augmentation can be dated back to Teh (2006), Zhang et al. (2006), Ceri et al. (2013), Zhao and Cho (2018), Chen et al. (2017), Gu et al. (2018), and Song et al. (2018), the work of Tian et al. (2019) was one of the first to offer a definition that corresponds with its modern understanding: Retrieval augmentation leverages results R from an information retrieval system to augment the input used in an LM. Their objectives are to maximize $P(r|q, R)$, where q is the input, r is a response, and R is one or a few retrieved documents (at most 100 in practice).

A series of earlier work has used a deep neural network together with a datastore consisting of a large text corpus, often designed or trained in a task-specific manner (Chen et al. 2017; Gu et al. 2018; Song et al. 2018; Tian et al. 2019). Notably, open-domain QA—answering a factoid question without a specific reference document given (Voorhees 1999)—is one of the tasks that greatly benefits from retrieval augmentation. Researchers have made progress on different components of retrieval augmentation for open-domain question answering, including the augmentation method (Chen et al. 2017; Yang et al. 2019a, b; Nie, Wang, and Bansal 2019; Wolfson et al. 2020, and our own work (Min et al. 2019a), reranking (Nogueira and Cho 2019, Humeau et al. 2020, and our own work (Iyer et al. 2021), augmentation (Chen et al. 2017; Yang et al. 2019a, b; Nie, Wang, and Bansal 2019; Wolfson et al. 2020, and our own work (Min et al. 2019a). Most retrieval augmentation used sparse retrieval including TF-IDF and BM25 (Robertson, Zaragoza et al. 2009), rather than neural-based retrieval.

Neural-based retrieval, called **dense retrieval** henceforth, has a long history since Latent Semantic Analysis (Deerwester et al. 1990). The progress has been centered around how to train the encoder that maps a query and a document into a vector space, typically using labeled pairs of queries and documents (Yih et al. 2011; Huang et al. 2013; Gillick et al. 2019). Such approaches complement the sparse vector methods as they can potentially give high similarity scores to semantically relevant text pairs, even without exact token matching. The dense representation alone, however, is typically inferior to the sparse one.

More recent work proposes a pre-training method for a joint training of a dense retrieval model and a language model (Lee, Chang, and Toutanova 2019; Guu et al. 2020). While effective, training of such models is very expensive and is difficult to train stably due to engineering complexities and hyperparameters coming from the necessity to asynchronously re-index the datastore during training.

4.1.2 A Nonparametric Softmax. The idea of using a nonparametric softmax is not new, and has been studied by a series of prior work (Khandelwal et al. 2020; Yogatama, de Masson d'Autume, and Kong 2021; Zhong, Lei, and Chen 2022; Lan et al. 2023)—so-called kNN-LM models. Our work is heavily inspired by the kNN-LM approach, and can be seen as an extreme version of it with no interpolation. However, NPM is the first that models a *fully* nonparametric distribution by entirely removing the softmax over a finite vocabulary. This offers a range of new functionalities, such as modeling a distribution over phrases, or predicting rare or unseen words.

Prior work has explored nonparametric inference without training (Khandelwal et al. 2020; He, Neubig, and Berg-Kirkpatrick 2021; Xu et al. 2022), or trained the

nonparametric model on the labeled data for a specific downstream task (Seo et al. 2018, 2019; Lee et al. 2021). In contrast, NPM is a fully nonparametric language model without the labeled data and performs a range of tasks zero-shot.

4.2 Retrieval Augmentation

In this section, we first discuss Dense Passage Retrieval (DPR), one of the first widely used neural retrieval models that led to a wide adaptation of retrieval augmentation. We cover the neural retrieval model (§4.2.1) and how results from this model are incorporated into a language model (§4.2.3). Following this, we outline the experimental setup (§4.2.4) and present the results (§4.2.5).

DPR sparked discussion on parametric versus nonparametric LMs, leading to a competition on open-domain question answering at NeurIPS 2021 called EfficientQA. We discuss the details of the competition, including the setup, participating models, results, and analyses (§4.3). EfficientQA demonstrated the impact of retrieval augmentation and DPR through the contributions of many external participants.

Since then, retrieval augmentation has seen rapid improvements. We discuss subsequent work that has driven these improvements in Section 4.5.

4.2.1 DPR Overview. Given a collection of M text passages, the goal of our DPR is to index all the passages in a low-dimensional and continuous space, such that it can retrieve efficiently the top k passages relevant to the input question for the reader at run-time. Note that M can be very large (e.g., 21 million passages in our experiments using the English Wikipedia) and k is usually small (e.g., 100 in our experiments).

DPR uses a dense encoder $E_P(\cdot)$ which maps any text passage to a d-dimensional real-valued vector and builds an index for all the M passages that we will use for retrieval. At run-time, DPR applies a different encoder $E_Q(\cdot)$ that maps the input question to a d-dimensional vector, and retrieves k passages of which vectors are the closest to the question vector. We define the similarity between the question and the passage using the dot product of their vectors: $\text{sim}(q, p) = E_Q(q)^\intercal E_P(p)$.

Encoders. Although in principle the question and passage encoders can be implemented by any neural network, in this work we use two independent BERT (Devlin et al. 2019) networks (base, uncased) and take the representation at the `[CLS]` token as the output, so $d = 768$.

Inference. We apply the passage encoder E_P to all the passages and index them using FAISS (Johnson, Douze, and Jégou 2019) offline. FAISS is an extremely efficient, open-source library for similarity search and clustering of dense vectors, which can easily be applied to billions of vectors. At run-time, given a question q, we derive its embedding $v_q = E_Q(q)$ and retrieve the top k passages with embeddings closest to v_q.

4.2.2 DPR Training. Training the encoders so that the dot-product similarity becomes a good ranking function for retrieval is essentially a *metric learning* problem (Kulis 2013). The goal is to create a vector space such that relevant pairs of questions and passages will have smaller distance (i.e., higher similarity) than the irrelevant ones, by learning a better embedding function.

Let $\mathcal{D} = \{\langle q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^- \rangle\}_{i=1}^m$ be the training data that consists of m instances. Each instance contains one question q_i and one relevant (positive) passage p_i^+, along

with n irrelevant (negative) passages $p^-_{i,j}$. We optimize the loss function as the negative log likelihood of the positive passage:

$$L(q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^-) = -\log \frac{e^{\text{sim}(q_i, p_i^+)}}{e^{\text{sim}(q_i, p_i^+)} + \sum_{j=1}^{n} e^{\text{sim}(q_i, p_{i,j}^-)}}$$

Positive and Negative Passages. For retrieval problems, it is often the case that positive examples are available explicitly, while negative examples need to be selected from an extremely large pool. For instance, passages relevant to a question may be given in a QA dataset, or can be found using the answer. All other passages in the collection, while not specified explicitly, can be viewed as irrelevant by default. In practice, how to select negative examples is often overlooked but could be decisive for learning a high-quality encoder. We consider three different types of negatives: (1) Random: any random passage from the corpus; (2) BM25: top passages returned by BM25 that don't contain the answer but match most question tokens; (3) Gold: positive passages paired with other questions which appear in the training set. Our best model uses gold passages from the same mini-batch and one BM25 negative passage. In particular, re-using gold passages from the same batch as negatives can make the computation efficient while achieving great performance. We discuss this approach below.

In-batch Negatives. Assume that we have B questions in a mini-batch and each one is associated with a relevant passage. Let $\mathbf{Q}$ and $\mathbf{P}$ be the $(B \times d)$ matrix of question and passage embeddings in a batch of size B. $\mathbf{S} = \mathbf{Q}\mathbf{P}^T$ is a $(B \times B)$ matrix of similarity scores, where each row of which corresponds to a question, paired with B passages. In this way, we reuse computation and effectively train on B^2 (q_i, p_j) question/passage pairs in each batch. Any (q_i, p_j) pair is a positive example when $i = j$, and negative otherwise. This creates B training instances in each batch, where there are $B - 1$ negative passages for each question.

The trick of in-batch negatives has been used in the full batch setting (Yih et al. 2011) and more recently for mini-batch (Henderson et al. 2017; Gillick et al. 2019). It has been shown to be an effective strategy for learning a dual-encoder model that boosts the number of training examples.

4.2.3 Augmentation. Now, we describe a method that incorporates retrieved passages into an LM, often called a *reader*. Given the top k retrieved passages (up to 100 in our experiments), the reader assigns a passage selection score to each passage. In addition, it extracts an answer span from each passage and assigns a span score. The best span from the passage with the highest passage selection score is chosen as the final answer. The passage selection model serves as a reranker through cross-attention between the question and the passage. Although cross-attention is not feasible for retrieving relevant passages in a large corpus due to its non-decomposable nature, it has more capacity than the dual-encoder model $\text{sim}(q, p)$. Applying it to selecting the passage from a small number of retrieved candidates has been shown to work well (Wang et al. 2019c, 2018b; Lin et al. 2018).

Specifically, let $\mathbf{P}_i \in \mathbb{R}^{L \times h}$ $(1 \leq i \leq k)$ be a BERT (base, uncased in our experiments) representation for the i-th passage, where L is the maximum length of the passage and h

the hidden dimension. The probabilities of a token being the starting/ending positions of an answer span and a passage being selected are defined as:

$$P_{\text{start},i}(s) = \text{softmax}\big(\mathbf{P}_i \mathbf{w}_{\text{start}}\big)_s$$
$$P_{\text{end},i}(t) = \text{softmax}\big(\mathbf{P}_i \mathbf{w}_{\text{end}}\big)_t$$
$$P_{\text{selected}}(i) = \text{softmax}\big(\hat{\mathbf{P}}^\intercal \mathbf{w}_{\text{selected}}\big)_i$$

where $\hat{\mathbf{P}} = [\mathbf{P}_1^{[\text{CLS}]}, \ldots, \mathbf{P}_k^{[\text{CLS}]}] \in \mathbb{R}^{h \times k}$ and $\mathbf{w}_{\text{start}}, \mathbf{w}_{\text{end}}, \mathbf{w}_{\text{selected}} \in \mathbb{R}^h$ are learnable vectors. We compute a span score of the s-th to t-th words from the i-th passage as $P_{\text{start},i}(s) \times P_{\text{end},i}(t)$, and a passage selection score of the i-th passage as $P_{\text{selected}}(i)$.

During training, we sample one positive and $\tilde{m} - 1$ negative passages from the top 100 passages returned by the retrieval system (BM25 or DPR) for each question. $\tilde{m}$ is a hyperparameter and we use $\tilde{m} = 24$ in all the experiments. The training objective is to maximize the marginal log-likelihood of all the correct answer spans in the positive passage (the answer string may appear multiple times in one passage), combined with the log-likelihood of the positive passage being selected. We use a batch size of 16 for large (NQ, TriviaQA, SQuAD) and 4 for small (TREC, WQ) datasets, and tune k on the development set. For experiments on small datasets under the *Multi* setting, in which using other datasets is allowed, we fine-tune the reader trained on Natural Questions to the target dataset. All experiments were done on eight 32GB GPUs.

4.2.4 DPR Experimental Setup. We describe the data we used for experiments and the basic setup.

Wikipedia Data Pre-processing. Following Lee, Chang, and Toutanova (2019), we use the English Wikipedia dump from 20 December 2018 as the source documents for answering questions. We first apply the pre-processing code released in DrQA (Chen et al. 2017) to extract the clean, text-portion of articles from the Wikipedia dump. This step removes semi-structured data, such as tables, info-boxes, lists, as well as the disambiguation pages. We then split each article into multiple, disjoint text blocks of 100 words as *passages*, serving as our basic retrieval units, following Wang et al. (2019c), which results in 21,015,324 passages in the end.[16] Each passage is also prepended with the title of the Wikipedia article from which the passage is pulled, along with an `[SEP]` token.

Question Answering Datasets. We use the same five QA datasets and training/dev/testing splitting method as in previous work (Lee, Chang, and Toutanova 2019). Below we briefly describe each dataset and refer readers to their paper for the details of data preparation.

Natural Questions (NQ) (Kwiatkowski et al. 2019) was designed for end-to-end question answering. The questions were mined from real Google search queries and the answers were spans in Wikipedia articles identified by annotators.
TriviaQA (Joshi et al. 2017) contains a set of trivia questions with answers that were originally scraped from the Web.

16 However, Wang et al. (2019c) also propose splitting documents into overlapping passages, which we do not find advantageous compared with the non-overlapping version.

WebQuestions (WQ) (Berant et al. 2013) consists of questions selected using Google Suggest API, where the answers are entities in Freebase.
CuratedTREC (TREC) (Baudiš and Šedivỳ 2015) sources questions from TREC QA tracks as well as various Web sources and is intended for open-domain QA from unstructured corpora.
SQuAD v1.1 (Rajpurkar et al. 2016) is a popular benchmark dataset for reading comprehension. Annotators were presented with a Wikipedia paragraph, and asked to write questions that could be answered from the given text. Although SQuAD has been used previously for open-domain QA research, it is not ideal because many questions lack context in absence of the provided paragraph. We still include it in our experiments for providing a fair comparison to previous work.

Selection of Positive Passages. Because only pairs of questions and answers are provided in TREC, WebQuestions, and TriviaQA, we use the highest-ranked passage from BM25 that contains the answer as the positive passage. If none of the top 100 retrieved passages has the answer, the question is discarded. For SQuAD and Natural Questions, since the original passages have been split and processed differently than our pool of candidate passages, we match and replace each gold passage with the corresponding passage in the candidate pool. We discard the questions when the matching is failed due to different Wikipedia versions or pre-processing.

4.2.5 DPR Experimental Results. Table 7 summarizes our QA results, measured by *exact match* with the reference answer after minor normalization as in Chen et al. (2017) and Lee, Chang, and Toutanova (2019). DPR performs consistently better than prior work using BM25 on all datasets except SQuAD. For large datasets like NQ and TriviaQA, models trained using multiple datasets (Multi) perform comparably to those trained using the individual training set (Single). Conversely, on smaller datasets like WQ and TREC, the multi-dataset setting has a clear advantage. Overall, our DPR-based models outperform the previous state-of-the-art results on four out of the five datasets, with 1% to 12% absolute differences in exact match accuracy.

Table 7
QA (Exact Match) Accuracy. The first block of results are copied from their cited papers. $\text{REALM}_{\text{Wiki}}$ and $\text{REALM}_{\text{News}}$ are the same model but pre-trained on Wikipedia and CC-News, respectively. *Single* and *Multi* denote that DPR is trained using individual or combined training datasets (all except SQuAD). For WQ and TREC in the *Multi* setting, we fine-tune the reader trained on NQ.

Training	Model	NQ	TriviaQA	WQ	TREC	SQuAD
Single	BM25+BERT (Lee, Chang, and Toutanova 2019)	26.5	47.1	17.7	21.3	33.2
Single	ORQA (Lee, Chang, and Toutanova 2019)	33.3	45.0	36.4	30.1	20.2
Single	HardEM (Min et al. 2019a)	28.1	50.9	–	–	–
Single	GraphRetriever (Min et al. 2019b)	34.5	56.0	36.4	–	–
Single	PathRetriever (Asai et al. 2020)	32.6	–	–	–	**56.5**
Single	$\text{REALM}_{\text{Wiki}}$ (Guu et al. 2020)	39.2	–	40.2	46.8	–
Single	$\text{REALM}_{\text{News}}$ (Guu et al. 2020)	40.4	–	40.7	42.9	–
Single	BM25	32.6	52.4	29.9	24.9	38.1
	DPR	**41.5**	56.8	34.6	25.9	29.8
	BM25+DPR	39.0	57.0	35.2	28.0	36.7
Multi	DPR	**41.5**	56.8	**42.4**	49.4	24.1
	BM25+DPR	38.8	**57.9**	41.1	**50.6**	35.8

It is worth contrasting our results to those of ORQA (Lee, Chang, and Toutanova 2019) and also the concurrently developed approach, REALM (Guu et al. 2020). While both methods include additional pre-training tasks and use an expensive end-to-end training regime, DPR manages to outperform them on both NQ and TriviaQA, simply by focusing on learning a strong passage retrieval model using pairs of questions and answers. The additional pre-training tasks are likely more useful only when the target training sets are small. Although the results of DPR on WQ and TREC in the single-dataset setting are less competitive, adding more question-answer pairs helps boost the performance, achieving the new state of the art.

4.3 NeurIPS 2020 EfficientQA Competition

In 2020, the year DPR has been introduced, there has been significant progress on large language models, raising the possibility of representing world knowledge solely in the parameters of a neural model instead of using a datastore (Roberts, Raffel, and Shazeer 2020). In this context, we organized the EfficientQA competition, held at NeurIPS 2020, which required contestants to build self-contained systems that contain all of the knowledge required to answer open-domain questions. Participants can explore either parameteric or nonparametric language models to optimize for model performance. However, the competition encouraged systems that store and access this knowledge using the smallest number of bytes, including code, corpora, and model parameters. Specifically, EfficientQA had four tracks: (1) best accuracy overall (unconstrained); (2) best accuracy, system size under 6GiB; (3) best accuracy, system size under 500MiB; and (4) smallest system to get 25% accuracy. These memory budgets were designed to encourage contestants to explore the trade-off between storing and accessing large datastores or the parameters of neural models.

Key Takeaways. The top submissions in each of EfficientQA's four tracks significantly outperformed the provided baselines. All top submissions use a retrieval corpus and a neural network answering module. However, the nature of the retrieval corpus and answering module differs drastically across the tracks (Table 8).

Table 8
A list of the baselines and systems from participants, along with the team affiliation and key distinction between systems. *Nonparam* means whether the model is nonparametric (✓) or parametric (✗). Key features indicate some of the key features, such as whether the datastore is pruned (*pruned*), whether the answer is extracted (*ext*) or generated (*gen*), and others. We refer to the original publication (Min et al. 2021a) for more detailed descriptions of each system.

Track	Model	Affiliation	Nonparam?	Key features
Unrestricted	REALM	Organizers	✓	ext
	DPR	Organizers	✓	ext
	MS UnitedQA	Microsoft & Dynamics 365	✓	ext+gen
	FB Hybrid	Facebook AI	✓	gen, lists/tables
6GiB	DPR-subset	Organizers	✓	ext, pruned
	T5-XL+SSM	Organizers	✗	gen
	FB system	FAIR-Paris&London	✓	gen, lists, pruned, lrzip compression
	Ousia-Tohoku Soseki	Studio Ousia, Tohoku U & RIKEN	✓	ext, pruned, ZPAQ compression
	BUT R2-D2	Brno U of Technology	✓	ext+gen, pruned
500MiB	T5-Small+SSM	Organizers	✗	gen
	UCLNLP-FB system	UCL & FAIR	✓,	data augmentation
	NAVER RDR	NAVER Clova	✓	ext, pruned, single Transformer
25% smallest	T5-XL+SSM	Organizers	✗	gen
	UCLNLP-FB system (29M)	UCL & FAIR	✓	data augmentation

Unrestricted and 6GiB Tracks. The top submissions to the unrestricted track and the 6GiB track outperformed the state-of-the-art baselines from April 2020 by nearly 20%. They achieved this improvement by combining an improved version of DPR (Karpukhin et al. 2020; Mao et al. 2020) with answer generation (Izacard and Grave 2020); leveraging the state-of-the-art in text generation (Raffel et al. 2020) and text encoding (Clark et al. 2020); including not only text but also tables and lists; and combining the extractive and generative answer prediction. The top submissions to the 6GiB track reduced the size of their indexed corpus and made use of the state-of-the-art in compression, with minimal impact on accuracy.

500MiB and Smallest Tracks. To get under 500MB, the systems made more drastic changes. The submission from NAVER Clova reduced the size of their indexed corpus and reused a single Transformer model for the retriever and the reader, winning the 500MiB track according to the human evaluation. The even smaller UCLNLP-FB system took a novel approach in generating a large corpus of question-answer pairs, indexing it, and retrieving the most similar question to the input question. This approach, with two systems with different sizes in question-answer corpus, won both the 500MiB track and the smallest 25% track according to the automatic evaluation.

Automatic vs. Human Evaluation. The human evaluation supports the observation that automatic metrics often incorrectly penalize correct predictions in the open-domain setting (Voorhees and Tice 2000; Roberts, Raffel, and Shazeer 2020). We also investigate the effect of question ambiguity on evaluation—the questions from NQ are often ambiguous without the associated evidence document (Min et al. 2020). We define an annotation scheme that supports multiple estimations of accuracy, corresponding to different definitions of correctness for answers to ambiguous questions. Almost all systems' accuracy increased by 20–25% under the strictest definition of correctness. The increase doubled when we relaxed the definition of correctness to permit any semantically valid interpretation of the question.

4.3.1 Competition Setup. The competition uses English questions and answers from the Natural Questions dataset (Kwiatkowski et al. 2019; NQ), the same dataset as the one used in Section 4.2.4. NQ was collected over the course of 2017 and 2018. For EfficientQA, we introduce a new test and development set constructed in the same way as the original NQ, but labeled slightly after (early 2019 rather than through 2018). Our test set was kept hidden from contestants, and submissions were made by uploading solutions to an automatic leaderboard.

Automatic Evaluation. The accuracy of systems' predicted answers is judged against reference annotations, annotated by five human workers. We follow the literature in using exact match between predicted and reference answers after minor normalization.

