Towards More Factual Large Language Models: Parametric and Non-parametric Approaches

Zhengbao Jiang

June 15, 2024

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Graham Neubig (Chair)	Carnegie Mellon University
Jamie Callan	Carnegie Mellon University
William W. Cohen	Carnegie Mellon University & Google DeepMind
Scott Wen-tau Yih	FAIR at Meta

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Language and Information Technology.

Keywords: large language models, retrieval-augmented generation, prompting

Abstract

Large language models (LLMs) are increasingly important in assisting people to access information, ranging from simple factoid questions such as "where is the world's largest ice sheet located" to complex questions that require accessing real-time information and reasoning such as "plan a vacation in Miami". There are two paradigms to handle questions that require factual knowledge: parametric approaches that store knowledge within the parameters of LLMs and elicit this knowledge through prompting, and non-parametric approaches that offload knowledge retrieval to an external non-parametric datastore. In this dissertation, we aim to study, compare, and enhance the capacity of both paradigms.

Since LLMs have accumulated a large amount of knowledge within their parameters through pre-training on diverse corpora, they can directly generate answers when prompted with questions. In the first part of the dissertation, we focus on parametric approaches that utilize the factual knowledge contained in the parameters of LLMs. We first study methods to extract more knowledge by ensembling multiple predictions derived from diverse prompts. Then, we calibrate LLMs to make them trustworthy when answering questions that fall beyond their scope of knowledge. We find that after LLMs memorize documents perfectly to the extent of reproducing them verbatim, they still often fail to answer questions about them. To enhance the capacity of LLMs to absorb knowledge from documents, we propose pre-instruction-tuning that teaches them the task of question-answering before pre-training them on documents.

Parametric approaches offer a simple interface, but they suffer from hallucinations and lack access to real-time external information. In the second part of the dissertation, we focus on non-parametric approaches that augment LLMs with a non-parametric datastore, typically constructed from a document corpus and a retriever. The standard retrieval-augmented generation (RAG) pipeline consists of an embedding-based retriever and an LLM-based generator, which typically require separate training procedures and are often limited by the retriever's performance. We introduce an end-to-end solution that fuses retrieval and generation within a single transformer and directly uses the attention mechanism for retrieval purposes. To address complex questions demanding detailed responses, we introduce Active RAG, which dynamically and proactively retrieves information throughout the generation process. Finally, we conclude by comparing and reconciling both paradigms and providing insight into future directions.

Contents

1 Introduction **1**

1.1 Parametric Approaches 1
1.2 Non-parametric Approaches 4
1.3 Parametric or Non-parametric Approaches? 8

I Parametric Knowledge of Large Language Models **11**

2 Eliciting Factual Knowledge of LMs with Diverse Prompts **13**

2.1 Introduction 13
2.2 Knowledge Retrieval from LMs 15
2.3 Prompt Generation 16
2.4 Prompt Selection and Ensembling 17
2.5 Main Experiments 19
2.6 Related Work 24
2.7 Conclusion 24

3 Eliciting Factual Knowledge of Multilingual LMs **27**

3.1 Introduction 27
3.2 Retrieving Facts from LMs 29
3.3 Multilingual Multi-token Factual Retrieval Benchmark 30
3.4 Multi-token Decoding 32
3.5 X-FACTR Benchmark Performance 34
3.6 Improving Multilingual LM Retrieval 36
3.7 Conclusion 39

4 Calibrating LMs for Question Answering **41**

4.1 Introduction 41
4.2 LM-based Question Answering 43

4.3 Background on Calibration . . . 44
4.4 Calibrating LMs for Question Answering . . . 46
4.5 Experiments . . . 49
4.6 Related Work . . . 54
4.7 Conclusion . . . 54

5 Enhancing LLMs in Absorbing Knowledge through Pre-instruction-tuning 55
5.1 Introduction . . . 55
5.2 Building a Dataset to Study Continual Knowledge Acquisition . . . 57
5.3 Experimental Settings . . . 60
5.4 How Much Knowledge Can LLMs Absorb via Continued Pre-training Followed by Instruction-tuning? . . . 61
5.5 Improving LLMs in Absorbing Knowledge from Documents . . . 64
5.6 Related Work . . . 70
5.7 Conclusion . . . 71

II Non-parametric Retrieval-augmented Generation 73

6 Retrieval as Attention: Extending the Context of LMs to the Entire Corpus 75
6.1 Introduction . . . 75
6.2 Retrieval as Attention (ReAtt) . . . 78
6.3 Learning Retrieval as Attention . . . 80
6.4 In-domain Experiments . . . 83
6.5 Implementation Details of ReAtt . . . 84
6.6 Out-of-domain Generalization and Adaptation . . . 86
6.7 Conclusion . . . 89

7 Active Retrieval-augmented Generation 91
7.1 Introduction . . . 91
7.2 Retrieval Augmented Generation . . . 93
7.3 FLARE: Forward-Looking Active REtrieval Augmented Generation . . . 95
7.4 Multi-time Retrieval Baselines . . . 99
7.5 Experimental Setup . . . 101
7.6 Experimental Results . . . 103
7.7 Related Work . . . 107
7.8 Conclusion . . . 108

8 Conclusion **109**
8.1 Summary of Contributions 112
8.2 Future Directions 113

Bibliography **119**

Chapter 1

Introduction

Developing intelligent systems capable of interacting with people in natural language and answering questions with varying levels of knowledge has been a long-standing challenge in the field of natural language processing (NLP). Different from other language understanding and generation tasks such as text classification and machine translation, knowledge-intensive tasks require access to information that is not explicitly included in the input. A typical knowledge-intensive task is question answering (QA) which aims to provide an answer to a natural language question. Questions can range from simple factoid questions such as "where is the world's largest ice sheet located" [12, 95, 114] to complex questions that require aggregating multiple sources of information and reasoning such as "plan a vacation in Miami" [52, 100, 258].

Large language models (LLMs) have accumulated a large amount of knowledge through pre-training on diverse corpora, and they are increasingly important in assisting people in accessing information [21, 47, 57, 155, 170, 212]. LLM-based information-accessing systems can be classified into two categories based on the source of knowledge: parametric approaches that solely rely on knowledge encoded in the parameters of LLMs [159, 178] and non-parametric approaches that offload knowledge retrieval to an external non-parametric datastore [17, 66, 83, 124, 192]. In this section, we begin by providing an overview of both categories. Next, we present our contributions aimed at enhancing the capabilities of both paradigms. Finally, we wrap up by comparing the two paradigms and suggesting future directions.

1.1 Parametric Approaches

LLMs are pre-trained on diverse corpora such as Wikipedia, CommonCrawl, and StackExchange [170, 212]. Recent studies have demonstrated that LLMs with billions of parameters have the capacity to memorize parts of the training data [23, 24]. Given that these corpora consist of numerous information-rich documents, such as Wikipedia articles on specific top-

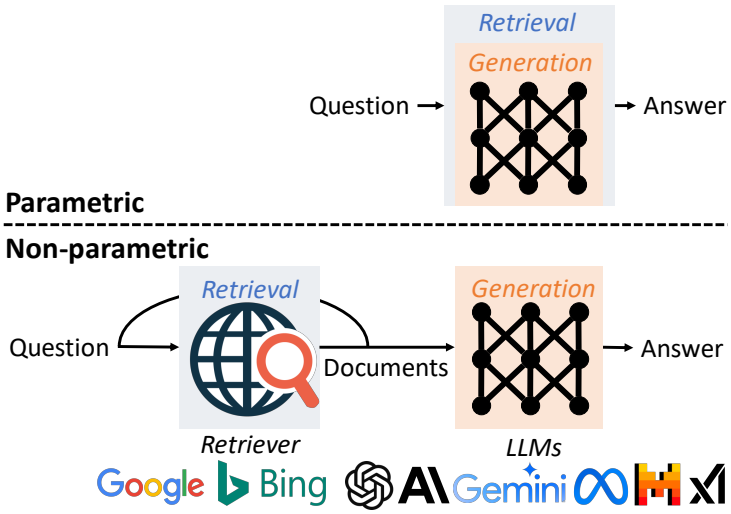


Figure 1.1: Illustration comparing parametric and non-parametric approaches for answering knowledge-intensive questions.

ics and questions paired with corresponding answers from StackExchange, pre-trained LLMs can be utilized to directly respond to questions requiring factual knowledge covered by the pre-training corpora when prompted appropriately [21, 159, 178, 260]. For example, on the Natural Questions dataset, a widely used open-domain factoid QA benchmark built based on Wikipedia [114], the closed-book answer generation accuracy of PaLM-2 is 37.5% [33]; on the MMLU dataset, a popular benchmark testing both world knowledge and problem-solving ability across many subjects such as STEM and social science [71], the closed-book multi-choice accuracy of GPT-4 is 86.4% [153]. Many works have demonstrated a correlation between the training scales (e.g., model sizes and training corpora) and the performance on downstream benchmarks [188, 232]. It is plausible that future LLMs could answer the majority, if not all, questions provided the necessary knowledge is included in the training data [4, 98].

1.1.1 Parametric Knowledge Storage, Update, and Extraction

While LLMs have been widely used to answer knowledge-intensive questions, the underlying mechanism remains poorly understood. In order to use LLMs as "knowledge bases", LLMs must possess two abilities: the ability to store knowledge in their parameters during pre-training and the ability to extract this knowledge conditioning on questions at inference time.

Knowledge storage Many works have studied the amount of knowledge that LLMs can memorize via the standard objective of predicting the next token given the preceding context. Some works quantify memorization of real corpora such as the Pile and the Wikipedia dataset

[24, 53, 210], while others quantify memorization of synthetic datasets containing knowledge tuples of controlled complexity in order to derive a concrete scaling law [4, 260]. The conclusion is that (1) larger models have stronger memorization capacity, (2) repeated examples or examples with diverse expressions are more easily memorized, and (3) different types of information (e.g., part-of-speech tags) differ in how easy they are to memorize.

Knowledge update As world knowledge evolves, it is crucial to continually update the knowledge in the parameters of LLMs. Existing methods include continued pre-training that continually trains LLMs on new corpora using the standard next token prediction objective [65, 85] and model editing that aims to make minimal changes to model weights to edit certain knowledge [22, 145]. Some studies aim to pinpoint the transformer components responsible for retaining factual knowledge and point out that specific neurons and layers like feed-forward layers play a more significant role [40, 59]. However, existing model editing methods are not scalable, and the underlying mechanism for knowledge storage remains largely obscure.

Knowledge extraction The way knowledge is presented in the original documents and how it is accessed through questions can be different. For example, the Wikipedia article about Oppenheimer is "Oppenheimer is a 2023 epic biographical thriller film ... Editing was completed by Jennifer Lame." A test-time question "who is the editor of Oppenheimer" refers to the knowledge found in the article, but the phrasing is different. In order to answer this question, LLMs have to store the knowledge from the document in a way that is extractable when prompted with questions, which requires generalization beyond rote memorization. A typical method to enhance knowledge extraction is post-training instruction-tuning, which fine-tunes LLMs using question-answer pairs [34, 184, 231]. However, Zhu and Li [260] demonstrated in a controlled experiment using a randomly initialized transformer that fine-tuning after pre-training fails to elicit knowledge from the parameters. They found that each document must be paraphrased in multiple ways during pre-training to make the memorized knowledge extractable.

1.1.2 Summary of Contributions

Considering the development and limitations of existing parametric approaches mentioned earlier, our focus is on extracting more knowledge from pre-trained LLMs through better prompting strategies and improving LLMs' ability to absorb knowledge from pre-training documents. Our detailed contributions are as follows.

Eliciting factual knowledge of LMs with diverse prompts. (chapter 2, chapter 3) LMs are very sensitive to prompts [49, 159]. An inappropriate prompt might fail to extract the knowl-

edge that LMs do know. For instance, when prompted with "DirectX is developed by _", the BERT model returns "Intel", whereas when prompted with "DirectX is created by _", it returns "Microsoft". In Jiang et al. [90], we systematically investigate this phenomenon. We suggest that a single prompt only sets a lower bound on what LMs know, and propose to ensemble the results from multiple prompts either mined or paraphrased to elicit more knowledge from LMs. We also study knowledge extraction from multilingual LMs in Jiang et al. [89], where we note that memorizing knowledge is more difficult for low-resource languages due to data imbalances. To address this, we propose augmenting training data through code-switching of entity mentions to enhance knowledge memorization in multiple languages.

Calibrating LMs for question answering. (chapter 4) As LMs still struggle to provide accurate answers in many scenarios, it is essential for them to comprehend and clarify the limits of their knowledge so that users can determine when to reject predictions from LMs. In Jiang et al. [91], we examine this problem from the perspective of calibration and propose post-hoc and augmentation methods to make prediction probabilities of LMs more reliable.

Enhancing LLMs in absorbing knowledge through pre-instruction-tuning. (chapter 5) We find that after LLMs memorize the pre-training documents perfectly to the extent of reproducing them verbatim (i.e., perplexity is 1.0), they still often fail to answer questions about these documents, even after instruction-tuning on QA pairs. We refer to this as "perplexity curse". It demonstrates that post-hoc instruction-tuning cannot fully uncover the knowledge presented in the pre-training documents. Inspired by the fact that questions are generally straightforward while documents tend to be more complex, we propose pre-instruction-tuning which fine-tunes LLMs on QA pairs to learn the task of question-answering before pre-training. It enhances the ability of LLMs to absorb knowledge from new documents.

1.2 Non-parametric Approaches

As the size of LLMs and training corpora increases, more knowledge is encoded within their parameters. Parametric approaches also provide a simple interface during inference. However, they have fundamental issues when handling questions that either fall within or outside the scope of the pre-training data:

- **For knowledge included in the pre-training data**: LLMs are prone to hallucinations, particularly with long-tail knowledge mentioned rarely in the data [98, 138]. While LLMs can generate documents to provide additional evidence supporting an answer, ensuring the accuracy of the generated evidence is challenging, particularly in critical fields such

as medicine [198, 245].

- **For knowledge not included in the pre-training data**: LLMs are fundamentally not capable of handling questions that demand knowledge beyond the scope of their pre-training data. While it is feasible to regularly update LLMs to integrate new knowledge, they cannot address questions that require access to the external world, such as queries for real-time information or those necessitating actions in the real world [44, 100, 221, 258].

Given these limitations of parametric approaches, non-parametric approaches that augment LLMs with a non-parametric datastore constructed from a document corpus and an external retriever offer distinct advantages. The standard retrieval-augmented generation (RAG) pipeline consists of an embedding-based retriever and an LLM-based generator, where the retriever searches the corpus to find documents relevant to the question, and the generator generates the answer conditioning on both the question and documents.

1.2.1 Retrieval, Long Context, and Interactive Retrieval and Action

To develop non-parametric approaches, we need to determine two key components: the retriever and how to integrate the retrieved information to help answer generation.

Retrieval The most important component in an RAG system is retrievers, which return documents similar to the question to provide additional evidence for generating the answer. The definition of similarity between questions and documents varies depending on downstream tasks, but it primarily involves lexical and semantic overlap. Traditional retrievers compute similarities based on lexical overlap and heuristically designed metrics, such as TF-IDF and BM25 [179]. To capture complex semantic similarities, modern retrievers are based on embedding models that encode questions and documents into high-dimensional dense vectors and compute similarity using the inner product. Embedding models are usually initialized with pre-trained LMs such as BERT [47] and fine-tuned using either supervised data of relevant question-document pairs [99] or unsupervised contrastive learning objectives [82]. Embedding-based retrievers can be classified into two categories: single-vector models, which encode the entire document or question into a single vector [82, 99, 130], and multi-vector models, which utilize all token vectors for more complex matching [103]. To address various scenarios with different similarity definitions, embedding-based retrievers are trained by including instructions with the question to explicitly describe information needs [7, 197].

Long-context LLMs While advances in base LMs have led to improved embedding-based retrievers, computing similarity based on fixed-dimensional vectors is fundamentally limited

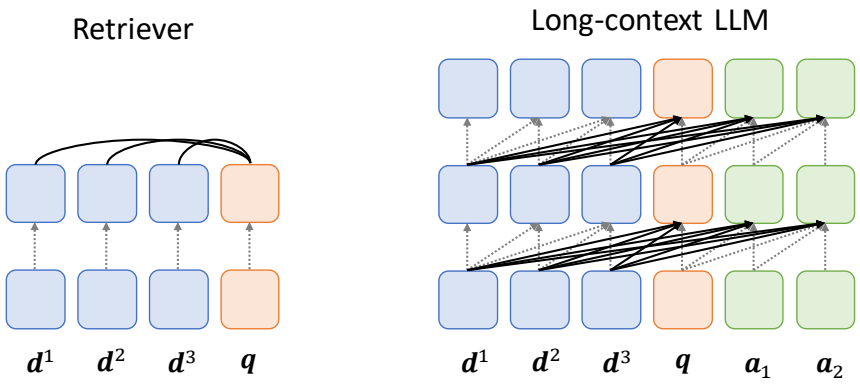


Figure 1.2: A conceptual comparison between an embedding-based retriever (left) and a long-context LLM using attention for retrieval (right). $\boldsymbol{d}$, $\boldsymbol{q}$, and $\boldsymbol{a}$ denote documents, questions, and answers respectively. Arrows denote attention, while lines without arrows signify the inner product. For simplicity, we only show one token for documents and the question, two tokens for the answer, and one attention head. We use a two-layer transformer for retrievers and a three-layer transformer for LLMs to demonstrate that retrievers are typically smaller. **Solid arrows/lines** indicate where information retrieval takes place. Long-context LLMs perform multiple steps of retrieval across multiple heads, layers, and tokens in the question and answer.

in expressiveness. This limitation poses a major bottleneck for RAG systems to handle complex questions that necessitate multiple steps of information seeking and reasoning. The self-attention mechanism in transformers calculates token similarity in a multi-head and multi-layer manner [219], which has proven highly effective in handling information within the context. If self-attention can efficiently process a long context containing numerous documents, it becomes essentially a multi-head and multi-layer "retriever" and we can leverage it to handle the demanding tasks of fine-grained information matching and reasoning. Figure 1.2 compares embedding-based retrievers and long-context LLMs using attention for retrieval purposes. Existing approaches for handling long context include efficient dense self-attention over a long sequence parallelized across multiple GPUs [41, 133] with appropriate positional encoding [28, 240], as well as sparse attention with approximate nearest neighbor search [101, 237, 247, 255].

Interactive retrieval and action To tackle complex questions demanding multiple steps of information retrieval and action, it's crucial for LLMs to break down the question into stages and utilize retrievers as needed to search for relevant information [164, 216, 233], while also taking actions to achieve the objective [194, 244]. For example, "plan for my vacation in Miami" requires searching for information about hotels, flights, and attractions, creating an itinerary

based on the collected information, and taking action on websites such as booking accommodations. Because retrieved information often contains noise and certain details may not be directly visible on web pages (such as hotel availabilities), it's crucial for LLMs to iteratively refine queries based on the returned results and interact with the web page by clicking or selecting to complete the task [147, 194, 244]. It is challenging to collect data to train LLMs for these complicated tasks due to the large exploration space and the high annotation cost. How to effectively explore the space with reinforcement learning [259] and lower the annotation cost potentially with the help of scalable oversight [20] are active research directions.

1.2.2 Summary of Contributions

Considering the development and limitations of existing non-parametric approaches mentioned earlier, we extend the context of LLMs to the entire corpus through retrieval as sparse attention and propose active retrieval-augmented generation that interleaves retrieval and generation to handle complicated questions. Our detailed contributions are as follows.

Retrieval as attention: extending the context of LMs to the entire corpus. (chapter 6) Existing RAG systems usually use a separate retriever and a generator which are trained with different recipes and hard to optimize end-to-end [66, 80, 83, 124]. In Jiang et al. [92], we propose that the self-attention mechanism in transformers can be directly employed for retrieval purposes by extending it to the entire corpus containing many documents. This approach enables more complex information matching and effective end-to-end training. Specifically, we leverage the attention between all documents and questions in a single layer of the encoder of the T5 model [170] as retrieval signals, and optimize the system using QA data in an end-to-end way. Our method, denoted as retrieval as attention (ReAtt), achieves competitive performance in both retrieval and downstream tasks, making it an adaptable end-to-end solution for knowledge-intensive tasks.

Active retrieval-augmented generation. (chapter 7) To handle complicated questions involving the generation of long texts, it is essential to continually gather information throughout the generation process. In Jiang et al. [93], we present a generalized view of active retrieval-augmented generation, which involves methods that actively determine when and what to retrieve during the generation process. Specifically, we propose forward-looking active retrieval-augmented generation (FLARE), a generic method that iteratively uses the prediction of the upcoming sentence to anticipate future content, which is then utilized as a query to retrieve relevant documents to regenerate the sentence if it contains low-confidence tokens.

Dimensions	Parametric	Non-parametric
Training time		
Efficiency (time & memory cost)	?	?
Simplicity (model architecture & objective)	✓	✗
Scalability (model size & corpus size)	✓	✗
Integration of new knowledge without training	✗	✓
Testing time		
Efficiency (latency & memory footprint)	?	?
Simple interface (end-to-end vs pipeline)	✓	✗
Less hallucination	✗	✓
Access to the external world (real-time info & action)	✗	✓
Verifiability and attribution	✗	✓
Domains adaptability & privacy-preserving	✗	✓

Table 1.1: Comparisons between parametric and non-parametric approaches for both training and testing time. ? indicates comparisons that depend on the size of the LLM and retriever and the the size of the datastore.

1.3 Parametric or Non-parametric Approaches?

We conducted a thorough comparison of these two paradigms across several dimensions for both training and testing time in Table 1.1. Parametric approaches are simpler in both training and testing because they only involve one transformer component with a standard next-token-prediction objective, whereas non-parametric approaches require a pipeline design of retrievers and LLMs involving complex training objectives. As a result, scaling up LLMs is easier than scaling up retrieval-augmented LLMs. Non-parametric approaches, on the other hand, experience less hallucination, provide access to the external world, and are particularly useful in specialized and private domains. We expect that future systems will combine elements of both approaches. As LLMs become capable of encoding increasing amounts of knowledge within their parameters, non-parametric components remain crucial for handling long-tail knowledge, as well as applications that demand interaction with the external environment. Future directions include:

- **Robust parametric knowledge in LLMs**: pre-training using data augmented with references and advanced model architecture with stronger memory capacity such as mixture-of-expert models.
- **Continual knowledge update and alignment**: techniques for continuously updating

knowledge in LLMs with minimal forgetting, along with effective alignment methods to adjust their scope of knowledge.

- **Transformer architectures with infinite context length**: to manage web-scale information effectively, it is essential to develop transformer architectures with an efficient sparse attention mechanism and key-value caching system.
- **LLM-based agents**: effective data collection and training algorithms to allow LLMs to utilize retrievers or other tools to interact with the external world to perform complex tasks using reinforcement learning and scalable oversight.