Human Evaluation. Because language and QA are inherently ambiguous, five reference answers are often insufficient, leading to correct predictions being marked incorrect by automatic metrics. To better estimate accuracy, predictions from each system were rated by three independent annotators. Annotators first interpreted the question (using web search if necessary), then assessed whether it was ambiguous (e.g., time-dependent, location-specific, or reliant on unstated intent), and finally judged each answer as "definitely correct" (correct given a usual interpretation of the question), "possibly correct"

Table 9
Summary of the result. For human evaluation result, relative improvements over the automatic evaluation are indicated in parenthesis. Following our analysis of the annotations, we use *Definitely correct* human ratings as a primary metric.

Track	Model	Automatic eval	Human eval	
			Definitely	Possibly
Unrestricted	MS UnitedQA	54.00	65.80 (+21.9%)	78.12 (+44.7%)
	FB Hybrid	53.89	67.38 (+25.0%)	79.88 (+48.2%)
6GiB	FB system	53.33	65.18 (+22.2%)	76.09 (+42.7%)
	Ousia-Tohoku Soseki	50.17	62.01 (+23.6%)	73.83 (+47.2%)
	BUT R2-D2	47.28	58.96 (+24.7%)	70.33 (+49.2%)
500MiB	UCLNLP-FB system	33.44	39.40 (+17.8%)	47.37 (+41.7%)
	NAVER RDR	32.06	42.23 (+31.7%)	54.95 (+71.4%)
25% smallest	UCLNLP-FB system (29M)	26.78	32.45 (+21.2%)	41.21 (+53.9%)

(could be correct, given some interpretation of the question),[17] or "definitely incorrect". Final labels were determined by majority vote: $\geq 2/3$ agreement on definitely correct yielded that label; $\geq 2/3$ agreement on definitely correct or possibly correct yielded possibly correct. Inter-annotator agreement was 69.2% (Cohen's $\kappa = 53.8$) for 3-way ratings, 85.7% ($\kappa = 71.4$) for definitely correct, and 76.7% ($\kappa = 53.4$) for possibly correct.

4.3.2 Competition Results. All five systems in the unrestricted track and the 6GiB track significantly outperform the previous state-of-the-art (Table 9): DPR (36.6%) and REALM (35.9%). Systems in the 6GiB track approach the unrestricted track's accuracy; for instance, the accuracy FB system is comparable to the accuracy of the top systems in the unrestricted track. The improvements in the 500MiB track are also impressive; both the top two systems significantly beat T5-small (17.6%).

Discrepancy Between Automatic Eval and Human Eval. Human raters find 13% and 17% of the predictions that do not match the reference answers to be definitely correct or possibly correct, respectively, overall increasing the accuracy of the systems. Most systems showed 17–25% and 41–54% improvement in accuracy when using definitely correct and possibly correct human evaluation, respectively, compared with automatic evaluation metrics which only consider exact string match to existing reference answers. An exception is NAVER RDR, which achieves significantly larger improvements (32% and 71%, respectively). When the gap in automatic measure between systems is marginal (around or smaller than 1%), human evaluation may change the rankings between the models.

Agreement Between System Predictions. Figure 16 (left) shows the agreement between system predictions, based on exact match in automatic evaluation. The largest agreement is made between FB Hybrid and FB system, likely because both are based on DPR and Fusion-in-Decoder. Agreements between systems in the unrestricted and the

17 Since the original NQ data was collected more than a year before the start of the EfficientQA competition, the denotation of some questions may have changed over time (e.g., "who won the last season of bake-off"). Rather than determine a single correct point in time for these questions, we asked our annotators to assume that the query could have been asked at any time and choose the "possibly correct" label for answers that may or may not have been correct when the question was asked.

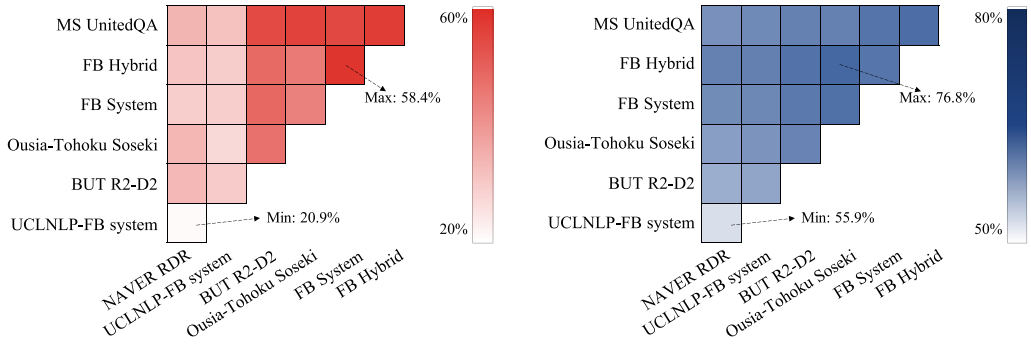

Figure 16
(Left) Agreement between system predictions. *(Right)* Ensemble oracle accuracy, which considers a prediction correct if at least one of the system predictions is correct (based on "definitely correct" human evaluation).

6GiB tracks are generally higher. In contrast, the two 500MiB-track systems show lower agreement with others, and even less with each other.

Ensemble Oracle Accuracy of the Systems. Figure 16 (right) reports the ensemble oracle accuracy for each system pair, which considers a prediction to be correct if either system prediction is correct. The FB Hybrid & Ousia-Tohoku Soseki pair achieves the highest ensemble oracle accuracy, indicating that their system predictions are substantially different from each other compared to other pairs with top performing systems.

Qualitative Analysis. We analyze 50 questions where at least one system prediction differed from the gold answer but was judged correct by humans as "definitely" or "possibly correct". We refer to Section 5.2 of the original publication (Min et al. 2021a) for the discussion, and summarize the main takeaways here.

First, automatic evaluation fails to capture many semantically equivalent answers, accounting for 60% of "definitely correct" predictions, consistent with the findings in Voorhees and Tice (2000). Second, question ambiguity is common, allowing multiple valid interpretations, as also reported in Min et al. (2020). Human annotations on ambiguity show low agreement (61.3%, $\kappa = 22.6$), with raters often disagreeing on whether answers were "definitely" or "possibly correct." Correctness thus reflects pragmatic interpretation rather than a binary distinction. For example, the question "who has the most superbowl rings" can validly refer to a person, player, or team, leading annotators to diverge on which answer counts as correct.

These findings highlight that many real-world questions require non-literal assumptions about the questioner's intent, and we believe that the natural language processing community should not shy away from that task. We also acknowledge that there is work to be done in creating better, non-binary, definitions of correctness.

Performance on Unambiguous Questions. To better understand the effect of ambiguity on the ranking of different solutions, we also evaluate system performance on a subset of unambiguous questions, defined as those that (1) have at least three out of five reference answers containing valid short answers,[18] and (2) are not labeled as ambiguous by any

18 This follows the original NQ approach of using annotator agreement to judge the answer quality.

Table 10
Human evaluation on the original set and a subset of unambiguous questions.

Model	All		Unambiguous Qs	
	Definitely	Possibly	Definitely	Possibly
MS UnitedQA	65.80	78.12	78.24	81.18
FB Hybrid	67.38	79.88	82.65	85.59
FB system	65.18	76.09	79.12	81.47
Ousia-Tohoku Soseki	62.01	73.83	72.94	75.00
BUT R2-D2	58.96	70.55	69.71	72.06
UCLNLP-FB system	39.40	47.37	42.06	43.24
NAVER RDR	42.23	54.95	49.71	53.24
UCLNLP-FB system (29M)	32.45	41.21	28.53	30.29

of three human raters. Table 10 shows human evaluation on the original set and this subset of unambiguous questions. Most systems, except the UCLNLP-FB system, achieve higher accuracy on unambiguous questions, with the first three systems achieving over or near 80%. The gap between "definitely correct" accuracy and "possibly correct" accuracy is marginal on unambiguous questions.

4.4 Nonparametric Prediction

So far, we discussed retrieval augmentation as one category of a nonparametric LM that operates as a function of the data given at test time. As we discussed in §4.2, these models retrieve text from the datastore which are subsequently fed into the parametric LM. However, their final predictions are still made by a parametric model. In particular, they still include a softmax over a finite vocabulary, which limits expressivity (Yang et al. 2018) and can make them reluctant to predict rare or unseen tokens (e.g., *Thessaloniki*).

In this section, we introduce NPM, the first **N**on**P**arametric **M**asked Language Model that predicts tokens solely based on a nonparametric distribution over *phrases* in a text corpus (Figure 17). NPM consists of an *encoder* that maps the text into a fixed-sized vector, and a *reference corpus* from which NPM retrieves a phrase and fills in the `[MASK]`. It, crucially, does not have a softmax over a fixed vocabulary, but instead has a *fully nonparametric* distribution over phrases. This is in contrast to retrieval augmentation that incorporates nonparametric components in a parametric model (Borgeaud et al. 2022; Izacard et al. 2022b).

Training such a nonparametric model introduces two key challenges: (1) full corpus retrieval during training is expensive, and (2) learning to predict an arbitrary length phrase without a decoder is non-trivial. We address the first challenge by using in-batch approximations to full corpus retrieval (Wu et al. 2020; Zhong, Lei, and Chen 2022), and the second by extending span masking (Joshi et al. 2020) and a phrase-level contrastive objective (Oord, Li, and Vinyals 2018; Lee et al. 2021).

We perform zero-shot evaluation on 16 tasks including classification, fact probing, and question answering. We also include temporal shift and word-level translation tasks that highlight the need to predict new facts or rare phrases. NPM is significantly more parameter-efficient, outperforming up to 500× larger parametric models and up to 37× larger retrieval-augmented models, particularly in predicting rare words.

4.4.1 Inference. NPM consists of an encoder and a reference corpus, and models a nonparametric distribution over a reference corpus (Figure 17). The key idea is to map

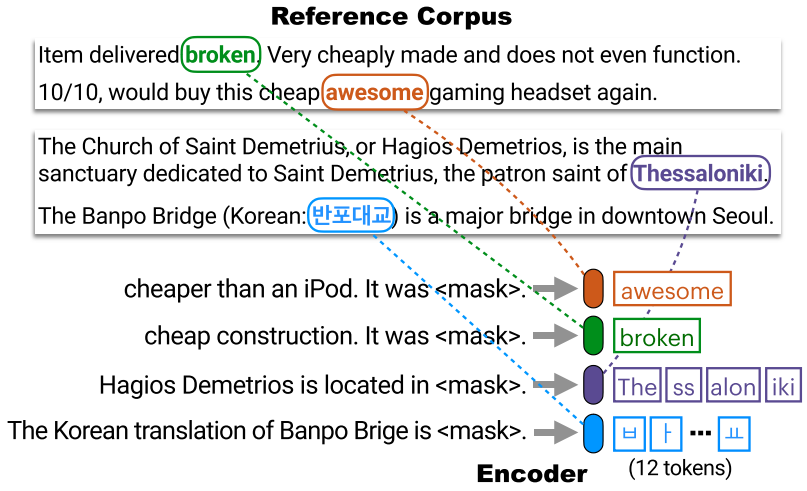

Figure 17
An illustration of NPM. The *encoder* maps a masked sentence into a dense vector, and retrieves the nearest phrase from a *reference corpus*. NPM can fill in the `[MASK]` with multiple tokens, e.g., *Thessaloniki* (4 BPE tokens) and unseen words, e.g., 반포대교 (12 BPE tokens).

all the phrases in the corpus into a dense vector space using the encoder and, when given a query with a `[MASK]` at inference, use the encoder to locate the nearest phrase from the corpus and fill in the `[MASK]`.

Encoder-only models are competitive representation models (Patel et al. 2022), outperforming the other two classes of models in classification tasks (§4.4.4). However, existing encoder-only models are unable to make a prediction whose number of tokens is unknown, making their use cases limited without fine-tuning. NPM addresses this issue, since it can fill in the `[MASK]` with an arbitrary number of tokens by retrieving a *phrase*.

This section describes inference of NPM assuming a learned encoder; then §4.4.2 describes how to train the encoder.

Overview. The encoder maps every distinct *phrase* in a reference corpus $\mathcal{C}$ into a dense vector space. At test time, the encoder maps the masked query into the same vector space and retrieves phrases from $\mathcal{C}$ to fill in the `[MASK]`. Here, $\mathcal{C}$ does not have to be the same as the training corpus, and can be replaced or scaled at test time without re-training the encoder.

In practice, there is a significant number of phrases in the corpus, and it is expensive to index all of them. We therefore use a technique from Lee et al. (2021) that represents a phrase with *token* representations of the start and the end of the phrase. In this approach, we index representations of each distinct token in $\mathcal{C}$, and then at test time, use a k nearest neighbor search for the start and the end of the phrase, separately.

Method. Formally, let $\mathcal{C} = \{c_1, \cdots, c_N\}$ be a reference corpus with N tokens. We first map each token c_i into a contextualized, h-dimensional vector $\mathbf{c}_i \in \mathbb{R}^h$ by feeding the text into the encoder and take the vector that corresponds to each token: $\mathbf{c}_1 ... \mathbf{c}_N = \text{Encoder}(c_1 ... c_N)$.

At inference time, NPM is given a query whose t-th token is masked: $q_1 ... q_{t-1}$, [MASK], $q_{t+1} ... q_L$. We replace [MASK] with two special tokens [MASK$_s$] [MASK$_e$] and feed it into the encoder to obtain a list of h-dimensional vectors:

$$\mathbf{q}_1 ... \mathbf{q}_{L+1} = \text{Encoder}(q_1 ... q_{t-1}, \texttt{[MASK}_s\texttt{]}, \texttt{[MASK}_e\texttt{]}, q_{t+1} ... q_L)$$

We then take the vector corresponding to [MASK$_s$] and [MASK$_e$] as $\mathbf{q}^{\text{start}}$ and $\mathbf{q}^{\text{end}}$, respectively.[19]

$$\mathbf{q}^{\text{start}} = \mathbf{q}_t, \mathbf{q}^{\text{end}} = \mathbf{q}_{t+1}$$

We then make a prediction via:

$$\arg\max_{v^* \in \mathcal{V}^*} \sum_{i \leq j} \mathbb{I}[v^* = c_{i:j}] \left(\exp(\text{sim}(\mathbf{q}^{\text{start}}, \mathbf{c}_i)) + \exp(\text{sim}(\mathbf{q}^{\text{end}}, \mathbf{c}_j)) \right)$$

where $\mathcal{V}^*$ is a set of possible n-grams defined by the vocabulary $\mathcal{V}$ and sim is a predefined similarity function that maps a pair of vectors into a scalar value. In practice, iterating over N tokens is infeasible. We thus use an approximation using a fast nearest neighbor search for the start and the end separately. Details are provided in the Appendix of the original publication (Min et al. 2023b).

The choice of similarity function can be flexible. We follow Zhong, Lei, and Chen (2022) in using a scaled inner product $\text{sim}(\mathbf{h}_1, \mathbf{h}_2) = \frac{\mathbf{h}_1 \cdot \mathbf{h}_2}{\sqrt{h}}$, where h is a dimension of the token vectors.

4.4.2 Training. We begin by marking spans (i.e., consecutive tokens) to be predicted, extending span masking from Joshi et al. (2020). Our span masking differs from Joshi et al. (2020) in two ways. First, we mask spans if they co-occur in the other sequences in the batch to guarantee in-batch positives during training (Section 4.4.2). Second, instead of replacing each token in the span with a [MASK], we replace the whole span with two special tokens [MASK$_s$] [MASK$_e$]. This is to obtain the start and the end vectors for each span as we do at inference.

Training Objective. We illustrate an example in Figure 18. The masked span is '*the Seattle Seahawks*', thus the model should retrieve a phrase '*the Seattle Seahawks*' from other sequences in the reference corpus when it is given a query like this at test time. Specifically, we should encourage the [MASK$_s$] vector to be closer to ...**the** Seattle Seahawks... and the [MASK$_e$] vector to be closer to ...the Seattle **Seahawks**..., while being distant from other tokens. We train the model to do so by approximating the full corpus as the other sequences in the batch. Concretely, we train the model to retrieve the start and the end of the span '*the Seattle Seahawks*' from other sequences in the same batch. Note that our masking strategy ensures that every masked span has a co-occurring span in the batch.

19 This allows obtaining two vectors without encoding the query twice, e.g., unlike Lee et al. (2021)

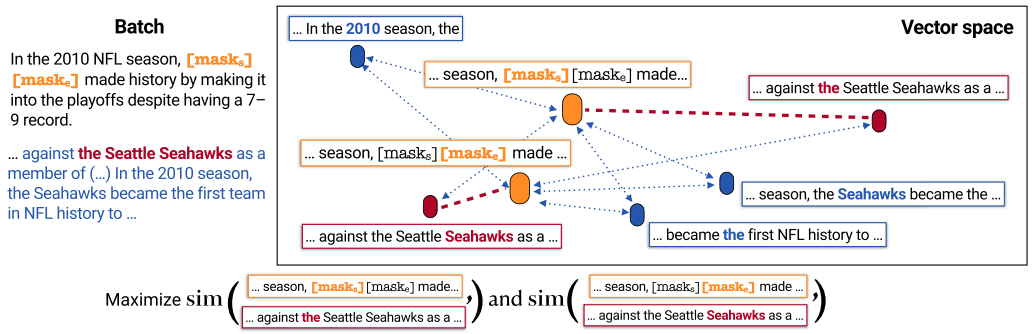

Figure 18
Training of NPM (Section 4.4.2). `[MASK_s] [MASK_e]` indicates the masked *span* whose original phrase is *the Seattle Seahawks*. We maximize the similarity scores between ... `[MASK_s] [MASK_e]` ... and ...**the** Seattle Seahawks..., and between ... `[MASK_s] [MASK_e]` ... and ...the Seattle **Seahawks**....

Formally, consider the i-th sequence in the batch with L tokens, $x^i = x^i_1 ... x^i_L$. We denote $\hat{x}^i = \hat{x}^i_1 ... \hat{x}^i_L$ as a consequence of span masking over x^i. Both x^i and $\hat{x}^i$ are fed into the encoder, and each token is mapped into an h-dimensional vector:[20]

$$\mathbf{x}^i_1 \cdots \mathbf{x}^i_L = \text{Encoder}(x^i_1 \cdots x^i_L)$$
$$\hat{\mathbf{x}}^i_1 \cdots \hat{\mathbf{x}}^i_L = \text{Encoder}(\hat{x}^i_1 \cdots \hat{x}^i_L)$$

Now, consider a masked span in x_i, represented with $\texttt{[MASK}_\text{s}\texttt{]}\ \texttt{[MASK}_\text{e}\texttt{]}$, denoted as $\hat{x}^i_t, \hat{x}^i_{t+1}$. We then denote g^i_t as the original n-gram that was replaced by $\hat{x}^i_t, \hat{x}^i_{t+1}$.

The training objective for this masked span is defined as

$$-\left(\log \frac{\sum_{\mathbf{y} \in \mathcal{Y}^+_\text{s}(g^i_t)} \exp(\text{sim}(\hat{\mathbf{x}}^i_t, \mathbf{y}))}{\sum_{\mathbf{y} \in \mathcal{Y}^+_\text{s}(g^i_t) \cup \mathcal{Y}^-_\text{s}(g^i_t)} \exp(\text{sim}(\hat{\mathbf{x}}^i_t, \mathbf{y}))} + \log \frac{\sum_{\mathbf{y} \in \mathcal{Y}^+_\text{e}(g^i_t)} \exp(\text{sim}(\hat{\mathbf{x}}^i_{t+1}, \mathbf{y}))}{\sum_{\mathbf{y} \in \mathcal{Y}^+_\text{e}(g^i_t) \cup \mathcal{Y}^-_\text{e}(g^i_t)} \exp(\text{sim}(\hat{\mathbf{x}}^i_{t+1}, \mathbf{y}))}\right)$$

Here, $\text{sim}(\cdot, \cdot)$ is the similarity function defined in Section 4.4.1, and $\mathcal{Y}^+_\text{s}(g^i_t)$, $\mathcal{Y}^-_\text{s}(g^i_t)$, $\mathcal{Y}^+_\text{e}(g^i_t)$, and $\mathcal{Y}^-_\text{e}(g^i_t)$ denote the start positives, start negatives, end positives, and end negatives of g^i_t, respectively. The start and end positives correspond to the boundaries of the target spans, while the start and end negatives are all other tokens that do not match these boundaries. This objective follows prior work on phrase-level contrastive learning (Lee et al. 2021; Ram et al. 2021; Deng et al. 2021; Kulkarni et al. 2022).

4.4.3 Training Details. We train on English Wikipedia (August 2019) and the English portion of CC-News (February 2019; Mackenzie et al. 2020), totaling 13B tokens. The data is segmented into sequences of up to 256 tokens.

20 The unmasked sequence and the masked sequence may have different lengths before padding, but we pad them to have the same length.

We use the model architecture and initial weights of RoBERTa large (Liu et al. 2019) (354M parameters), trained for 100K steps on thirty-two 32GB GPUs. Each batch contains 512 sequences (131K tokens). We use Adam (Kingma and Ba 2014) with learning rate 3×10^{-5}, weight decay 0.01, and 4K warm-up steps.

Batch construction is crucial for in-batch approximations. While Zhong, Lei, and Chen (2022) use BM25 to group related sequences, building such an index over billions of tokens is expensive. Instead, we batch sequences from the same document (or group short documents together). This ensures (a) positives (spans with shared strings) also share context, reducing false positives, and (b) negatives are contextually similar, making training more effective. We also aggregate sequences across GPUs to enlarge the effective batch size.

4.4.4 Experiments: Closed-set Tasks. We evaluate in a zero-shot setting on closed-set tasks—tasks where a small candidate set is given for each question.

Evaluation Datasets. We include nine classification datasets that are known for not necessarily requiring factual knowledge: AGNews (Zhang, Zhao, and LeCun 2015), Yahoo (Zhang, Zhao, and LeCun 2015), Subj (Pang and Lee 2004), SST-2 (Socher et al. 2013), MR (Pang and Lee 2004), Rotten Tomatoes (RT), CR (Hu and Liu 2004), Amazon polarity (Amz) (McAuley and Leskovec 2013), and RTE (Dagan, Glickman, and Magnini 2005). The tasks range from topic classification and sentiment analysis to subjectivity classification and textual entailment.

Baselines. We include RoBERTa (Liu et al. 2019) as the encoder-only, T5 (Raffel et al. 2020) as the encoder-decoder, and GPT-2/3 (Radford et al. 2019; Brown et al. 2020a) as the decoder-only model. For the decoder-only models, we additionally apply PMI (Holtzman et al. 2021) for better calibration of the model output. We also compare with Shi et al. (2022) who use *k*NN inference using GPT-2 with PMI. In particular, (1) GPT-2 *k*NN uses *k*NN inference without training, and (2) GPT-2 *k*NN-LM interpolates distributions from GPT-2 and GPT-2 *k*NN.

Setup. We use the templates and verbalizers from Shi et al. (2022) for all models. When available, we use fuzzy verbalizers from Shi et al. (2022). We use a domain-specific reference corpus: A union of the English Wikipedia and CC News for AGN, Yahoo, and RTE, a subjectivity corpus for Subj, and a review corpus for sentiment classification datasets. Their sizes vary from 15M tokens to 126M tokens. More details are provided in the Appendix of the original paper (Min et al. 2023b).

Results (Table 11). Among parametric baselines, RoBERTa achieves the best performance, outperforming larger models including GPT-3. This is likely because the encoder-only model benefits from bidirectionality, as claimed in Patel et al. (2022). The *k*NN-LM approach from Shi et al. (2022), which incorporates the nonparametric component to the parametric model, outperforms all other baselines. Nonetheless, solely relying on retrieval (*k*NN) performs poorly with GPT-2, suggesting that using *k*NN at inference only is limited. Nonetheless, NPM significantly outperforms all baselines, achieving consistently competitive performance over all datasets. This indicates that, even for tasks that do not explicitly require external knowledge, nonparametric models are very competitive.

Table 11
Zero-shot results on closed-set tasks. # *Params* indicates the relative number of model parameters compared to RoBERTa large (354M). RoBERTa, T5, and GPT-2 are their *large* variants unless specified otherwise; GPT-3 is from *Davinci, non-instruct*. Numbers with citations are taken from the corresponding papers; numbers without citations are from our own experiments. We run the code by Shi et al. (2022) and Holtzman et al. (2021) for datasets not included in the original paper. As a reference, we provide results of fine-tuning on the full training dataset in the last row. † indicates a reference corpus is used. NPM outperforms larger parameters models.

Model	# Params	AGN	Yahoo	Subj	SST-2	MR	RT	CR	Amz	RTE	Avg
Baselines (encoder-only)											
RoBERTa (Gao, Fisch, and Chen 2021)	1.0×	–	–	51.4	83.6	80.8	–	79.5	–	51.3	–
RoBERTa	1.0×	71.3	41.4	67.6	84.5	81.7	81.1	80.4	83.5	57.4	72.1
Baselines (encoder-decoder)											
T5	2.2×	72.0	51.3	54.9	57.5	57.7	59.1	56.4	59.3	55.6	58.2
T5 3B	8.5×	**80.5**	53.6	54.8	59.6	58.6	57.3	53.7	57.0	58.5	59.3
Baselines (decoder-only)											
GPT-2 (Shi et al. 2022)	2.2×	67.4	49.7	60.8	55.3	54.6	53.0	66.2	57.6	53.1	57.5
+ PMI (Shi et al. 2022)	2.2×	65.1	48.8	62.5	76.5	74.6	74.1	82.8	76.2	54.2	68.3
GPT-2 kNN† (Shi et al. 2022)	2.2×	29.8	37.0	50.0	47.1	49.9	49.1	69.3	57.4	54.1	49.3
GPT-2 kNN-LM† (Shi et al. 2022)	2.2×	78.8	51.0	62.5	84.2	78.2	80.6	84.3	85.7	55.6	73.4
GPT-3 (Holtzman et al. 2021)	500×	75.4	53.1	66.4	63.6	57.4	57.0	53.8	59.4	56.0	60.2
+ PMI (Holtzman et al. 2021)	500×	74.7	54.7	64.0	71.4	76.3	75.5	70.0	75.0	**64.3**	69.5
Ours (encoder-only, nonparametric)											
NPM†	1.0×	74.5	53.9	**75.5**	**87.2**	**83.7**	**86.0**	81.2	83.4	61.7	**76.4**
Full fine-tuning (reference)											
RoBERTa (Gao, Fisch, and Chen 2021)	1.0×	–	–	97.0	95.0	90.8	–	89.4	–	80.9	–

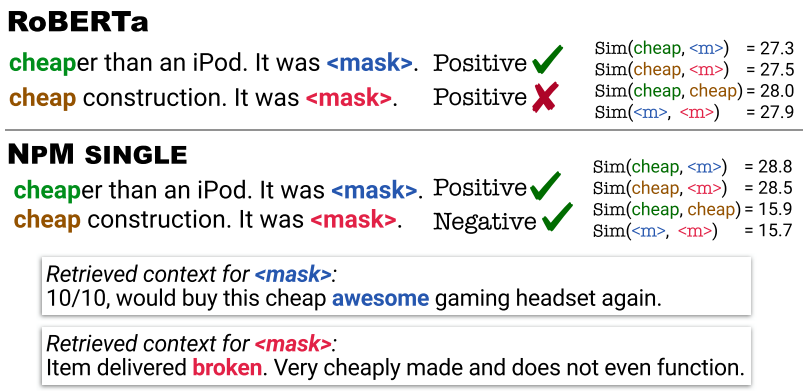

Figure 19
Predictions from RoBERTa (baseline) and NPM. The bottom indicates the context NPM retrieves to fill in `[MASK]`. Note that the fuzzy verbalizer maps *broken* to `Negative` and *awesome* to `Positive`.

Qualitative Analysis. Figure 19 depicts predictions from RoBERTa and NPM on a sentiment analysis task. The first example uses *cheap* to indicate *inexpensive*, and the second example uses *cheap* to indicate *of very poor quality*. RoBERTa predicts `Positive` to both, while NPM makes correct predictions by retrieving the context that uses *cheap* in the same context as the input.

We also find that representations from NPM lead to better word sense disambiguation. For instance, RoBERTa assigns a high similarity score between *cheap* (*inexpensive*) and *cheap* (*of very poor quality*). On the other hand, NPM successfully assigns a low similarity score between *cheap* and *cheap*, even though their surface forms are the same.

4.4.5 Experiments: Open-set Tasks. We now evaluate on open-set tasks whose answer can be any arbitrary-length string.

Evaluation Datasets. We evaluate on seven datasets: T-REx and Google-RE from LAMA (Petroni et al. 2019), KAMEL (Kalo and Fichtel 2022), Natural Questions (NQ; Kwiatkowski et al. 2019), TriviaQA (TQA; Joshi et al. 2017), TempLAMA$^{22}_{19}$, and an entity translation task. In particular, TempLAMA requires probing knowledge with temporal updates, motivated by Dhingra et al. (2022) and Jang et al. (2022). The entity translation task involves a translation of an entity from English to other, non-Latin languages, requiring the model to predict extremely rare (if not unseen) characters.

Baselines. We compare with T5 (Raffel et al. 2020) as the encoder-decoder, and GPT-3 (Brown et al. 2020a) and OPT (Zhang et al. 2022) as the decoder-only models. The encoder-only models are not applicable for open-set tasks since the number of tokens to predict is unknown. We also compare against retrieval augmentation methods by using BM25 (Robertson, Zaragoza et al. 2009) as our retrieval model, feeding the LMs with up to five passages from BM25.

Setup. For all datasets, we report Exact Match (EM). The LAMA test data is biased toward frequent entities because they are filtered to only include answers that are single tokens based on BERT (Devlin et al. 2019). Because we do not want our evaluation to be biased toward overly frequent entities, we report a micro-averaged accuracy over the data whose answers are 1, 2, 3, and 4+ grams, respectively. Other datasets do not have such filtering, therefore we report average EM.

As a reference corpus, we use the English Wikipedia from 1 August 2019, consisting of 810M tokens. For TempLAMA$^{22}_{19}$, we use the English Wikipedia from 1 August 2022, consisting of 858M tokens. For NPM, we find combining with sparse retrieval significantly helps: We reduce the search space to the top 3 passages based on BM25 and perform dense search as done in Kassner and Schütze (2020).

Main Results. Figure 20 shows results on five knowledge tasks. First, performance of parametric models largely depends on the number of parameters, as it has been claimed in much of prior work (Brown et al. 2020a; Kandpal et al. 2022). Retrieval-augmented models with BM25 significantly improve performance.

NPM outperforms or is on par with significantly larger baselines across all datasets. It substantially outperforms all models on two LAMA datasets, including 500x larger GPT-3 either with or without BM25. On KML, TQA, and NQ, NPM consistently outperforms 37$\times$ larger models with or without BM25. This is impressive given that NPM is not trained on data with questions.

It is also worth noting that sparse retrieval is critical in NPM, e.g., without sparse retrieval, performance on LAMA-TREx drops from 34.5 to 16.1. We think this is because (1) sparse retrieval and dense retrieval capture complementary features, and (2) the removal of approximation in search improves search quality. We think future work can explore completely removing sparse retrieval, as has been done in Lee et al. (2021).

Impact of the Reference Corpus Size. Figure 21 reports the impact of the size of the reference corpus, from 41M tokens (5%) to 810M tokens (100%). Performance of NPM is highly correlated with the size of the reference corpus, strongly suggesting that using a larger reference corpus is important.

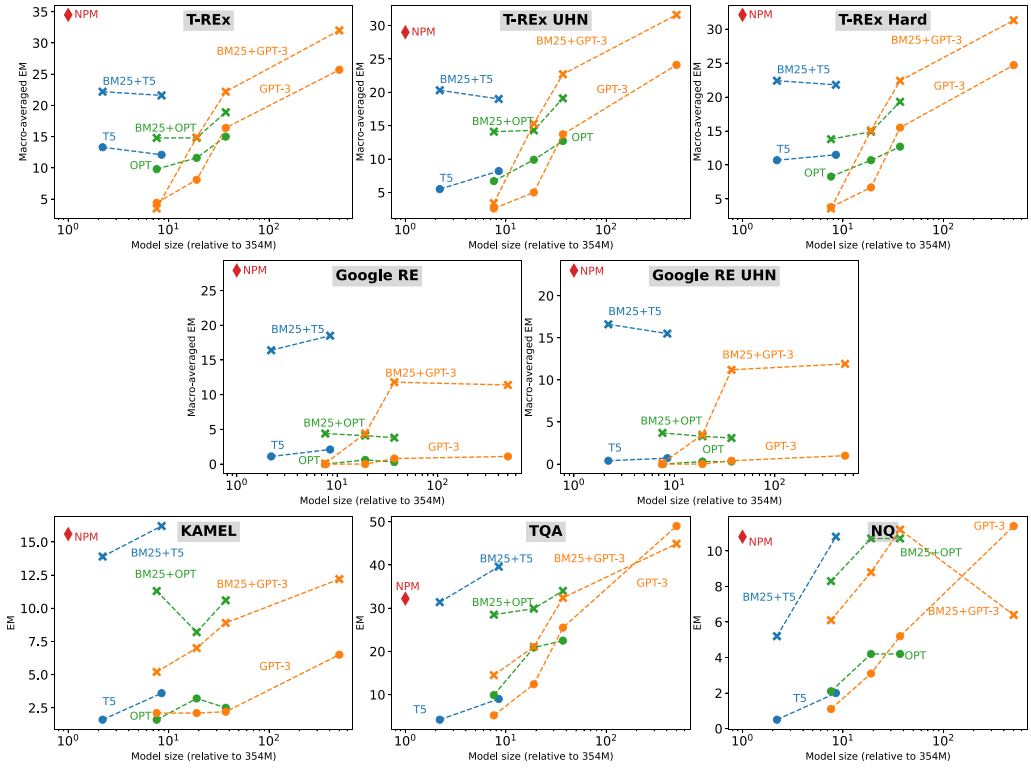

Figure 20
Zero-shot results on knowledge tasks. The *x*-axis indicates the relative number of model parameters in log scale compared to RoBERTa large (354M). NPM outperforms significantly larger parameters models, either with or without BM25. See the original publication (Min et al. 2023b) for the raw numbers.

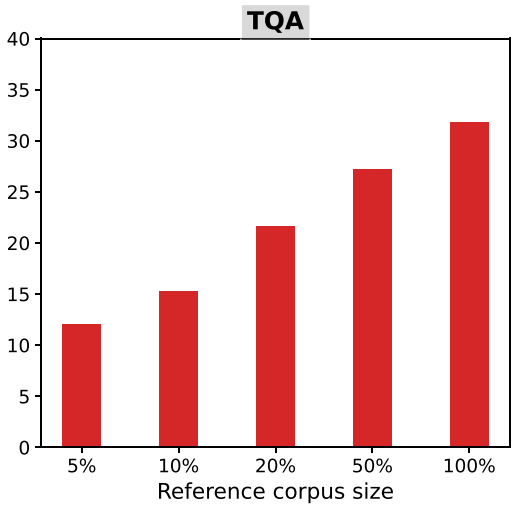

Figure 21
Ablation on the size of the reference corpus, from 41M tokens (5%) to 810M tokens (100%), evaluated on TriviaQA. There is a strong correlation between the corpus size and performance.

Table 12
Results on TempLAMA$^{22}_{19}$ on an unchanged set, a changed set, and a macro-average over two, respectively. xx→xx indicates performance when using the outdated and the updated Wikipedia, respectively.

Model	#Params	Unchanged	Changed	AVG
Baselines				
T5	2.2×	1.9	0.4	1.1
T5 3B	8.5×	1.8	0.4	1.1
OPT 6.7B	19×	2.5	1.0	1.7
OPT 13B	37×	4.9	2.1	3.5
BM25 + T5	2.2×	13.7→14.9	3.0→**20.1**	17.5
BM25 + T5 3B	8.5×	11.9→12.0	2.2→17.8	14.9
BM25 + OPT 6.7B	19×	10.2→8.2	1.7→11.3	9.7
BM25 + OPT 13B	37×	14.8→14.4	2.8→16.6	15.5
Ours				
NPM	1.0×	18.9→**19.5**	2.9→17.5	**18.5**

Results on Temporal Knowledge Tasks. Table 12 reports results on TempLAMA. NPM retains its performance on the unchanged set (18.9→19.5) and successfully updates its answers on the changed set (2.9→17.5). Its performance is significantly better than the performance of parametric models with up to 13B parameters, and is on par with a larger model with the retrieval augmentation approach, which also successfully updates its answer by leveraging the updated corpus. This is in agreement with prior work that shows the model with a nonparametric component adapts to temporal updates by replacing the reference corpus at test time (Izacard et al. 2022b). Nonetheless, the retrieval augmentation approach is still significantly worse than NPM when the target entities are rare, which we show next.

Performance on Rare Entities. We break down the instances on LAMA and TempLAMA based on the number of BPE splits of the target entity, e.g., *Thessaloniki* is one word that is split into 4 BPE tokens, thus the number of splits is 3. Since BPE splits a word if it is rare, the number of BPE splits indicates the rarity of the entity. We compare NPM with GPT-3 and BM25+GPT-3 on LAMA, and BM25+T5 (770M and 3B) on TempLAMA, the two most competitive baselines on each dataset.

Figure 22 reports results. On LAMA, NPM outperforms GPT-3 fairly consistently, with larger gains as the number of BPE splits increases. On TempLAMA, although

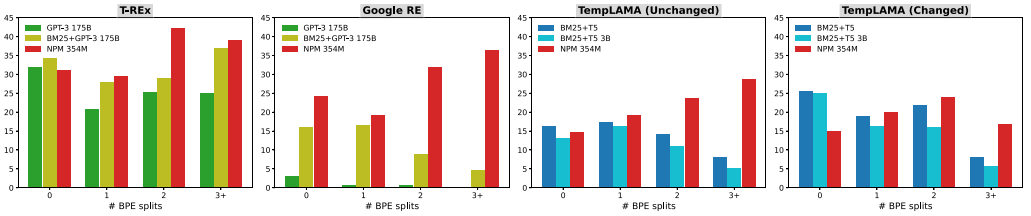

Figure 22
Performance on LAMA and TempLAMA tasks, broken down based on the number of BPE splits of the target entity, which is an indication of rarity of the entities (L:Frequent→R:Rare). Gains from NPMover GPT-3 or T5 are larger when the target entities are rare.

Table 13
Results on the entity translation task. *#L* indicates the number of languages multilingual models are trained on. **Bold** and **Bold** indicate the best among monolingual models and the best including multilingual models, respectively.

Model	#Params	#L	w/o BM25	w/ BM25
Baselines, English-only				
T5	2.2×		0.2	1.9
T5 3B	8.5×		0.5	4.4
OPT 6.7B	19×		0.4	22.3
OPT 13B	37×		1.0	24.6
Ours, English-only				
NPM	1.0×		**52.4**	
References, Multilingual				
mT5	3.4×	101	1.3	19.0
mT5 XL	11×	101	4.1	**56.6**
BLOOM 3B	8.5×	46	0.0	17.4
BLOOM 7.1B	20×	46	0.1	26.0

BM25+T5 is competitive on frequent entities with zero BPE split, it consistently lags behind NPM with ≥ 1 BPE splits. This suggests that NPM is particularly good at addressing rare entities, compared to not only parametric models without retrieval but also the retrieval augmented models.

Results in Entity Translation. Results on the entity translation task are shown in Table 13. See the Appendix of the original publication (Min et al. 2023b) for per-language results. T5 and OPT struggle to perform the task, both with and without BM25 retrieval, since non-English text was extremely rare or unseen during their training. In contrast, NPM performs well across all languages.

To better calibrate performance of NPM, we provide reference performance of models that are purposely trained on the multilingual data: mT5 (Xue et al. 2021) and BLOOM (Scao et al. 2022). NPM outperforms 3.4× larger mT5 and 20× larger BLOOM, and approaches 11× larger mT5, even though it is trained on English. We think strong cross-lingual transferability of NPM is likely because it can retrieve a phrase based on its surrounding context, even if it has not seen the exact word during training.

4.4.6 Limitations & Future Work Ideas. NPM has several limitations.

Scaling with Corpus Size. Our experiments use up to 1B tokens, smaller than the training data of large LMs (Brown et al. 2020a; Rae et al. 2021). Future work can scale further with efficient retrieval libraries such as Distributed FAISS (Johnson, Douze, and Jégou 2019) or ScaNN (Guo et al. 2020).

Memory Usage. While NPM reduces GPU cost, storing corpus embeddings is RAM- and disk-intensive (e.g., Wikipedia requires 70GB RAM and 1.4TB disk). Future work can explore more compact indexing methods (Jegou, Douze, and Schmid 2010; Norouzi, Punjani, and Fleet 2012; Ge et al. 2014; Izacard et al. 2020; Yamada, Asai, and Hajishirzi 2021).

Exploration of Larger Vocabulary. Large vocabulary is known to lead performance gains (Conneau et al. 2020) but is bounded in memory costs. Previous work explored more efficient softmax approximations (Morin and Bengio 2005; Chen, Grangier, and Auli 2016; Grave et al. 2017). Our nonparametric training offers an alternative by removing the softmax over the vocabulary. With the RoBERTa architecture, increasing the vocab size by 2× makes the baseline training 50% more memory expensive, but does not increase the memory in training NPM. This article does not include more systematic evaluation on this, but future work can explore training NPM with a significantly larger vocabulary to further boost performance.

Extension for Generation. NPM is encoder-only and evaluated on prediction tasks. Future work can explore autoregressive generation as done in Patel et al. (2022) or use NPM for editing (Schick et al. 2022; Gao et al. 2022).

Extension to Few-shot Learning and Fine-tuning. We focus on zero-shot evaluation. Extending NPM to few-shot or fine-tuning setups is straightforward and likely cheaper than with very large models (e.g., T5, OPT, GPT-3).

Cross-lingual Transfer. Our work explored cross-lingual transfer in a limited setup where the model is trained on monolingual data. Future work can train multilingual NPM and explore more comprehensive cross-lingual evaluation. Nonparametric training may alleviate the burden of collecting large-scale multilingual corpora since it makes the model less sensitive to the language coverage in the training data, and may lead to significantly better cross-lingual transfer.