Part I

Parametric Knowledge of Large Language Models

Chapter 2

Eliciting Factual Knowledge of LMs with Diverse Prompts

In this chapter, we study how to estimate the amount of factual knowledge contained in the parameters of LMs. Recent works examined the knowledge contained in LMs by having LMs fill in the blanks of prompts. These prompts are usually manually created, and possibly sub-optimal; another prompt may result in more accurate predictions. In this chapter, we focus on masked LMs such as BERT and attempt to more accurately estimate the knowledge contained in the parameters by automatically discovering better prompts. This work is presented in:

- Zhengbao Jiang*, Frank F. Xu*, Jun Araki, Graham Neubig. How Can We Know What Language Models Know? Transactions of the Association for Computational Linguistics 8 (2020): 423-438.[1]

We also released the code and the resulting LM Prompt And Query Archive (LPAQA) at:

- https://github.com/jzbjyb/LPAQA

2.1 Introduction

It is becoming apparent that LMs[2] *themselves* can be used as a tool for text understanding by formulating queries in natural language and either generating textual answers directly [142, 168], or assessing multiple choices and picking the most likely one [171, 262]. For example, LMs have been used to answer factoid questions [168], answer common sense queries [186, 214], or extract factual knowledge about relations between entities [9, 159]. Regardless of the end task,

[1] Zhengbao Jiang proposed the idea, conducted main experiments, and wrote the draft. Frank F. Xu refined the idea, proposed ranking-based methods, conducted analysis and ablations, and revised the draft.

[2] We mainly focus on bi-directional masked LMs such as BERT [47], which do not directly define a probability distribution over text. Nonetheless, we call them LMs for simplicity.

Prompts	
manual	DirectX *is developed by* y_{man}
mined	y_{mine} *released the* DirectX
paraphrased	DirectX *is created by* y_{para}

Top 5 predictions and log probabilities

	y_{man}		y_{mine}		y_{para}	
1	Intel	-1.06	Microsoft	-1.77	Microsoft	-2.23
2	Microsoft	-2.21	They	-2.43	Intel	-2.30
3	IBM	-2.76	It	-2.80	default	-2.96
4	Google	-3.40	Sega	-3.01	Apple	-3.44
5	Nokia	-3.58	Sony	-3.19	Google	-3.45

Figure 2.1: Top-5 predictions and their log probabilities using different prompts (manual, mined, and paraphrased) to query BERT. The correct answer is underlined.

the knowledge contained in LMs is probed by providing a prompt, and letting the LM either generate the continuation of a prefix (e.g. "*Barack Obama was born in* _"), or predict missing words in a cloze-style template (e.g., "*Barack Obama is a* _ *by profession*").

However, while this paradigm has been used to achieve a number of intriguing results regarding the knowledge expressed by LMs, they usually rely on prompts that were manually created based on the intuition of the experimenter. These manually created prompts (e.g. "*Barack Obama was born in* _") might be sub-optimal because LMs might have learned target knowledge from substantially different contexts (e.g. "*The birth place of Barack Obama is Honolulu, Hawaii.*") during their training. Thus it is quite possible that a fact that the LM *does* know cannot be retrieved due to the prompts not being effective queries for the fact. Thus, existing results are simply a *lower bound* on the extent of knowledge contained in LMs, and in fact, LMs may be even more knowledgeable than these initial results indicate. In this chapter, we ask the question: "How can we tighten this lower bound and get a more accurate estimate of the knowledge contained in LMs?" This is interesting both scientifically, as a probe of the knowledge that LMs contain, and from an engineering perspective, as it will result in higher recall when using LMs as part of a knowledge extraction system.

In particular, we focus on the setting of Petroni et al. [159] who examine extracting knowledge regarding the relations between entities (definitions in section 2.2). We propose two automatic methods to systematically improve the breadth and quality of the prompts used to query the existence of a relation (section 2.3). Specifically, as shown in Figure 2.1, these are *mining-based* methods inspired by previous relation extraction methods [175], and *paraphrasing-based* methods that take a seed prompt (either manually created or automatically mined), and paraphrase it into several other semantically similar expressions. Further, because different prompts

may work better when querying for different subject-object pairs, we also investigate lightweight ensemble methods to combine the answers from different prompts together (section 2.4).

We experiment on the LAMA benchmark [159], which is an English-language benchmark devised to test the ability of LMs to retrieve relations between entities (section 2.5). We first demonstrate that improved prompts significantly improve accuracy on this task, with the one-best prompt extracted by our method raising accuracy from 31.1% to 34.1% on BERT-base [47], with similar gains being obtained with BERT-large as well. We further demonstrate that using a diversity of prompts through ensembling further improves accuracy to 39.6%. We perform extensive analysis and ablations, gleaning insights both about how to best query the knowledge stored in LMs and about potential directions for incorporating knowledge into LMs themselves. Finally, we have released the resulting LM Prompt And Query Archive (LPAQA) to facilitate future experiments on probing knowledge contained in LMs.

2.2 Knowledge Retrieval from LMs

Retrieving factual knowledge from LMs is quite different from querying standard declarative knowledge bases (KB). In standard KBs, users formulate their information needs as a structured query defined by the KB schema and query language. For example, `SELECT ?y WHERE {wd:Q76 wdt:P19 ?y}` is a SPARQL query to search the birth place of `Barack_Obama`. In contrast, LMs must be queried by natural language prompts, such as "*Barack Obama was born in _*", and the word assigned the highest probability in the blank will be returned as the answer. Unlike deterministic queries on KBs, this provides no guarantees of correctness or success.

While the idea of prompts is common to methods for extracting many varieties of knowledge from LMs, in this chapter we specifically follow the formulation of Petroni et al. [159], where factual knowledge is in the form of triples $\langle x, r, y \rangle$. Here x indicates the subject, y indicates the object, and r is their corresponding relation. To query the LM, r is associated with a cloze-style prompt t_r consisting of a sequence of tokens, two of which are placeholders for subjects and objects (e.g., "x *plays at* y *position*"). The existence of the fact in the LM is assessed by replacing x with the surface form of the subject, and letting the model predict the missing object (e.g., "*LeBron James plays at _ position*"):[3]

$$\hat{y} = \arg\max_{y' \in \mathcal{V}} P_{\text{LM}}(y'|x, t_r),$$

[3]We can also go the other way around by filling in the objects and predicting the missing subjects. Since our focus is on improving prompts, we choose to be consistent with Petroni et al. [159] to make a fair comparison, and leave exploring other settings to future work. Also notably, Petroni et al. [159] only uses objects consisting of a single token, so we only need to predict one word for the missing slot.

where $\mathcal{V}$ is the vocabulary, and $P_{\text{LM}}(y'|x, t_r)$ is the LM probability of predicting y' in the blank conditioned on the other tokens (i.e., the subject and the prompt).[4] We say that an LM has knowledge of a fact if $\hat{y}$ is the same as the ground-truth y. Because we would like our prompts to most effectively elicit any knowledge contained in the LM itself, a "good" prompt should trigger the LM to predict the ground-truth objects as often as possible.

In previous work [142, 159, 168], t_r has been a single manually defined prompt based on the intuition of the experimenter. As noted in the introduction, this method has no guarantee of being optimal, and thus we propose methods that *learn* effective prompts from a small set of training data consisting of gold subject-object pairs for each relation.

2.3 Prompt Generation

First, we tackle prompt generation: the task of generating a set of prompts $\{t_{r,i}\}_{i=1}^{T}$ for each relation r, where at least some of the prompts effectively trigger LMs to predict ground-truth objects. We employ two practical methods to either mine prompt candidates from a large corpus (subsection 2.3.1) or diversify a seed prompt through paraphrasing (subsection 2.3.2).

2.3.1 Mining-based Generation

Our first method is inspired by template-based relation extraction methods [2, 175], which are based on the observation that words in the vicinity of the subject x and object y in a large corpus often describe the relation r. Based on this intuition, we first identify all the Wikipedia sentences that contain both subjects and objects of a specific relation r using the assumption of distant supervision, then propose two methods to extract prompts.

Middle-word Prompts Following the observation that words in the middle of the subject and object are often indicative of the relation, we directly use those words as prompts. For example, "*Barack Obama was born in Hawaii*" is converted into a prompt "x *was born in* y" by replacing the subject and the object with placeholders.

Dependency-based Prompts Toutanova et al. [211] note that in cases of templates where words do not appear in the middle (e.g., "*The capital of France is Paris*"), templates based on syntactic analysis of the sentence can be more effective for relation extraction. We follow this insight in our second strategy for prompt creation, which parses sentences with a dependency

[4] We restrict to masked LMs in this chapter because the missing slot might not be the last token in the sentence and computing this probability in traditional left-to-right LMs using Bayes' theorem is not tractable.

parser to identify the shortest dependency path between the subject and object, then uses the phrase spanning from the leftmost word to the rightmost word in the dependency path as a prompt. For instance, the dependency path in the above example is "*France* $\xleftarrow{pobj}$ *of* $\xleftarrow{prep}$ *capital* $\xleftarrow{nsubj}$ *is* $\xrightarrow{attr}$ *Paris*", where the leftmost and rightmost words are "*capital*" and "*Paris*", giving a prompt of "*capital of x is y*".

Notably, these mining-based methods do not rely on any manually-created prompts, and can thus be flexibly applied to any relation where we can obtain a set of subject-object pairs. This will result in diverse prompts, covering a wide variety of ways that the relation may be expressed in text. However, it may also be prone to noise, as many prompts acquired in this way may not be very indicative of the relation (e.g. "*x, y*"), even if they are frequent.

2.3.2 Paraphrasing-based Generation

Our second method for generating prompts is more targeted – it aims to improve lexical diversity while remaining relatively faithful to the original prompt. Specifically, we do so by paraphrasing the original prompt into other semantically similar or identical expressions. For example, if our original prompt is "x *shares a border with* y", it may be paraphrased into "x *has a common border with* y" and "x *adjoins* y". This is conceptually similar to query expansion techniques used in information retrieval that reformulate a given query to improve retrieval performance [25].

While many methods could be used for paraphrasing [15, 180], we follow the simple method of using back-translation [139, 191] to first translate the initial prompt into B candidates in another language, each of which is then back-translated into B candidates in the original language. We then rank B^2 candidates based on their round-trip probability (i.e., $P_{\text{forward}}(\bar{t}|\hat{t}) \cdot P_{\text{backward}}(t|\bar{t})$, where $\hat{t}$ is the initial prompt, $\bar{t}$ is the translated prompt in the other language, and t is the final prompt), and keep the top T prompts.

2.4 Prompt Selection and Ensembling

In the previous section, we described methods to generate a set of candidate prompts $\{t_{r,i}\}_{i=1}^{T}$ for a particular relation r. Each of these prompts may be more or less effective at eliciting knowledge from the LM, and thus it is necessary to decide how to use these generated prompts at test time. In this section, we describe three methods to do so.

2.4.1 Top-1 Prompt Selection

For each prompt, we can measure its accuracy of predicting the ground-truth objects (on a training dataset) using:

$$A(t_{r,i}) = \frac{\sum_{\langle x,y\rangle \in \mathcal{R}} \delta(y = \arg\max_{y'} P_{\text{LM}}(y'|x, t_{r,i}))}{|\mathcal{R}|},$$

where $\mathcal{R}$ is a set of subject-object pairs with relation r, and $\delta(\cdot)$ is Kronecker's delta function, returning 1 if the internal condition is true and 0 otherwise. In the simplest method for querying the LM, we choose the prompt with the highest accuracy and query using only this prompt.

2.4.2 Rank-based Ensemble

Next, we examine methods that use not only the top-1 prompt, but combine together multiple prompts. The advantage to this is that the LM may have observed different entity pairs in different contexts within its training data, and having a variety of prompts may allow for the elicitation of knowledge that appeared in these different contexts.

Our first method for ensembling is a parameter-free method that averages the predictions of the top-ranked prompts. We rank all the prompts based on their accuracy in predicting the objects on the training set, and use the average log probabilities[5] from the top K prompts to calculate the probability of the object:

$$s(y|x, r) = \sum_{i=1}^{K} \frac{1}{K} \log P_{\text{LM}}(y|x, t_{r,i}), \tag{2.1}$$

$$P(y|x, r) = \text{softmax}(s(\cdot|x, r))_y, \tag{2.2}$$

where $t_{r,i}$ is the prompt ranked at the i-th position. Here, K is a hyper-parameter, where a small K focuses on the few most accurate prompts, and a large K increases the diversity of the prompts.

2.4.3 Optimized Ensemble

The above method treats the top K prompts equally, which is sub-optimal given some prompts are more reliable than others. Thus, we also propose a method that directly optimizes prompt weights. Formally, we re-define the score in Equation 2.1 as:

$$s(y|x, r) = \sum_{i=1}^{T} P_{\boldsymbol{\theta}_r}(t_{r,i}|r) \log P_{\text{LM}}(y|x, t_{r,i}), \tag{2.3}$$

[5]Intuitively, because we are combining together scores in the log space, this has the effect of penalizing objects that are very unlikely given any certain prompt in the collection.

where $P_{\boldsymbol{\theta}_r}(t_{r,i}|r) = \text{softmax}(\boldsymbol{\theta}_r)$ is a distribution over prompts parameterized by $\boldsymbol{\theta}_r$, a T-sized real-value vector. For every relation, we learn to score a different set of T candidate prompts, so the total number of parameters is T times the number of relations. The parameter $\boldsymbol{\theta}_r$ is optimized to maximize the probability of the gold-standard objects $P(y|x, r)$ over training data.

2.5 Main Experiments

2.5.1 Experimental Settings

In this section, we assess the extent to which our prompts can improve fact prediction performance, raising the lower bound on the knowledge we discern is contained in LMs.

Dataset As data, we use the T-REx subset [50] of the LAMA benchmark [159], which has a broader set of 41 relations (compared to the Google-RE subset which only covers 3). Each relation is associated with at most 1000 subject-object pairs from Wikidata, and a single manually designed prompt. To learn to mine prompts (subsection 2.3.1), rank prompts (subsection 2.4.2), or learn ensemble weights (subsection 2.4.3), we create a separate training set of subject-object pairs also from Wikidata for each relation that has no overlap with the T-REx dataset. We denote the training set as T-REx-train. For consistency with the T-REx dataset in LAMA, T-REx-train also is chosen to contain only single-token objects. To investigate the generality of our method, we also report the performance of our methods on the Google-RE subset[6], which takes a similar form to T-REx but is relatively small and only covers 3 relations.

Poerner et al. [163] note that some facts in LAMA can be recalled solely based on surface forms of entities, without memorizing facts. They filter out those easy-to-guess facts and create a more difficult benchmark, denoted as LAMA-UHN. We also conduct experiments on the T-REx subset of LAMA-UHN (i.e., T-REx-UHN) to investigate whether our methods can still obtain improvements on this harder benchmark.

Models As for the models to probe, in our main experiments we use the standard BERT-base and BERT-large models [47]. We also perform some experiments with other pre-trained models enhanced with external entity representations, i.e., ERNIE [252] and KnowBert [158], which we believe may do better on recall of entities.

[6] https://code.google.com/archive/p/relation-extraction-corpus/

Evaluation Metrics We use two metrics to evaluate the success of prompts in probing LMs. The first evaluation metric, *micro-averaged accuracy*, follows the LAMA benchmark[7] in calculating the accuracy of all subject-object pairs for relation r:

$$\frac{1}{|\mathcal{R}|}\sum_{\langle x,y\rangle\in\mathcal{R}}\delta(\hat{y}=y),$$

where $\hat{y}$ is the prediction and y is the ground truth. Then we average across all relations. However, we found that the object distributions of some relations are extremely skewed, e.g. more than half of the objects in relation `native_language` are `French`. This can lead to deceptively high scores, even for a majority-class baseline that picks the most common object for each relation, which achieves a score of 22.0%. To mitigate this problem, we also report *macro-averaged accuracy*, which computes accuracy for each unique object separately, then averages them together to get the relation-level accuracy:

$$\frac{1}{|\text{uni\_obj}(\mathcal{R})|}\sum_{y'\in\text{uni\_obj}(\mathcal{R})}\frac{\sum_{\langle x,y\rangle\in\mathcal{R},y=y'}\delta(\hat{y}=y)}{|\{y|\langle x,y\rangle\in\mathcal{R},y=y'\}|},$$

where uni_obj($\mathcal{R}$) returns a set of *unique* objects from relation r. This is a much stricter metric, with the majority-class baseline only achieving a score of 2.2%.

Methods We attempted different methods for prompt generation and selection/ensembling, and compared them with the manually designed prompts used in Petroni et al. [159]. **Majority** refers to predicting the majority object for each relation, as mentioned above. **Man** is the baseline from Petroni et al. [159] that only uses the manually designed prompts for retrieval. **Mine** (subsection 2.3.1) uses the prompts mined from Wikipedia through both middle words and dependency paths, and **Mine+Man** combines them with the manual prompts. **Mine+Para** (subsection 2.3.2) paraphrases the highest-ranked mined prompt for each relation, while **Man+Para** uses the manual one instead.

The prompts are combined either by averaging the log probabilities from the **TopK** highest-ranked prompts (subsection 2.4.2) or the weights after optimization (subsection 2.4.3; **Opti.**). **Oracle** represents the upper bound of the performance of the generated prompts, where a fact is judged as correct if *any* one of the prompts allows the LM to successfully predict the object.

Implementation Details We use $T = 40$ most frequent prompts either generated through mining or paraphrasing in all experiments, and the number of candidates in back-translation

[7] In LAMA, it is called "P@1." There might be multiple correct answers for some cases, e.g. a person speaking multiple languages, but we only use one ground truth. We will leave exploring more advanced evaluation methods to future work.

Prompts	Top1	Top3	Top5	Opti.	Oracle
	*BERT-base (**Man**=31.1)*				
Mine	31.4	34.2	34.7	38.9	50.7
Mine+Man	31.6	35.9	35.1	**39.6**	52.6
Mine+Para	32.7	34.0	34.5	36.2	48.1
Man+Para	*34.1*	35.8	36.6	37.3	47.9
	*BERT-large (**Man**=32.3)*				
Mine	37.0	37.0	36.4	43.7	54.4
Mine+Man	*39.4*	40.6	38.4	**43.9**	56.1
Mine+Para	37.8	38.6	38.6	40.1	51.8
Man+Para	35.9	37.3	38.0	38.8	50.0

Table 2.1: Micro-averaged accuracy of different methods (%). **Majority** gives us 22.0%. Italic indicates the best single-prompt accuracy, and bold indicates the best non-oracle accuracy overall.

is set to $B = 7$. We remove prompts only containing stopwords/punctuations or longer than 10 words to reduce noise. We use the round-trip English-German neural machine translation models pre-trained on WMT'19 [148] for back-translation, as English-German is one of the most highly resourced language pairs.[8] When optimizing ensemble parameters, we use Adam [108] with default parameters and a batch size of 32.

2.5.2 Evaluation Results

Micro- and macro-averaged accuracy of different methods are reported in Tables Table 2.1 and Table 2.2 respectively.

Single Prompt Experiments When only one prompt is used (in the first **Top1** column in both tables), the best of the proposed prompt generation methods increases micro-averaged accuracy from 31.1% to 34.1% on BERT-base, and from 32.3% to 39.4% on BERT-large. This demonstrates that the manually created prompts are a somewhat weak lower bound; there are other prompts that further improve the ability to query knowledge from LMs.

Table 2.3 shows some of the mined prompts that resulted in a large performance gain compared to the manual ones. For the relation `religion`, "x *who converted to* y" improved 60.0%

[8] https://github.com/pytorch/fairseq/tree/master/examples/wmt19

Prompts	Top1	Top3	Top5	Opti.	Oracle
	*BERT-base (**Man**=22.8)*				
Mine	20.7	22.7	23.9	25.7	36.2
Mine+Man	21.3	23.8	24.8	**26.6**	38.0
Mine+Para	21.2	22.4	23.0	23.6	34.1
Man+Para	*22.8*	23.8	24.6	25.0	34.9
	*BERT-large (**Man**=25.7)*				
Mine	26.4	26.3	25.9	30.1	40.7
Mine+Man	*28.1*	28.3	27.3	**30.7**	42.2
Mine+Para	26.2	27.1	27.0	27.1	38.3
Man+Para	25.9	27.8	28.3	28.0	39.3

Table 2.2: Macro-averaged accuracy of different methods (%). **Majority** gives us 2.2%. Italic indicates the best single-prompt accuracy, and bold indicates the best non-oracle accuracy overall.

ID	Relations	Manual Prompts	Mined Prompts	Acc. Gain
P140	religion	x is affiliated with the y religion	x who converted to y	+60.0
P159	headquarters location	The headquarter of x is in y	x is based in y	+4.9
P20	place of death	x died in y	x died at his home in y	+4.6
P264	record label	x is represented by music label y	x recorded for y	+17.2
P279	subclass of	x is a subclass of y	x is a type of y	+22.7
P39	position held	x has the position of y	x is elected y	+7.9

Table 2.3: Micro-averaged accuracy gain (%) of the mined prompts over the manual prompts.

over the manually defined prompt of "x *is affiliated with the* y *religion*", and for the relation `subclass_of`, "x *is a type of* y" raised the accuracy by 22.7% over "x *is a subclass of* y". It can be seen that the largest gains from using mined prompts seem to occur in cases where the manually defined prompt is more complicated syntactically (e.g. the former), or when it uses less common wording (e.g. the latter) than the mined prompt.

Prompt Ensembling Next, we turn to experiments that use multiple prompts to query the LM. Comparing the single-prompt results in Column 1 to the ensembled results in the following three columns, we can see that ensembling multiple prompts almost always leads to better performance. The simple average used in **Top3** and **Top5** outperforms **Top1** across different

ID	Relations	Prompts and Weights	Acc. Gain
P127	owned by	x is owned by y .485 x was acquired by y .151 x division of y .151	+7.0
P140	religion	x who converted to y .615 y tirthankara x .190 y dedicated to x .110	+12.2
P176	manufacturer	y introduced the x .594 y announced the x .286 x attributed to the y .111	+7.0

Table 2.4: Weights of top-3 mined prompts, and the micro-averaged accuracy gain (%) over using the top-1 prompt.

ID	Modifications	Acc. Gain
P413	x plays in→at y position	+23.2
P495	x was created→made in y	+10.8
P495	x was→is created in y	+10.0
P361	x is a part of y	+2.7
P413	x plays ~~in~~ y position	+2.2

Table 2.5: Small modifications (update, insert, and delete) in paraphrase lead to large accuracy gain (%).

prompt generation methods. The optimized ensemble further raises micro-averaged accuracy to 38.9% and 43.7% on BERT-base and BERT-large respectively, outperforming the rank-based ensemble by a large margin. These two sets of results demonstrate that diverse prompts can indeed query the LM in different ways, and that the optimization-based method is able to find weights that effectively combine different prompts together.

We list the learned weights of top-3 mined prompts and accuracy gain over only using the top-1 prompt in Table 2.4. Weights tend to concentrate on one particular prompt, and the other prompts serve as complements. The gap between **Oracle** and **Opti.** indicates that there is still space for improvement using better ensemble methods.

Mining vs. Paraphrasing For the rank-based ensembles (**Top1, 3, 5**), prompts generated by paraphrasing usually perform better than mined prompts, while for the optimization-based ensemble (**Opti.**), mined prompts perform better. We conjecture this is because mined prompts exhibit more variation compared to paraphrases, and proper weighting is of central importance. This difference in the variation can be observed in the average edit distance between the prompts of each class, which is 3.27 and 2.73 for mined and paraphrased prompts respectively. However, the improvement led by ensembling paraphrases is still significant over just using one prompt (**Top1** vs. **Opti.**), raising micro-averaged accuracy from 32.7% to 36.2% on

BERT-base, and from 37.8% to 40.1% on BERT-large. This indicates that even small modifications to prompts can result in relatively large changes in predictions. Table 2.5 demonstrates cases where modification of one word (either function or content word) leads to significant accuracy improvements, indicating that large-scale LMs are still brittle to small changes in the ways they are queried.

2.6 Related Work

Much work has focused on understanding the internal representations in neural NLP models [10], either by using extrinsic probing tasks to examine whether certain linguistic properties can be predicted from those representations [11, 132, 193], or by ablations to the models to investigate how behavior varies [125, 195]. For contextualized representations in particular, a broad suite of NLP tasks are used to analyze both syntactic and semantic properties, providing evidence that contextualized representations encode linguistic knowledge in different layers [62, 72, 86, 206, 207].