Limitation in Speed. Retrieval makes inference considerably slower than the counterpart without retrieval. We think that (1) retrieval can significantly be faster with better engineering (we use the default hyperparameters of the FAISS index with no tuning) or better index, and (2) the speed of NPM is still on par with the speed of significantly larger parametric models that NPM outperforms. We leave improving inference speed to future work.

4.5 Discussion of Subsequent Work

4.5.1 Dense Retrieval. Since the introduction of DPR, researchers have significantly improved dense retrieval models through multiple axes, including the model architecture, the training objective, and the method to obtain better hard negatives (Xiong et al. 2021a, b; Ding et al. 2021; Gao and Callan 2021; Zhan et al. 2021; Gao and Callan 2022). Researchers also scaled up dense retrieval, either in terms of training data (made possible with the introduction of a self-supervised contrastive learning objective [Izacard et al. 2022a; Ram et al. 2022]) or encoder size (Ni et al. 2022; Neelakantan et al. 2022). Researchers also combined dense retrieval with sparse retrieval (Mao et al. 2020), added late-interaction operations on top of the dual encoder (Khattab and Zaharia 2020; Santhanam et al. 2021), and made dense retrieval cross-lingual (Asai et al. 2021). There has also been work that tunes retrieval using signals from an LM after augmentation, including our own work (Shi et al. 2024b), which showed its effectiveness over state-of-the-art LLMs like GPT-3, Instruct GPT-3, and CoDeX.

4.5.2 Augmentation. Subsequent work (Izacard and Grave 2020; Lewis et al. 2020b) has shown that DPR can be combined with generation models such as BART (Lewis

et al. 2020a) and T5 (Raffel et al. 2020). In particular, Lewis et al. (2020b) showed the effectiveness of this approach for a range of knowledge-intensive tasks beyond open-domain QA, including fact verification, dialogue, and slot filling. Izacard et al. (2022b) proposed a pre-training objective that jointly trains a retrieval model and an LM, further improving performance.

While this line of work assumes the model can be fine-tuned on a labeled dataset, more recent work explores the use of retrieval-augmented LMs without fine-tuning. Ram et al. (2023) and Shi et al. (2024b), concurrently, showed that with prompting of a frozen, large LM with block-box access, retrieval augmentation can yield improvements over a range of tasks.

Other papers report that out-of-box language models do not necessarily know how to use retrieval results; they often ignore retrieval results when they need to, or become easily distracted by incorrect retrieval results. Recent work proposed various ways to manage these issues, e.g., through post hoc verification (Asai et al. 2024) or by filtering retrieved results before the augmentation stage (Wang et al. 2023b).

4.5.3 Pre-training. More recently, we proposed a new pre-training objective, called *in-context pre-training* (Shi et al. 2024a), that closely resembles retrieval augmentation. The key idea is to train an LM on a sequence of relevant documents obtained from an off-the-shelf retrieval model, thereby explicitly encouraging the LM to learn to condition on a set of retrieved documents. Unlike prior work in joint training (Lee, Chang, and Toutanova 2019; Guu et al. 2020; Izacard et al. 2022b), in-context pre-training is very straightforward since it does not require asynchronously re-indexing the datastore, and can be achieved simply by changing the document ordering and directly applying existing pre-training pipelines. We trained LMs with 1B and 7B parameters from scratch and showed significant improvements in open-domain question answering with retrieval augmentation, while also achieving benefits such as in-context learning and long-context language modeling.

4.6 Summary and Future Work

In this section, we discuss nonparametric LMs, a new class of LMs with both learned parameters and a datastore. These models are significantly more parameter efficient and better at capturing long-tail data distributions because they can retrieve relevant information from a datastore at test time without relying solely on memorization of the training data through enormous amounts of parameters. Moreover, since the datastore can be updated at any time with no further training, these models can easily stay up-to-date (e.g., by keeping the datastore up-to-date) and support opt-out (e.g., by removing datapoints from the datastore). We highlighted our key contributions to nonparametric LMs: including one of the first widely used dense retrieval models (DPR; §4.2), one of the first studies that compare parametric and nonparametric models in various conditions (EfficientQA; §4.3), and one of the first models trained with a nonparametric softmax (NPM; §4.4), exploring an alternative method for incorporating the datastore beyond conventional retrieval augmentation.

Still many open questions remain, which we highlight below.

4.6.1 Model Architecture for Nonparametric LMs. Nonparametric LMs have seen little change beyond retrieval augmentation, which is limited by the number of passages that can be incorporated and the overhead of attending to them. The nonparametric softmax model, while promising in avoiding this issue, introduces significant increases

in runtime and memory costs due to a substantial increase in the number of vectors in a datastore.

We believe it is necessary to explore alternative architectures that can incorporate a larger retrieval of results more frequently and more efficiently. One such attempt is the RETRO model (Borgeaud et al. 2022), which modifies the attention layers of Transformers to integrate retrieval results in parallel with the input. This architecture allows for the efficient handling of a vast number of documents with greater frequency, incorporating many documents in parallel. However, follow-up on this model has been limited because it is not open-sourced and requires pre-training from scratch. Table 14 includes detailed comparisons between these approaches.

Table 14
Comparisons of various architectures for nonparametric LMs. Further discussion can be found in the slides for nonparametric model architectures from the ACL 2023 Tutorial (Asai et al. 2023).

Retrieval augmentation	Nonparametric softmax	RETRO
	Pros	
• Easy to use off-the-shelf retrieval models and LMs with a black-box access.	• A single model, without two separate models. • Scales better than retrieval augmentation.[a] • Generalizes better to OOD datastore.[a] • Can efficiently encode a large # of passages, more frequently.	• Easy to use an off-the-shelf retrieval model with a black-box access.[b] • Can efficiently encode a large # of passages, more frequently.
	Cons	
• Two models, each trained in isolation.[c] • Error propagation issues. • The # of passages is strictly limited by the sequence length limit of LMs.[d] • Frequent retrieval significantly increases runtime. • Has many unnecessary operations, e.g., attention across multiple retrieved passages.	• Needs a vector per token (or phrase) instead of vector per passage, making the model more expensive in runtime and memory.[e] • Computation used for incorporation is the least.[f]	• The LM has to be trained from scratch. • Error propagation issues.

[a] : Discussed in Section 5.5.
[b] : However, using a different retrieval model may require re-training of a language model.
[c] : Joint training (Guu et al. 2020; Izacard et al. 2022b) or sequential training (Shi et al. 2024a) can potentially address this issue.
[d] : A massive long context language model can potentially address this issue.
[e] : For instance, the number of vectors in a datastore increases by 100x if each passage has 100 tokens. This is equally applicable to retrieval augmentation or RETRO with multi-vector retrieval (Khattab and Zaharia 2020; Santhanam et al. 2021; Lee et al. 2024).
[f] : This is because the incorporation is through a single inner product operation at the final prediction stage, unlike other models that incorporate retrieval results in every attention layer.

Another direction is long-context LMs, which use sparse or alternative attention mechanisms to scale context windows. If long-context LMs could scale to process inputs as large as a trillion tokens, they might effectively serve as nonparametric LMs by processing a datastore of that magnitude. Such models could approximate performing nearest neighbor searches within the attention layers, which essentially equates to the use of retrieval operations for each attention operation.

4.6.2 Scaling Nonparametric LMs. Scaling has driven the success of large LMs, typically through parameters and training data. A central question here is whether nonparametric LMs can provide an alternative pathway for scaling—the datastore size. To answer this question, we should ideally scale the datastores to accommodate Internet-scale data consisting of trillions of words—the scale of modern LMs' training data. Currently, they accommodate relatively small-scale data (e.g., a few billions of words) and are thus limited to a specific domain (e.g., Wikipedia) in many cases.

A key obstacle in scaling the datastore lies in scaling the nearest neighbor search algorithm to handle trillions of vectors with manageable inference speed and memory requirement. Addressing this challenge requires joint research in algorithms, systems, and databases.

4.6.3 Runtime Efficiency of Nonparametric LMs. Nonparametric LMs are slower than comparable parametric ones due to nearest neighbor search. Improving efficiency will require (1) better distributed search, (2) approximate nearest neighbor methods with improved speed–memory–accuracy trade-offs, and (3) caching mechanisms tailored for next-token prediction.

5. Responsible Language Models

Large language models (LMs) are under widespread legal scrutiny, in large part because they are trained on copyrighted content, which may infringe on the rights of data producers (Metz 2022; Vincent 2023; Silverman et al. v. OpenAI 2023; Brittain 2023). At the heart of this discussion is the inherent trade-off between legal risk and model performance. Training only on data sources such as public domain, non-copyrightable or otherwise permissively licensed data significantly degrades performance (as we show in §5.5). This limitation arises from the scarcity of permissive data and its narrow specificity to sources such as copyright-expired books, government documents, and permissively licensed code, which are largely different from common LM corpora that cover more diverse domains (Raffel et al. 2020; Gao et al. 2020; Together 2023).

In this section, we demonstrate it is possible to improve the risk-performance trade-off by segregating training data into two distinct parts of the model: parametric and nonparametric (Figure 23). We learn LM parameters on *low-risk* data (i.e., data under the most permissive licenses), and then use *high-risk* data (i.e., data under copyright, restrictive licenses, or unknown licenses) as a datastore used only at inference time (*nonparametric*). With nonparametric datastores, we can *retrieve* high-risk data to improve model predictions without training on it. The datastore can be easily updated at any time, and allows creators to remove their data from the model entirely, at the level of individual examples. This approach also attributes model predictions at the sentence-level, enabling credit assignment to data owners. These new capabilities enable better alignment of the model with various data-use regulations, e.g., the *fair use* doctrine in the United States (Henderson et al. 2023) and the GDPR in the European Union (Zhang et al. 2023a), as detailed in §5.1. This is in contrast to parametric models, where

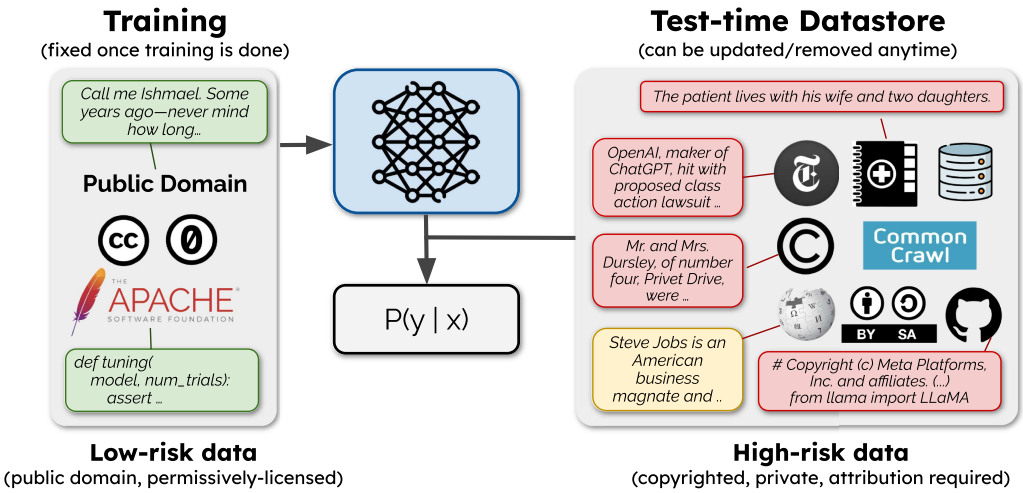

Figure 23
An overview of SILO. We train a *parametric* language model on low-risk datasets that contain public domain text (e.g., copyright-expired books) and permissively licensed code. At inference time, we use a *nonparametric datastore* that can include high-risk data, including medical text with personally identifiable information, copyrighted news, copyrighted books, data requiring attribution, and code under non-permissive licenses. The datastore can be modified at any time, e.g., to respond to opt-out requests.

removing data after training is infeasible (Bourtoule et al. 2020; Carlini et al. 2021) and data attribution at scale is difficult (Feldman and Zhang 2020; Zhang et al. 2021; Han et al. 2023).

We introduce SILO, a new nonparametric language model that follows our proposal (§5.3). The parametric component in SILO is trained on a new pre-training corpus, the OPEN LICENSE CORPUS (**OLC**, §5.2), which we curate to include data under three types of permissive licenses, from public domain to Creative Commons. OLC is diverse but has a domain distribution that is very different from typical pre-training corpora; it is dominated by code and government text. This leads to a new challenge of generalizing a model trained on highly specific domains, which we call *extreme domain generalization*. We train three 1.3B-parameter LMs on varying subsets of OLC, and then construct a test-time datastore that can include high-risk data, employing a retrieval method to make use of the datastore's contents during inference. We compare two widely studied retrieval methods: a nearest-neighbors approach (kNN-LM) that uses a nonparametric next-token prediction function (Khandelwal et al. 2020) and a retrieval-in-context approach (RIC-LM) that retrieves text blocks and feeds them to the parametric LM in context (Shi et al. 2024b; Ram et al. 2023).

We evaluate SILO in language modeling perplexity on 14 different domains, covering both in-domain and out-of-domain data with respect to OLC (§5.5). These domains highlight specific legal risks, e.g., copyrighted materials such as books, news, and user reviews, or private data such as emails and clinical notes. We compare SILO to Pythia (Biderman et al. 2023), a parametric LM with a similar parameter count but trained mostly on high-risk data (Gao et al. 2020).[21] We first show that parametric-only SILO

21 The Pile contains a large amount of copyrighted or restrictively licensed data, e.g., most content in its Books3, ArXiv, GitHub, OpenWebText, YoutubeSubtitles, and Common Crawl subsets.

is competitive on domains covered by OLC but falls short out-of-domain, confirming the challenge of extreme domain generalization. However, adding an inference-time datastore to SILO effectively addresses this challenge. Comparing the two methods of retrieving over this datastore, we find that while both *k*NN-LM and RIC-LM significantly improve out-of-domain performance, the former generalizes better than the latter, allowing SILO to reduce the gap with the Pythia baseline by 90% on average across all domains. Further analysis attributes these improvements to two factors: (1) *k*NN-LM strongly benefits from scaling the datastore and (2) the nonparametric next-token prediction in *k*NN-LM is robust to domain shift. Altogether, our study suggests that in the few domains where SILO has not yet matched Pythia performance levels, the remaining gaps can likely be closed by scaling the datastore size and further enhancing the nonparametric model.

5.1 Background

5.1.1 Training Datasets for LMs. State-of-the-art LMs are trained on vast text corpora that consist of billions or even trillions of tokens (Brown et al. 2020b; Raffel et al. 2020; Gao et al. 2020; Together 2023). These training sets are built by combining (1) manually selected sources such as Wikipedia, book collections, and GitHub and (2) web pages collected through web-crawling services such as Common Crawl. Most LM training efforts ignore copyright and intellectual property regulations that apply to these texts. For example, sources such as GitHub repositories and book collections typically contain text with highly restrictive licenses (Bandy and Vincent 2021).

5.1.2 Legality of Language Models. The legality of training LMs this way has become a subject of intense debate, with numerous lawsuits being filed in the United States, United Kingdom, and beyond (Gershgorn 2021; Metz 2022; Vincent 2023; De Vynck 2023; Kadrey et al. v. Meta Platforms, Inc. 2023; Silverman et al. v. OpenAI 2023; Tremblay et al. v. OpenAI 2023). While the outcome of the lawsuits is uncertain, it is likely that such legal issues will continue to be a major factor in future LMs, especially since each country has its own data regulations. For example,

- In the United States, the *fair use doctrine* allows the public to use copyrighted data in certain cases, even without a license (Henderson et al. 2023). Deciding whether or not a model's use of copyrighted work constitutes fair use involves multiple dimensions, including whether the trained model is intended for commercial use, whether or not the work is factual or creative, the amount of the copyright content used, and the value of the copyrighted work. There are claims that using parametric language models for *generative* use-cases does *not* constitute fair use, because the technology may output the copyrighted text verbatim (Lemley and Casey 2020), which also has been shown empirically (Carlini et al. 2021, 2023; Kandpal, Wallace, and Raffel 2022; Chang et al. 2023). This is in contrast to *transformative* technologies, such as classifiers, which may use the copyrighted text but do not directly generate content, which the fair use doctrine favors. We refer readers to Henderson et al. (2023) for a more comprehensive discussion.
- The *General Data Protection Regulation (GDPR)* is a comprehensive data protection and privacy law in the European Union (EU). It grants individuals more control over their data by regulating organizations and

businesses. The obligations include (1) obtaining consent from users before processing their data, (2) providing transparency about data processing, (3) ensuring data security, and (4) allowing individuals to access, correct, and erase their data. GDPR has global impact, as many international companies handle EU citizens' data. While it is under debate how GDPR is applied to training language models, compliance with GDPR is expensive (e.g., requiring retraining for every data correction or erasure). See Zhang et al. (2023a) for more discussion on challenges for compliance with the GDPR's *Right to Erasure* (and the *Right to be Forgotten* in general).

The goal of our work is not to weigh in on legal discussions; instead, we study the feasibility of developing technologies that explicitly manage legal risk. In particular, our technique places all copyrighted data in a nonparametric datastore. While the data is still used in service of a generative model, restricting copyrighted data in a datastore and providing instance-level attribution and data opt-out can increase the likelihood of a successful fair use defense (Henderson et al. 2022).[22] Moreover, GDPR's requirement regarding user data access, correction, and erasure aligns well with the capabilities of the datastore. Attribution and opt-out are fundamental features of our model (§5.3.2). This is in contrast to other techniques like post-hoc training data attribution (Koh and Liang 2017; Han et al. 2023) and the removal of the effect of particular training examples from parameters (Cao and Yang 2015; Jang et al. 2023b), which lack inherent guarantees and are hard to scale.

5.1.3 Prior Work in Copyright Risk Mitigation. The most straightforward approach to avoid copyright infringement is to filter training data to only include permissive licenses. This has been done in prior work, primarily for code (e.g., Kocetkov et al. 2023; Fried et al. 2023; Together 2023) and scientific text (e.g., Soldaini and Lo 2023). Extending a similar approach to a wider range of domains remains unclear, because permissive data is extremely scarce in most domains. For the same reason, Henderson et al. (2023) has suggested that restricting the training data to public domain or otherwise permissively licensed data may be impractical. In this work, we show that there is in fact a large number of tokens from data sources with permissive licenses, but the key challenge instead arises from the highly skewed domain distribution. See §5.6 for other copyright mitigation strategies that are more technical in nature.

5.2 Open License Corpus: A Permissively-Licensed Pre-training Corpus

Our study focuses on addressing the legal risk of copyright violation in language models by separating *low-risk* data sources (i.e., those in the public domain or under permissive licenses) from *high-risk* ones (i.e., those with unknown licenses or under copyright). We introduce the Open License Corpus (OLC), a large collection of permissive textual datasets across multiple domains, comprising 228B tokens. This curated data is then used to train model parameters and highlights the challenge of extreme domain generalization due to its skewed domain distribution.

22 Our model on its own does not entirely remove legal risk. Rather, it provides functionalities that, when used appropriately, lower legal risk and strengthen a fair use defense. See §5.6 for a discussion.

5.2.1 A Disclaimer. Our license taxonomy and categorization are imperfect, and OLC should not be viewed as universally safe to use. Licenses may vary by time and jurisdiction (e.g., some Gutenberg books are public domain in the U.S. but copyrighted elsewhere). Additional restrictions, such as the DMCA,[23] may also apply. Finally, we do not filter personally identifiable information, so privacy risks may remain. We encourage users of OLC to consult legal professionals on the suitability of each data source for their application.

5.2.2 Taxonomy of Data Licenses. As discussed in §5.1, determining what data one is permitted to use from a copyright perspective is an ongoing topic of debate, and is context- and country-dependent (Henderson et al. 2023). In this section, we take a conservative approach where we train models using only text with the most permissible licenses, thus enabling widespread downstream use. We focus on four broad categories: (1) **Public domain** (PD), which has no restrictions; (2) **Permissively licensed software** (SW), which includes MIT, Apache, and BSD;[24] (3) **Attribution licenses** (BY), which are free to use as long as *credit is given to the creator*, e.g., Creative Commons Attribution (CC-BY); and (4) **All other data**, which we assumed to be non-permissive, e.g., any text that is explicitly protected by copyright or licenses that are non-commercial (e.g., CC-NC), any software without clear MIT, BSD, or Apache licenses, and any generic web-crawled data where the license or copyright information is unclear.

5.2.3 Building the OPEN LICENSE CORPUS. Based on this taxonomy of licenses, we collect OLC, a 228B token corpus of PD, SW, and BY data. The text generally falls into eight different domains:

- PD BY **Legal:** We curate legal text from the Pile of Law (Henderson et al. 2022), an amalgation of 31 different sources of text related to civil court cases, patents, and other legal and governmental works, either licensed as public domain or CC-BY. We also gather public domain text from the Case Law Access Project (Caselaw Access Project 2018), which covers over 6.5 million decisions published by state and federal courts throughout U.S. history.
- SW **Code:** We use the GitHub subset of the RedPajama dataset (Together 2023), which contains code from GitHub repositories with three permissive software licenses: MIT, Apache, and BSD.
- SW BY **Conversation:** We source conversational text under permissive software licenses from the HackerNews (MIT license) and the Ubuntu IRC (Apache license) subsets of the Pile (Gao et al. 2020). We also use the Stackexchange subset of the RedPajama dataset (Together 2023) and a Stackoverflow corpus from Kaggle,[25] both under the CC-BY-SA license.

23 https://www.copyright.gov/dmca/.

24 Unlike public domain text, these licenses typically carry some basic stipulations such as requiring one to include a copy of the original license (although, it is debatable whether it is still required when the associated text is used as data or treated as a software). The code is otherwise free to use, and code is generally well protected by fair use clauses (Lemley and Casey 2020).

25 https://www.kaggle.com/datasets/stackoverflow/stackoverflow.

- SW **Math:** We source mathematical text from the Deepmind Mathematics (Saxton et al. 2019) and the AMPS (Hendrycks et al. 2021) datasets, both of which are under the Apache license.
- PD BY **Science:** We source scientific text from ArXiv abstracts that are in the public domain (ArXiv 2023). We also collect full-text articles from the Semantic Scholar Research Corpus (Lo et al. 2020, S2ORC), either licensed as public domain or CC-BY.
- PD **Books:** We source books from the Gutenberg corpus (Project Gutenberg), which are copyright-expired books that are in the public domain.
- PD BY **News:** We collect public domain news text from the English subset of the MOT corpus (Palen-Michel, Kim, and Lignos 2022). We also collect text from Wikinews, which is under CC BY-SA.
- BY **Encyclopedic:** Finally, we include a large set of Wikipedia from the subset included in RedPajama (Together 2023). We follow RedPajama in using Wikipedia snapshots from 20 languages even though the model primarily focuses on English.

Following Kandpal, Wallace, and Raffel (2022) and Lee et al. (2022), we deduplicate text using Groeneveld (2023), a document-level filter that considers n-gram overlap. We first deduplicate within each domain to remove redundant documents from similar sources (e.g., Case Law and the Pile of Law), and then perform deduplication against the validation and test datasets of the Pile to avoid test leakage.

In Table 15, we compare the distribution of domains in OLC to that of the Pile (Gao et al. 2020), a popular pre-training corpus that includes data under copyright restrictions (e.g., Books, web crawl).[26] These statistics convey a number of research challenges when working with OLC. First, while we tried our best to collect public domain or permissively licensed data, the size of OLC is still 31% smaller than the Pile. In addition, while the majority of the Pile is sourced from scientific text, web crawl, and books, OLC is dominated by code, scientific text, and legal text. This highlights that models designed for use outside these specific domains will likely struggle and may require special techniques for extreme domain generalization.

5.3 SILO

We introduce SILO, which combines an LM trained on permissive data with a non-parametric datastore based on less restricted data. Our goal with SILO is to build an LM—i.e., a model that takes a prefix of text x and outputs a next-word probability distribution over the vocabulary $P(y \mid x)$—but to do so in a legally safe way. We first describe the general methodology from prior work (§5.3.1–5.3.2) and then how we build SILO upon them by placing low-risk data and high-risk data to model parameters and a nonparametric datastore, respectively (§5.4). Implementation details are provided in §5.4.

26 This comparison also dovetails with our experiments in §5.5, where we compare SILO to Pythia, a model trained on the Pile.

Table 15
OLC is large but its distribution is different from that of typical pre-training corpora. Data distribution of OLC (PD, PDSW, PDSWBY) in comparison to the Pile (Gao et al. 2020), a common LM training dataset that is not specifically designed for legal permissibility. We report the number of tokens in billions, and the relative frequency. †: There is no explicit news domain in the Pile, but news sites are some of the most representative data sources in Common Crawl (Dodge et al. 2021).

	PD		PDSW		PDSWBY		*The Pile*	
Domain	**Tokens (B)**	**%**	**Tokens (B)**	**%**	**Tokens (B)**	**%**	**Tokens (B)**	**%**
Code	0.0	0.0	58.9	59.1	58.9	25.8	32.6	9.8
Legal	27.1	86.2	27.1	27.2	27.2	11.9	30.8	9.3
Conversation	0.0	0.0	5.9	5.9	27.2	11.9	33.1	10.0
Math	0.0	0.0	3.5	3.5	3.5	1.50	7.1	2.1
Books	2.9	9.3	2.9	2.9	2.9	1.3	47.1	14.2
Science	1.2	3.8	1.2	1.2	71.5	31.3	86.0	26.0
News	0.2	0.7	0.2	0.2	0.2	0.1	–†	–†
Wikipedia	0.0	0.0	0.0	0.0	37.0	16.2	12.1	3.7
Unverified web	0.0	0.0	0.0	0.0	0.0	0.0	83.1	25.0
Total	31.4	100.0	99.6	100.0	228.3	100.0	331.9	100.0

5.3.1 The Parametric Component. For the parametric component of SILO, we use a standard, dense, decoder-only Transformer LM (Vaswani et al. 2017) using the LLaMA architecture (Touvron et al. 2023). This model uses a fixed set of parameters at both training and inference time.

5.3.2 The Nonparametric Component. We experiment with two widely used retrieval methods for the nonparametric component (Figure 24): the k-nearest neighbors LM (kNN-LM; Khandelwal et al. 2020) and the retrieval-in-context approach (RIC-LM; Shi et al. 2024b; Ram et al. 2023). Each approach constructs a datastore from the raw text data offline, and then uses it on-the-fly at inference time.

The k-Nearest Neighbors Language Model (kNN-LM). A kNN-LM (Khandelwal et al. 2020) interpolates the next-token probability distribution from a parametric LM with a nonparametric distribution based on every token that is stored in a datastore. Given a text dataset consisting of N tokens $c_1 ... c_N$, a datastore is built by creating a key-value pair for every token c_i ($1 \leq i \leq N$). Specifically, a value is c_i and a key k_i is $...c_{i-1}$, a prefix preceding c_i. At test time, given an input prefix x, the nonparametric distribution is computed by:

$$P_{k\text{NN}}(y \mid x) \propto \sum_{(k,v)\in\mathcal{D}} \mathbb{I}[v = y]\,(-d(\text{Enc}(k), \text{Enc}(x)))$$

Here, Enc is an encoder that maps a text into $\mathbb{R}^h$ and $d : \mathbb{R}^h \times \mathbb{R}^h \to \mathbb{R}$ is a distance function, where h is the hidden dimension. We follow Khandelwal et al. (2020) and use the output vector from the last layer of the Transformers in the parametric LM as Enc, L2 distance as d, and an approximate nearest neighbor search using FAISS (Johnson, Douze, and Jégou 2019; details in §5.4). The final model takes the kNN-LM output and

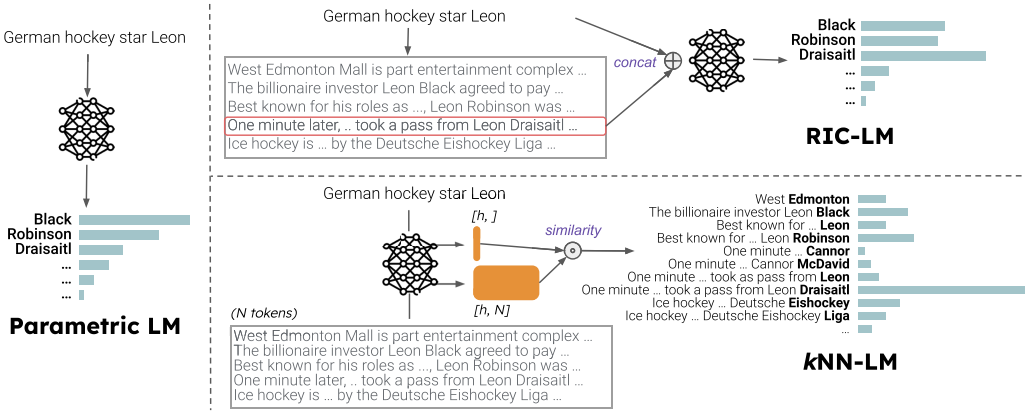

Figure 24
An illustration of a parametric model and two retrieval methods we compare: RIC-LM and kNN-LM. The orange boxes indicate representations of the input prefix and the tokens in the datastore, each in $\mathbb{R}^h$ and $\mathbb{R}^{h \times N}$, where h is a hidden dimension and N is the number of tokens in the datastore. The distribution from kNN-LM in the figure describes $P_{k\text{NN}}$; while omitted in the figure, the final output distribution from kNN-LM is an interpolation between $P_{k\text{NN}}$ and the distribution from the parametric LM. See §5.3.2 for more details of each method.

interpolates it with the output from the parametric LM: $\lambda P_{\text{LM}}(y \mid x) + (1 - \lambda) P_{k\text{NN}}(y \mid x)$, where λ is a hyperparameter between 0 and 1.

The Retrieval-in-Context Language Model (RIC-LM). As an alternative to kNN-LM, RIC-LM (Shi et al. 2024b; Ram et al. 2023) retrieves text blocks from a datastore and feeds them to the parametric LM in context. Specifically, given a dataset consisting of N tokens $c_1 ... c_N$, an index $\mathcal{D}$ is constructed by splitting the data into text blocks $b_1 ... b_M$, optionally with a sliding window. At test time, given an input prefix x, RIC-LM retrieves the most similar paragraph to the prefix $\hat{p} = \arg\max_{b \in \mathcal{D}} \text{sim}(b, x)$ and concatenates it to the prefix to produce $P_{\text{LM}}(y \mid \hat{b}, x)$. Here, sim is a function that computes a similarity score between two pieces of text; we use BM25 following Ram et al. (2023) who show that BM25 outperforms alternative dense retrieval methods.

Future work can improve RIC-LM, e.g., by using multiple text blocks through ensembling (Shi et al. 2024b) or reranking (Ram et al. 2023), by tuning the retrieval system (Shi et al. 2024b), or by training the LM to use retrieved blocks in context (Guu et al. 2020; Izacard et al. 2022b).

Comparison Between kNN-LM and RIC-LM. The key difference between kNN-LM and RIC-LM lies in how the nonparametric component influences the output. In kNN-LM, it directly impacts the output distribution, while in RIC-LM, it indirectly influences the output by affecting the input to the parametric model. kNN-LM intuitively benefits more from a datastore as it provides direct influence to the output and relies less on the parametric component. Nonetheless, RIC-LM interacts more easily with a parametric model (i.e., it is applicable to a black-box LM) and offers better speed and memory efficiency.

Empirical comparisons between kNN-LM and RIC-LM have been largely unexplored; in fact, we are unaware of such work. In our experiments (§5.5.2), we present a

series of such comparisons, with varying sizes of the datastore, and with and without distribution shift.

Attribution and Opt-out. Since elements in the datastore that contribute to the model prediction are transparent, both *k*NN-LM and RIC-LM offer inherent attributions. Moreover, data removed from the datastore is guaranteed not to contribute to any model predictions, allowing data owners to remove their data at the level of individual examples. Both are unique characteristics of nonparametric language models. While prior work studies post-hoc attribution to the data used for training model parameters (Koh and Liang 2017; Han et al. 2023) and removing the effect of specific training examples from parameteric models (Cao and Yang 2015; Jang et al. 2023b), they are arguably not fundamental due to lack of inherent guarantees, and are difficult to scale.

5.4 Building SILO

SILO is built upon the general methodology of *k*NN-LM and RIC-LM. However, unlike prior work that uses the same data for learning model parameters and a nonparametric datastore, SILO uses distinct datasets for these two components.

The key idea behind SILO is to use low-risk data to estimate model parameters, and to use high-risk data only in a nonparametric datastore. This is based on the motivation that model parameters should be learned conservatively, since training data is difficult to remove or trace after model training is completed. In contrast, a nonparametric datastore offers greater flexibility, as it can be easily updated, grown, or filtered, supports data opt-out at the level of individual examples, and provides attributions for free to every model prediction. These functions enable adherence to data-use regulations (§5.1).

Training Datasets. We train each of our LMs on one of the three datasets of OLC: PD data, PDSW data, and PDSWBY data. Each of the resulting models constitutes a different level of possible copyright infringement risk.

Datastore. We assume in-distribution data for each test domain is available at inference time, and construct a datastore for each domain. These test-time datasets can be either in-domain or out-of-domain with respect to the data used to train model parameters.

Implementation Details. We use 1.3B-parameter Transformer LMs based on the LLaMA architecture (Touvron et al. 2023). Each model is trained with 128 A100 GPUs across 16 nodes. Following Muennighoff et al. (2023), we train for multiple epochs in each dataset and perform early stopping. We train our PD, PDSW, and PDSWBY models for 60B, 250B, and 350B tokens in total, respectively. We refer to the Appendix of the original publication (Min et al. 2024) for more details. Because the distribution of OLC is highly skewed (§5.2), we perform a simple upweighting scheme where we upsample all data that accounts for less than 5% by a factor of 3×, which we found to work well after a sweep of different settings.

Evaluation. We benchmark our models using language modeling perplexity on 14 domains that represent both in-domain and out-of-domain data with respect to different levels of OLC. This includes: public-domain legal documents from the **FreeLaw** Project subset of the the Pile (Gao et al. 2020), a held-out collection of books from the **Gutenberg** collection (Project Gutenberg), conversational text from the **Hacker News** subset of the Pile, held-out code files from the **GitHub** subset of the Pile (most of which are

non-permissive licensed), scientific text of NIH Grant abstracts that are taken from the **NIH ExPorter** subset of the PILE, philosophy papers taken from the **PhilPapers** of the PILE, held-out English **Wikipedia** articles from the PILE, news articles from **CC-News** (Mackenzie et al. 2020), books from **BookCorpus2** which is an expanded version of Zhu et al. (2015), books from **Books3** by Presser (2020), random web-crawled pages from **OpenWebText2** (Gokaslan and Cohen 2019; Gao et al. 2020), emails from the **Enron Emails** corpus (Klimt and Yang 2004), **Amazon** product reviews from He and McAuley (2016), and finally clinical notes from **MIMIC-III** (Johnson et al. 2016) with personal identifiable information (PII) masked out. Our choice of domains highlights legal risks discussed in the earlier sections, e.g., CC-News, BookCorpus2, Books3, and Amazon reviews are mostly copyrighted, GitHub is mostly not permissively licensed,[27] and Enron Emails and MIMIC-III include private text. We merge all text into one stream of text and split them into batches with a maximum sequence length of 1,024 and a sliding window of 512, a setup that is standard in prior language modeling literature (Baevski and Auli 2019; Khandelwal et al. 2020). For MIMIC-III, which includes masked PII, we filter out notes where more than 50% of tokens correspond to PII, and then exclude tokens corresponding to PII when computing perplexity.

Datastore. We construct an in-domain datastore for each test domain based on their training data. For datasets from the PILE, we consider 10% of the training data. For *k***NN-LM**, each datastore consists of up to 1 billion h-dimensional vectors ($h = 2{,}048$). We build an index for fast nearest neighbor search using FAISS (Johnson, Douze, and Jégou 2019). For **RIC-LM**, each datastore consists of text blocks with a length of 1,024 and a sliding window of 512. We use BM25 from Pyserini (Lin et al. 2021). More details, including ablations on different implementations of RIC-LM and other hyperparameters can be found in the Appendix of the original publication (Min et al. 2024).

5.5 Experiments

We first evaluate the parametric-only component of SILO trained on the OPEN LICENSE CORPUS (§5.5.1), and then show the effect of adding a datastore that may contain high-risk text (§5.5.2). For all experiments, we use the 1.4B Pythia model (Biderman et al. 2023) as a baseline because it is trained with a similar amount of compute (data size and model parameters), but is trained on mostly high-risk data.

5.5.1 Results: Parametric Component. Table 16 reports results for our 1.3B LMs trained on varying levels of permissively licensed data (PD, PDSW, PDSWBY) compared to Pythia. Despite relying only on permissive data, our models are competitive with Pythia: They match quality on in-domain sources (e.g., FreeLaw, Gutenberg, HackerNews, Wikipedia) and perform comparably on GitHub due to overlap with permissive code in SW. The largest gaps appear on domains excluded from our training, e.g., news, books, OpenWebText, and emails, and Wikipedia in the case of models besides PDSWBY. This illustrates the extreme domain generalization challenge that is present when training on only permissive data, as we hint in §5.2. Further ablations, such as upsampling low-resource data and the impact of including or excluding source code, are reported in the Appendix of the original publication (Min et al. 2024).

27 Kocetkov et al. (2023) estimates about 13% of the GitHub data is under MIT, Apache, and BSD.

Table 16
Perplexity (the lower the better) of the parametric-only SILO trained on PD, PDSW, and PDSWBY (without a datastore), compared to Pythia-1.4B, a model trained with similar amounts of compute but on mostly non-permissive data. We use ■, ■, and ■ to indicate text that is in-domain, out-of-domain, or out-of-domain but has relevant data in-domain (e.g., high-risk GitHub code vs. our permissive GitHub code). Our parametric LMs are competitive to Pythia in-domain but fall short out-of-domain.

Eval data	PD	PDSW	PDSWBY	Pythia
FreeLaw	5.3	5.7	6.5	5.6
Gutenberg	15.2	12.5	14.1	13.1
HackerNews	38.0	13.7	14.5	13.3
GitHub	13.5	2.7	2.8	2.4
NIH ExPorter	28.2	19.2	15.0	11.1
PhilPapers	31.7	17.6	15.0	12.7
Wikipedia	28.9	20.3	11.3	9.1
CC News	34.0	23.3	21.2	12.0
BookCorpus2	25.3	19.2	19.6	13.2
Books3	27.2	19.3	18.6	12.6
OpenWebText2	37.8	21.1	18.8	11.5
Enron Emails	18.6	13.2	13.5	6.9
Amazon	81.1	34.8	37.0	22.9
MIMIC-III	22.3	19.0	15.5	13.1
Average	29.1	17.3	16.0	11.4

5.5.2 Results: Adding the Nonparametric Component. Because building legally permissive LMs poses a challenge of extreme domain generalization, our next question is whether using an in-domain, nonparametric datastore can reduce the gap. We explore this question with our parametric LM trained on the PDSW subset of OLC; see Appendix of the original publication (Min et al. 2024) for results of models trained on PD or PDSWBY. All models are evaluated on a subset of 8 out-of-domain datasets to the parametric model: GitHub, NIH ExPorter, Wikipedia, CC News, Books3, Enron Emails, Amazon, and MIMIC-III.

Main Results. Table 17 shows that adding the datastore with either *k*NN-LM- or RIC-LM-based retrieval improves performance over just using the parameteric component on all domains, but *k*NN-LM is more effective than RIC-LM. In most domains, *k*NN-LM reduces the gap between SILO and Pythia by more than 50% (on NIH ExPorter, Wikipedia, Amazon) or even outperforms Pythia (on GitHub, CC News, Enron Emails, MIMIC-III). Books3 is the domain with the least benefit, on which *k*NN-LM still reduces the gap by 28%.

Impact of Scaling the Datastore. Figure 25 demonstrates that both *k*NN-LM and RIC-LM-based retrieval consistently improves performance as the datastore size increases, with a strong log-linear trend. However, *k*NN-LM improves performance more rapidly than RIC-LM does, consistently over all datasets. Extrapolating the trend suggests that, on the domains that SILO has not outperformed Pythia yet, scaling the datastore even further (with *k*NN-LM retrieval) may enable it to match Pythia.

Why Does kNN-LM Outperform RIC-LM?. Our next question is why *k*NN-LM is better than RIC-LM—is it (a) because *k*NN-LM is better than RIC-LM in general, or (b) because

Table 17
Perplexity (the lower the better) of parametric LMs (Prm-only), *k*NN-LM, and RIC-LM. % in parentheses indicate a reduction in the gap between the parametric-only SILO and Pythia. As in Table 16, ■ indicates in-domain; ■ indicates out-of-domain; ■ indicates out-of-domain but has relevant data in-domain, all with respect to the training data of the parametric LM. Adding a datastore, with *k*NN-LM, effectively reduces the gap between SILO and Pythia.