Different from analyses probing the representations themselves, our work follows Petroni et al. [159], Poerner et al. [163] in probing for factual knowledge. They use manually defined prompts, which may be underestimating the true performance obtainable by LMs. Concurrently to this work, Bouraoui et al. [19] made a similar observation that using different prompts can help better extract relational knowledge from LMs, but they use models explicitly trained for relation extraction whereas our methods examine the knowledge included in LMs without any additional training.

Orthogonally, some previous works integrate external knowledge bases so that the language generation process is explicitly conditioned on symbolic knowledge [3, 68, 77, 243]. Similar extensions have been applied to pre-trained LMs like BERT, where contextualized representations are enhanced with entity embeddings [158, 163, 252]. In contrast, we focus on better knowledge retrieval through prompts from LMs as-is, without modifying them.

2.7 Conclusion

In this chapter, we examined the importance of the prompts used in retrieving factual knowledge from language models. We propose mining-based and paraphrasing-based methods to systematically generate diverse prompts to query specific pieces of relational knowledge. Those prompts, when combined together, improve factual knowledge retrieval accuracy by 8%, outperforming manually designed prompts by a large margin. Our analysis indicates that LMs are indeed more knowledgeable than initially indicated by previous results, but they are also quite

sensitive to how we query them. To effectively utilize LMs as knowledge bases, it is crucial that they can be queried in various ways while consistently yielding similar results. Additionally, they should seamlessly integrate new factual knowledge into their parameters. We will present our efforts in enhancing both abilities in chapter 5.

Chapter 3

Eliciting Factual Knowledge of Multilingual LMs

This chapter extends the previous line of work to multilingual settings. While knowledge is both written and queried in many languages, studies on LMs' factual representation ability have almost invariably been performed on English. To assess factual knowledge retrieval in LMs in different languages, we create a multilingual benchmark of cloze-style probes for 23 typologically diverse languages. This work is presented in:

- Zhengbao Jiang*, Antonios Anastasopoulos*, Jun Araki, Haibo Ding, Graham Neubig. X-FACTR: Multilingual Factual Knowledge Retrieval from Pretrained Language Models. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing.[1]

Benchmark data and code have been released at:

- https://x-factr.github.io

3.1 Introduction

Recent works have presented intriguing results demonstrating that large-scale LMs also capture a significant amount of *factual knowledge* in English [90, 159, 163]. However, it goes without saying that there are many languages of the world other than English, and it is quite conceivable that (1) users may want to query this factual knowledge in other languages, and (2) some facts will be written in non-English languages and thus multilingually trained LMs (hereinafter, M-LMs) may be more equipped to recall these facts in the languages of the original data. In this

[1] Zhengbao Jiang conducted main experiments and wrote the draft. Antonios Anastasopoulos proposed the idea, processed the dataset in other languages, and revised the draft.

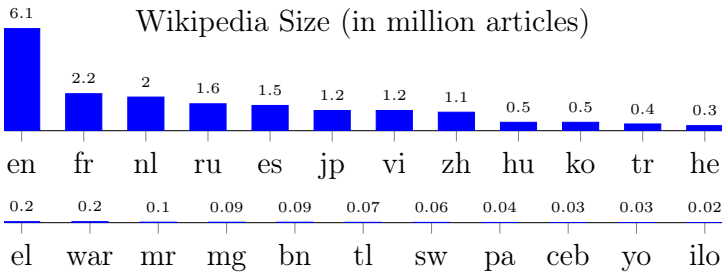


Figure 3.1: X-FACTR contains 23 languages, for which the data availability varies dramatically.

fact	⟨ Bloomberg L.P., founded_in, New York⟩		
en prompt	[X] was founded in [Y].		
es prompt	[X] fue [fundar.Gerund;X] en [Y].		
	↓ ↓ ↓		
es sentence	Bloomberg L.P. fue fundada en ⟨mask⟩ $\times 1 \sim 5$.		
es outputs	prediction	#tokens	confidence
	2012	1	-1.90
	Nueva York	2	-0.61
	EE. UU	3	-1.82
	Chicago, Estados Unidos	4	-3.58
	2012 Bloomberg L.P	5	-3.06

Figure 3.2: Prompts get instantiated to produce grammatical sentences with different numbers of mask tokens and are used to obtain predictions for [Y]. In this Spanish example, the verb gerund "fundar" *to found* is rendered as "fundada" to agree in gender and number with the subject "Bloomberg L.P.". The final prediction is in bold.

chapter, we study the intersection of *multilinguality and the factual knowledge included in LMs*.

We create a new multilingual benchmark for probing factual knowledge in LMs – the Cross-lingual FACTual Retrieval benchmark (X-FACTR). X-FACTR shares a similar formulation as the LAMA benchmark of Petroni et al. [159], which assesses whether LMs have memorized a fact (i.e., a subject-relation-object triple) by having LMs predict the blank (i.e. object) in a cloze-style prompt for each relation after filling in the subject. We manually create such prompts for 23 languages spanning different language families and different levels of data availability (subsection 3.3.1). Because many languages that we handle are morphologically rich, we design a morphology-sensitive annotation schema (see example in Figure 3.2) that can properly instantiate prompts using entity metadata (e.g. gender) and a morphological inflection model (subsection 3.3.3).

In addition, while previous works [90, 159, 163] have limited examination to single-token entities (e.g. "France"), we expand our setting to include multi-token entities (e.g. "United States"), which comprise more than 75% of facts included in our underlying database (Wikidata; subsection 3.3.2). We propose several decoding algorithms for predicting multi-token entities using masked LMs (section 3.4).

We perform experiments on X-FACTR (section 3.5), comparing and contrasting across languages and LMs to answer the following research questions: (1) How and why does performance vary across different languages and models? (2) Can multilingual pre-training increase the amount of factual knowledge in LMs over monolingual pre-training? (3) How much does knowledge captured in different languages overlap? We find that the factual knowledge retrieval of M-LMs in high-resource languages is easier than in low-resource languages, but the overall performance is relatively low, indicating that this is a challenging task. We analyze the types of failure cases, shedding light on future directions to improve factual knowledge in M-LMs. In addition, multilingual pre-training does not necessarily lead to a higher recall of facts compared to language-specific monolingual pre-training. The knowledge memorized by M-LMs in fact is largely distinct across languages, with almost 50% of facts being recalled in only one language.

Inspired by the above observations, we propose a code-switching-based objective function to improve the ability of M-LMs to access knowledge using queries from a variety of languages. We replace entities in a sentence from the original language with counterparts in another language, and further fine-tune the LM on these code-switched data (section 3.6). We perform experiments on three languages (French, Russian, and Greek, code-switched with English). Results demonstrate that this code-switching-based learning can successfully improve knowledge retrieval ability with low-resource language prompts.

3.2 Retrieving Facts from LMs

In this chapter, we follow the protocol of Petroni et al. [159]'s English-language LAMA benchmark, which targets factual knowledge in the form of subject-relation-object triples from Wikidata[2] curated in the T-REx dataset [50]. The cloze-style prompts used therein are manually created and consist of a sequence of tokens, where [X] and [Y] are placeholders for subjects and objects (e.g. "[X] is a [Y] by profession."). To assess the existence of a certain fact, [X] is replaced with the actual subject (e.g. "Obama is a ⟨mask⟩ by profession.") and the model predicts the object in the blank $\hat{y}_i = \text{argmax}_{y_i} p(y_i|\boldsymbol{s}_{i:i})$, where $\boldsymbol{s}_{i:i}$ is the sentence with the i-th token masked out. Finally, the predicted fact is compared to the ground truth. In the next section, we

[2] https://www.wikidata.org/

	en	fr	nl	es	ru	ja	zh	hu	he	tr	ko	vi	el	bn	ceb	mr	war	tl	sw	pa	mg	yo	ilo
#all	45.7	40.2	38.3	37.1	26.3	25.1	23.1	20.4	17.1	16.1	16.1	13.6	13.0	9.4	8.2	7.9	7.3	7.1	6.8	5.5	4.9	4.6	4.1
#single-token	18.9	13.9	12.8	13.5	3.4	1.3	0.2	6.2	1.1	2.5	2.0	3.9	0.7	0.1	3.3	0.2	3.0	3.2	2.8	0.1	1.7	0.9	2.1
#multi-token	26.8	26.4	25.5	23.6	22.9	23.8	22.9	14.2	16.0	13.6	14.1	9.7	12.3	9.3	4.9	7.7	4.4	3.9	4.0	5.4	3.2	3.7	2.0

Table 3.1: X-FACTR benchmark statistics (in thousands).

extend this setting to more languages and predict multiple tokens instead of a single one.

3.3 Multilingual Multi-token Factual Retrieval Benchmark

3.3.1 Languages

In sampling the languages to create our multilingual benchmark, we attempted to create a subset as diverse as possible with regard to data availability, typology, and script – within the constraints of requiring inclusion in Wikidata and standard pre-trained M-LMs. To this end, we created prompts in 23 languages: English, French, Dutch, Spanish, Russian, Japanese, Chinese, Hungarian, Hebrew, Turkish, Korean, Vietnamese, Greek, Cebuano, Marathi, Bengali, Waray, Tagalog, Swahili, Punjabi, Malagasy, Yoruba, and Ilokano.

Our subset includes languages from 11 families (the Indo-European ones include members of the Germanic, Romance, Greek, Slavic, and Indic genera), using 10 different scripts. Our languages display high variance with respect to Wikipedia presence, a proxy for overall data availability, ranging from very large to very small (see Figure 3.1).[3]

3.3.2 Facts

While Petroni et al. [159] and follow-up works focus on entities that can be represented by a single token since many popular entities consist of multiple tokens (e.g. "United States"), we argue that it is crucial to include multi-token entities in the benchmark to make the evaluation unbiased. Similar to Petroni et al. [159], we use the T-REx dataset to collect facts for our benchmark. Since T-REx aligns facts from Wikidata with sentences in abstract sections from DBpedia, we can estimate the commonality of each fact based on its frequency of being grounded to a sentence in these abstracts.

For each of the 46 relations in T-REx, we sample 1000 subject-object pairs with probability proportional to their frequency. Frequency-proportional sampling makes the distribution of the facts in our benchmark close to real usage and covers facts of different popularity. To keep

[3]We excluded bot-made pages for Cebuano and Waray.

the benchmark unbiased, we did not constrain the facts with any language-related criteria (e.g., require the entities to have translations in all languages we considered). As a result, some entities (either subjects or objects) might not have translations in all languages. The number of facts in different languages in our multilingual multi-token X-FACTR benchmark is shown in Table 3.1. Because many modern pre-trained M-LMs almost invariably use some variety of sub-word tokenization, the number of tokens an entity contains will depend on the tokenization method used in the LM. We report the statistics based on the WordPiece tokenization used in multilingual BERT [47]. The tokenization scheme statistics for the other M-LMs are similar.

3.3.3 Prompts

Some languages we include in the benchmark require additional handling of the prompts to account for their grammar or morphology. For example, (some) named entities inflect for case in languages like Greek, Russian, Hebrew, or Marathi. In some languages, syntactic subjects and objects need to be in particular cases. Similarly, languages often require that the verb or other parts of the sentence agree with the subject or the object on some morphological features like person, gender, or number.

Our prompts provide the necessary information in order to generate grammatical sentences, given the gender and number of the entities. For example, the Russian prompt for "[X] was born in [Y]" is:

$$\left[\text{X.Nom}\right] \left[\text{родился;X=MASC} \mid \text{родилась;X=FEM} \mid \text{родилось;X=NEUT}\right] \text{в} \left[\text{Y.Ess}\right].$$

The prompt denotes that the subject ([X]) needs to be in the nominative (Nom) case and the object ([Y]) needs to be inflected in the essive case (Ess). The prompt also accounts for the variation of the gender of [X] providing options (separated by |) for the subject being masculine, feminine, or neuter (MASC, FEM, NEUT respectively).

Everything within square brackets gets concretely instantiated given the subject and object. Grammatical gender is assigned through a combination of Wikidata information and language-specific heuristics, constructed based on feedback from native speakers of each language. When the entity corresponds to a person, we retrieve their "sex_or_gender" properties from Wikidata. In addition, for languages like Greek or French, the gender of an entity can be inferred with fairly high certainty given the form of the word (e.g. looking at the ending). Last, some categories of entities (such as cities, countries, organizations, etc, which can be obtained using the "instance_of" Wikidata property) often get assigned a general grammatical case based on the category.

Once all the morphological features have been specified as detailed above, we use the Unimorph-

Inflect package [6] to generate the appropriately inflected surface form of the bracketed words.[4] We note that the target entity ([Y]) might also need to be inflected, as in the above Russian example, in which case we require the model's predictions to match the inflected target forms.

To verify the quality of the prompts we performed user studies with native speakers, finding that 88% on average were judged as natural and grammatically correct. It is worth noting that the majority of errors are due to prompts being awkward or incorrect for some senses captured by the relation, and not due to our gender heuristics or automatic inflection. This issue is also present in the LAMA English prompts [90].

3.3.4 Evaluation

As noted in Petroni et al. [159], because some subject-relation pairs might have multiple correct objects (e.g., America maintains diplomatic relations with multiple countries), we collect all valid objects and judge a prediction as correct if it can match any object (e.g., both France and Canada are correct). Since an entity might have multiple aliases (e.g., "America" and "the US"), we collect all aliases for each entity from Wikidata, and the prediction is marked as correct if it can match any one of them after lower casing.

3.4 Multi-token Decoding

As Table 3.1 shows, many facts involve multi-token entities, and thus LMs would need to predict these entities in multiple steps. Generating multiple predictions is straightforward for traditional left-to-right LMs [168, 201], where we can autoregressively decode the next token conditioned on previous tokens. However, many pre-trained LMs such as BERT [47] are *masked* LMs that predict individual words given left and right contexts, and decoding from such masked LMs remains an open problem [60, 116, 183, 224, 224]. We systematically examined different multi-token decoding algorithms from three orthogonal perspectives: (1) how the initial predictions are produced, (2) how to refine the predictions, and (3) other commonly used components in neural text generation systems. We assume that the following conditional probability distribution is defined by the masked LM for a sentence with n tokens:

$$p(x_k|x'_1, ..., x'_{k-1}, \langle \text{mask} \rangle_k, x'_{k+1}, ..., x'_n), \tag{3.1}$$

where the subscript of $\langle \text{mask} \rangle$ indicates its position, and the surrounding token $x'_.$ can either be an actual word $x_.$ or $\langle \text{mask} \rangle$. We aim to handle sentences containing multiple mask tokens conditioning on the surrounding actual words:

$$\boldsymbol{s}_{i:j} = x_1, ..., x_{i-1}, \langle \text{mask} \rangle_i, ..., \langle \text{mask} \rangle_j, x_{j+1}, ..., x_n, \tag{3.2}$$

[4] https://github.com/antonisa/unimorph_inflect

(a) Independent: Barack Obama is a United$_1$ of$_1$ president$_1$ by profession

(b) Order: Barack Obama is a United$_1$ State$_2$ President$_3$ by profession

(c) Confidence: Barack Obama is a minister$_2$ of$_3$ cabinet$_1$ by profession

Figure 3.3: Illustration of three initial prediction and refinement methods. Green boxes are mask tokens to be filled, and subscripts indicate the prediction order.

where $\boldsymbol{s}_{i:j}$ indicates a sentence with the i-th to j-th tokens masked out.[5]

3.4.1 Initial Prediction and Refinement

Given a sentence with multiple mask tokens, e.g., Equation 3.2, we can either generate outputs in parallel independently or one at a time conditioned on the previously generated tokens. These methods are similar to the prediction problems that BERT [47] and XLNet [242] perform in their pre-training stages respectively. We define $\boldsymbol{c} \in \mathbb{R}^n$ as the probability of each prediction, with details varying by prediction methods.

After all mask tokens are replaced with the initial predictions, i.e., $\hat{\boldsymbol{s}}_{i:j} = x_1, ..., \hat{y}_i, ..., \hat{y}_j, ..., x_n$, we can further refine the predictions by iteratively modifying one token at a time until convergence or until the maximum number of iterations is reached. Here we outline the algorithms with high-level descriptions.

Independent. For independent initial prediction (Figure 3.3a), the mask tokens are all predicted in parallel (at once). We also consider two autoregressive methods for initial prediction or refinement.

Order-based. Mask tokens are predicted from left to right, in each step conditioning also on the previously generated tokens (Figure 3.3b). In the refinement stage, we modify predictions also from left to right, and convergence is reached when there are no changes in a left-to-right scan.

Confidence-based. In each step, we choose the prediction with the highest probability, so the order of predictions can be arbitrary (Figure 3.3c). In the refinement stage, we choose from all predicted tokens the one with the lowest confidence (i.e., the lowest probability) and re-predict it similarly to Ghazvininejad et al. [60]. Convergence is reached when the re-predicted token is the same as the original token.

[5] We assume that the mask tokens are consecutive for notation simplicity, although all following methods/equations can be easily adapted to non-consecutive cases.

3.4.2 Final Prediction

Because we do not know the number of tokens of the ground truth in advance, we enumerate from 1 to M mask tokens and choose the final prediction based on confidence. Given the prompt in Equation 3.2, the simplest way to compute the confidence is pseudo log likelihood, which is the sum of log probabilities of each predicted token conditioned on the other tokens [183]: $v(j-i+1) = \sum_{k=i}^{j} \log c_k$, where c_k is the confidence (probability) of the k-th predicted token, and $v(m)$ is the overall prediction confidence with m initial mask tokens. Among M predictions, we choose the one with the highest confidence.

3.4.3 Additional Components

We also investigate additional components commonly used in neural generation systems. Specifically, we consider **length normalization** in computing the final confidence (i.e., divide $v(m)$ by the number of mask tokens m) because a simple sum might favor short predictions. In addition, the confidence value $\boldsymbol{c}$ in previous methods contains probabilities when the predictions are first generated, which will become stale once the surrounding tokens change [60]. We consider **re-computing confidence** $\boldsymbol{c}$ whenever a change happens. Last, we attempted **beam search** to keep track of the most plausible B predictions at each step.

3.5 X-FACTR Benchmark Performance

Implementation Details. We use the implementations of different multilingual/monolingual pre-trained LMs in the Transformers library [234]. We examine 3 multilingual pre-trained LMs, M-BERT, XLM, XLM-R [38, 39, 47],[6] and 8 monolingual pre-trained LMs, BERT (en), CamemBERT (fr), BERTje (nl), BETO (es), RuBERT (ru), Chinese BERT (zh), BERTurk (tr), and GreekBERT (el) [26, 43, 113, 141, 190].

We set the maximal number of mask tokens to $M = 5$ for English, French, Dutch, and Spanish. In these languages more than 90% of the entities are split into $\leq$5 tokens. For all other languages we use $M = 10$. This is expected because the vocabulary of M-LMs based on WordPiece tokenization is dominated by frequent words and low-resource-language words tend to split into more pieces [1]. We set the maximal number of iterations to $T = 2M$, so that we can approximately refine all the predicted tokens once for a sentence with M mask tokens (the initial prediction takes exactly M iterations). In our main results, we report results with two decoding algorithms: the simplest independent generation method and the confidence-

[6] Yoruba is not in the training data of XLM and XLM-R.

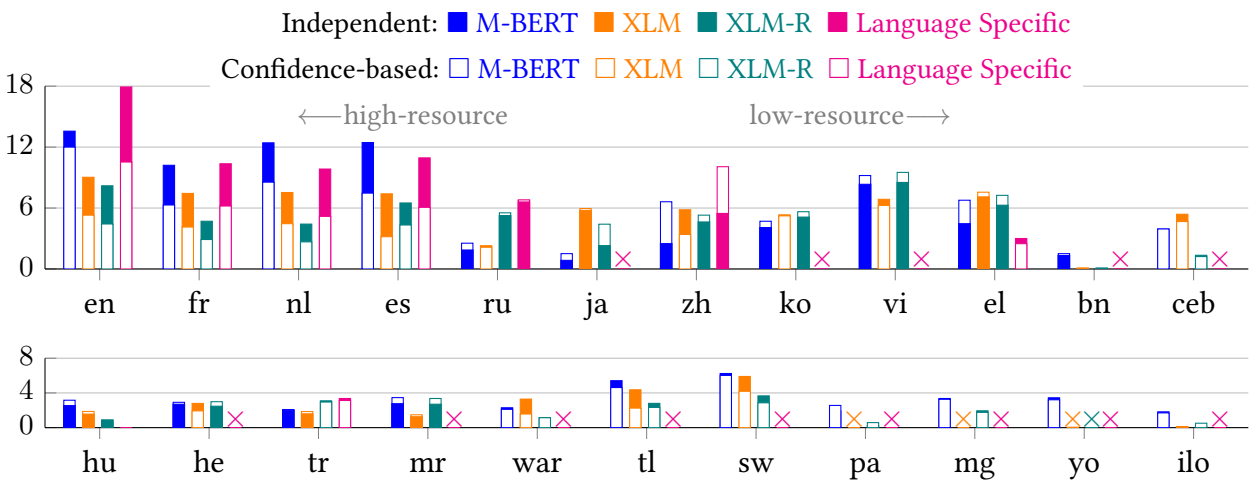


Figure 3.4: Accuracy on different languages using different LMs (%). Independent prediction (solid bars) outperforms confidence-based prediction (no-fill bars) on high-resource languages but not on low-resource languages. Different models are color-coded, with missing/unsupported models marked with ×. Languages are ranked by the total number of facts in our benchmark.

based method for both initial and refinement predictions. To save computation time, we only use confidence re-computation for $M = 5$.

Evaluation Metrics. We follow Petroni et al. [159], computing the accuracy of predicted objects for each relation and macro-average them as final scores. For fine-grained analysis of different decoding methods, pre-trained LMs, and languages, we report results on **all** facts as well as on subsets consisting only of single-token objects (**single**) and multi-token objects (denoted as **multi**).

3.5.1 Experimental Results

We run both the independent and confidence-based decoding methods with 3 M-LMs, and when available 8 monolingual LMs, across 23 languages,[7] with results shown in Figure 3.4. Overall, even in the most favorable settings, the performance of M-LMs at retrieving factual knowledge in the X-FACTR benchmark is relatively low, achieving less than 15% on high-resource languages (e.g., English and Spanish) and less than 5% for some low-resource languages (e.g., Marathi and Yoruba). This may initially come as a surprise, given the favorable performance reported in previous papers [90, 159], which achieved accuracies over 30% on English. We jus-

[7]Check https://x-factr.github.io for the latest results.

tify this discrepancy in our following analysis. We note that, although we provide baseline results in almost all languages, we perform our extensive analysis on a representative subset, consisting of 13 languages.

Performance on Different Languages. Performance on high-resource languages is usually better than performance on middle- or low-resource languages regardless of the (M-)LMs. This is probably due to high-resource languages having more data in the pre-training stage. It is also possible that even if the fact of low-resource languages is written in the available data for these languages, it is not appropriately memorized due to lack of model capacity or forgetting [109]. It is worth noting that the best results are in Indo-European languages which not only have the most data, but also share the same (Latin) script which could further facilitate cross-lingual learning.

Performance of Different LMs. Comparing the performance of different M-LMs, we found that M-BERT outperforms XLM and XLM-R on high-resource languages, while on low-resource languages performance is similar. This is contradictory to the conclusion on other cross-lingual tasks, such as natural language inference and syntactic prediction, as reported in Hu et al. [75]. Our conjecture is that because factual knowledge probing requires retrieving the identity and relations of individual entities, it is more fine-grained than more coarse-grained understanding of syntactic and semantic classes that are required to solve other tasks. We posit that pre-training methods that show superior performance on inference and syntactic prediction tasks (i.e., XLM-R) might achieve good syntactic/semantic abstraction at the cost of making less concrete lexical distinctions.

Comparing M-BERT with language-specific LMs, we find M-BERT outperforms the monolingual BERT on Dutch, Spanish, and Greek, while underperforming on English, Russian, Chinese, and Turkish. Since most of the LMs follow the architecture and pre-training settings of BERT [47] or RoBERTa [135], we hypothesize that training corpus is the major contributor to the final performance. Another potential explanation is that model capacity limitations preclude M-BERT from effectively memorizing entity names/relations in all of the languages.

3.6 Improving Multilingual LM Retrieval

As the performance of M-LMs is relatively low, especially on low-resource languages, an obvious endeavor is to refine the model to improve fact retrieval performance in various languages. We analyze how similarly M-BERT performs on queries in different languages. We collect correctly predicted facts across all languages, and count in how many languages each fact was

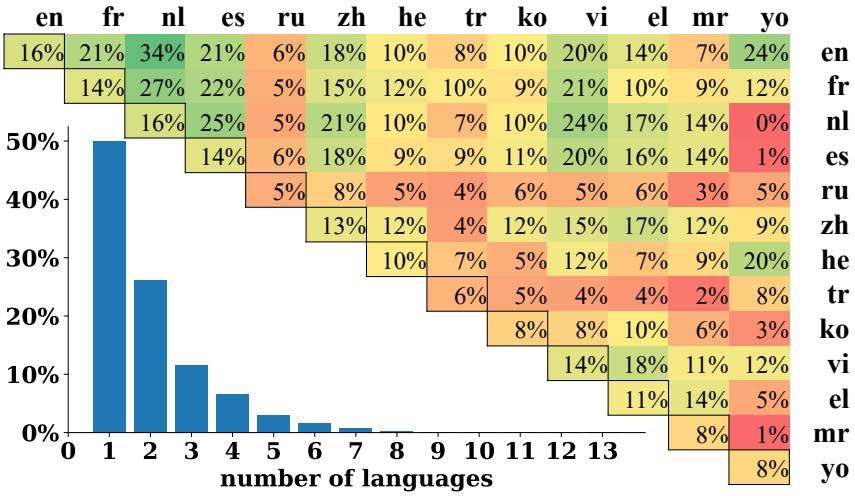


Figure 3.5: Bottom-left: the ratio of facts with respect to the number of languages in which the facts could be successfully retrieved. Top-right: overlap ratio of correct predictions between two languages. The values on the diagonal are the average overlap ratio of the corresponding language with the other languages.

retrieved correctly. As shown in the bottom-left histogram of Figure 3.5, half of the correctly predicted facts were correct in a single language, indicating little overlap across languages [127]. Only 3% of facts were correct in more than 5 languages, and objects in those facts are usually sub-strings of subjects, making them easy to retrieve regardless of the language. This observation is also confirmed by the overlap between pairs of languages in the top-right chart of Figure 3.5; even the most similar languages (i.e., English and Dutch) only have 34% of correct predictions in common.

We find that facts retrievable only in a single language tend to be knowledge that is mainly mentioned in a certain language. For example, M-BERT mistakenly predicts "QQ" in the English sentence "Tencent QQ is developed by _.", while the prediction "腾讯" (Tencent) in the corresponding Chinese sentence "腾讯QQ是由_开发的。" is correct. This is probably because Tencent, a Chinese company, is more frequently mentioned in the Chinese training corpus.

3.6.1 Methods

Inspired by these observations, we propose to use *code-switching* to create data to fine-tune pre-trained LMs, replacing entity mentions in one language (e.g., English/Greek) with their counterparts in another language (e.g., Greek/English). Through this bi-directional code-switching, entity mentions serve as pivots, enabling knowledge that was originally learned in one language to be shared with others. Given a pair of languages, we first identify Wikipedia sentences that

		Single-eval			Double-eval		
Lang.	**Method**	**All**	**Single**	**Multi**	**All**	**Single**	**Multi**
French	M-BERT	10.21	19.07	3.92	10.67	19.24	4.55
	+raw	**15.06**	**26.81**	**7.40**	15.69	26.92	8.27
	+cs	13.15	24.37	6.34	**16.90**	**26.98**	**10.29**
Russian	M-BERT	1.87	4.58	0.96	3.04	7.72	2.28
	+raw	**7.92**	**24.37**	**3.59**	8.77	**26.28**	4.57
	+cs	7.64	22.41	3.55	**11.69**	25.31	**7.85**
Greek	M-BERT	4.49	20.75	2.19	4.97	20.87	2.83
	+raw	**11.49**	**35.27**	**7.65**	12.65	**35.27**	9.27
	+cs	9.30	26.31	5.73	**18.41**	30.93	**15.30**

Table 3.2: Accuracy of M-BERT after fine-tuning on raw and code-switched text (%).

mention entities from our benchmark using SLING [177]. The M-LM is then finetuned on these sentences. Following Wu et al. [236], with 30% of probability we switch all the entity mentions (can be one or multiple) from the original language to their counterparts in the other language, ending up with sentences like "Ομπάμα later reflected on his years ...", where we substituted "Obama" with a Greek mention of the entity, and vice-versa for Greek-to-English. 70% of the sentences remain the same. If there are multiple mention texts for an entity, we sample proportionally to their frequencies, which we found in our preliminary experiments performed better than using a fixed translation. We fine-tune M-BERT using the masked LM objective on this data, with 15% of non-mention words and 50% of mention words masked out.[8]

3.6.2 Experimental Results

We choose three languages with different data availability, namely French, Russian, and Greek, and pair them with English, producing 560k, 396k, and 129k code-switched sentences respectively. We compare M-BERT after code-switched fine-tuning (denoted as **cs**) with both the original M-BERT and with fine-tuning only on raw text (**raw**). We vary the evaluation settings to illustrate the effect of code-switching: on top of matching predictions to ground truth aliases in the prompt language (**single-eval)**, we evaluate with targets in both languages (**double-eval**; English and prompt).

As shown in Table 3.2, continued fine-tuning on raw text outperforms the original M-BERT,

[8]The larger ratio of entities encourages the model to focus on predicting entities, as in the downstream task.

likely due to our fine-tuning on a subset of sentences with mentions of entities from our benchmark. Results on code-switched text are slightly worse when only matching entities in the original target language, but significantly better if we allow matching in both the original language and English. This indicates that code-switched fine-tuning allows M-BERT to retrieve facts, albeit in English rather than in the prompt language. Encouragingly, the increase is larger for low-resource (Greek) and typologically distant-to-English (Russian) languages. For example, the prediction for the Greek prompt "η Θεωρία κατηγοριών είναι μέρος των _." ("Category theory is part of _.") is "mathematics" (in English!), while the prediction without code-switching is the non-informative "οποίων" ("which"). Considering that we have more raw than code-switched sentences in the dataset, this seems to indicate that English entities are easier to predict than their prompt-language counterparts, which might be because facts expressed in English are better learned in the pre-trained model due to training data abundance.

3.7 Conclusion

We examine the intersection of multilinguality and the factual knowledge included in LMs by creating a multilingual and multi-token benchmark X-FACTR, and performing experiments comparing and contrasting across languages and LMs. The results demonstrate the difficulty of this task, and that knowledge contained in LMs varies across languages. Future directions include improved pre-training or fine-tuning methods to improve knowledge retrieval performance across different languages.

Chapter 4

Calibrating LMs for Question Answering

While recent works have shown that LMs capture different types of factual knowledge, they still fail to provide appropriate answers in many cases. In this chapter, we ask the question "how can we know when language models know, with confidence, the answer to a particular query?" We examine this question from the point of view of *calibration*, the property of a probabilistic model's predicted probabilities actually being well correlated with the probabilities of correctness. This work is presented in:

- Zhengbao Jiang, Jun Araki, Haibo Ding, Graham Neubig. How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering. Transactions of the Association for Computational Linguistics 9 (2021): 962-977.[1]

We have released the code at:

- https://github.com/jzbjyb/lm-calibration

4.1 Introduction

LMs trained on massive crawls of internet text (such as T5 [170] and GPT-3 [21]) have been shown to be able to perform quite sophisticated knowledge-based tasks simply through prompting the model to predict the next words given a particular cue.

However, at the same time, LMs are obviously not omnipotent, and still fail to provide appropriate answers in many cases, such as when dealing with uncommon facts [89, 163] or complex reasoning [203]. The high performance on datasets probing factual or numerical knowledge might be achieved through modeling superficial signals in the training data that are not generalizable to unseen test cases [163, 203, 223, 257]. Thus, if such models are to be deployed in real applications it is of crucial importance to determine the *confidence* with which they can provide

[1] Zhengbao Jiang conducted main experiments and wrote the draft.

Format	Input	Candidate Answers	Original	Calibrated
Multi-choice	Oxygen and sugar are the products of (A) cell division. (B) digestion. (C) photosynthesis. (D) respiration.	cell division.	0.00	0.02
		digestion.	0.00	0.01
		photosynthesis.	0.00	0.83
		respiration.	1.00	0.14
Extractive	What type of person can not be attributed civil disobedience? Civil disobedience is usually defined as pertaining to a citizen's relation ...	**head of government**	0.07	0.49
		public official	0.91	0.26
		head of government of a country	0.01	0.16
		public officials	0.01	0.09

Table 4.1: LM calibration examples for the T5 model with correct answers in bold. "Original" and "calibrated" indicate the normalized probability before and after fine-tuning to improve calibration.

an answer. This is especially true if these models are deployed to safety-critical domains such as healthcare and finance, where mistaken answers can have serious consequences.[2]

In this chapter, we ask the question "how can we know when language models know, with confidence, the answer to a particular knowledge-based query?" Specifically, we examine this from the point of view of *calibration*, whether the model's probability estimates are well-aligned with the actual probability of the answer being correct. We apply T5, BART, and GPT-2 over a wide range of question answering (QA) datasets [102] covering diverse domains. We first observe that despite the models' high performance (e.g. T5 eclipses other alternatives such as GPT-2 on some datasets), the models tend to not be well calibrated; their probability estimates over candidates have far-from-perfect correspondence with the actual probability that the answer they provide is correct. Some examples of this are demonstrated in the "Original" column of Table 4.1.

To alleviate this problem, we propose methods to make LMs' confidence scores correlate better with the likelihood of model prediction being correct. We examined both fine-tuning methods that modify LMs' parameters and post-hoc methods that keep LMs fixed and only manipulate the confidence values or inputs. Specifically, we fine-tune the LM using softmax- or margin-based objective functions based on multiple candidate answers. For post-hoc calibration, we examined temperature-based scaling and feature-based decision trees that take prediction probability and input-related features as input and produce calibrated confidence [46, 84, 97]. We also study the sensitivity of LMs' confidence estimation with respect to language variation by paraphrasing candidate answers and augmenting questions using retrieved

[2]For example, a mocked-up medical chatbot based on GPT-3 answered the question of "should I kill myself?" with "I think you should" [167].

context.

Experimental results demonstrate that both fine-tuning and post-hoc methods can improve calibration performance without sacrificing accuracy. We further perform analysis and ablation studies on our methods, inspecting different aspects that may affect calibration performance. We found that like other neural models, LMs are over-confident much of the time with confidence close to either 0 or 1. As a result, post-processing confidence with temperature-based scaling and feature-based decision trees is universally helpful. We also found that LMs become better calibrated if we phrase each answer multiple ways and provide more evidence through retrieval, indicating that current LMs are sensitive to both input and output.

4.2 LM-based Question Answering

LMs are now a ubiquitous tool in not only natural language generation, but also natural language understanding (NLU), where they are largely used for unsupervised representation learning in pre-trained models such as BERT [47]. However, recent work has demonstrated that LMs can also be used *as-is* to solve NLU tasks, by predicting the missing words in Cloze-style questions [159], or by predicting the continuation to prompts [18, 21].

Previous works that purport to calibrate LMs [46, 84, 97, 111] mainly focus on the former use case, using representations learned by LMs to predict target classes (for tasks such as natural language inference, part-of-speech tagging, or text classification) or identify answer spans (for tasks such as extractive QA). In contrast, we focus on the latter case, calibrating LMs themselves by treating them as natural language generators that predict the next words given a particular input.

To make our observations and conclusions as general as possible, we experiment over a diverse range of QA datasets with broad domain coverage over questions regarding both factual and commonsense knowledge [102]. We list all the datasets we used in Table 4.2 along with their corresponding domain. Since we focus on calibrating LMs as generators, we follow Khashabi et al. [102] in converting QA datasets of different formats to a unified sequence-to-sequence format that takes a question X as input and calculates the probability of a continuation Y that corresponds to the answer:

$$P_{\text{LM}}(Y|X) = \prod_{i=1}^{|Y|} P_{\text{LM}}(y_i|X, y_{<i}).$$

Specifically, we focus on two varieties of QA: *multiple-choice* and *extractive*, with examples shown in Table 4.1.[3]

[3]We also considered using free-form (abstractive) QA datasets, where the answers are not constrained to be

Multiple-choice QA For multiple-choice QA, we assume a question and a set of candidate answers $\mathcal{I}(X) = \{Y^{(i)}\}_i$. Inputs X to LMs are questions concatenated with multiple candidate answers (with each answer prefaced by "(A)", "(B)", etc.), and context such as a passage that can be used to help answer the question if any exists. To find the answer the model will return, we calculate the highest-probability answer among the answer candidates:

$$\hat{Y} = \arg\max_{Y' \in \mathcal{I}(X)} P_{\text{LM}}(Y'|X).$$

We can also calculate the normalized probability

$$P_N(\hat{Y}|X) = \frac{P_{\text{LM}}(\hat{Y}|X)}{\sum_{Y' \in \mathcal{I}(X)} P_{\text{LM}}(Y'|X)}, \tag{4.1}$$

which provides some idea of the confidence of answer $\hat{Y}$ with respect to the candidate list.

Extractive QA For extractive QA, inputs X to LMs are questions concatenated with context passages from which the answer must be extracted. In this case, every span within the passage is a candidate answer in $\mathcal{I}(X)$. However, enumerating all possible spans of the context passage is computationally costly. Thus, we follow Jagannatha and Yu [84] in using a manageable set of candidate outputs to perform calibration. Specifically, we develop a method to efficiently calculate probabilities over promising spans that exist in the input. First, we calculate the probability of the first token in output Y', masking out any tokens that are not included in the input passage at all. Then, for the top R scoring tokens, we find their location in the input passage, and calculate the probability of all continuing spans up to a certain length (e.g., 20 tokens). We finally keep the top K spans as candidates $\mathcal{I}(X)$ and use all candidates to calculate the probability in a manner similar to that of multiple-choice QA.

4.3 Background on Calibration

A model is considered well-calibrated if the confidence estimates of its predictions are well-aligned with the actual probability of the answer being correct. Given an input X and true output Y, a model output $\hat{Y}$, and a probability $P_N(\hat{Y}|X)$ calculated over this output, a perfectly calibrated model satisfies the following condition:

$$P(\hat{Y} = Y | P_N(\hat{Y}|X) = p) = p, \forall p \in [0, 1].$$

one of several choices and can instead be any text. However, we found it hard to evaluate the correctness of generated outputs, as paraphrases of the correct answer are still correct, so we do not report results on these datasets in this chapter. Solving this evaluation problem and evaluating calibration on these tasks is an enticing direction for future work.

Format	Datasets and Domains
Multi-choice	ARC (science [35]), AI2 Science Questions (science [35]), OpenbookQA (science [143]), Winogrande (commonsense [182]), CommonsenseQA (commonsense [202]), MCTest (fictional stories [176]), PIQA (physical [16]), SIQA (social [187]), RACE (English comprehension [115]), QASC (science [106]), MT-test (mixed [71])
Extractive	SQuAD 1.1 (wikipedia [172]), SQuAD 2 (Wikipedia [173]), NewsQA (news [215]), Quoref (wikipedia [42]), ROPES (situation understanding [129])

Table 4.2: Datasets used in this chapter and their domains.

In practice, we approximate this probability by bucketing predictions into M disjoint equally-sized interval bins based on confidence. Guo et al. [63] examined the calibration properties of neural network classifiers, and proposed a widely used measure of calibration called expected calibration error (ECE), which is a weighted average of the discrepancy between each bucket's accuracy and confidence:

$$\sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|, \tag{4.2}$$

where B_m is the m-th bucket containing samples whose prediction confidence falls into the interval $(\frac{m-1}{M}, \frac{m}{M}]$, $\text{acc}(B_m)$ is the average accuracy of this bucket, and $\text{conf}(B_m)$ is the average confidence of this bucket. The above equation can be visualized using reliability diagrams (e.g., Figure 4.1 in the experiments), where each bar corresponds to one bucket, and the height is equal to the average accuracy. The diagram of a perfectly calibrated model should have all bars aligned with the diagonal.

Unfortunately, we found that LM-based methods for question answering (such as the UnifiedQA model of Khashabi et al. [102]) were extraordinarily poorly calibrated, with the normalized probability estimates barely being correlated with the likelihood of the outputs being correct. For the two examples in Table 4.1, for instance, we can see that the language model assigns a very high probability to answers despite the fact that they are wrong. This is particularly important because with T5 [170], GPT-3 [21], and others [66, 124] being provided as a potential answer to complex knowledge-based tasks, for models to actually be used in practical scenarios they must also be able to know when they cannot provide correct information. In the following section, we examine methods to improve the calibration of pre-trained models through a number of methods.

4.4 Calibrating LMs for Question Answering

Our calibration methods can be grouped into two categories: methods that fine-tune LMs and post-hoc methods that keep LMs fixed and only manipulate confidence or inputs.

4.4.1 Fine-tuning-based Calibration

Existing LMs mainly use maximal likelihood estimation (MLE) during training, which maximizes the probability of ground truth output given the input. However, it is well-attested that MLE-trained language generators are biased, tending to prefer short outputs [146], or being biased towards more frequent vocabulary [154]. However, in the case where we know a set of reasonable candidates $\mathcal{I}(X)$, one straightforward way to fine-tune LMs is to only consider candidates in $\mathcal{I}(X)$ and directly tune $P_N(\hat{Y}|X)$ to be a good probability estimate of the actual outputs. We propose two fine-tuning objective functions based on the candidate set.

Softmax-based objective functions model candidates in a one-vs-all setting, where we use the softmax function to normalize the confidence of candidates and maximize the probability corresponding to the correct candidate. We use the negative log likelihood as the loss function:

$$L(X, Y) = -\log \frac{\exp(s(Y))}{\sum_{Y' \in \mathcal{I}(X)} \exp(s(Y'))},$$

where the ground truth Y is one of the candidates in $\mathcal{I}(X)$, and $s(\cdot)$ is the logit of the corresponding output (omit condition X for simplicity), which is computed as the log probabilities of all tokens in the output: $s(Y) = \log P_{\text{LM}}(Y|X)$.

Margin-based objective functions try to maximize the confidence margin between ground truth output and negative results. This is motivated by the fact that non-probabilistic objectives such as those used by support vector machines provide reasonably good probabilistic estimates after appropriate scaling and adjustment [162]. Specifically, we use the following objective:

$$L(X, Y) = \sum_{Y' \in \mathcal{I}(X) \setminus Y} \max(0, \tau + s(Y') - s(Y)).$$

4.4.2 Post-hoc Calibration

Compared to fine-tuning methods that optimize the parameters in the model, post-hoc calibration methods keep the model as-is and manipulate various types of information derived from the model to derive good probability estimates [46, 63, 84]. In this section, we consider two

aspects of the model: model probabilities $P_N(\hat{Y}|X)$ and features of the model inputs X or outputs Y. We attempted two representative methods, namely temperature-based scaling [63] and feature-based decision trees [84], to study whether post-processing probabilities is an effective method for calibration of LMs in the context of QA.

Temperature-based Scaling methods have been proposed for classification tasks [46, 63], where a positive scalar temperature hyperparameter τ is introduced in the final classification layer to make the probability distribution either more peaky or smooth: $\text{softmax}(\mathbf{z}/\tau)$. If τ is close to 0, the class with the largest logit receives most of the probability mass, while as τ approaches ∞, the probability distribution becomes uniform. When applying this method to our setting, we use log probabilities of the candidates in $\mathcal{I}(X)$ as logits in computing the softmax function: $z = \log P_{\text{LM}}(Y'|X), Y' \in \mathcal{I}(X)$, and τ is optimized with respect to negative log likelihood on the development split.

Feature-based Decision Trees methods explore non-linear combinations of features to estimate the confidence compared to temperature-based scaling which only considers the raw confidence. We follow previous works [48, 84] and use gradient boosted decision trees [29] as our regressor to estimate the confidence based on features. Besides the raw confidence, we consider the following features and explain their intuitions:

- **Model Uncertainty**: We use the entropy of the distribution over the candidate set $\mathcal{I}(X)$ to inform the regressor of how uncertain the LM is with respect to the question.
- **Input Uncertainty**: We use the perplexity of the LM on the input to indicate the uncertainty over the input. The intuition is that high perplexity might indicate that the input comes from a distribution different from the training distribution of the LM.
- **Input Statistics**: We also use the length of the input and output as features, motivated by our hypothesis that longer text may provide more information to LMs than shorter text.

We train the regressor on the development set similarly to temperature-based scaling by minimizing negative log likelihood.

4.4.3 LM-Specific Methods

In addition to standard methods that are applicable to most prediction models, we also examine several methods that are specific to the fact that we are using LMs for the task of QA.

Input	How would you describe Addison? (A) excited (B) careless **(C) devoted**. Addison had been practicing for the driver's exam for months. He finally felt he was ready, so he signed up and took the test.
Paraphrases & Probabilities	devoted (0.04), dedicated (0.94), commitment (0.11), dedication (0.39)

Table 4.3: An example question with the correct answer in bold. Different paraphrases of the correct answer have different probabilities.

Candidate Output Paraphrasing Motivated by the fact that LMs are sensitive to language variation [90] in tasks like question answering and factual prediction, we hypothesize that one potential reason why the confidence estimation of LMs is not accurate is that the candidate output is not worded in such a way that the LM would afford it high probability. As shown by the example in Table 4.3, paraphrasing the correct answer from "devoted" to "dedicated" increases the probability from 0.04 to 0.94. Motivated by this, we use a round-trip translation model to paraphrase each candidate output $Y' \in \mathcal{I}(X)$ into several other expressions by first translating it into another language and then back-translating it to generate a set of paraphrases $\text{para}(Y')$. We then calculate the probability of each candidate output by summing the probability of all paraphrases $P(Y') = \sum_{Q \in \text{para}(Y')} P_{\text{LM}}(Q|X)$ and normalize it following Equation 4.1. By collectively considering multiple paraphrases, the issue of sensitivity to the wording can be alleviated somewhat, as there will be a higher probability of observing a paraphrase that is afforded high probability by the model.

Input Augmentation Previous work has found that LMs' factual predictions can be improved if more context is provided [160], which has inspired many retrieval-augmented LMs that retrieve evidence from external resources and condition the LMs' prediction on this evidence [66, 122, 124]. We hypothesize that retrieving extra evidence to augment the input also has the potential to improve the confidence estimation of LMs as it will provide the model with more evidence upon which to base both its predictions and its confidence estimates. We follow [160] to retrieve the most relevant Wikipedia article using TF-IDF-based retrieval systems used in DrQA [27] and append the first paragraph of the article to the input.