Eval data	SILO (PDSW)			Pythia
	Prm-only	*k*NN-LM	RIC-LM	Prm-only
GitHub	2.7	2.4 (−100%)	2.4 (−100%)	2.4
NIH ExPorter	19.2	15.0 (−52%)	18.5 (−9%)	11.1
Wikipedia	20.3	14.5 (−52%)	19.4 (−8%)	9.1
CC News	23.3	8.0 (−135%)	16.8 (−58%)	12.0
Books3	19.3	17.4 (−28%)	18.6 (−10%)	12.6
Enron Emails	13.2	5.9 (−116%)	9.9 (−68%)	6.9
Amazon	34.9	26.0 (−75%)	33.7 (−10%)	23.0
MIMIC-III	19.0	6.6 (−210%)	15.6 (−58%)	13.1
Average	19.0	12.0 (−91%)	16.9 (−27%)	11.3

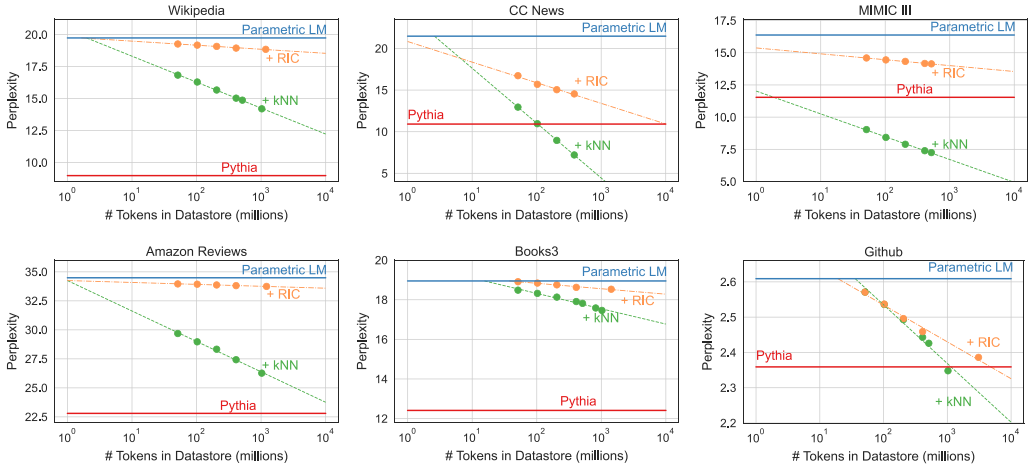

Figure 25
Impact of scaling the datastore of SILO (PDSW). Perplexity on random 128K tokens from the validation data reported. The rightmost dots for *k*NN-LM and RIC-LM in each figure correspond to the final models used in Table 17. Scaling the test-time datastore consistently improves performance over all domains.

*k*NN-LM generalizes out-of-domain better than RIC-LM does? We refer to the Appendix of the original publication (Min et al. 2024) for this discussion. The short answer is both: *k*NN scales better than RIC-LM in the in-domain setting, supporting (a); while both *k*NN-LM and RIC-LM improve performance when the test data is out-of-domain, the gains are larger with *k*NN-LM, supporting (b).

Where Does the Remaining Gap Come From?. Even with a large datastore, SILO lags behind Pythia on a few domains. Moreover, a Pythia-based *k*NN-LM outperforms our model since *k*NN-LM improves Pythia as well. There are two possible points of failure in our model for these cases: Either the parametric component (which outputs P_{LM}) struggles

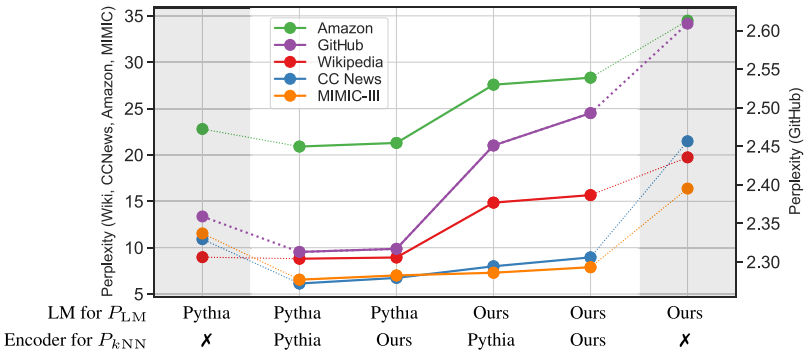

Figure 26
Impact of using different parameters on SILO. The left-most and the right-most models are parametric models, and the other four models are *k*NN-LMs, using a datastore with 204.8 million tokens (20% of the datastore we use for the main experiments). *Ours* indicates our parametric model trained on the PDSW subset of OPEN LICENSE CORPUS. Most of the performance degradation comes from using the out-of-domain parametric LM, rather than using the out-of-domain encoder.

out-of-domain, or the encoder (that outputs $P_{k\text{NN}}$) struggles out-of-domain. To better understand which part of the model contributes to the gap we observe, we vary SILO with different choices for the parametric component and the encoder. We compare replacing either the parametric component or the encoder with Pythia. This setup allows us to measure the effects of the out-of-domain nature of our parametric component (which is only trained on PDSW subset of OLC) in each of these components.

Results in Figure 26 reveal that most performance gaps come from the LM: Performance improves significantly when the parametric component is replaced with Pythia, given a fixed encoder. In contrast, performance improvement is relatively marginal when the encoder is replaced with Pythia, given a fixed parametric component. These results indicate that the parametric component, which gives P_{LM}, is quite sensitive to domain shift, but the encoder, which provides the nonparametric distribution $P_{k\text{NN}}$, is fairly robust to extreme domain shift. This also explains why *k*NN-LM generalizes better than RIC-LM, since RIC-LM is bottlenecked by the parametric component.

In summary, our analysis points to two promising directions to reduce the gap: (1) Scaling the datastore beyond 1 billion tokens, e.g., at the scale of trillions of tokens as in Borgeaud et al. (2022), as demonstrated by Figure 25. (2) Improving the robustness of the model by improving nonparametric techniques or designing a model that only uses a nonparametric distribution (Min et al. 2023b), as demonstrated by Figure 26.

5.5.3 Examples of Data Attribution and Opt-Out. As discussed in §5.1, the design of SILO can better align with various data-use regulations by providing mechanisms for data attribution during inference and for data owners to remove their data from the model at any time.

Data Opt-out. To showcase the impact of opt-out on model performance, we conduct experiments with J.K. Rowling's Harry Potter series. We first identify all seven Harry Potter books from the Books3 corpus of the Pile. For each book, we calculate the perplexity of SILO using two 1.024B token datastores on Books3, but one including the remaining six Harry Potter books and the other excluding any Harry Potter books. This

experiment is to see whether excluding Harry Potter books from the former datastore can reduce the likelihood of generating the leave-out Harry Potter book.

Table 18 shows the results. SILO with Harry Potter books in the datastore effectively improves perplexity over all seven books, closing the gap between the PDSW model and Pythia. However, when the Harry Potter books are removed from the datastore, the perplexity gets worse, approaching that of the parametric-only LM. This illustrates that

Table 18
The effect of data opt-out. *w/ HP* and *w/o HP* indicate that the datastore includes or excludes Harry Potter books, respectively. The number (1 to 7) indicates a different book from the Harry Potter series used as the eval data (excluded from the datastore in each row). ■ indicates in-domain; ■ indicates out-of-domain.

Eval	SILO (PDSW)			Pythia
	Prm-only	*k*NN-LM w/o HP	*k*NN-LM w/ HP	Prm-only
1	15.9	15.2	13.0	9.6
2	17.7	16.7	12.4	10.0
3	16.5	15.6	11.4	9.5
4	17.7	16.8	12.9	10.1
5	17.8	16.9	13.2	10.2
6	17.4	16.5	12.8	10.1
7	18.8	17.8	15.1	10.9
Avg	17.4	16.5	12.9	10.1

Table 19
Attribution examples on Harry Potter books. We show the top-1 retrieved context of SILO (PDSW). **Red underline text** indicates the next token that immediately follows the prefix. In both examples, the test data is from the sixth novel and the retrieved context is from the fourth novel in the Harry Potter series. In the series, *Azkaban* is the notorious wizarding prison, and the *green light* is a distinct characteristic of the Killing Curse, *Avada Kedavra*.

`Test Prefix` 'I - what - *dragons*?' spluttered the Prime Minister. 'Yes, three,' said Fudge. 'And a sphinx. Well, good day to you.' The Prime Minister hoped beyond hope that dragons and sphinxes would be the worst of it, but no. Less than two years later, Fudge had erupted out of the fire yet again, this time with the news that there had been a mass breakout from
`Test Continuation` **Azkaban**. 'A *mass* breakout?' the Prime Minister had repeated hoarsely.
`Retrieved Prefix` 'D' you know Crouch, then?' said Harry. Sirius' face darkened. He suddenly looked as menacing as the night when Harry had first met him, the night when Harry had still believed Sirius to be a murderer. 'Oh, I know Crouch all right,' he said quietly. 'He was the one who gave me the order to be sent to
`Retrieved Continuation` **Azkaban** - without a trial.'

`Test Prefix` Terror tore at Harry's heart…he had to get to Dumbledore and he had to catch Snape…somehow the two things were linked…he could reverse what had happened if he had them both together…Dumbledore could not have died…(…) Harry felt Greyback collapse against him; with a stupendous effort he pushed the werewolf off and onto the floor as a jet of
`Test Continuation` **green** light came flying toward him; he ducked and ran, headfirst, into the fight.
`Retrieved Prefix` Voldemort was ready. As Harry shouted, "Expelliarmus!" Voldemort cried, "Avada Kedavra!"
`Retrieved Continuation` **green** light issued from Voldemort's wand just as a jet of red light blasted from …

eliminating the effect of the Harry Potter books from the model substantially reduces the likelihood of generating the leave-out book.

Attribution Examples. To show the attribution feature of our model, Table 19 provides qualitative examples on the top-1 context retrieved by SILO. The model is able to assign a high probability to the ground truth token by retrieving highly relevant context. It achieves this by leveraging the unique characteristics of the text within the datastore, such as recognizing that *Azkaban* refers to the prison and *green light* is associated with the *Killing Curse* in the Harry Potter books. This highlights that a nonparametric approach addresses specific legal risks that we have discussed earlier, e.g., it offers per-token attribution for free, and can provide a copyright notice when part of copyrighted text is being used for the probability distribution.

5.6 Discussion & Future Work

Our work suggests that it is possible to improve the trade-off between legal risk and model performance when training LMs. Our approach provides new options for model designers to mitigate the legal risk of LMs, and empowers stakeholders to have more control over the data that drives these systems. We point out a number of rich areas for future work, beyond what was mentioned throughout the article.

5.6.1 Addressing the Limitations of SILO. SILO does not eliminate legal risk; rather, it provides users more control over the model's generated content and functionalities to better align with legal regulations. For instance, SILO does not replace permissions to use copyrighted content in a datastore when providing attribution is not sufficient, but its opt-out capabilities can strengthen fair use defense. Moreover, SILO does not prevent copying copyright content from a datastore, but it offers a way to prevent generating sensitive text (Huang et al. 2023) or prevent copying the content verbatim. These functionalities increase the likelihood of a successful fair use defense if used appropriately. Moreover, although effective for copyright and privacy, SILO may worsen fairness concerns, e.g., toxicity and racial bias, due to reliance on older copyright-expired texts. Balancing legal risk mitigation with fairness remains an open challenge. Finally, our study depends on explicit metadata to identify licenses, likely underestimating the supply of permissively licensed text. Future work may explore inferring licenses at scale to build larger, more diverse corpora.

5.6.2 Introducing Novel Data Licensing Approaches. SILO introduces the possibility for data owners to set different levels of permissivity for learning parameters and for including in a nonparametric datastore. A data owner might choose to be more permissive about including data in the datastore due to its ease of removal, ensuring that the excluded data has no influence on model predictions anymore, and its ability to provide per-prediction attribution. Moreover, we envision that SILO could provide a path forward for data owners to get properly credited (or be paid directly) every time their data in a datastore contributes to a prediction. This is orthogonal to recent work that circumvented copyright issues by licensing out training data from data creators (Yu et al. 2023a).

5.6.3 Investigating Other Copyright Risk Mitigation Strategies. It is critical to continue to develop new techniques that use copyrighted data while protecting the rights of data owners and subjects. In addition to nonparametric approaches, there are many other

ways to achieve these goals. First, one could train LMs on copyrighted content but filter and guide their outputs towards text that is non-infringing (Henderson et al. 2023). Second, training models with differential privacy (Dwork et al. 2006; Abadi et al. 2016) or near-access freeness (Vyas, Kakade, and Barak 2023) may prevent them from regenerating individual details of copyright data. Finally, one could provide attributions for standard base LMs using post-hoc attribution methods, e.g., influence functions (Koh and Liang 2017), rather than switching the model class to a retrieval-based model. All of these methods are complementary and orthogonal to our proposed approach.

5.6.4 Generalizing SILO *as a Modular LM.* Our work is closely related to recent studies on modular LMs, which have specialized parameters (or *experts*) trained on different domains (Gururangan et al. 2022; Li et al. 2022; Gururangan et al. 2023), languages (Pfeiffer et al. 2020, 2022), or tasks (Chen et al. 2022c; Jang et al. 2023a). Our work extends modular LMs to include nonparametric datastores, and focuses on *specializing* different parts of the model to low- and high-risk subsets of the training data. Legal risks may also be mitigated with a collection of parametric expert models that are specialized to low- and high-risk data. Future work may explore this possibility as well as the usefulness of combining a nonparametric datastore with parametric experts.

5.6.5 Extending SILO *to Other Modalities.* While this work focuses on text-only models, similar methods to ours could apply to other domains and modalities. For instance, it might be possible to build permissive text-to-image generative models (Rombach et al. 2022) using compartmentalized public domain pre-training and retrieval augmentation (Chen et al. 2022a; Golatkar et al. 2023). We believe such approaches are especially promising because there are many sources of public domain data in other modalities, e.g., images, speech, video, and more.

5.7 Summary

We introduce SILO, a language model that mitigates legal risk by training only on permissively licensed data (OPEN LICENSE CORPUS) while leveraging an unrestricted nonparametric datastore at inference. This design enables use of high-risk data without training on it, supports sentence-level attribution, and allows opt-outs by removing datastore entries. On language modeling, the parametric-only SILO performs competitively in-domain but struggles out-of-domain, underscoring the challenge of extreme generalization. Adding a nonparametric datastore with kNN-LM retrieval closes this gap, often surpassing the unrestricted Pythia baseline. Performance scales with datastore size, and the nonparametric encoder proves more robust to distribution shift than the parametric model. These results highlight promising directions for building legally compliant AI systems.

6. Conclusion & Future Work

LMs, such as ChatGPT, continue to significantly transform the fields of NLP and AI. By training deep neural models, such as Transformers (Vaswani et al. 2017), with 100 billion or more parameters using extensive text data (e.g., 100 billion tokens or more), we have developed models capable of performing a wide range of NLP tasks without the need for task-specific architecture design or training. These models are already being widely integrated into user-facing production systems, including ChatGPT, Claude, and Gemini.

This article focused on (1) *understanding* how these models work and (2) *rethinking* how the next generation of models should use data at scale.

Understanding Current Language Models (§3). We discussed how to better understand current "black-box" LMs and demonstrated how far they can perform on these tasks, focusing on ICL.

1. We improved LMs' ICL (§3.2) by training LMs with a multi-task objective on a large collection of tasks and evaluating them on a new task during inference. The key is to train LMs to condition on k task examples (as ICL typically does) *during training* and then, at inference time, condition on *new* examples about the given task (Min et al. 2022b). This helps LMs learn to better use the given examples in order to activate required patterns.
2. We then challenged the widespread belief that ICL allows an LM to acquire new abilities on the fly with no training. We showed that LMs accurately perform tasks with ICL even when the given examples are incorrect (Min et al. (2022c); Lyu et al. (2023); Wang et al. (2023a); §3.6). This indicates that LMs perform tasks by relying on patterns present in their training data, which can be activated in a specific way, rather than by obtaining a new ability on the fly.
3. Our work has led to a significant body of follow-up work discovering unexpected behaviors of LMs (Madaan and Yazdanbakhsh 2022; Wei et al. 2023b; Jang, Ye, and Seo 2022; Wies, Levine, and Shashua 2023; Lampinen et al. 2022; Pan et al. 2023; Kim et al. 2022; Schaeffer et al. 2023; Zhang et al. 2023b; She et al. 2023; Turpin et al. 2023) and has also inspired the development of improved models (Chung et al. 2022; Ivison et al. 2023), as discussed in §3.9.

Nonparametric Language Models (§4). We discussed nonparametric LMs, a new class of LMs that includes not only learned parameters but also a datastore, i.e., a massive collection of raw text documents.

1. We focused on retrieval augmentation, which first retrieves text pieces from a datastore and feeds them to the LM (§4.2). We introduced DPR (Karpukhin et al. 2020), one of the first neural retrieval models that opened up a new era in retrieval augmentation. Its effect is further demonstrated by a NeurIPS Competition on open-domain question answering (§4.3; Min et al. 2021a), and has led to a large body of subsequent work (Lewis et al. 2020b; Xiong et al. 2021a, b; Mao et al. 2020; Ding et al. 2021; Gao and Callan 2021; Izacard et al. 2022a; Zhan et al. 2021; Gao and Callan 2022; Asai et al. 2021; Ni et al. 2022; Ram et al. 2022).
2. We next focused on LMs with *a nonparametric softmax* (§4.4). Instead of outputting a categorical distribution over words, this method assigns

scores to every word or phrase in the datastore as a nonparametric distribution (Khandelwal et al. 2020). Our work, NPM (Min et al. 2023b), is one of the first to train an LM with a nonparametric softmax. NPM outperforms alternatives on a range of tasks, is especially effective in handling rare concepts (such as rare entities and facts), and can grow and be updated by expanding and replacing the datastore.

3. We summarized open problems in nonparametric LMs (§4.6), including exploring alternative architectures that incorporate larger retrieval results more frequently and more efficiently (Table 14) and improving runtime efficiency.

Responsible Language Models (§5). We discussed how to leverage new functionalities of nonparametric LMs to advance responsible data use (Min et al. 2024).

1. We claimed that using all available data on the web raises concerns related to crediting data creators and complying with legal constraints such as copyrights and the Right to be Forgotten.
2. We introduced SILO, a nonparametric LM that was trained exclusively on permissively licensed data and uses copyrighted data in a datastore during inference. SILO improves legal compliance because it (1) helps data creators receive appropriate credit for their contribution by providing data attributions for every model prediction and (2) enables support for data opt-out requests through the removal of data from the datastore at any time.
3. We demonstrated that parametric-only SILO struggles with extreme domain shift; however, adding an inference-time datastore using a nonparametric softmax addresses this challenge, allowing SILO to reduce the gap with an existing LM by 90% on average.

Future Directions

Every future LM should be flexible, up-to-date, and performant with fewer parameters than they currently have. While we believe a nonparametric LM is one of the most promising directions to achieve this goal, there are still many open challenges, e.g., how to design these models to be more expressive and efficient (as explored Section 4.4), and whether they can ultimately open up a new dimension in *scaling*—a crucial aspect in the remarkable success of LMs. We believe future research in our field should tackle these challenges and, ultimately, expand model capabilities to address critical problems, both within natural language (e.g., factuality) and beyond (e.g., legality and privacy)—areas that current LMs are far from solving.

Expanding Nonparametric LMs. One of the key questions is whether nonparametric LMs can provide an alternative pathway for scaling—the datastore size. To answer this question, we should scale the datastores to accommodate Internet-scale data consisting of trillions of words—the scale of modern LMs' training data. Currently, they

accommodate relatively small-scale data (e.g., a few billions of words) and are thus limited to a specific domain (e.g., Wikipedia) in many cases. A key obstacle in scaling the datastore lies in scaling the nearest neighbor search algorithm to handle trillions of vectors while improving its runtime speed. Addressing this challenge necessitates not only NLP and machine learning research but also collaborative efforts with researchers specializing in computer systems, algorithms, and databases. Furthermore, nonparametric LMs should be expanded to handle a wider range of tasks. One notable area is higher-level reasoning, i.e., aggregating information from different parts of the data and generalizing to novel scenarios. By removing the need for memorizing the data in their parameters, nonparametric LMs can sharpen their focus on higher-level reasoning abilities, which was briefly explored in our work (Press et al. 2023).

Making LMs Factual. One of the most critical issues with today's LMs lies in *factuality*, e.g., they often generate false information. This poses a serious concern since the pursuit of acquiring new information is one of the primary use cases for LMs, and the long-tail distribution of facts severely challenges LMs. There are three critical challenges. First, we should design models that improve the trade-off between *precision* (how accurate a response is) and *recall* (how much information a response covers). In particular, improving (or even defining) recall is largely underexplored—we intend to work on this aspect, with AmbigQA (Min et al. 2020, 2021b) as part of an initial effort. Second, we need evaluation methods that are more systematic, rigorous, and scalable. The challenge arises because a response often consists of a mixture of true and false pieces of information, each piece requiring costly verification. We built initial work toward this goal (Min et al. 2023a), which we have made accessible through a user-friendly package and has gained wide adoption within just four months of its initial release (Ye et al. 2023c; Sun, Gupta, and Iyyer 2023; Malaviya et al. 2023; Dhuliawala et al. 2023; Tian et al. 2023; Asai et al. 2024). Finally, we should develop models that further relax the assumptions about the queries, e.g., the validity of presuppositions made within a question, which we initiated in (Yu et al. 2023b). For instance, the question "How much of the EU's development fund is the UK contributing to?" presupposes that the UK is still a part of the EU, which should be challenged.

Optimizing LMs for Societal Objectives. As LMs become increasingly prevalent in real-world applications, researchers are responsible for studying their societal impact, identifying potential harms, and providing technical solutions, which we plan to actively work on. For instance, developing technical solutions to enhance LM compliance with legal constraints and enable proper attribution to data creators is very critical, as explored in Min et al. (2024). Moreover, it is critical to develop analysis techniques to understand LMs' gender, geographical, or political bias, e.g., how the training data and datastores affect model bias differently, and develop mitigation methods. Finally, we need more work in designing new algorithms that identify and mitigate privacy concerns in LMs and protect the interests of data creators, LM developers, and users. One of unexplored research questions is how to protect the data in nonparametric LMs' datastores in both training and inference, where techniques like federated learning and differential privacy can be leveraged.