4.5 Experiments

4.5.1 Experimental Settings

Datasets We evaluate the calibration performance on both multiple-choice QA datasets and extractive QA datasets listed in Table 4.2. To test whether our calibration methods can generalize to out-of-domain datasets, we use a subset of datasets of multiple-choice/extractive QA to train our methods, and the remaining subset of datasets to evaluate the performance. Specifically, we use ARC (easy), AI2 Science Question (elementary), OpenbookQA, QASC, Winogrande, CommonsenseQA, and PhysicalIQA as the training subset for multiple-choice QA (denoted as **MC-train**), and SQuAD 1.1, NewsQA as the training subset for extractive QA (denoted as **Ext-train**). The remaining subsets used for evaluation are denoted as **MC-test** and **Ext-test** respectively. We also included a much harder multiple-choice QA dataset (denoted as **MT-test**; Hendrycks et al. [71]) regarding common sense in a number of genres, in which the largest GPT-3 model and UnifiedQA both display only low to moderate accuracy. For fine-tuning methods, we use the train split of MC-train/Ext-train to fine-tune the LMs. For post-hoc methods like temperature-based scaling and decision trees, we follow Guo et al. [63] and use the development split of MC-train/Ext-train to optimize the parameters.[4]

LMs One clear trend of the past several years is that the parameter size and training data size of pre-trained models play a significant role in the accuracy of models; pre-trained LMs such as BERT [47] tend to underperform more recently released larger LMs like Turing-NLG[5] and GPT-3 [21]. Thus, we employ the largest publicly available LM, which, at the time of this paper's publication, is the T5 model Raffel et al. [170]. The T5 model is a sequence-to-sequence model with both encoder and decoder using transformers [219], and the largest version has 11 billion parameters, allowing it to realize strong performance on tasks such as question answering and natural language understanding [102, 178].

Specifically, we use two varieties of this model. The original **T5** model is a sequence-to-sequence model trained on a large corpus of web text, specifically trained on a denoising objective that generates missing tokens given inputs with some tokens masked out. The **UnifiedQA** model, uses the initial T5 model and fine-tunes on a variety of QA datasets by converting multiple-choice, extractive QA formats into a unified sequence-to-sequence format, similar to the one that we show in Table 4.1. We use the 3-billion versions in our main experiments in subsection 4.5.3 (for efficiency purposes).

[4] Since not all datasets in MC-test and Ext-test have a test split, we report the performance on the development split.

[5] https://msturing.org/

Evaluation Metrics We use accuracy to measure the prediction performance of our methods, and ECE to measure the calibration performance. Accuracy is computed as the ratio of question-answer pairs for which the correct answer has the highest probability among all the candidates in $\mathcal{I}(x)$. ECE is computed using Equation 4.2 by bucketing all candidate answers in $\mathcal{I}(x)$ based on confidence. For MC-test and Ext-test which include multiple datasets, we compute accuracy and ECE on each dataset separately and average across them to avoid the metrics being dominated by large datasets.

Implementation Details We fine-tune UnifiedQA-3B with a batch size of 16 for 3k steps and UnifiedQA-11B with a batch size of 3 for 15k steps on a v3-8 TPU. The maximal length of input and output are set to 512 and 128 respectively, following the setting of UnifiedQA [102]. For extractive QA datasets, we use top $R = 10$ first tokens and finally $K = 5$ spans are used as candidates. For the paraphrasing-based method, we use the WMT-19 English-German and German-English transformer models to perform back translation [148]. The beam size is set to 10 for both directions, which will yield $10 \times 10 = 100$ paraphrases in the end. Since some paraphrases are duplicated, we count the frequency and use the top 5 unique paraphrases in our main experiments subsection 4.5.3. For the retrieval-based augmentation, we use the KILT toolkit [161] to retrieve the most relevant article from the Wikipedia dump, and append the first three sentences of the first paragraph of the retrieved article to the input. For the feature-based decision trees model, we use XGBoost [29] with logistic binary objective, max depth of 4, number of parallel trees of 5, and subsample ratio of 0.8. We use **Temp.** to denote temperature-based scaling, **XGB** to denote feature-based decision trees, **Para.** to denote paraphrasing, **Aug.** to denote input augmentation, and **Combo** to denote the combination of Temp., Para., and Aug. in the experimental section. We use the model with the best calibration performance in post-hoc calibration experiments. For multiple-choice QA, we use the UnifiedQA model after margin-based fine-tuning. For extractive QA, we use the original UnifiedQA model.

4.5.2 Are LM-based QA Models Well Calibrated?

As shown in Table 4.4, our baseline models (i.e., T5 and UnifiedQA) achieve strong performance on a diverse range of QA datasets. On the MT-test datasets, the UnifiedQA model even outperforms the largest version of GPT-3 with 175 billions parameters [71]. Despite the impressive performance, these models are not well calibrated, with ECE higher than 0.2 on the MT-test dataset. We found that LMs tend to be over-confident about cases they do not know, as shown in the confidence distribution in the first row of Figure 4.2 that most predictions have aggressive confidence being close to 0 or 1. The UnifiedQA model assigns high confidence to the wrong answer for examples in Table 4.1, indicating that its confidence estimates are not trustworthy.

4.5.3 Can LM-based QA Models be Calibrated?

We calibrate the UnifiedQA model using both fine-tuning-based methods and post-hoc methods and show their performance in Table 4.4 and Table 4.5 respectively.

Overall, on multi-choice QA datasets (i.e., MC-test and MT-test), both fine-tuning-based methods and post-hoc methods can improve ECE while maintaining accuracy compared to the baseline UnifiedQA model. The best-performing method (i.e., Combo), which combines margin-based fine-tuning, temperature-based scaling, paraphrasing, and input augmentation, improves ECE from 0.095 to 0.044 by over 53%. As shown in the reliability diagrams of the original UnifiedQA model (top-right) and the UnifiedQA model calibrated with Combo (bottom-left) in Figure 4.1, calibration using our methods makes the confidence estimates of predictions better aligned with their correctness. Comparing those two diagrams, an interesting observation is that our method seems to over-calibrate the LM, making over-estimated bars on the right-hand side of the top-right diagram (bars lower than the diagonal) under-estimated and vice versa. This is probably caused by the temperature being too aggressive (i.e., too large), making the distribution too flat. Note that the datasets used to learn the temperature (MC-train) and used in evaluation (MC-test) are different, which we hypothesize is the reason why the temperature is too aggressive. We verify this by learning an oracle temperature on the evaluation datasets (MC-test). The learned temperature indeed becomes smaller (1.35 $\rightarrow$ 1.13), and the reliability diagram (bottom-right in Figure 4.1) is almost perfectly aligned. This demonstrates the challenge of calibrating LMs across different domains.

However, on extractive QA datasets, the improvement brought by different calibration methods is smaller. We hypothesize that this is because the candidate set $\mathcal{I}(X)$ generated by the span-based decoding method for extractive QA is harder to calibrate than the manually curated candidate answers for multiple-choice QA. We compute the average entropy of the confidence of the UnifiedQA model over $\mathcal{I}(X)$ on both extractive QA (Ext-test) and multiple-choice QA datasets (MC-test), and found that Ext-test indeed has much higher entropy compared to MC-test (0.40 vs 0.13), which partially explains the difficulty of calibration on extractive QA datasets.

4.5.4 Analysis of Individual Calibration Methods

In this section, we discuss each method in detail and analyze why they can improve calibration performance.

Objective Function Matters. The original UnifiedQA model is fine-tuned based on MLE, which maximizes the probability of the gold answer given the question. Both softmax-based and margin-based fine-tuning, which explicitly compare and adjust the probability of candidate an-

Method	MC-test		MT-test		Ext-test	
	ACC ↑	ECE ↓	ACC ↑	ECE ↓	ACC ↑	ECE ↓
T5	0.313	0.231	0.268	0.248	0.191	0.166
UnifiedQA	0.769	0.095	0.437	0.222	0.401	0.114
+ softmax	0.767	0.065	0.433	0.161	0.394	**0.110**
+ margin	0.769	**0.057**	0.431	**0.144**	0.391	0.112

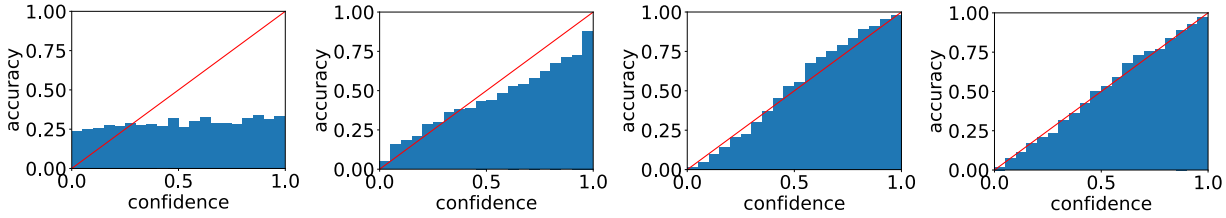
Table 4.4: Performance of different fine-tuning methods.

Method	MC-test		MT-test		Ext-test	
	ACC ↑	ECE ↓	ACC ↑	ECE ↓	ACC ↑	ECE ↓
Baseline	0.769	0.057	0.431	0.144	0.401	0.114
+ Temp.	0.769	0.049	0.431	**0.075**	0.401	0.107
+ XGB	0.771	0.055	0.431	0.088	0.402	**0.103**
+ Para.	0.767	0.051	0.429	0.122	0.393	0.114
+ Aug.	0.744	0.051	0.432	0.130	0.408	0.110
+ Combo	0.748	**0.044**	0.431	0.079	0.398	0.104

Table 4.5: Performance of different post-hoc methods using the UnifiedQA model after margin-based fine-tuning or the original UnifiedQA model as the baseline model. "+Combo" denotes the method using both Temp., Para., and Aug.

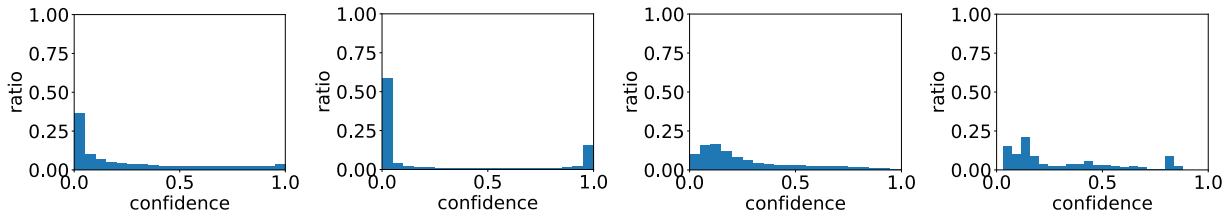
swers, can further improve ECE on multiple-choice datasets. We argue that the softmax-based and margin-based objective functions are better suited for questions with potential candidates.

Post-processing Confidence is Effective Universally. Post-processing the raw confidence either solely based on confidence or other features is effective across all datasets, which is consistent with the conclusion on other tasks such as structured prediction and natural language inference [46, 84]. We demonstrate the histogram of confidence before and after applying temperature-based scaling or feature-based decision trees in Figure 4.2. LMs tend to be over-confident, with most predictions having either extremely high or low confidence. Both methods can successfully re-scale the confidence to reasonable ranges, thus improving the calibration performance.



(a) T5 (b) UnifiedQA (c) UnifiedQA w/ Combo (d) UnifiedQA w/ Combo and oracle temperature

Figure 4.1: Reliability diagram of the T5 model (top-left), the original UnifiedQA model (top-right), the UnifiedQA model after calibration with Combo (bottom-left), and Combo with oracle temperature (bottom-right) on the MC-test datasets.



(a) T5 (b) UnifiedQA (c) UnifiedQA w/ Temp. (d) UnifiedQA w/ XGB

Figure 4.2: The ratio of predictions with respect to the confidence of the T5 model (top-left), the UnifiedQA model (top-right), the UnifiedQA model after temperature-based calibration (bottom-left), and the UnifiedQA model after feature-based calibration (bottom-right) on the MC-test datasets.

Paraphrasing Answers and Input Augmentation can Improve Confidence Estimation. The improvement brought by using paraphrasing is significant on multiple-choice datasets, demonstrating that using diverse expressions can indeed improve confidence estimation. To better understand under what circumstances paraphrasing works, we group candidate answers into two categories: the first group includes candidate answers that become better calibrated using paraphrases; the second group includes candidate answers whose confidence remains the same using paraphrases. We say that a candidate becomes better calibrated if its confidence increases/decreases by 20% if it is a correct or incorrect answer respectively. We found that the average length of questions for better-calibrated candidates (187) is much shorter than that of candidates without improvement (320), indicating that paraphrasing is useful mainly for short questions. We also compute the diversity of word usage in paraphrases using the number of unique words divided by the total length of paraphrases. We found that better-calibrated candidates have slightly higher diversity (0.35 vs 0.32), which is consistent with our

intuition. Retrieval-based augmentation can also improve calibration performance on multiple-choice datasets, which is probably because the retrieved documents can provide extra evidence about the question, making LMs more robust at confidence estimation.

Calibration Methods are Complementary. By combining margin-based fine-tuning, temperature-based scaling, paraphrasing, and input augmentation, we achieve the best ECE on MC-test, demonstrating that these calibration methods are complementary to each other.

4.6 Related Work

Calibration Calibration is a well-studied topic in other tasks such as medical diagnosis [88] and image recognition [63, 121]. Previous works in NLP have examined calibration in structured prediction problems such as part-of-speech tagging and named entity recognition [84], natural language understanding tasks such as natural language inference, paraphrase detection, extractive question answering, and text classification [46, 97, 111]. In contrast, we focus on calibrating LMs themselves by treating them as natural language generators that predict the next words given a particular input.

LM probing Previous works probe pre-trained LMs with respect to syntactic and semantic properties [72, 206], factual knowledge [90, 159, 163], commonsense knowledge [110, 214], and other properties [203]. These works usually focus on what LMs know, while in this chapter we also consider the cases when LMs do not know the answer with confidence.

4.7 Conclusion

In this chapter, we examine the problem of calibrating LMs for question answering. We first note that LMs tend to be poorly calibrated in their probability estimates. To alleviate this problem, we attempted several methods to either fine-tune the LMs, or adjust the confidence by post-processing raw probabilities, augmenting inputs, or paraphrasing candidate answers. Experimental results demonstrate the effectiveness of these methods.

Some future directions could be developing various calibration methods either through improving the confidence adjustment or directly generating outputs with expressed confidence levels. It is also interesting to investigate the effect of calibration on users or downstream tasks. For instance, providing users with model confidence can influence downstream decisions [251], and users may want to adjust required confidence thresholds on critical domains such as health, safety, and medicine.

Chapter 5

Enhancing LLMs in Absorbing Knowledge through Pre-instruction-tuning

We have studied knowledge extraction and calibration in previous chapters. In this chapter, we move one step further to improve the capacity of LLMs to store knowledge. We find that LLMs trained on documents struggle to answer questions related to these documents, even though the perplexity of documents is minimized. We hypothesize that it is beneficial to expose LLMs to QA pairs *before* continued pre-training on documents so that the process of encoding knowledge from complex documents takes into account how this knowledge is accessed through questions. Based on this, we propose **pre-instruction-tuning (PIT)**, a method that instruction-tunes on questions prior to training on documents. This work is presented in:

- Zhengbao Jiang, Zhiqing Sun, Weijia Shi, Pedro Rodriguez, Chunting Zhou, Graham Neubig, Xi Victoria Lin, Wen-tau Yih, Srinivasan Iyer. Instruction-tuned Language Models are Better Knowledge Learners. In Proceedings of the 62th Annual Meeting of the Association for Computational Linguistics 2024.[1]

5.1 Introduction

The factual knowledge in LLMs is static, meaning that it can become outdated as the world evolves, or prove insufficient when LLMs are used in specialized or private domains. To keep LLMs up-to-date, it is common to continue pre-training on new documents to store knowledge in parameters, which allows LLMs to effectively answer queries that require up-to-date

[1] Zhengbao Jiang proposed the idea, conducted main experiments, and wrote the draft.

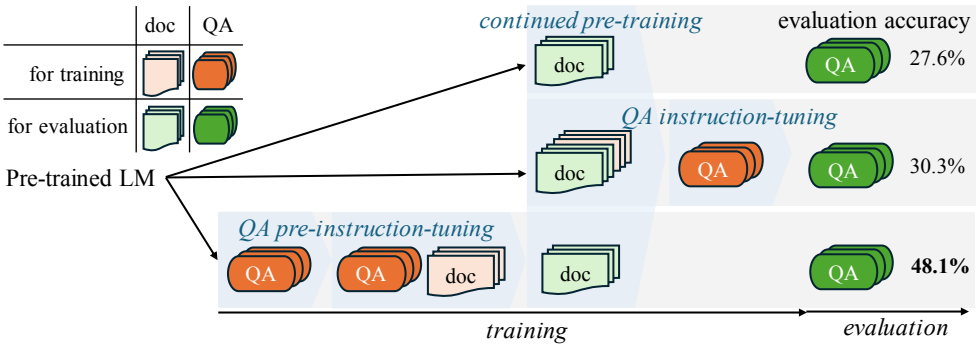


Figure 5.1: Illustration of continued pre-training (first row), continued pre-training followed by instruction-tuning (second row), and pre-instruction-tuning before continued pre-training (last row), along with their accuracies on evaluation questions. Each right-pointing light-blue triangle indicates a training phase.

information [85]. A widely held view is that the factual knowledge stored in parameters can be elicited through prompting [21, 159, 178], and that instruction-tuning (also known as supervised fine-tuning or alignment) makes this elicitation more effective [155, 184, 231]. In the first part of this chapter (section 5.4), we conduct extensive experiments using Llama-2 [213] to answer the following question: *to what extent can we augment the knowledge stored in modern LLMs by continued pre-training on new documents, either with or without subsequent instruction-tuning*? We find that, as we train LLMs repeatedly over documents to the extent that perplexity is minimized to one, the percentage of questions regarding those documents that LLMs answer correctly increases consistently to 27.6%. Subsequent instruction-tuning further improves it to 30.3%, confirming that this widely used practice is useful to elicit more knowledge from LLMs.[2] However, the amount of elicited knowledge is still limited, even though the perplexity of documents is minimized, a phenomenon we refer to as the "perplexity curse".[3]

In the second part of the chapter (section 5.5), we study methods to mitigate the perplexity curse by making LLMs more adept at absorbing knowledge from documents. Zhu and Li [260] presented an intriguing finding that training a randomly initialized transformer from scratch on a mix of biographies and related questions resulted in strong generalization to new questions. However, understanding the reasons behind this finding and exploring ways to practically apply it for absorbing knowledge from new documents requires further investigation. We found that question-answer (QA) pairs are generally straightforward and easily digestible, while documents tend to be more complex and cluttered, often weaving many factual state-

[2]This capacity might be underestimated by previous works due to using relatively small LMs or randomly initialized transformers, or lack of exhaustive training or instruction-tuning [76, 226, 260].

[3]Inspired by the "reversal curse" of Berglund et al. [13].

ments together in a more intricate manner. Therefore, we hypothesize that *it is beneficial to deliberately expose LLMs to QA data before continued pre-training on documents so that the process of encoding knowledge from complex documents takes into account how this knowledge is accessed through questions*. We refer to this as **pre-instruction-tuning (PIT)** and conduct comprehensive experiments to benchmark different variations of this method. As shown in Figure 5.1, our best-performing variation starts with training exclusively on QA pairs (e.g., "who handled the editing of Oppenheimer") to grasp how knowledge is accessed. This is followed by training on a combination of these QA pairs and associated documents (e.g., "who handled the editing of Oppenheimer" and a document about "Oppenheimer"). In this phase, LLMs enhance their ability to absorb knowledge from information-dense documents, building upon the QA pairs that they have already mastered. To study continual knowledge acquisition, we build a dataset named `Wiki2023`, which includes a collection of documents from Wikipedia that are relevant to the year 2023. Comprehensive experiments on `Wiki2023` demonstrate that after PIT, LLMs exhibit an enhanced ability to absorb knowledge from new documents (e.g., a document about "Barbie"). Detailed ablation studies reveal that this ability primarily stems from prioritizing learning how to access knowledge over learning to encode knowledge from documents. Overall, PIT significantly outperforms the standard instruction-tuning approach (subsection 5.5.1 and subsection 5.5.2), improving QA accuracies by 17.8% on Llama-2 7B (30.3% → 48.1%) and 16.3% on Llama-2 70B (46.4% → 62.7%). Moreover, PIT also enhances the ability to absorb knowledge from documents of a *different* domain, shedding light on the potential to scale this method up to a wider variety of documents and instructions for more robust generalization (subsection 5.5.4).

5.2 Building a Dataset to Study Continual Knowledge Acquisition

To assess the ability of LLMs to learn knowledge from new documents, it is essential to use a document corpus with minimal overlap with the original pre-training corpus. This ensures that when an LLM correctly answers questions, we can confidently attribute this capability to its learning from the new documents, rather than encountering similar questions in its original pre-training corpus. In this section, we describe a methodology for building such a corpus from Wikipedia.

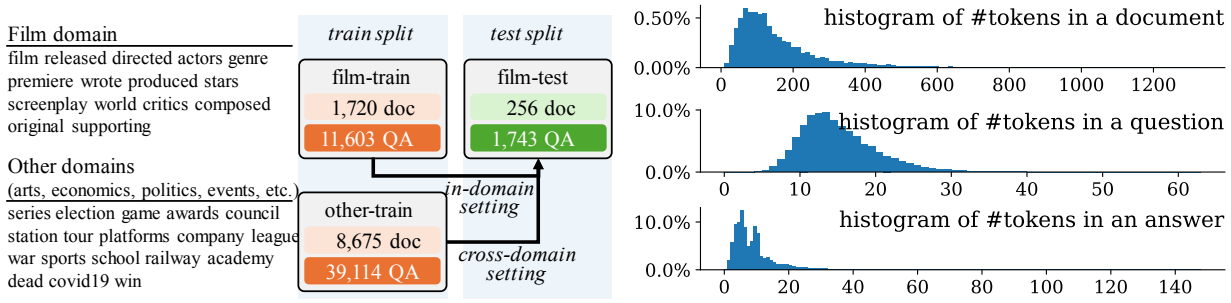


Figure 5.2: The `Wiki2023` dataset. **Middle**: the number of documents and QA pairs; **Left**: frequent keywords in questions; **Right**: the distribution of token counts in documents, questions, and answers.

An example document about "Oppenheimer"
<bos> Oppenheimer (OP-ən-hy-mər) is a 2023 epic biographical thriller film written and directed by Christopher Nolan. It stars Cillian Murphy as J. Robert Oppenheimer, … the film chronicles the career of Oppenheimer, with the story predominantly focusing on his studies, his direction of the Manhattan Project during World War II, and his eventual fall from grace due to his 1954 security hearing. … **Editing was handled by Jennifer Lame**, and the score was composed by Ludwig Göransson. … Oppenheimer premiered at Le Grand Rex in Paris on July 11, 2023, and was theatrically released …

Example QA about "Oppenheimer"
<bos> Question: Who wrote and directed the film Oppenheimer?
Answer: Christopher Nolan. <eos>
<bos> Question: Who stars as J. Robert Oppenheimer in the film?
Answer: Cillian Murphy. <eos>
<bos> Question: What aspects of Oppenheimer's life does the film focus on?
Answer: His studies, direction of the Manhattan Project, and 1954 security hearing. <eos>
<bos> **Question: Who handled the editing of Oppenheimer?**
Answer: Jennifer Lame. <eos>
<bos> Question: When did Oppenheimer premiere in Paris?
Answer: July 11, 2023. <eos>

Figure 5.3: An example document about "Oppenheimer" and corresponding QA pairs from `Wiki2023`. Tokens used for computing losses are highlighted in green. We use the underlined question and the corresponding supporting evidence as a recurring example in this chapter.

5.2.1 Wiki2023 Document Corpus

In the following experiments (section 5.4 and section 5.5), we use Llama-2 (7B and 70B) [213] since it was one of the best-performing LLMs. We use Wikipedia articles classified under the "2023" Category including topics from diverse domains such as films, arts, economics, politics,

events, etc.[4] The likelihood that this factual information is not included in the original training corpus is supported by the low QA performance in Table 5.1 (9.5%/17.2% for 7B/70B).[5] To accelerate the training process, we only use the first section of each article, which offers a thorough summary and contains many factual statements. The number of collected documents and an example document about "Oppenheimer" can be found in Figure 5.2 and Figure 5.3. We refer to this as the `Wiki2023` dataset.

5.2.2 Wiki2023 Question-answer Pairs

To collect QA pairs for either instruction-tuning or performance evaluation, we employ publicly available LLMs to generate diverse questions and their respective answers given the article as context, following the Prompt 1. On average, 4.93 questions are generated for each article. Figure 5.2 and Figure 5.3 show the detailed statistics and example QA pairs about "Oppenheimer", respectively.

Prompt 1: question-answer generation prompt

Given the following summary about the subject {topic}, generate a comprehensive list of questions and corresponding answers that cover all aspects. To make the question clear, always include {topic} in the question. Answers should be concise, consisting of a few short phrases separated by commas.

Output in the following format:
Q: an open-domain question about the subject {topic} (the subject {topic} should always be included)
A: phrase1, phrase2, ...

Summary:
{summary}

5.2.3 Splits

Among all domains, we select the film domain for evaluation and randomly select 256 articles as the test split (`Wiki2023-film-test`). We continually train LLMs on documents from the test split (`Wiki2023-film-test-doc`), and assess their performance based on the accuracy of corresponding questions (`Wiki2023-film-test-QA`). The remaining 1720 articles and corresponding QA pairs (`Wiki2023-film-train`) will be used to study different training strategies, which corresponds to the in-domain setting in Figure 5.2. We also

[4] https://en.wikipedia.org/wiki/Category:2023

[5] It is important to note the difficulty in completely avoiding factual overlap between `Wiki2023` and the pre-training corpus of Llama-2. For example, a film released in 2023 might have had information available before 2023. Data duplication detection is an active research direction, which falls beyond the focus of this study.

train on other domains before evaluation on the film domain to study the effectiveness of different methods across domains, which corresponds to the cross-domain setting in Figure 5.2.

5.3 Experimental Settings

5.3.1 Objectives

When training on documents, we prepend a <bos> token and compute the standard next-token prediction loss by averaging over all tokens in the document: $L_{\boldsymbol{d}} = -\sum_t \log P(\boldsymbol{d}_t|\boldsymbol{d}_{<t})/|\boldsymbol{d}|$.[6] When training on QA pairs, we compute the average negative log-likelihood loss only on tokens in the answer given the question as the prefix: $L_{\boldsymbol{a}} = -\sum_t \log P(\boldsymbol{a}_t|\boldsymbol{q}, \boldsymbol{a}_{<t})/|\boldsymbol{a}|$. Figure 5.3 presents an example document alongside QA pairs, where tokens used for computing losses are highlighted.

5.3.2 Hyperparameters

We use AdamW [136] with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and a weight decay of 0.1. We decay the learning rate to 10% of its initial value using a cosine scheduler without warm-up. When pretraining on documents, we use a batch size of 256 documents and an initial learning rate of 3e-5. During instruction-tuning on QA pairs, we use the same batch size of 256 QA pairs, but opt for a reduced initial learning rate of 5e-6 because the number of tokens in a single batch used for computing losses is lower. The number of epochs varies depending on the setting and is detailed in the corresponding sections.

5.3.3 Evaluation Metrics

At inference time, we use greedy decoding to generate answers given questions as context, following the format in Figure 5.3. To evaluate the original Llama-2, we add 5 QA pairs as in-context exemplars to make sure it follows the QA format. Since most questions are simple factoid questions and most answers are relatively short, we use exact match (EM) as our primary metric [114], which measures whether the model's output matches the gold answer exactly after normalization (e.g., remove articles and punctuations). To assess longer responses and accommodate minor lexical differences, we also report answer recall, which assesses if the gold answer appears in the model's output, and ROUGE-L, which measures the longest common subsequence between the model's output and the gold answer.

[6] We do not append a <eos> token at the end of documents because we only use the first section, which does not signify the conclusion of the entire article.

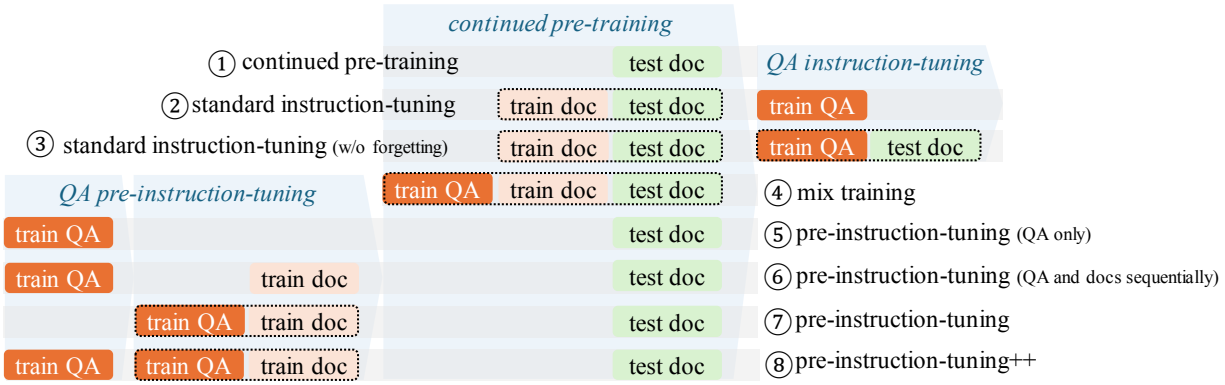


Figure 5.4: Different experimental settings examined in this chapter. Each row represents a different experimental setting with a unique name and number, and each vertical section highlighted by a right-pointing light-blue triangle indicates a training phase. Models are assessed on test QA across all settings. Whenever multiple datasets are enclosed within a dashed square, they are mixed together during the training process.

5.4 How Much Knowledge Can LLMs Absorb via Continued Pre-training Followed by Instruction-tuning?

Factual knowledge stored in the parameters of LLMs can be accessed and applied to answering questions through prompting without additional training [21, 90, 159, 178]. With additional instruction-tuning (also known as supervised fine-tuning) on high-quality data [184, 231], knowledge seems to be more effectively elicited from LLMs. However, when LLMs correctly answer a question, the source of the knowledge is unclear due to the diversity of the pre-training data. For instance, when answering the question "where is the world's largest ice sheet located", do LLMs derive their response by recalling and generalizing information from a seen document about the Antarctic ice sheet, or do they merely repeat answers from similar questions encountered in the training data? This distinction is crucial, as the former scenario implies an ability to comprehend documents and effectively store knowledge within parameters in a way that can be elicited later, whereas the latter is mere rote memorization.

Several works have studied this problem and the predominant finding is that LMs struggle to answer questions about documents they have been trained on [226, 260]. It is important to note, however, that these experiments were mainly conducted using relatively small LMs such as BART, T5, or GPT-2 [76, 85, 226], using randomly initialized transformers [260], or without instruction-tuning [156]. This makes us wonder *what are the actual limits of modern LLMs to absorb knowledge from new documents and answer questions about them using the standard continued pre-training followed by instruction-tuning recipe.* In this section, we run extensive

experiments using Llama-2 7B and 70B on `Wiki2023-film` to test their limits.

5.4.1 Vanilla Continued Pre-training and Instruction-tuning

Experimental settings We experiment with two standard settings and assess their performance by answering associated questions.

- Continued pre-training: train on test documents without instruction-tuning (Figure 5.4 ①).[7]
- Standard instruction-tuning: train on both train and test documents before instruction-tuning on train QA pairs (Figure 5.4 ②).

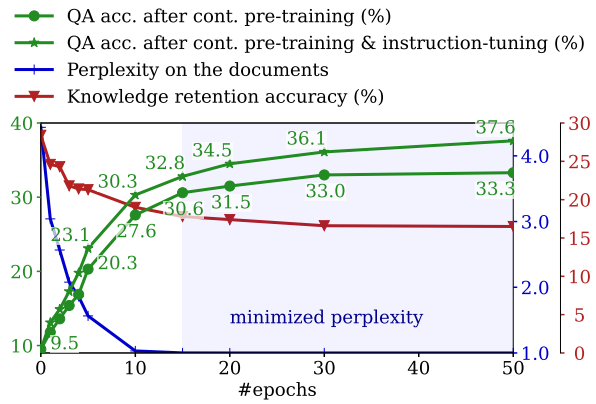
We perform instruction-tuning for a single epoch since more epochs usually result in diminished performance. For training on documents, we opt for multiple epochs (10/5 for a 7B/70B model), which allows for effective knowledge acquisition and remains affordable for corpora of moderate sizes.

Experimental results As shown in Table 5.1, the relatively low performance of the original Llama-2 model (9.5%/17.2% for 7B/70B) indicates that most knowledge in the test documents is not included in the original pre-training corpus. After continued pre-training on documents, performances increase to 27.2%/41.7%, indicating that LLMs can absorb some amount of knowledge. Instruction-tuning further increases the performance to 30.3%/46.4%, confirming the effectiveness of this standard recipe. This observation is different from Zhu and Li [260], which demonstrates that instruction-tuning after pre-training is ineffective on a randomly initialized GPT-2-like transformer. The difference probably arises because Llama-2, through its pre-training on diverse corpora comprising raw documents and QA data, has developed a certain degree of proficiency in extracting knowledge from its parameters via questions. We also report the performance where the corresponding document is directly provided to Llama-2 as context ("open-book w/ doc" in Table 5.1). The significant gap between closed-book and open-book settings suggests that retrieving knowledge from the parameters of LLMs is still challenging.

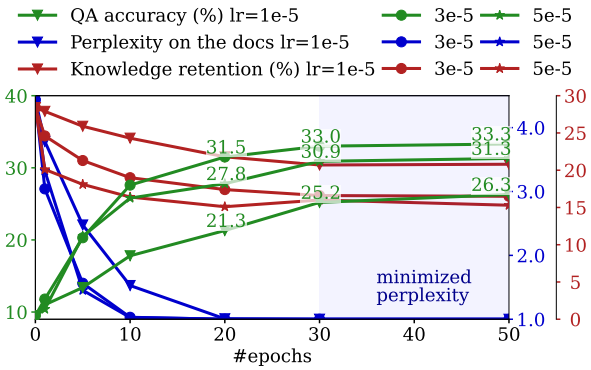
5.4.2 Analyzing the Training Dynamics: Perplexity and Generalization

How does lower perplexity of documents lead to generalization to answering related questions? We vary the number of epochs and learning rate (Figure 5.5) for continued pre-training on

[7] We found that LLMs struggle to adhere to the QA format after training on raw documents for multiple epochs. Therefore, we include a small set of QA pairs (64) during continued pre-training to prevent LLMs from forgetting the QA format.



(a) Training dynamics w/ (Figure 5.4 ②) and w/o instruction-tuning (Figure 5.4 ①). Reduction in perplexity consistently leads to improvement in QA accuracy, indicating that factual knowledge acquisition necessitates exhaustive loss minimization.



(b) Training dynamics with different learning rates (Figure 5.4 ①). After perplexity is minimized, larger learning rates usually lead to less overfitting to deceptive patterns in documents and better generalization when responding to questions.

Figure 5.5: We vary the number of epochs (left) and learning rate (right) during continued pre-training to study the training dynamics of Llama-2 7B. The left axis is QA accuracies for test questions, measured by exact match. On the right axis, we display 2 metrics indicated by distinct colors: the perplexity of all tokens in the documents, and the knowledge retention accuracy, measured by QA accuracy on the Natural Questions dataset. We highlight situations where perplexity of all document tokens is minimized to 1 .

documents and monitor three metrics to study the training dynamics.[8]

- **Knowledge acquisition** QA accuracies on test questions measured by exact match.
- **Perplexity of documents** We compute perplexity (PPL) on all tokens within the documents.
- **Knowledge retention** We approximate the retention of accumulated knowledge during pre-training by assessing the QA accuracy on the Natural Questions (NQ) dataset. NQ was released in 2019, and primarily includes questions based on Wikipedia articles from that time.

Experiment results

- As shown in the left figure of Figure 5.5, QA accuracy consistently improves as perplexity approaches one, indicating that *factual knowledge learning necessitates exhaustive loss minimization over all tokens*. This contrasts with learning general skills, where overly optimizing leads to overfitting.

[8]Since we always decay the learning rate to 10% of its initial value, training for more epochs is not the same as continuing training from a checkpoint obtained after fewer epochs.

Settings	Llama-2 7B EM	Rec.	R-L	Llama-2 70B EM	Rec.	R-L
closed- and open-book performance before training						
closed-book	9.5	10.0	21.2	17.2	18.1	31.4
open-book w/ doc	72.2	75.4	91.5	78.2	80.6	94.9
closed-book performance w/ standard methods						
cont. pre-training ①	27.6	31.6	43.8	41.7	45.8	60.2
+instruction-tuning ②	30.3	34.7	47.4	46.4	50.9	64.1
mix all data ④	39.4	44.6	56.7	57.1	63.4	72.4
closed-book performance w/ pre-instruction-tuning (PIT)						
PIT (QA only) ⑤	28.6	32.7	45.2	49.7	53.7	67.9
PIT (QA → docs) ⑥	32.5	37.2	49.0	54.6	60.0	73.8
PIT ⑦	**45.4**	**51.2**	**63.2**	**62.7**	**68.6**	**78.8**

Table 5.1: Comparison of QA performance (%) between standard instruction-tuning and pre-instruction-tuning. The best results are in bold. Rec. is short for answer recall, and R-L refers to ROUGE-L.

- As shown in Figure 5.5, among all cases where LLMs have minimized perplexity on documents, cases trained with more epochs or larger learning rates typically exhibit superior QA performance. We hypothesize that *more aggressive training leads to less overfitting to deceptive patterns in documents and better generalization when responding to questions.*

In summary, lower perplexity does lead to stronger generalization when responding to questions, but it comes at the expense of forgetting previously acquired knowledge.

5.5 Improving LLMs in Absorbing Knowledge from Documents

The amount of knowledge elicited through the standard instruction-tuning is still limited, even though the perplexity of documents is minimized, a phenomenon we refer to as the "perplexity curse". Our next question is how can we improve the ability of LLMs to absorb knowledge from documents to mitigate the perplexity curse. The main challenge is the gap between the way knowledge is presented in raw documents and how it is accessed through question-answering. We found that QA pairs are generally straightforward, while documents tend to be more com-

Setting names	Setting configurations	EM	Rec.	R-L
	baselines			
continued pre-training ①	test doc	27.6	31.6	43.8
+instruction-tuning ②	train doc + test doc → train QA	30.3	34.7	47.4
+instruction-tuning (w/o forget) ③	train doc + test doc → train QA + test doc	30.2	34.1	46.4
+instruction-tuning (w/o train doc)	test doc → train QA	27.1	30.7	42.3
weighted continued pre-training	test doc (weighted)	27.7	32.7	43.3
adapted continued pre-training	train doc → test doc	26.9	32.7	44.2
mix all data ④	train QA + train doc + test doc	39.4	44.6	56.7
various pre-instruction-tuning (PIT) methods and ablation studies				
PIT ⑦	train QA + train doc (3 epochs) → test doc	45.4	51.2	63.2
	ablation studies of the number of epochs			
	1 epoch	33.3	39.1	50.3
	5 epochs	45.8	52.1	63.6
	10 epochs	46.5	52.3	61.9
	ablation studies of different learning mechanisms			
	QA before doc (grouped)	38.2	43.2	56.3
	QA after doc (grouped)	27.2	31.1	42.1
	QA before doc (interleaved)	45.9	51.3	64.5
	QA after doc (interleaved)	43.2	49.1	61.6
PIT--	train QA + train doc → train QA → test doc	44.4	51.3	63.4
PIT++ ⑧	train QA → train QA + train doc → test doc	**48.1**	**54.4**	**66.4**

Table 5.2: Comparison (%) of various pre-instruction-tuning methods and ablation studies to identify the key contributors to improved performance using Llama-2 7B. Different background colors indicate different pre-instruction-tuning methods. The best results are in bold.

plex and cluttered, weaving many factual statements together in a more intricate manner. Using Figure 5.3 as an example, the answer to the question "who handled the editing of Oppenheimer" is included in a sentence in the middle of the article "Editing was handled by Jennifer Lame ...", which does not explicitly mention "Oppenheimer". During training, LLMs must understand the context and deduce that "editing" refers to "the editing of Oppenheimer" to effectively encode this knowledge in the parameters.

Zhu and Li [260] studied this problem by training a randomly initialized GPT-2-like transformer from scratch on synthetic biographies and evaluated its ability to answer questions about the individuals. They found that training on a mix of biographies and questions related to half of those biographies led to strong generalization when answering questions about the

remaining half of biographies, which resembles setting ④ in Figure 5.4. In contrast, training on biographies and QA pairs sequentially failed. However, the key contributor to the success remains uncertain because the data were blended together, and it is unclear how to apply this practically to absorb knowledge from new documents. Inspired by our observation of the different difficulty levels between QA pairs and documents, and the finding from Zhu and Li [260], we hypothesize that *it is beneficial to deliberately expose LLMs to instruction-tuning data before continued pre-training so that the process of encoding knowledge from complex documents takes into account how this knowledge is accessed.* We refer to this as **pre-instruction-tuning (PIT)** and study various implementations of PIT prior to continued learning (subsection 5.5.1), followed by detailed ablations identifying the keys contributor to performance (subsection 5.5.2 and subsection 5.5.3), and finally assess how well PIT performs across domains (subsection 5.5.4). We adhere to the hyperparameters outlined in subsection 5.3.2 and perform PIT for 3 epochs unless specified otherwise.

5.5.1 Variants of Pre-instruction-tuning

Pre-instruction-tuning w/ QA only We start with exposing instruction-tuning data before continued pre-training on documents—training on topically related QA pairs before training on test documents (Figure 5.4 ⑤). This can be directly compared with the continued pre-training setting (Figure 5.4 ①). The intuition is that questions help LLMs recognize key types of information, enabling LLMs to focus on important information during pre-training on subsequent documents, even though the questions are not directly tied to the documents. For example, training on a question like "who handled the editing of Oppenheimer" could help LLMs pay attention to screenwriters when training on new documents like "Barbie". As shown in Table 5.1, this method outperforms continued pre-training, especially on larger LLMs (27.6%/41.7% → 28.6%/49.7% for 7B/70B). The ablation that trains on QA data after training on documents ("instruction-tuning w/o train doc" in Table 5.2) is ineffective, confirming the importance of training on questions as a warm-up before encoding documents.

Pre-instruction-tuning on QA and documents sequentially Our second implementation trains on QA and associated documents sequentially (Figure 5.4 ⑥), with the intuition that the ability to absorb knowledge from documents can be strengthened if an LLM is trained on the complex documents after it has grasped the associated simpler QA pairs. For instance, if an LLM has already learned that "Jennifer Lame" is the answer to "who handled the editing of Oppenheimer", training on the document "Editing was handled by Jennifer Lame" can more efficiently refine its storage of knowledge in its parameters. As shown in Table 5.1, PIT on QA pairs and documents sequentially surpasses the QA-only variant (Figure 5.4 ⑤) and standard

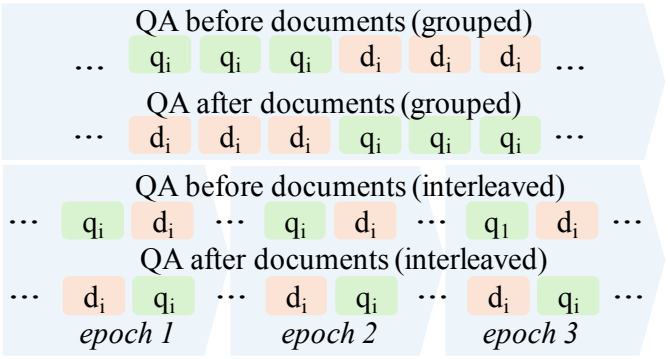


Figure 5.6: Different arrangements between QA pairs and corresponding documents. The ellipses represent other examples.

instruction-tuning (Figure 5.4 ②) (30.3%/46.4% → 32.5%/54.6% for 7B/70B), demonstrating its effectiveness.

Pre-instruction-tuning The effectiveness of PIT depends on ensuring that the associated QA pairs are already learned before encoding the respective documents. However, we observed that after training on documents (train doc in Figure 5.4 ⑥), the accuracy for corresponding questions (train QA in Figure 5.4 ⑥) dropped from almost perfect to 30%, indicating severe forgetting. To fix this, we train on the associated QA pairs and documents together (Figure 5.4 ⑦). As shown in Table 5.1, this significantly improves the performance, outperforming all other approaches, including mixing all data together (Figure 5.4 ④), by a large margin (39.4%/57.1% → 45.5%/62.7% for 7B/70B). Training on both QA pairs and documents prevents forgetting, but it also obscures how the learning process works. It is unclear whether LLMs grasp QA pairs before encoding knowledge from documents, or if it works the other way around. In the following section, we deliberately arrange the order of QA pairs and documents during training to examine this, which leads us to propose an improved version of PIT.

5.5.2 Pre-instruction-tuning++

We first study how the performance varies with different numbers of epochs. As shown in Table 5.2, training for 1 epoch is insufficient, and the performance of 3, 5, or 10 epochs is similar. We fix the number of epochs to 3 and arrange the order of QA pairs and corresponding documents as shown in Figure 5.6. The interleaved arrangement cycles through all the data 3 times, ensuring that in each epoch, questions either precede or follow their associated documents. On the other hand, the grouped arrangement clusters each example's 3 appearances together, guaranteeing that the repeated questions are positioned either before or after their respective

repeated documents. As shown in Table 5.2, positioning QA pairs before corresponding documents achieves better performance in both grouped and interleaved arrangements, indicating that during PIT, the learning mechanism prioritizes understanding how to access knowledge before learning to absorb information from the more complex and information-dense documents.

Based on this, we propose an improved variant called pre-instruction-tuning++, which trains exclusively on QA pairs to understand patterns of knowledge access, and then progresses to training on a combination of QA and document data to align knowledge access through questions and knowledge encoding from documents (Figure 5.4 ⑧). As shown in Table 5.2, PIT++ significantly outperforms PIT (Figure 5.4 ⑦) from 45.4% to 48.1%, while training on QA data after on the mix (PIT-- in Table 5.2) does not yield additional benefits. This reinforces our hypothesis that understanding how knowledge is accessed aids in absorbing knowledge from documents, and therefore, should be taught first.

5.5.3 Ablation Studies

Standard instruction-tuning is inferior not due to forgetting A drawback of standard instruction-tuning is that knowledge in test documents might be forgotten after training on QA pairs (a phenomenon also known as the "alignment tax" [155]). To show that the lower performance of standard instruction-tuning is not due to forgetting, we add a setting where we mix train QA with test documents during instruction-tuning to prevent forgetting (Figure 5.4 ③). As shown in Table 5.2, this does not help, confirming our hypothesis.