Acknowledgments

SM would like to thank her thesis committee members, Noah A. Smith, Shane Steinert-Threlkeld, and Graham Neubig, for their guidance and support. She is also grateful to her collaborators: Scott Wen-tau Yih, Danqi Chen, Mike Lewis, Weijia Shi, Barlas Oğuz, Xinxi Lyu, Vladimir Karpukhin, Minjoon

Seo, Pang Wei Koh, Daniel Khashabi, Patrick Lewis, Noah A. Smith, Akari Asai, Ari Holtzman, Muru Zhang, Srini Iyer, Victor Zhong, Michael Duan, Niloofar Mireshghallah, Ali Farhadi, Kalpesh Krishna, Mohit Iyyer, Mikel Artetxe, Ofir Press, Ludwig Schmidt, Yashar Mehdad, Matt Gardner, Sameer Singh, Rulin Shao, Yejin Choi, Boshi Wang, Eric Wallace, Michihiro Yasunaga, Anshuman Suri, Zexuan Zhong, Belinda Z. Li, Julian Michael, Iz Beltagy, Jonathan Berant, Alon Talmor, Jiacheng Liu, Tim Dettmers, Zeqiu (Ellen) Wu, Anas Awadalla, Huan Sun, Chenglei Si, Arman Cohan, Jacqueline He, Jungsoo Park, Sachin Gururangan, Mitchell Wortsman, Xilun Chen, Chen Zhao, Jordan Boyd-Graber, David Wadden, Douwe Kiela, Xi Victoria Lin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Sang Michael Xie, Xinyan Velocity Yu, Lianhui Qin, Prithviraj Ammanabrolu, Tong Chen, Gabriel Ilharco, Qingqing Cao, Yizhong Wang, and Yanai Elazar. Finally, she extends her deepest gratitude to the members of UW-NLP, Meta AI, and AI2 for their support.

References

Abadi, Martin, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *ACM SIGSAC*. https://doi.org/10.1145/2976749.2978318

Aghajanyan, Armen, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. *arXiv preprint arXiv:2101.11038*. https://doi.org/10.18653/v1/2021.emnlp-main.468

Aiden, Erez Lieberman and Jean-Baptiste Michel. 2011. Quantitative analysis of culture using millions of digitized books. *Science*, 331:176–182. https://doi.org/10.1126/science.1199644, PubMed: 21163965

Artetxe, Mikel, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, et al. 2021. Efficient large scale language modeling with mixtures of experts. *arXiv preprint arXiv:2112.10684*. https://doi.org/10.18653/v1/2022.emnlp-main.804

ArXiv. 2023. Arxiv dataset.

Asai, Akari, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over Wikipedia graph for question answering. In *Proceedings of the International Conference on Learning Representations*.

Asai, Akari, Sewon Min, Zexuan Zhong, and Danqi Chen. 2023. ACL 2023 Tutorial: Retrieval-based language models and applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pages 41–46. https://doi.org/10.18653/v1/2023.acl-tutorials.6

Asai, Akari, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of the International Conference on Learning Representations*.

Asai, Akari, Xinyan Yu, Jungo Kasai, and Hannaneh Hajishirzi. 2021. One question answering model for many languages with cross-lingual dense passage retrieval. In *Proceedings of Advances in Neural Information Processing Systems*.

Baevski, Alexei and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *Proceedings of the International Conference on Learning Representations*.

Baeza-Yates, Ricardo, Berthier Ribeiro-Neto, et al. 1999. *Modern information retrieval*, volume 463. ACM Press.

Bandy, Jack and Nicholas Vincent. 2021. Addressing "documentation debt" in machine learning: A retrospective datasheet for BookCorpus. In *Proceedings of Advances in Neural Information Processing Systems*.

Baudiš, Petr and Jan Šedivỳ. 2015. Modeling of the question answering task in the YodaQA system. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 222–228. https://doi.org/10.1007/978-3-319-24027-5_20

Bengio, Yoshua, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13.

Berant, Jonathan, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/D13-1160

Biderman, Stella, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the International Conference on Learning Representations*.

Borgeaud, Sebastian, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the International Conference of Machine Learning*.

Bourtoule, Lucas, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2020. Machine unlearning. *arxiv preprint arXiv:1912.03817*. https://doi.org/10.1109/SP40001.2021.00019

Brants, T., Ashok Popat, Peng Xu, Franz Josef Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Conference on Empirical Methods in Natural Language Processing*.

Brittain, Blake. 2023. U.S. copyright office says some AI-assisted works may be copyrighted. *Reuters*, 15 March.

Brown, Tom, 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. 2020a. Language models are few-shot learners. In *Proceedings of Advances in Neural Information Processing Systems*.

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, et al. 2020b. Language models are few-shot learners. In *Proceedings of Advances in Neural Information Processing Systems*.

Burns, Collin, Hao-Tong Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Cao, Yinzhi and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pages 463–480. https://doi.org/10.1109/SP.2015.35

Carlini, Nicholas, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. In *Proceedings of the International Conference on Learning Representations*.

Carlini, Nicholas, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *USENIX Security Symposium*.

Ceri, Stefano, Alessandro Bozzon, Marco Brambilla, Emanuele Della Valle, Piero Fraternali, Silvia Quarteroni, Stefano Ceri, Alessandro Bozzon, Marco Brambilla, Emanuele Della Valle, et al. 2013. An introduction to information retrieval. *Web Information Retrieval*, pages 3–11. https://doi.org/10.1007/978-3-642-39314-3_1

Chang, Kent K., Mackenzie Cramer, Sandeep Soni, and David Bamman. 2023. Speak, memory: An archaeology of books known to ChatGPT/GPT-4. *arXiv preprint arXiv:2305.00118*. https://doi.org/10.18653/v1/2023.emnlp-main.453

Chen, Danqi, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the Association for Computational Linguistics*, pages 1870–1879. https://doi.org/10.18653/v1/P17-1171

Chen, Mark, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chen, Wenhu, Hexiang Hu, Chitwan Saharia, and William W. Cohen. 2022a. Re-Imagen: Retrieval-Augmented Text-to-Image Generator. In *Proceedings of Advances in Neural Information Processing Systems*.

Chen, Wenlin, David Grangier, and Michael Auli. 2016. Strategies for training large vocabulary neural language models. In *Proceedings of the Association for Computational Linguistics*, pages 1975–1985. https://doi.org/10.18653/v1/P16-1186

Chen, Yanda, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2022b. Meta-learning via language model in-context tuning. In *Proceedings of the Association for Computational Linguistics*, pages 719–730. https://doi.org/10.18653/v1/2022.acl-long.53

Chen, Zitian, Yikang Shen, Mingyu Ding, Zhenfang Chen, Hengshuang Zhao, Erik Learned-Miller, and Chuang Gan. 2022c. Mod-squad: Designing mixture of experts as modular multi-task learners. *arXiv preprint arXiv:2212.08066*. https://doi.org/10.1109/CVPR52729.2023.01138

Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Chung, Hyung Won, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Clark, Kevin, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceedings of the International Conference on Learning Representations*.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the Association for Computational Linguistics*, pages 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747

Dagan, Ido, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*. https://doi.org/10.1007/11736790_9

De Vynck, Gerrit. 2023. ChatGPT maker OpenAI faces a lawsuit over how it used people's data. *Washington Post*, 28 June.

Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407. https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

Deng, Xiang, Yu Su, Alyssa Lees, You Wu, Cong Yu, and Huan Sun. 2021. ReasonBERT: Pre-trained to reason with distant supervision. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 6112–6127. https://doi.org/10.18653/v1/2021.emnlp-main.494

Dettmers, Tim, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. In *Proceedings of the International Conference on Learning Representations*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186. https://doi.org/10.18653/v1/N19-1423

Dhingra, Bhuwan, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273. https://doi.org/10.1162/tacl_a_00459

Dhuliawala, Shehzaad, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*. https://doi.org/10.18653/v1/2024.findings-acl.212

Ding, Yingqi, Qu Yuchen, Jing Liu, Kai Liu, Ruiyang Ren, Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 5835–5847.

Dodge, Jesse, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/2021.emnlp-main.98

Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284. https://doi.org/10.1007/11681878_14

Efrat, Avia and Omer Levy. 2020. The Turking Test: Can language models understand instructions? *arXiv preprint arXiv:2010.11982*.

Evgeniou, Theodoros and Massimiliano Pontil. 2004. Regularized multi–task learning. In *Knowledge Discovery and Data Mining*. https://doi.org/10.1145/1014052.1014067

Feldman, Vitaly and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Proceedings of Advances in Neural Information Processing Systems*.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference of Machine Learning*.

Franz, Alex and Thorsten Brants. 2006. All our n-gram are belong to you. *Google Machine Translation Team*, 20.

Freeman, William T., Thouis R. Jones, and Egon C. Pasztor. 2002. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22:56–65. https://doi.org/10.1109/38.988747

Fried, Daniel, Armen Aghajanyan, Jessy Lin, Sida Wang, Eric Wallace, Freda Shi, Ruiqi Zhong, Scott Yih, Luke Zettlemoyer, and Mike Lewis. 2023. InCoder: A generative model for code infilling and synthesis. In *Proceedings of the International Conference on Learning Representations*.

Gao, Leo, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The Pile: An 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Gao, Luyu and Jamie Callan. 2021. Condenser: A pre-training architecture for dense retrieval. In *Proceedings of Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/2021.emnlp-main.75

Gao, Luyu and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the Association for Computational Linguistics*, pages 2843–2853. https://doi.org/10.18653/v1/2022.acl-long.203

Gao, Luyu, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y. Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. 2022. Attributed text generation via post-hoc research and revision. *arXiv preprint arXiv:2210.08726*.

Gao, Tianyu, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the Association for Computational Linguistics*, pages 3816–3830. https://doi.org/10.18653/v1/2021.acl-long.295

Ge, Tiezheng, Kaiming He, Qifa Ke, and Jian Sun. 2014. Optimized product quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. https://doi.org/10.1109/TPAMI.2013.240, PubMed: 26353197

Gershgorn, David. 2021. GitHub's automatic coding tool rests on untested legal ground. *The Verge*, 7 July.

Gillick, Daniel, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Conference on Computational Natural Language Learning*. https://doi.org/10.18653/v1/K19-1049

Gokaslan, Aaron and Vanya Cohen. 2019. OpenWebText corpus. http://Skylion007.github.io/

Golatkar, Aditya, Alessandro Achille, Ashwin Swaminathan, and Stefano Soatto. 2023. Training data protection with compositional diffusion models. *arXiv preprint arXiv:2308.01937*.

Grave, É.douard, Armand Joulin, Moustapha Cissé, David Grangier, and Hervé Jégou. 2017. Efficient softmax approximation for GPUs. In *Proceedings of the International Conference of Machine Learning*.

Groeneveld, Dirk. 2023. The big friendly filter. https://github.com/allenai/bff

Gu, Jiatao, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In *Association for the Advancement of Artificial Intelligence*. https://doi.org/10.1609/aaai.v32i1.12013

Gu, Yuxian, Li Dong, Furu Wei, and Minlie Huang. 2023. Pre-training to learn in context. In *Proceedings of the Association for Computational Linguistics*, pages 4849–4870. https://doi.org/10.18653/v1/2023.acl-long.267

Guo, Ruiqi, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the International Conference of Machine Learning*.

Gururangan, Suchin, Mike Lewis, Ari Holtzman, Noah A. Smith, and Luke Zettlemoyer. 2022. DEMix layers: Disentangling domains for modular language modeling. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 5557–5576. https://doi.org/10.18653/v1/2022.naacl-main.407

Gururangan, Suchin, Margaret Li, Mike Lewis, Weijia Shi, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. 2023. Scaling expert language models with unsupervised domain discovery. *arXiv preprint arXiv:2303.14177*.

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

Halawi, Danny, Jean-Stanislas Denain, and Jacob Steinhardt. 2023. Overthinking the truth: Understanding how language models process false demonstrations. *arXiv preprint arXiv:2307.09476*.

Han, Xiaochuang, Daniel Simig, Todor Mihaylov, Yulia Tsvetkov, Asli Celikyilmaz, and Tianlu Wang. 2023. Understanding in-context learning via supportive pretraining data. In *Proceedings of the Association for Computational Linguistics*, pages 12660–12673. https://doi.org/10.18653/v1/2023.acl-long.708

He, Junxian, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021. Efficient nearest neighbor language models. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 5703–5714. https://doi.org/10.18653/v1/2021.emnlp-main.461

He, Ruining and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the World Wide Web Conference*. https://doi.org/10.1145/2872427.2883037

Henderson, Matthew, Rami Al-Rfou, Brian Strope, Yun-hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *ArXiv*, abs/1705.00652.

Henderson, Peter, Mark Krass, Lucia Zheng, Neel Guha, Christopher D. Manning, Dan Jurafsky, and Daniel Ho. 2022. Pile of Law: Learning responsible data filtering from the law and a 256GB open-source legal dataset. In *Proceedings of Advances in Neural Information Processing Systems*.

Henderson, Peter, Xuechen Li, Dan Jurafsky, Tatsunori Hashimoto, Mark A. Lemley, and Percy Liang. 2023. Foundation models and fair use. *arXiv preprint arXiv:2303.15715*. https://doi.org/10.2139/ssrn.4404340

Hendrycks, Dan, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Proceedings of Advances in Neural Information Processing Systems*.

Hoffmann, Jordan, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Hollander, Myles, Douglas A. Wolfe, and Eric Chicken. 2013. *Nonparametric Statistical Methods*. John Wiley & Sons.

Holtzman, Ari, Peter West, Vered Schwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 7038–7051. https://doi.org/10.18653/v1/2021.emnlp-main.564

Hu, Minqing and Bing Liu. 2004. Mining and summarizing customer reviews. In *Knowledge Discovery and Data Mining*. https://doi.org/10.1145/1014052.1014073

Huang, Po Sen, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for Web search using clickthrough data. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 2333–2338. https://doi.org/10.1145/2505515.2505665

Huang, Yangsibo, Samyak Gupta, Zexuan Zhong, Kai Li, and Danqi Chen. 2023. Privacy implications of retrieval-based language models. *arXiv preprint arXiv:2305.14888*. https://doi.org/10.18653/v1/2023.emnlp-main.921

Humeau, Samuel, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *Proceedings of the International Conference on Learning Representations*.

Ivison, Hamish, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, et al. 2023. Camels in a changing climate: Enhancing LM adaptation with Tulu 2. *arXiv preprint arXiv:2311.10702*.

Iyer, Srinivasan, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, et al. 2022. OPT-IML: Scaling language model instruction meta learning through the lens of generalization. *arXiv preprint arXiv:2212.12017*.

Iyer, Srinivasan, Sewon Min, Yashar Mehdad, and Scott Wen-tau Yih. 2021. ReConsider: Re-ranking using span-focused cross-attention for open domain question answering. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1280–1287. https://doi.org/10.18653/v1/2021.naacl-main.100

Izacard, Gautier, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022a. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Izacard, Gautier and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the European Chapter of the Association for Computational Linguistics*, pages 874–880. https://doi.org/10.18653/v1/2021.eacl-main.74

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

Izacard, Gautier, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Sebastian Riedel, and Edouard Grave. 2020. A memory efficient baseline for open domain question answering. *arXiv preprint arXiv:2012.15156*.

Jang, Joel, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023a. Exploring the benefits of training expert language models over instruction tuning. In *Proceedings of the International Conference of Machine Learning*.

Jang, Joel, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. 2022. TemporalWiki: A lifelong benchmark for training and evaluating ever-evolving language models. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 6237–6250. https://doi.org/10.18653/v1/2022.emnlp-main.418

Jang, Joel, Seonghyeon Ye, and Minjoon Seo. 2022. Can large language models truly understand prompts? A case study with negated prompts. In *NeurIPS 2022 Workshop on Transfer Learning for NLP (TL4NLP)*.

Jang, Joel, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2023b. Knowledge unlearning for mitigating privacy risks in language models. *Proceedings of the Association for Computational Linguistics*, pages 14389–14408. https://doi.org/10.18653/v1/2023.acl-long.805

Jegou, Herve, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128. https://doi.org/10.1109/TPAMI.2010.57, PubMed: 21088323

Johnson, Alistair E. W., Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data*. https://doi.org/10.1038/sdata.2016.35

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

Joshi, Mandar, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77. https://doi.org/10.1162/tacl_a_00300

Joshi, Mandar, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A

large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the Association for Computational Linguistics*, pages 1601–1611. https://doi.org/10.18653/v1/P17-1147

Jurafsky, Dan and James H. Martin. 2000. Speech and language processing—An introduction to natural language processing, computational linguistics, and speech recognition. In *Prentice Hall Series in Artificial Intelligence*.

Kadrey et al. v. Meta Platforms, Inc. 2023. Case 3:23-cv-03417, n.d. Cal.

Kalo, Jan Christoph and Leandra Fichtel. 2022. KAMEL: Knowledge analysis with multitoken entities in language models. In *Proceedings of the Conference on Automated Knowledge Base Construction*.

Kandpal, Nikhil, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2022. Large language models struggle to learn long-tail knowledge. *arXiv preprint arXiv:2211.08411*.

Kandpal, Nikhil, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. In *Proceedings of the International Conference of Machine Learning*.

Karpukhin, Vladimir, Barlas Oğuz, 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 Empirical Methods in Natural Language Processing*, pages 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

Kassner, Nora and Hinrich Schütze. 2020. BERT-kNN: Adding a kNN search component to pretrained language models for better QA. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3424–3430. https://doi.org/10.18653/v1/2020.findings-emnlp.307

Katz, Slava M. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35400–401. https://doi.org/10.1109/TASSP.1987.1165125

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

Khashabi, Daniel, 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 EMNLP*, pages 1896–1907. https://doi.org/10.18653/v1/2020.findings-emnlp.171

Khattab, Omar and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development In Information Retrieval*, pages 39–48. https://doi.org/10.1145/3397271.3401075

Kim, Junyeob, Hyuhng Joon Kim, Hyunsoo Cho, Hwiyeol Jo, Sang-Woo Lee, Sang goo Lee, Kang Min Yoo, and Taeuk Kim. 2022. Ground-truth labels matter: A deeper look into input-label demonstrations. In *Proceedings of Empirical Methods in Natural Language Processing*.

Kingma, Diederik P. and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, Diederik P. and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

Klimt, Bryan and Yiming Yang. 2004. The Enron corpus: A new dataset for email classification research. In *Proceedings of European Conference of Machine Learning*. https://doi.org/10.1007/978-3-540-30115-8_22

Kobayashi, Mei and Koichi Takeda. 2000. Information retrieval on the web. *ACM Computing Surveys (CSUR)*, 32(2):144–173. https://doi.org/10.1145/358923.358934

Kocetkov, Denis, Raymond Li, Loubna Ben allal, Jia Li, Chenghao Mou, Yacine Jernite, Margaret Mitchell, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro Von Werra, and Harm de Vries. 2023. The Stack: 3TB of permissively licensed source code. *Transactions on Machine Learning Research*.

Kodirov, Elyor, Tao Xiang, Zhenyong Fu, and Shaogang Gong. 2015. Unsupervised domain adaptation for zero-shot learning. In *Proceedings of the IEEE International Conference on Computer Vision*. https://doi.org/10.1109/ICCV.2015.282

Koh, Pang Wei and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the International Conference of Machine Learning*.

Kulis, Brian. 2013. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364. https://doi.org/10.1561/9781601986979

Kulkarni, Mayank, Debanjan Mahata, Ravneet Arora, and Rajarshi Bhowmik. 2022. Learning rich representation of keyphrases from text. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 891–906. https://doi.org/10.18653/v1/2022.findings-naacl.67

Kwiatkowski, Tom, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, pages 452–466. https://doi.org/10.1162/tacl_a_00276

Lampinen, Andrew Kyle, Ishita Dasgupta, Stephanie C. Y. Chan, Kory Matthewson, Michael Henry Tessler, Antonia Creswell, James L. McClelland, Jane X. Wang, and Felix Hill. 2022. Can language models learn from explanations in context? In *Proceedings of Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/2022.findings-emnlp.38

Lan, Tian, Deng Cai, Yan Wang, Heyan Huang, and Xian-Ling Mao. 2023. Copy is all you need. In *Proceedings of the International Conference on Learning Representations*.

Lee, Jinhyuk, Zhuyun Dai, Sai Meher Karthik Duddu, Tao Lei, Iftekhar Naim, Ming-Wei Chang, and Vincent Zhao. 2024. Rethinking the role of token retrieval in multi-vector retrieval. *Advances in Neural Information Processing Systems*, 36.

Lee, Jinhyuk, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning dense representations of phrases at scale. In *Proceedings of the Association for Computational Linguistics*, pages 6634–6647. https://doi.org/10.18653/v1/2021.acl-long.518

Lee, Katherine, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating training data makes language models better. In *Proceedings of the Association for Computational Linguistics*, pages 8424–8445. https://doi.org/10.18653/v1/2022.acl-long.577

Lee, Kenton, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the Association for Computational Linguistics*, pages 6086–6096. https://doi.org/10.18653/v1/P19-1612

Lemley, Mark A. and Bryan Casey. 2020. Fair learning. *Texas Law Review*. https://texaslawreview.org/fair-learning/

Lewis, Mike, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the Association for Computational Linguistics*, pages 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of Advances in Neural Information Processing Systems*.