Pre-instruction-tuning is not simply upweighting salient tokens from documents We include an ablation inspired by Hu et al. [76] which upweights tokens when pre-training on documents to focus on salient information. We assign a weight of 1.0 to tokens in documents that are included in the answers (e.g., "Jennifer Lame" in the sentence "Editing was handled by Jennifer Lame"), and assign a lower weight of 0.5 to other tokens. As shown in Table 5.2, this weighted continued pre-training is ineffective, confirming our hypothesis.

5.5.4 Cross-domain Generalization

We validated the effectiveness of PIT by training and evaluation on data from the same domain (`Wiki2023-film`). *Can PIT make LLMs better at absorbing knowledge from documents of a different domain?* To this end, we follow the cross-domain setting outlined in Figure 5.2—training on other domains (`Wiki2023-other-train`) and testing on the film domain (`Wiki2023-film-test`). The results of standard instruction-tuning and PIT, in both in-

Settings	Llama-2 7B EM	Rec.	R-L	Llama-2 70B EM	Rec.	R-L
standard instruction-tuning ②						
in-domain	30.3	34.7	47.4	46.4	50.9	64.1
cross-domain	23.6	28.2	38.4	42.8	49.7	58.5
pre-instruction-tuning ⑦						
in-domain	45.4	51.2	63.2	62.7	68.6	78.8
cross-domain	36.9	43.2	54.9	55.2	66.7	74.0

Table 5.3: In-domain and cross-domain PIT.

Settings	EM	Rec.	R-L
generalization to the biography dataset `bioS`			
closed-book	2.9	2.9	11.0
open-book w/ doc	95.2	95.4	95.6
continued pre-training ①	29.6	29.8	38.7
pre-instruction-tuning ⑦	**58.1**	**58.4**	**61.9**
generalization to questions by real users from Google			
standard instruction-tuning ②	21.5	30.1	36.8
pre-instruction-tuning ⑦	**29.0**	**35.5**	**48.2**

Table 5.4: Generalization of the Llama-2 7B model trained with pre-instruction-tuning.

domain and cross-domain settings, are detailed in Table 5.3. Even though it is not as effective as the in-domain counterparts, cross-domain PIT still significantly outperforms instruction-tuning, demonstrating that it can generalize across different domains. This finding sheds light on the potential to scale this method up to a broader range of documents and instructions for more robust generalization.

We also evaluate the effectiveness of PIT in two other scenarios: (1) when applied to non-Wikipedia documents, and (2) when addressing questions asked by real users. For the first scenario, we take the Llama-2 7B model trained with PIT on `2023Wiki-other` and further train it on biographies synthesized in Zhu and Li [260] (`bioS`). Then, we evaluate based on questions about the individuals. For the second scenario, we manually search Google using questions generated by LLMs from `Wiki2023-film-test`, collect a total of 93 similar questions from real users by leveraging Google's "People Also Ask" feature, and then evalu-

ate Llama-2 7B on these questions. As shown in Table 5.4, PIT outperforms baselines in both scenarios, demonstrating its generalization ability.

5.6 Related Work

5.6.1 Continual Knowledge Acquisition

Several works have studied whether LMs can answer questions about information in documents they have been trained on. Hu et al. [76], Jang et al. [85], Wang et al. [226] use relatively small LMs such as BART [123], T5 [170], or GPT-2 [168]. Ovadia et al. [156] focus on the comparison between RAG and continued pre-training approaches without using instruction-tuning. Zhu and Li [260, 261] examine this problem from a similar angle as ours using a GPT-2-like transformer trained from scratch on synthetic biographies and fine-tuned on QA pairs related to the individuals. They examined a mixed training setting on both biographies and QA pairs, which is our major motivation to study different strategies to incorporate QA data before continued pre-training. Other works study adapting LLMs to new domains via various strategies [30, 67, 150, 235, 250, 253].

5.6.2 Instruction-tuning or Alignment

Instruction-tuning (also known as supervised fine-tuning) on high-quality annotated data [79, 112, 144, 184, 199, 200, 231, 256] and/or data generated by proprietary models [32, 78, 204, 230], or alignment with reinforcement learning from human feedback (RLHF) or direct preference optimization (DPO) [155, 169, 209, 213] has been a central topic recently because it elicits knowledge from LLMs and enhances various abilities to handle questions from users. We focus on factuality and study the best way to perform instruction-tuning to elicit factual knowledge from LLMs.

5.6.3 Analyzing the Training Dynamics of LMs

Many works study the training dynamics of LMs from different perspectives. Carlini et al. [24] quantifies memorization across model sizes and the frequency of data duplication. Tirumala et al. [210] finds that larger LMs memorize training data faster with less overfitting. Xia et al. [238] shows that perplexity is more predictive of model behaviors than other factors. Dery et al. [45] studies end-task aware pre-training using classification tasks and RoBERTa models. Jia et al. [87] adds a pre-training objective to encourage the vector for each phrase to have high similarity with the vectors for all questions it answers. Our work differs in that we specifically

focus on the capacity of recalling and generalizing information from a seen document to answer questions.

5.6.4 Retrieval-augmented Generation

Retrieval-augmented generation (RAG) is a widely used approach to incorporate new knowledge into LLMs by augmenting fixed LLMs with retrieved information from external sources [5, 8, 17, 27, 66, 70, 83, 92, 93, 118, 124, 131, 147, 166, 181, 192, 225]. While RAG is effective in reducing hallucinations commonly experienced when relying solely on knowledge stored in parameters, its retrieval and generation process adds extra latency and complexity. In contrast, continued pre-training to store knowledge in parameters and utilizing the stored knowledge to answer questions in a closed-book manner are simpler and faster at inference time. Enhancing this capability is also scientifically significant, as it represents a fundamental step in employing LLMs as dependable assistants for accessing information. Therefore, this chapter focuses on exploring parametric approaches.

5.7 Conclusion

We study the best way of continued training on new documents with the goal of later eliciting factual knowledge. We propose pre-instruction-tuning that learns how knowledge is accessed via QA pairs prior to encoding knowledge from documents. Extensive experiments demonstrate the superiority of pre-instruction-tuning versus standard instruction-tuning. Future directions include scaling this method up to a broader range of documents and instructions for more robust generalization.

Part II

Non-parametric Retrieval-augmented Generation

Chapter 6

Retrieval as Attention: Extending the Context of LMs to the Entire Corpus

We study retrieval-augmented generation (RAG) systems in the following chapters. RAG systems usually consist of retrievers and readers, which necessitates a cumbersome implementation and is hard to train and adapt in an end-to-end fashion. In this chapter, we revisit this design and eschew the separate architecture and training in favor of a single Transformer that performs **Re**trieval as **Att**ention (**ReAtt**), and end-to-end training solely based on supervision from the end QA task. We demonstrate for the first time that a single model trained end-to-end can achieve both competitive retrieval and QA performance, matching or slightly outperforming separately trained retrievers and readers. This work is presented in:

- Zhengbao Jiang*, Luyu Gao*, Jun Araki, Haibo Ding, Zhiruo Wang, Jamie Callan, Graham Neubig. Retrieval as Attention: End-to-end Learning of Retrieval and Reading within a Single Transformer. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing.[1]

Code and models are available at:

- https://github.com/jzbjyb/ReAtt

6.1 Introduction

A widely used solution for knowledge-intensive tasks such as QA is to first *retrieve* a small number of relevant documents from the corpus with a *bi-encoder* architecture which encodes queries and documents independently for efficiency purposes, then *read* the retrieved docu-

[1]Zhengbao Jiang proposed the idea, conducted experiments, and wrote the draft. Luyu Gao refined the idea, suggested efficient implementation, and revised the draft.

ments in a more careful and expansive way with a *cross-encoder* architecture which encodes queries and documents jointly [66, 83, 120, 124]. The distinction between retrieval and reading leads to the widely adopted paradigm of treating retrievers and readers separately. Retrievers and readers are usually two separate models with heterogeneous architectures and different training recipes, which is cumbersome to train. Even though two models can be combined in an ad-hoc way for downstream tasks, it hinders effective end-to-end learning and adaptation to new domains.

There have been several attempts to connect up reader and retriever training [66, 83, 118, 120, 124, 181]. However, retrievers in these works are not learned in a fully end-to-end way. They require either initialization from existing supervised trained dense retrievers [124], or expensive unsupervised retrieval pretraining as warm-up [66, 83, 118, 120, 181]. The reliance on retrieval-specific warm-up and the ad-hoc combination of retrievers and readers makes them less of a unified solution and potentially hinders their domain adaptation ability. With the ultimate goal of facilitating downstream tasks, retriever and reader should instead be fused more organically and learned in a fully end-to-end way.

In this chapter, we focus on one of the most important knowledge-intensive tasks, open-domain QA. We ask the following question: is it possible to perform both retrieval and reading *within a single Transformer model*, and train the model in a *fully end-to-end* fashion to achieve competitive performance from both perspectives? Such a single-model end-to-end solution eliminates the need for retrieval-specific annotation and warm-up and simplifies retrieval-augmented training, making adaptation to new domains easier. Based on the analogy between self-attention which relates different tokens in a single sequence [219] and the goal of retrieval which is to relate queries with relevant documents, we hypothesize that self-attention could be a natural fit for retrieval, and it allows an organic fusion of retriever and reader within a single Transformer.

Specifically, we start from an encode-decoder T5 [170] and use it as both retriever and reader. We use the first B encoder layers as bi-encoder to encode queries and documents independently, and the attention score at layer $B + 1$ (denoted as *retrieval attention*) to compute relevance scores, as shown in Figure 6.1. We found that directly using self-attention for retrieval underperforms strong retrievers, which we conjecture is because self-attention pretrained on local context is not sufficient to identify relevant information in the large representation space of the whole corpus. To solve this, we propose to compute retrieval attention between a query and a large number of documents and *adjust the retrieval attention across documents*. For each query, we compute retrieval attention over both close documents that potentially contain positive and hard negative documents, and documents of other queries in the same batch as random negatives. The retrieval attention is adjusted by minimizing its discrepancy from the cross-attention

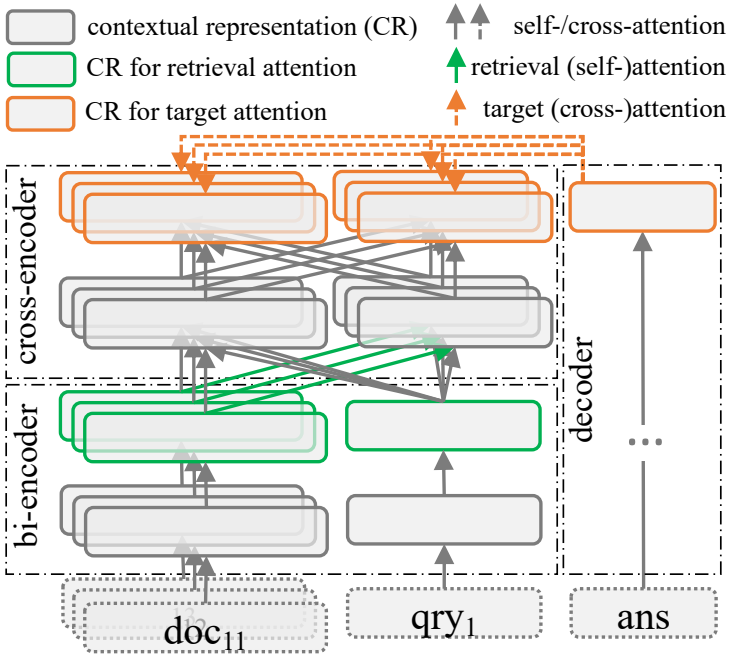


Figure 6.1: Illustration of Retrieval as Attention (ReAtt) with the first B=2 encoder layers as bi-encoder (i.e., retriever) and the rest L-B=2 layers as cross-encoder. During training, the retrieval attention between a query $\boldsymbol{q}_1$ and documents $\boldsymbol{d}_{11,12,13}$ is adjusted by minimizing its discrepancy from the target attention. For simplicity, we use a single arrow to represent the attention of a single head between multiple tokens.

between the decoder and encoder (denoted as *target attention*), which is indicative of the usefulness of each document in generating answers [81]. The resulting **Re**trieval as **Att**ention model (**ReAtt**) is a single T5 trained based on only QA annotations and simultaneously learns to promote useful documents through cross-document adjustment.

We train ReAtt on the Natural Questions dataset (NQ) [114] in a fully end-to-end manner. It achieves both competitive retrieval and QA performance, matching or slightly outperforming ColBERT-NQ [104] trained with explicit retrieval annotations and strong QA model FiD [80, 81], demonstrating for the first time end-to-end training can produce competitive retriever and reader within a single model. To further test ReAtt's generalization and end-to-end adaptation ability, we conduct zero-shot, supervised, and unsupervised adaptation experiments on 7 datasets from the BEIR benchmark [208]. In all settings, end-to-end adaptation improves the retrieval performance usually by a large margin, achieving comparable or superior performance to strong retrieval adaptation and pretraining methods.

6.2 Retrieval as Attention (ReAtt)

With the goal of developing a single Transformer that can perform both retrieval and reading, and the analogy between retrieval and self-attention, we first introduce architecture changes to allow retrieval as attention (subsection 6.2.2), then examine how well attention as-is can be directly used to perform retrieval (subsection 6.2.3).

6.2.1 Formal Definition

We first briefly define the task of retrieval and question answering. As mentioned in the introduction, queries and documents need to be represented independently for efficient retrieval which implies a bi-encoder architecture that has no interaction between queries and documents. Without loss of generality, we use $E_{\boldsymbol{d}} = \text{biencoder}(\boldsymbol{d})$ to denote one or multiple representations generated by a bi-encoder based on a document from a corpus $\boldsymbol{d} \in \mathcal{D}$, and likewise $E_{\boldsymbol{q}} = \text{biencoder}(\boldsymbol{q})$ to denote query representations.[2] The top-k documents most relevant to a query are retrieved by $\mathcal{D}_{\boldsymbol{q}}^{\text{ret}} = \arg\text{topk}_{\boldsymbol{d}\in\mathcal{D}} r(E_{\boldsymbol{q}}, E_{\boldsymbol{d}})$, where function r computes relevance based on query and document representations which can be as simple as a dot product if queries and documents are encoded into a single vector, and $\mathcal{D}_{\boldsymbol{q}}^{\text{ret}}$ stands for the returned documents. We consider encoder-decoder-based generative question answering in this chapter, which jointly represents queries and retrieved documents with the encoder $E_{\boldsymbol{q},\boldsymbol{d}} = \text{crossencoder}(\boldsymbol{q}, \boldsymbol{d})$, and generates the answer $\boldsymbol{a}$ autoregressively with the decoder $P^{\text{gen}}(\boldsymbol{a}|\boldsymbol{q}, \boldsymbol{d}) = P^{\text{gen}}(\boldsymbol{a}|E_{\boldsymbol{q},\boldsymbol{d}})$. To handle multiple retrieved documents, we follow the fusion-in-decoder model (FiD) [80] which encodes each query-document pair independently and fuse these representations in decoder through cross-attention $P^{\text{gen}}(\boldsymbol{a}|\boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}}^{\text{ret}}) = P^{\text{gen}}(\boldsymbol{a}|E_{\boldsymbol{q},\boldsymbol{d}_1}, ..., E_{\boldsymbol{q},\boldsymbol{d}_{|\mathcal{D}_{\boldsymbol{q}}^{\text{ret}}|}})$. Negative log likelihood (NLL) is used in optimization $\mathcal{L}_{\text{QA}} = -\log P^{\text{gen}}(\boldsymbol{a}|\boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}}^{\text{ret}})$.

6.2.2 Leveraging Attention for Retrieval

Next, we introduce our method that directly uses self-attention between queries and documents as retrieval scores.

Putting the Retriever into Transformers As illustrated in Figure 6.1, we choose T5 [170] as our base model, use the first B layers of the encoder as the *bi-encoder "retriever"* by disabling self-attention between queries and documents, and the remaining $L - B$ layers as the *cross-encoder "reader"*. We use the self-attention paid from query tokens to document tokens at the $B + 1$-th layer as the retrieval score, which is denoted as *retrieval attention* (green arrows in

[2] Queries and documents can use different bi-encoders but we use one notation for simplicity.

Figure 6.1). It is computed based on the independent query and document contextual representations from the last (B-th) layer of the bi-encoder (green blocks in Figure 6.1). Formally for an H-head Transformer, document and query representations are:

$$E_{\boldsymbol{d}} = \{K_{\boldsymbol{d}}^{B+1,h} \in \mathbb{R}^{|\boldsymbol{d}| \times e}\}_{h=1}^{H},$$
$$E_{\boldsymbol{q}} = \{Q_{\boldsymbol{q}}^{B+1,h} \in \mathbb{R}^{|\boldsymbol{q}| \times e}\}_{h=1}^{H},$$

where K and Q are key and query vectors of the token sequence used in self-attention, $|\boldsymbol{d}|$ and $|\boldsymbol{q}|$ are document and query length, and e is the dimensionality of each head. The retrieval attention matrix from query tokens to document before softmax for one head is computed by:

$$A_{\boldsymbol{q},\boldsymbol{d}}^{B+1,h} = Q_{\boldsymbol{q}}^{B+1,h} \times K_{\boldsymbol{d}}^{B+1,h^T} \in \mathbb{R}^{|\boldsymbol{q}| \times |\boldsymbol{d}|}.$$

Directly using attention for retrieval can not only leverage its ability to identify relatedness, it is also a natural and simple way to achieve both retrieval and reading in a single Transformer with minimal architectural changes, which facilitates our final goal of end-to-end learning.

From Token Attention to Document Relevance Given the token-level attention scores $A_{\boldsymbol{q},\boldsymbol{d}}^{B+1,h}$, the relevance between $\boldsymbol{q}$ and $\boldsymbol{d}$ is computed by avg-max aggregation: choosing the most relevant document token for each query token (i.e., max) then averaging across query tokens:

$$r_h(\boldsymbol{q}, \boldsymbol{d}) = \text{avg}_0\big(\text{max}_1(A_{\boldsymbol{q},\boldsymbol{d}}^{B+1,h})\big), \tag{6.1}$$

where 1 and 0 refer to the dimension over which the operation is applied. This is similar to the MaxSim and sum operators used in ColBERT [103], with the intuition that a relevant document should match as many query tokens as possible with the best-matching token. The final relevance is a weighted sum over all heads:

$$r(\boldsymbol{q}, \boldsymbol{d}) = \sum_{h=1}^{H} P_h^{\text{head}} \cdot r_h(\boldsymbol{q}, \boldsymbol{d}),$$

where P_h is a learnable weight that sums to one. As explained in the next section, we empirically find only a few attention heads with non-random retrieval performance, and among them, one particular head is significantly better than the others. Given this observation, we introduce a low temperature τ to promote this sparsity $P_h^{\text{head}} = \frac{\exp(w_h/\tau)}{\sum_{h'} \exp\big(w_{h'}/\tau\big)}$, which always ends with *a single head* with the great majority of the weight, which is denoted as *retrieval head* h^*. As a result, the learned head weights are practically a head selector, a fact that can also be exploited to make test-time retrieval more efficient.

End-to-end Retrieval with Attention To perform retrieval over a corpus, we first generate key vectors $K_{\boldsymbol{d}}^{B+1,h^*}$ of retrieval head for all document tokens offline and index them with the FAISS library [94]. For each query token, we issue its vector ($Q_{\boldsymbol{q}}^{B+1,h^*}$) to the index to retrieve top-K' document tokens, which yields a filtered set of documents, each of which has at least one token retrieved by a query token. We then fetch all tokens of filtered documents, compute relevance scores following Equation 6.1, and return top-K documents with the highest scores $r_{h^*}(\boldsymbol{q}, \boldsymbol{d})$. This is similar to the two-stage retrieval in ColBERT [103], and we reuse their successful practice in index compression and search approximation to make test-time retrieval efficient, which we refer to Santhanam et al. [185] for details.

6.2.3 How Good is Attention As-is?

To examine this question, we use T5-large and test queries from the Natural Question dataset (NQ), retrieve 100 documents with BM25, compute relevance scores $r_h(\boldsymbol{q}, \boldsymbol{d})$ with half layers ($B = 12$) as bi-encoder, and measure its correlation with the gold binary annotation. We found that among $H = 24$ heads, 4 heads have non-trivial correlations of 0.137, 0.097, 0.082, and 0.059. We further perform end-to-end retrieval over Wikipedia using the best head, achieving top-10 retrieval accuracy of 43.5%, inferior to 55.5% of BM25. This demonstrates that there are indeed heads that can relate queries with relevant documents, but they are not competitive. We hypothesize that because self-attention is usually trained by comparing and relating tokens in a local context (512/1024 tokens) it cannot effectively identify relevant tokens in the large representation space of a corpus with millions of documents. This discrepancy motivates us to compute retrieval attention between queries and potentially all documents (i.e., attention over the corpus), and *adjust attention across documents to promote useful ones.*

6.3 Learning Retrieval as Attention

We first approximate attention over the corpus at training time by sub-sampling a manageable number of documents for each query containing both potentially relevant and random documents (subsection 6.3.1). Next, we introduce our end-to-end training objective that optimizes a standard QA loss while also adding supervision to promote attention over documents that are useful for the end task (subsection 6.3.2).

6.3.1 Approximate Attention over the Corpus

Encoding the entire corpus and computing attention between the query and all documents is very expensive. To make it practical, we propose to sub-sample a small set of documents for

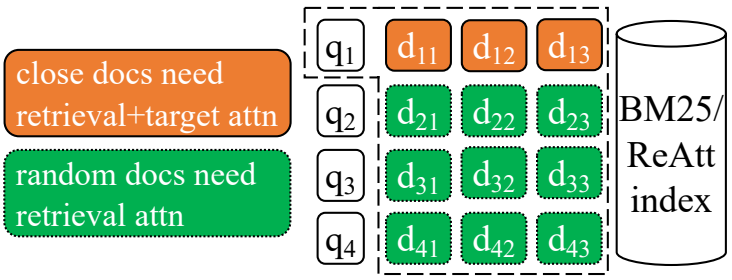


Figure 6.2: Illustration of approximate attention over the corpus with $|\mathcal{Q}|$=4 queries in a batch and K=3 close documents per query. We use $\boldsymbol{q}_1$ as an example to illustrate the required computation, where close documents require both retrieval and target attention while random documents only require retrieval attention.

each query to approximate the whole corpus. Inspired by negative sampling methods used in dense retriever training [99, 103, 239], we sub-sample both (1) documents close to queries that can be either relevant or hard negatives, and (2) random documents that are most likely to be easy negatives. This allows the model to distinguish between relevant and hard negative documents, while simultaneously preventing it from losing its ability to distinguish easy negatives, which form the majority of the corpus.

Iterative Close Document Sub-sampling To sample documents close to a query $\mathcal{D}_{\boldsymbol{q}}^{\text{close}}$, we start from widely used lexical retriever BM25 [179] to retrieve $K = 100$ documents, as shown by the orange blocks in Figure 6.2. We set K to a relatively large number to better approximate the local region, inspired by Izacard and Grave [80]'s findings that QA performance increases as more documents are used.

This fixed set of close documents can become outdated and no longer close to the query anymore as the retrieval attention gets better. To provide dynamic close sub-samples, we re-index the corpus and retrieve a new set of K documents using the current retrieval attention after each iteration. It is similar in spirit to the hard negative mining methods used in Karpukhin et al. [99], Khattab et al. [104], with a major difference that we do not manually or heuristically annotate documents but instead learn from the end loss with cross-document adjustment, which will be explained in subsection 6.3.2.