Li, Margaret, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*.

Lin, Jimmy, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*. https://doi.org/10.1145/3404835.3463238

Lin, Yankai, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the Association for Computational Linguistics*, pages 1736–1745. https://doi.org/10.18653/v1/P18-1161

Lin, Yuri, Jean-Baptiste Michel, Erez Aiden Lieberman, Jon Orwant, William S. Brockman, and Slav Petrov. 2012. Syntactic annotations for the Google Books Ngram Corpus. In *Annual Meeting of the*

Association for Computational Linguistics, pages 169–174.
Liu, Jiachang, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for GPT-3? *arXiv preprint arXiv:2101.06804*. https://doi.org/10.18653/v1/2022.deelio-1.10
Liu, Yinhan, 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. *arXiv preprint arXiv:1907.11692*.
Lo, Kyle, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the Association for Computational Linguistics*, pages 4969–4983. https://doi.org/10.18653/v1/2020.acl-main.447
Lockman, Abe and David Klappholz. 1983. The control of inferencing in natural language understanding. In *Computational Linguistics*, 9:59–70. https://doi.org/10.1016/B978-0-08-030253-9.50010-X
Lu, Yao, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*. https://doi.org/10.18653/v1/2022.acl-long.556
Lyu, Xinxi, Sewon Min, Iz Beltagy, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Z-ICL: Zero-shot in-context learning with pseudo-demonstrations. In *Proceedings of the Association for Computational Linguistics*, pages 2304–2317. https://doi.org/10.18653/v1/2023.acl-long.129
Mackenzie, Joel, Rodger Benham, Matthias Petri, Johanne R. Trippas, J. Shane Culpepper, and Alistair Moffat. 2020. CC-News-En: A large English news corpus. In *Proceedings of the ACM International Conference on Information and Knowledge Management*. https://doi.org/10.1145/3340531.3412762
Madaan, Aman and Amir Yazdanbakhsh. 2022. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*.
Malaviya, Chaitanya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2023. ExpertQA: Expert-curated questions and attributed answers. https://doi.org/10.18653/v1/2024.naacl-long.167
Mallen, Alex, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. In *Proceedings of the Association for Computational Linguistics*, pages 9802–9822. https://doi.org/10.18653/v1/2023.acl-long.546
Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press. https://doi.org/10.1017/CBO9780511809071
Mao, Yuning, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2020. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the Association for Computational Linguistics*, pages 4089–4100.
McAuley, Julian and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 165–172. https://doi.org/10.1145/2507157.2507163
McCarthy, John, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon. 2006. A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955. *AI Magazine*, 27(4):12–12.
Metz, Cade. 2022. Lawsuit takes aim at the way A.I. is built. *New York Times*, 23 November.
Micikevicius, Paulius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2017. Mixed precision training. In *Proceedings of the International Conference on Learning Representations*.
Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in Neural Information Processing Systems*.
Min, Sewon, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, Tom Kwiatkowski, and more. 2021a. NeurIPS 2020

EfficientQA competition: Systems, analyses and lessons learned. In *Machine Learning Research (PMLR)*.

Min, Sewon, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 2851–2864. https://doi.org/10.18653/v1/D19-1284

Min, Sewon, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Min, Sewon, Suchin Gururangan, Eric Wallace, Hannaneh Hajishirzi, Noah A. Smith, and Luke Zettlemoyer. 2024. SILO language models: Isolating legal risk in a nonparametric datastore. In *Proceedings of the International Conference on Learning Representations*.

Min, Sewon, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023a. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 12076–12100. https://doi.org/10.18653/v1/2023.emnlp-main.741

Min, Sewon, Kenton Lee, Ming-Wei Chang, Kristina Toutanova, and Hannaneh Hajishirzi. 2021b. Joint passage ranking for diverse multi-answer retrieval. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 6997–7008. https://doi.org/10.18653/v1/2021.emnlp-main.560

Min, Sewon, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022a. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the Association for Computational Linguistics*, pages 5316–5330. https://doi.org/10.18653/v1/2022.acl-long.365

Min, Sewon, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022b. MetaICL: Learning to learn in context. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2791–2809. https://doi.org/10.18653/v1/2022.naacl-main.201

Min, Sewon, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022c. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of Empirical Methods in Natural Language Processing*, pages 11048–11064. https://doi.org/10.18653/v1/2022.emnlp-main.759

Min, Sewon, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 5783–5797. https://doi.org/10.18653/v1/2020.emnlp-main.466

Min, Sewon, Weijia Shi, Mike Lewis, Xilun Chen, Wen-tau Yih, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2023b. Nonparametric masked language modeling. In *Findings of ACL*, pages 2097–2118. https://doi.org/10.18653/v1/2023.findings-acl.132

Mishra, Swaroop, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2021. Reframing instructional prompts to GPTK's language. *arXiv preprint arXiv:2109.07830*. https://doi.org/10.18653/v1/2022.findings-acl.50

Mishra, Swaroop, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the Association for Computational Linguistics*, pages 3470–3487. https://doi.org/10.18653/v1/2022.acl-long.244

Morin, Frederic and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *Proceedings of the International Workshop on Artificial Intelligence and Statistics*.

Muennighoff, Niklas, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling data-constrained language models. *arXiv preprint arXiv:2305.16264*.

Neelakantan, Arvind, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas A. Tezak, Jong Wook Kim, Chris Hallacy, et al. 2022. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*.

Ni, Jianmo, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith B. Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of Empirical*

Methods in Natural Language Processing, pages 9844–9855. https://doi.org/10.18653/v1/2022.emnlp-main.669

Nie, Yixin, Songhe Wang, and Mohit Bansal. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 2553–2566. https://doi.org/10.18653/v1/D19-1258

Nogueira, Rodrigo and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *ArXiv*, abs/1901.04085.

Norouzi, Mohammad, Ali Punjani, and David J. Fleet. 2012. Fast search in hamming space with multi-index hashing. In *Computer Vision and Pattern Recognition*, pages 3108–3115. https://doi.org/10.1109/CVPR.2012.6248043

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Palen-Michel, Chester, June Kim, and Constantine Lignos. 2022. Multilingual open text release 1: Public domain news in 44 languages. In *Proceedings of the Language Resources and Evaluation Conference*.

Pan, Jane, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of ACL*, pages 8298–8319. https://doi.org/10.18653/v1/2023.findings-acl.527

Pang, Bo and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the Association for Computational Linguistics*, pages 271–278. https://doi.org/10.3115/1218955.1218990

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Proceedings of Advances in Neural Information Processing Systems*.

Patel, Ajay, Bryan Li, Mohammad Sadegh Rasooli, Noah Constant, Colin Raffel, and Chris Callison-Burch. 2022. Bidirectional language models are also few-shot learners. *arXiv preprint arXiv:2209.14500*.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. https://doi.org/10.3115/v1/D14-1162

Perez, Ethan, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Proceedings of Advances in Neural Information Processing Systems*.

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2227–2237. https://doi.org/10.18653/v1/N18-1202

Petroni, Fabio, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of Empirical Methods in Natural Language Processing*, pages 2463–2473. https://doi.org/10.18653/v1/D19-1250

Pfeiffer, Jonas, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the curse of multilinguality by pre-training modular transformers. In *Conference of the North American Chapter of the Association for Computational Linguistics*. https://doi.org/10.18653/v1/2022.naacl-main.255

Pfeiffer, Jonas, Ivan Vuli?, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673. https://doi.org/10.18653/v1/2020.emnlp-main.617

Press, Ofir, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of EMNLP*, pages 5687–5711. https://doi.org/10.18653/v1/2023.findings-emnlp.378

Project Gutenberg. Project Gutenberg. www.gutenberg.org

Presser, Shawn. 2020. Books3 corpus.

Project, Caselaw Access. 2018. Caselaw access project.

Radford, Alec, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Rae, Jack W., Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:5485–5551.

Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 2383–2392. https://doi.org/10.18653/v1/D16-1264

Ram, Ori, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. In *Proceedings of the Association for Computational Linguistics*, pages 3066–3079. https://doi.org/10.18653/v1/2021.acl-long.239

Ram, Ori, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331. https://doi.org/10.1162/tacl_a_00605

Ram, Ori, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. Learning to retrieve passages without supervision. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2687–2700. https://doi.org/10.18653/v1/2022.naacl-main.193

Razeghi, Yasaman, Robert L. Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*. https://doi.org/10.18653/v1/2022.findings-emnlp.59

Reynolds, Laria and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3411763.3451760

Roberts, Adam, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of Empirical Methods in Natural Language Processing*, pages 5418–5426. https://doi.org/10.18653/v1/2020.emnlp-main.437

Robertson, Stephen, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*. https://doi.org/10.1561/1500000019

Rombach, Robin, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR52688.2022.01042

Rong, Frieda. 2021. Extrapolating to unnatural language processing with GPT-3's in-context learning: The good, the bad, and the mysterious. https://ai.stanford.edu/blog/in-context-learning

Rubin, Ohad, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*. https://doi.org/10.18653/v1/2022.naacl-main.191

Ruder, Sebastian. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Sanh, Victor, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *Proceedings of the International Conference on Learning Representations*.

Santhanam, Keshav, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2021. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. *arXiv preprint arXiv:2112.01488*. https://doi.org/10.18653/v1/2022.naacl-main.272

Saxton, David, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *Proceedings of the International Conference on Learning Representations*.

Scao, Teven Le, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel

Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. BLOOM: A 176B-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Schaeffer, Rylan, Kateryna Pistunova, Samar Khanna, Sarthak Consul, and Sanmi Koyejo. 2023. Invalid logic, equivalent gains: The bizarreness of reasoning in language model prompting. In *ICML 2023 Workshop: Knowledge and Logical Reasoning in the Era of Data-driven Learning*.

Schick, Timo, Jane Dwivedi-Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. 2022. PEER: A collaborative language model. *arXiv preprint arXiv:2208.11663*.

Schick, Timo and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2339–2352. https://doi.org/10.18653/v1/2021.naacl-main.185

Seo, Minjoon, Tom Kwiatkowski, Ankur P. Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2018. Phrase-indexed question answering: A new challenge for scalable document comprehension. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 559–564. https://doi.org/10.18653/v1/D18-1052

Seo, Minjoon, Jinhyuk Lee, Tom Kwiatkowski, Ankur P. Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the Association for Computational Linguistics*, pages 4430–4441. https://doi.org/10.18653/v1/P19-1436

She, Jingyuan S., Christopher Potts, Samuel R. Bowman, and Atticus Geiger. 2023. ScoNe: Benchmarking negation reasoning in language models with fine-tuning and in-context learning. In *Proceedings of the Association for Computational Linguistics*, pages 1803–1821. https://doi.org/10.18653/v1/2023.acl-short.154

Shi, Weijia, Julian Michael, Suchin Gururangan, and Luke Zettlemoyer. 2022. Nearest neighbor zero-shot inference. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 3254–3265. https://doi.org/10.18653/v1/2022.emnlp-main.214

Shi, Weijia, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Xi Victoria Lin, Noah A. Smith, Luke Zettlemoyer, Wen tau Yih, and Mike Lewis. 2024a. In-context pretraining: Language modeling beyond document boundaries. In *Proceedings of the International Conference on Learning Representations*.

Shi, Weijia, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024b. REPLUG: Retrieval-augmented black-box language models. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 8371–8384. https://doi.org/10.18653/v1/2024.naacl-long.463

Siegel, Sidney. 1957. Nonparametric statistics. *The American Statistician*, 11:13–19. https://doi.org/10.1080/00031305.1957.10501091

Silverman et al. v. OpenAI. 2023. Case 3:23-cv-03416, n.d. Cal.

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 1631–1642. https://doi.org/10.18653/v1/D13-1170

Soldaini, Luca and Kyle Lo. 2023. peS2o (pretraining efficiently on S2ORC) dataset. Technical report, Allen Institute for AI. ODC-By. https://github.com/allenai/pes2o

Song, Yiping, Rui Yan, Cheng-Te Li, Jian-Yun Nie, Ming Zhang, and Dongyan Zhao. 2018. An ensemble of retrieval-based and generation-based human-computer conversation systems. In *International Joint Conference on Artificial Intelligence*. https://doi.org/10.24963/ijcai.2018/609

Sun, Simeng, Dhawal Gupta, and Mohit Iyyer. 2023. Exploring the impact of low-rank adaptation on the performance, efficiency, and regularization of RLHF. *arXiv preprint arXiv:2309.09055*.

Teh, Yee Whye. 2006. A Bayesian interpretation of interpolated Kneser-Ney. NUS School of Computing Technical Report tra2/06. National University of Singapore. https://www.stats.ox.ac.uk/∼teh/research/compling/hpylm.pdf

Tian, Katherine, Eric Mitchell, Huaxiu Yao, Christopher D. Manning, and Chelsea Finn. 2023. Fine-tuning language models

for factuality. *arXiv preprint arXiv:2311.08401*.

Tian, Zhiliang, Wei Bi, Xiaopeng Li, and Nevin L. Zhang. 2019. Learning to abstract for memory-augmented conversational response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3816–3825. https://doi.org/10.18653/v1/P19-1371

Together. 2023. RedPajama: An open source recipe to reproduce LLaMA training dataset. https://www.together.ai/blog/redpajama

Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Tremblay et al. v. OpenAI, Inc. 2023. Case 3:23-cv-03223 , N.D. Cal.

Turing, Alan M. 1980. Computing machinery and intelligence. *Creative Computing*, 6(1):44–53.

Turpin, Miles, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *arXiv preprint arXiv:2305.04388*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of Advances in Neural Information Processing Systems*.

Vilalta, Ricardo and Youssef Drissi. 2002. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18:77–95. https://doi.org/10.1023/A:1019956318069

Vincent, James. 2023. Getty images sues AI art generator stable diffusion in the US for copyright infringement.

Voorhees, Ellen M. 1999. The TREC-8 question answering track report. In *TREC*, volume 99, pages 77–82. https://doi.org/10.6028/NIST.SP.500-246.qa-overview

Voorhees, Ellen M. 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*, pages 200–207. https://doi.org/10.1145/345508.345577

Vyas, Nikhil, Sham Kakade, and Boaz Barak. 2023. Provable copyright protection for generative models. In *Proceedings of the International Conference of Machine Learning*, pages 35277–35299.

Wang, Alex, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of Advances in Neural Information Processing Systems*.

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355. https://doi.org/10.18653/v1/W18-5446

Wang, Ben and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 billion parameter autoregressive language model. https://github.com/kingoflolz/mesh-transformer-jax

Wang, Boshi, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023a. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *Proceedings of the Association for Computational Linguistics*, pages 2717–2739. https://doi.org/10.18653/v1/2023.acl-long.153

Wang, Shuohang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018b. R3: Reinforced ranker-reader for open-domain question answering. In *Association for the Advancement of Artificial Intelligence*. https://doi.org/10.1609/aaai.v32i1.12053

Wang, Wei, Vincent W. Zheng, Han Yu, and Chunyan Miao. 2019b. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*. https://doi.org/10.1145/3293318

Wang, Yizhong, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP

tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109. https://doi.org/10.18653/v1/2022.emnlp-main.340

Wang, Zhiguo, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019c. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 5878–5882. https://doi.org/10.18653/v1/D19-1599

Wang, Zhiruo, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, and Graham Neubig. 2023b. Learning to filter context for retrieval-augmented generation. *arXiv preprint arXiv:2311.08377*.

Webson, Albert and Ellie Pavlick. 2022. Do prompt-based models really understand the meaning of their prompts? In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2300–2344. https://doi.org/10.18653/v1/2022.naacl-main.167

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

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Wei, Jerry, Le Hou, Andrew Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, et al. 2023a. Symbol tuning improves in-context learning in language models. *arXiv preprint arXiv:2305.08298*. https://doi.org/10.18653/v1/2023.emnlp-main.61

Wei, Jerry W., Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. 2023b. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*.

Wies, Noam, Yoav Levine, and Amnon Shashua. 2023. The learnability of in-context learning. *arXiv preprint arXiv:2303.07895*.

Winograd, Terry. 1971. Procedures as a representation for data in a computer program for understanding natural language. https://dspace.mit.edu/handle/1721.1/7095

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Wolfson, Tomer, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, pages 183–198. https://doi.org/10.1162/tacl_a_00309

Wu, Ledell, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 6397–6407. https://doi.org/10.18653/v1/2020.emnlp-main.519

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

Xiong, Lee, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021a. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the International Conference on Learning Representations*.

Xiong, Wenhan, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oğuz. 2021b. Answering complex open-domain questions with multi-hop dense retrieval. In *Proceedings of the International Conference on Learning Representations*.

Xu, Frank F., Junxian He, Graham Neubig, and Vincent Josua Hellendoorn. 2022. Capturing structural locality in

non-parametric language models. In *Proceedings of the International Conference on Learning Representations*.

Xue, Linting, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 483–498. https://doi.org/10.18653/v1/2021.naacl-main.41

Yamada, Ikuya, Akari Asai, and Hannaneh Hajishirzi. 2021. Efficient passage retrieval with hashing for open-domain question answering. In *Proceedings of the Association for Computational Linguistics*, pages 979–986. https://doi.org/10.18653/v1/2021.acl-short.123

Yang, Wei, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019a. End-to-end open-domain question answering with BERTserini. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 72–77. https://doi.org/10.18653/v1/N19-4013

Yang, Wei, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019b. Data augmentation for BERT fine-tuning in open-domain question answering. *ArXiv*, abs/1904.06652. https://doi.org/10.18653/v1/N19-4013

Yang, Zhilin, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2018. Breaking the softmax bottleneck: A high-rank RNN language model. In *Proceedings of the International Conference on Learning Representations*.

Ye, Qinyuan, Bill Yuchen Lin, and Xiang Ren. 2021. CrossFit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 7163–7189. https://doi.org/10.18653/v1/2021.emnlp-main.572

Ye, Seonghyeon, Hyeonbin Hwang, Sohee Yang, Hyeongu Yun, Yireun Kim, and Minjoon Seo. 2023a. In-context instruction learning. https://deepai.org/publication/in-context-instruction-learning

Ye, Seonghyeon, Doyoung Kim, Joel Jang, Joongbo Shin, and Minjoon Seo. 2023b. Guess the instruction! Flipped learning makes language models stronger zero-shot learners. In *Proceedings of the International Conference on Learning Representations*.

Ye, Seonghyeon, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2023c. FLASK: Fine-grained language model evaluation based on alignment skill sets. *arXiv preprint arXiv:2307.10928*.

Yih, Wen tau, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Conference on Computational Natural Language Learning*, pages 247–256.

Yogatama, Dani, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, pages 362–373. https://doi.org/10.1162/tacl_a_00371

Yu, Lili, Bowen Shi, Ram Pasunuru, and Benjamin Miller. 2023a. Scaling Autoregressive Multi-Modal Models: Pretraining and Instruction Tuning. *arXiv:2309.02591*

Yu, Xinyan Velocity, Sewon Min, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023b. CREPE: Open-domain question answering with false presuppositions. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 10457–10480. https://doi.org/10.18653/v1/2023.acl-long.583

Zhan, Jingtao, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, M. Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*. https://doi.org/10.1145/3404835.3462880

Zhang, Chiyuan, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2021. Counterfactual memorization in neural language models. *arXiv preprint arXiv:2112.12938*.

Zhang, Dawen, Pamela Finckenberg-Broman, Thong Hoang, Shidong Pan, Zhenchang Xing, Mark Staples, and Xiwei Xu. 2023a. Right to be forgotten in the era of large language models: Implications, challenges, and solutions. *arXiv preprint arXiv:2307.03941*. https://doi.org/10.1007/s43681-024-00573-9

Zhang, Leizhong, Qiong Yang, Ta Bao, Dave Vronay, and Xiaoou Tang. 2006. Imlooking: image-based face retrieval in online dating profile search. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*,

pages 1577–1582. https://doi.org/10.1145/1125451.1125739

Zhang, Susan, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhang, Xiang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of Advances in Neural Information Processing Systems*.

Zhang, Yuhui, Michihiro Yasunaga, Zhengping Zhou, Jeff Z. HaoChen, James Zou, Percy Liang, and Serena Yeung. 2023b. Beyond positive scaling: How negation impacts scaling trends of language models. In *Findings of ACL*, pages 7479–7498. https://doi.org/10.18653/v1/2023.findings-acl.472

Zhao, Jake and Kyunghyun Cho. 2018. Retrieval-augmented convolutional neural networks for improved robustness against adversarial examples. *arXiv preprint arXiv:1802.09502*. https://doi.org/10.1109/CVPR.2019.01183

Zhao, Tony Z., Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the International Conference of Machine Learning*.

Zhong, Ruiqi, 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 EMNLP*, pages 2856–2878. https://doi.org/10.18653/v1/2021.findings-emnlp.244

Zhong, Zexuan, Tao Lei, and Danqi Chen. 2022. Training language models with memory augmentation. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 5657–5673. https://doi.org/10.18653/v1/2022.emnlp-main.382

Zhu, Yukun, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Computer Vision and Pattern Recognition*. https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Zhu_Aligning_Books_and_ICCV_2015_paper.pdf, https://doi.org/10.1109/ICCV.2015.11