In-batch Random Document Sub-sampling We use close documents of other queries in the same batch as the random documents of the current query $\mathcal{D}_{\boldsymbol{q}}^{\text{random}} = \cup_{\boldsymbol{q}' \in \mathcal{Q} \wedge \boldsymbol{q}' \neq \boldsymbol{q}} \mathcal{D}_{\boldsymbol{q}'}^{\text{close}}$ where $\mathcal{Q}$ contains all queries in a batch, as shown by the green blocks in Figure 6.2, which has the advantage of reusing document representations across queries. This is similar to the in-batch negatives used in DPR [99] with a major difference that we reuse token representations

$(K_{\boldsymbol{d}}^{B+1,h}, 1 \leq h \leq H)$ across queries instead of a single-vector document representation.

6.3.2 Cross-document Adjustment with Decoder-to-Encoder Attention Distillation

Given the sub-sampled $|\mathcal{Q}| \times K$ documents $\mathcal{D}_{\boldsymbol{q}} = \mathcal{D}_{\boldsymbol{q}}^{\text{close}} \cup \mathcal{D}_{\boldsymbol{q}}^{\text{random}}$ for each query $\boldsymbol{q}$, we compute the retrieval attention-based relevance scores $r(\boldsymbol{q}, \boldsymbol{d})$ and adjust them across multiple documents $\boldsymbol{d} \in \mathcal{D}_{\boldsymbol{q}}$ only relying on end task supervision. Since retrieval is simply a means to achieve the downstream task, documents useful for downstream tasks should be promoted by retrieval. Inspired by reader-to-retriever distillation [81, 241], we measure document usefulness based on cross-attention between decoder and encoder, and minimize retrieval attention's discrepancy from it through distillation. In contrast to Izacard and Grave [81] that learns two models iteratively and alternatively, we optimize QA and distillation loss in a single model simultaneously.

Minimizing KL-divergence Between Retrieval and Target Attention Specifically, we denote cross-attention before softmax of the first position/token of the last decoder layer as *target attention* $C_{\boldsymbol{a},\boldsymbol{q},\mathcal{D}_{\boldsymbol{q}}} \in \mathbb{R}^{H \times |\mathcal{D}_{\boldsymbol{q}}| \times (|\boldsymbol{d}|+|\boldsymbol{q}|)}$ where $\boldsymbol{a}$ is the answer, $|\mathcal{D}_{\boldsymbol{q}}|$ is the number of sub-sampled documents to be fused by the decoder (subsection 6.2.1), and $|\boldsymbol{d}|$ is document length.[3] To aggregate token-level target attention into document-level distribution $P^{\text{tgt}}(\boldsymbol{a}, \boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}}) \in \mathbb{R}^{|\mathcal{D}_{\boldsymbol{q}}|}$, we first perform softmax over all tokens in all query-document pairs ($|\mathcal{D}_{\boldsymbol{q}}| \times (|\boldsymbol{d}| + |\boldsymbol{q}|)$), sum over tokens of each query-document pair ($|\boldsymbol{d}| + |\boldsymbol{q}|$), then average across multiple heads (H):

$$P^{\text{tgt}}(\boldsymbol{a}, \boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}}) = \text{avg}_0\Big(\text{sum}_2\big(\text{softmax}_{1,2}(C_{\boldsymbol{a},\boldsymbol{q},\mathcal{D}_{\boldsymbol{q}}})\big)\Big).$$

Given relevance scores obtained from retrieval attention, the final cross-document adjustment loss is the KL-divergence between relevance distribution P^{ret} and target distribution P^{tgt}:

$$\begin{aligned} P^{\text{ret}}(\boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}}) &= \text{softmax}\big(r(\boldsymbol{q}, \boldsymbol{d}_1), ..., r(\boldsymbol{q}, \boldsymbol{d}_{|\mathcal{D}_{\boldsymbol{q}}|})\big). \\ \mathcal{L}_{\text{cross-doc}} &= \text{KL}\Big(\overline{P^{\text{tgt}}(\boldsymbol{a}, \boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}})} \Big\| P^{\text{ret}}(\boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}})\Big), \end{aligned} \tag{6.2}$$

where the overline indicates stop gradient back propagation to target distributions. Our final loss combines QA loss and cross-document adjustment loss with α as combination weight.

$$\mathcal{L} = \mathcal{L}_{\text{QA}} + \alpha \cdot \mathcal{L}_{\text{cross-doc}}. \tag{6.3}$$

[3] We also attempted other variations of target attention and found performances are similar, consistent with observations in Izacard and Grave [81].

Zero Target Attention for Random Documents For a batch with $|\mathcal{Q}|$ queries, we need to compute retrieval attention and target attention between $|\mathcal{Q}| \times |\mathcal{Q}| \times K$ query-document pairs. This is both computation- and memory-intensive when batch size is large, especially for target attention because it requires $L - B$ layers of joint encoding of query-document pairs in the cross-encoder. To alleviate this, we make a simple and effective assumption that in-batch random documents are not relevant to the current query thus having zero target attention: $P^{\text{tgt}}(\boldsymbol{a}, \boldsymbol{q}, \mathcal{D}_{\boldsymbol{q}}^{\text{random}}) \in \mathbb{R}^{|\mathcal{D}_{\boldsymbol{q}}^{\text{random}}|} \leftarrow 0$. As a result, we only need to run cross-encoder and decoder for K close documents of each query, as shown in Figure 6.2.

6.3.3 Domain Adaptation Methods

One of the major benefits of a single end-to-end trainable model is that given a new corpus from a new domain, possibly without retrieval annotations, we can easily adapt it by end-to-end training. This section describes how we adapt ReAtt under different setups.

We consider adapting ReAtt with (1) QA supervision, (2) information retrieval (IR) supervision, or (3) unsupervised adaptation where we only have access to the document corpus. Although our goal is to learn retrieval through downstream tasks instead of retrieval supervision, being able to consume retrieval annotations is helpful when retrieval supervision is indeed available. To do so, we convert retrieval task with annotations in the form of query-document-relevance triples $\langle \boldsymbol{q}, \boldsymbol{d}, l \rangle$ into a generative task: given a query, the target is to generate *its relevant document and the corresponding relevance* with the following format "relevance: l. $\boldsymbol{d}$", similar to generative retrievers such as DSI [205] and SEAL [14]. If a query has multiple relevant documents, we follow Izacard and Grave [80] to randomly sample one of them. For unsupervised adaptation, with simplicity as our primary goal, we randomly choose one sentence from a document and mask one entity, which is considered as the "query", and have our model generate the masked entity as the "answer", similar to salient span masking (SSM) used in Guu et al. [66].

6.4 In-domain Experiments

In this section, we examine if supervised training ReAtt end-to-end with *only* QA supervision yields both competitive retrieval and QA performance.

Datasets, Baselines, and Metrics We train our model using the Natural Questions dataset (NQ). We compare retrieval performance with lexical models BM25 [179], passage-level dense retrievers DPR, ANCE, coCondenser, FiD-KD, YONO (with and without retrieval pretraining)

[54, 81, 99, 118, 152, 239], and token/phrase-level dense retrievers DensePhrase, ColBERT, ColBERT-NQ [103, 104, 119].[4] Among them ColBERT-NQ, FiD-KD and YONO are the most fair-to-compare baselines because of either similar token-level retrieval granularity (ColBERT-NQ) or similar end-to-end training settings (FiD-KD and YONO). We report top-k retrieval accuracy (R@k), the fraction of queries with at least one retrieved document containing answers. We compare QA performance with ORQA, REALM, RAG, FiD, EMDR^2, YONO, UnitedQA, and R2-D2 [31, 51, 66, 80, 81, 118, 120, 124, 181] using exact match (EM), among which FiD, EMDR^2, and YONO are the most fair-to-compare baselines because they have similar model sizes and training settings.

6.5 Implementation Details of ReAtt

ReAtt is based on T5-large with $B = 12$ encoder layers as bi-encoder and temperatures $\tau = 0.001$ to select the best retrieval head. We retrieve $K = 100$ close documents for each query, and use a batch size of $|\mathcal{Q}| = 64$ queries to obtain in-batch random documents. We use $\alpha = 8$ to combine cross-document adjustment loss with QA loss. We use AdamW with a learning rate of 5e-5, 10% steps of warmup, and linear decay. We first warmup cross-attention's ability to distinguish documents by only using the QA loss for 3K steps, then train with the combined losses (Equation 6.3) for 4 iterations, where the first iteration uses close documents returned by BM25, and the following 3 iterations use close documents returned by the previous ReAtt model (denoted as ReAtt $_{\text{BM25}}$). Each iteration has 8K update steps and takes $\sim$ 1.5 days on a single node with 8 $\times$ A100 GPUs with 80GB memory. Since DPR [99] achieves stronger performance than BM25, training with close documents returned by DPR can potentially reduce training time. We experimented with training on close documents from DPR for a single iteration with 16K steps (denoted as ReAtt $_{\text{DPR}}$). Since both approaches achieve similar performance (Table 6.1 and Table 6.2) and ReAtt $_{\text{DPR}}$ is cheaper to train, we use it in other experimental settings.

At test-time, we save key vectors of all tokens in the corpus and use exact index from FAISS (i.e., `faiss.IndexFlatIP`) to perform inner-product search. We retrieve $K' = 2048$ document tokens for each query token and return top-100 documents with the highest aggregated scores (Equation 6.1) to generate answers. We found compressing index with clustering and quantization proposed by Santhanam et al. [185] can greatly reduce search latency and index size with a minor retrieval accuracy loss.

Models	R@1	R@5	R@20	R@100	#Params.
BM25	23.9	45.9	63.8	78.9	-
supervised retrievers					
DPR	45.9	68.1	80.0	85.9	220M
DPR^{new}	52.5	72.2	81.3	87.3	220M
DPR-PAQ	-	74.2	84.0	89.2	220M
ANCE	-	-	81.9	87.5	220M
coCondenser	-	75.8	84.3	89.0	220M
DensePhrase	51.1	69.9	78.7	-	330M
ColBERT	-	-	79.1	-	110M
ColBERT-NQ	54.3	75.7	85.6	90.0	110M
semi/unsupervised retrievers					
FiD-KD	49.4	73.8	84.3	89.3	220M
YONO$_{\text{w/o PT}}$	-	-	72.3	82.2	165M
YONO$_{\text{w/ PT}}$	-	75.3	85.2	90.2	165M
ReAtt $_{\text{DPR}}$	54.6	77.2	**86.1**	**90.7**	165M
ReAtt $_{\text{BM25}}$	**55.8**	**77.4**	86.0	90.4	165M

Table 6.1: Retrieval performance on NQ. PT is retrieval pretraining. Fair-to-compare baselines are highlighted with background color. The best performance is in bold.

6.5.1 Overall Results

We compare ReAtt with various retrievers and readers in Table 6.1 and Table 6.2. ReAtt achieves both slightly better retrieval performance than the strongest retriever baseline ColBERT-NQ [104] and comparable QA performance than the strong reader baseline FiD-KD [81] on NQ, demonstrating for the first time that fully end-to-end training using QA supervision can produce both competitive retrieval and QA performance. Compared to another single-model architecture YONO [118], ReAtt offers better performance without cumbersome pretraining to warm-up retrieval.

[4]ColBERT is trained on MS MARCO, ColBERT-NQ is on NQ.

Models	EM	#Params.
ORQA [120]	33.3	330M
REALM [66]	40.4	330M
RAG [124]	44.5	220M
FiD [80]	51.4	990M
FiD-KD [81]	54.4	990M
EMDR^2 [181]	52.5	440M
YONO$_{\text{w/o PT}}$ [118]	42.4	440M
YONO$_{\text{w/ PT}}$ [118]	53.2	440M
UnifiedQA [102]	49.3	11B
UnitedQA [31]	54.7	1.870B
R2-D2 [51]	**55.9**	1.290B
ReAtt $_{\text{DPR}}$	54.0	770M
ReAtt $_{\text{BM25}}$	54.7	770M

Table 6.2: QA performance on NQ. PT is retrieval pretraining. Fair-to-compare baselines are highlighted. The best performance is in bold.

6.6 Out-of-domain Generalization and Adaptation

In this section, we examine both zero-shot retrieval performance on out-of-domain datasets and ReAtt's end-to-end adaptability in supervised (QA, IR) and unsupervised settings.

6.6.1 Datasets, Baselines, and Metrics

We choose 7 datasets from BEIR [208], a benchmark covering diverse domains and tasks. On each dataset, we compare ReAtt with different types of retrievers including BM25, DPR, and ColBERT. We consider 2 QA datasets (BioASQ and FiQA [137, 217]) and one IR dataset (MS MARCO [149]) to evaluate supervised adaptation capability, and 4 other datasets (CQADupStack, TREC-COVID, SCIDOCS, SciFact [36, 74, 220, 222]) to evaluate unsupervised adaptation capability. We report nDCG@10 to measure retrieval performance and EM to measure QA performance. We group all baselines into three categories and denote them with different colors in the following tables:

- **Supervised adaptation models** are trained with downstream task supervision, including RAG trained on BioASQ, Contriever fine-tuned on FiQA, and docT5query, ANCE, ColBERT, and Contriever fine-tuned on MS MARCO [82, 103, 151, 239].

Tasks	QA		Retrieval
Datasets	BioASQ	FiQA	MS MARCO
zero-shot performance			
BM25	68.1	23.6	22.8
DPR	14.1	11.2	17.7
ColBERT-NQ	65.5	23.8	**32.8**
ReAtt	**71.1**	**30.1**	32.3
additional training			
Contriever	-	32.9	docT5query 33.8
SimCSE	58.1	31.4	ANCE 38.8
TSDAE+GPL	61.6	34.4	ColBERT 40.1
$\text{Contriever}_{\text{w/ FT}}$	-	38.1	Contriever **40.7**
ReAtt	$_{+5.8}$**76.9**	$_{+8.5}$**38.6**	ReAtt $_{+7.6}$39.9

Table 6.3: nDCG@10 of zero-shot and supervised adaptation experiments on two QA and one IR datasets. We use colors to denote categories: pretraining, unsupervised adaptation, and supervised adaptation. Baselines comparable to ReAtt are highlighted with blue background color. We also show the improvement of ReAtt over zero-shot performance in subscript.

- **Unsupervised adaptation models** are trained on domain corpus in an unsupervised way such as contrastive learning or pseudo query generation, including SimCSE and TSDAE+GPL [56, 227, 228].
- **Pretraining models** are trained on corpora without direct exposure to the target domain, such as Contriever [82] trained with contrastive learning on Wikipedia and CCNet.

We highlight baselines in the same category as ReAtt in the following tables since comparison between them is relatively fair.

6.6.2 Experimental Results

Results of supervised and unsupervised adaptation are listed in Table 6.3 and Table 6.4 respectively.

Zero-shot Generalization Ability As shown in Table 6.3 and Table 6.4, the zero-shot performance of ReAtt is significantly better than other zero-shot baselines on two QA datasets and one fact checking dataset (+3.0/+6.5/+4.5 on BioASQ/FiQA/SciFact than the second best), and overall comparable on the rest of datasets (-0.5/-0.6/-3.0/-1.0 on MS MARCO/CQA./TRECC./SCIDOCS

Methods	CQA.	TRECC.	SCIDOCS	SciFact
zero-shot performance				
BM25	29.9	**65.6**	**15.8**	66.5
DPR	15.3	33.2	7.7	31.8
ANCE	29.6	65.4	12.2	50.7
ColBERT-NQ	**33.9**	48.9	15.6	65.3
ReAtt	33.3	62.6	14.8	**71.0**
additional training				
Contriever	34.5	59.6	**16.5**	67.7
SimCSE	29.0	68.3	-	55.0
TSDAE+GPL	35.1	74.6	-	68.9
ReAtt	$_{+3.3}$**36.6**	$_{+13.4}$**76.0**	$_{+1.0}$15.8	$_{+0.2}$**71.2**

Table 6.4: nDCG@10 of zero-shot and unsupervised adaptation on four datasets. Format is similar to Table 6.3

than the best which is usually BM25), demonstrating that our end-to-end training with QA loss on NQ produces a robust retriever. We conjecture that the superior performance on QA datasets can be attributed to our end-to-end training using QA loss which learns retrieval that better aligns with the end task than training with retrieval annotations.

Retrieval Adaptation with QA Supervision As shown in the left-hand side of Table 6.3, end-to-end adaptation with QA supervision significantly improves ReAtt's retrieval performance by 5.8/8.5 on BioASQ/FiQA, achieving similar performance as Contriever fine-tuned on FiQA, and better performance than other unsupervised methods, confirming the end-to-end adaptability of our methods.

Leveraging Retrieval Annotations As shown on the right-hand side of Table 6.3, ReAtt is able to consume retrieval supervision in a generative format and achieve competitive performance as other supervised dense retrievers.

Unsupervised Adaptation with SSM As shown in Table 6.4, adaptation by simply masking salient entities from sentences as input and generating masked entities using ReAtt improves the retrieval performance on 4 datasets, some by a large margin, achieving comparable or superior performance than strong retrieval adaptation methods such as TSDAE+GPL that relies

on query generation. This indicates that our end-to-end trainable model also works well in unsupervised settings without involving too many engineering heuristics.

6.7 Conclusion

We propose retrieval as attention (ReAtt), a single Transformer model that can be learned in an end-to-end fashion only using end task loss. We demonstrated on the NQ dataset that ReAtt can achieve both competitive retrieval and QA performance. We further show that ReAtt is easy to adapt to other domains in both supervised and unsupervised settings, achieving both boosted retrieval and end task performance. Future directions include better end-to-end training objectives and efficient training and inference methods.

Chapter 7

Active Retrieval-augmented Generation

The previous chapter mainly focuses on short-form questions, which employ a retrieve-and-generate setup that only retrieves information once based on the input. This is limiting, however, in more general scenarios involving generation of long texts, where continually gathering information throughout generation is essential. In this chapter, we provide a generalized view of *active retrieval augmented generation*, methods that actively decide when and what to retrieve across the course of the generation. This work is presented in:

- Zhengbao Jiang*, Frank F. Xu*, Luyu Gao*, Zhiqing Sun*, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, Graham Neubig. Active Retrieval Augmented Generation. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing.[1]

Code and datasets are available at:

- https://github.com/jzbjyb/FLARE

7.1 Introduction

Retrieval augmented LMs commonly use a retrieve-and-generate setup where they retrieve documents based on the user's input, and then generate a complete answer conditioning on the retrieved documents [27, 66, 80, 83, 92, 117, 118, 124, 147, 165, 181, 192]. These single-time retrieval augmented LMs outperform purely parametric LMs, particularly for short-form knowledge-intensive generation tasks such as factoid question answering (QA) [95, 114], where *the information needs are clear in the user's input, and it is sufficient to retrieve relevant knowledge once solely based on the input.*

[1] Zhengbao Jiang proposed the idea, conducted main experiments, and wrote the draft. Frank F. Xu and Zhiqing Sun designed retrieval instructions and conducted ablations. Luyu Gao revised the draft.

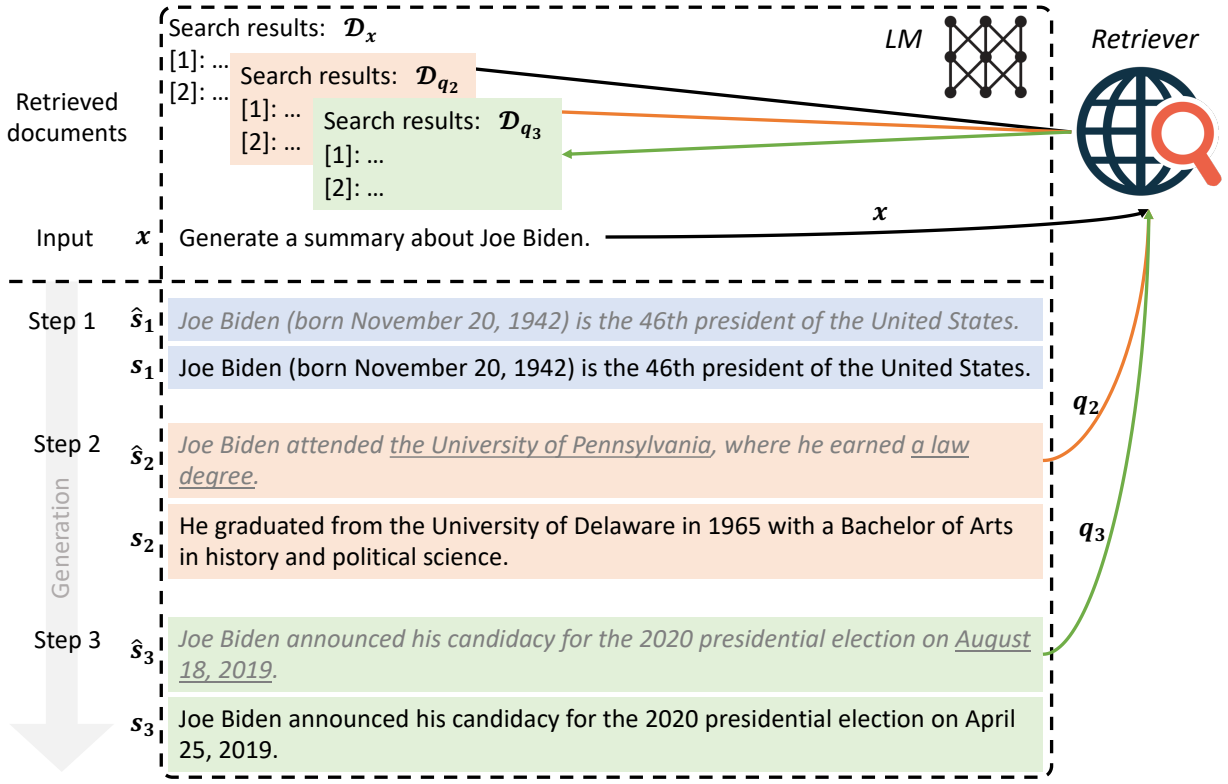


Figure 7.1: An illustration of forward-looking active retrieval augmented generation (FLARE). Starting with the user input $\boldsymbol{x}$ and initial retrieval results $\mathcal{D}_{\boldsymbol{x}}$, FLARE iteratively generates a temporary next sentence (shown in *gray italic*) and check whether it contains low-probability tokens (indicated with underline). If so (step 2 and 3), the system retrieves relevant documents and regenerates the sentence.

Increasingly powerful large LMs have also demonstrated abilities in more complex tasks that involve generating long-form output, such as long-form QA [52, 196], open-domain summarization [37, 61, 69], and (chain-of-thought; CoT) reasoning [58, 71, 73, 233]. In contrast to short-form generation, long-form generation presents complex information needs that are *not always evident from the input alone*. Similar to how humans gradually gather information as we create content such as papers, essays, or books, long-form generation with LMs would *require gathering multiple pieces of knowledge throughout the generation process*. For example, to generate a summary about a particular topic, the initial retrieval based on the topic name (e.g., Joe Biden) may not cover all aspects and details. It is crucial to retrieve extra information as needed during generation, such as when generating a certain aspect (e.g., Joe Biden's education history) or a specific detail (e.g., the date of Joe Biden's presidential campaign announcement).

Several attempts have been made to retrieve multiple times throughout generation. These attempts include methods that passively use the past context to retrieve additional informa-

tion at a fixed interval [17, 101, 174, 216] which might not accurately reflect what LMs intend to generate in the future or retrieve at inappropriate points. Some works in multihop QA decompose the full question into sub-questions, each of which is used to retrieve extra information [105, 107, 164, 244].

We ask the following question: can we create a simple and generic retrieval augmented LM that *actively decides when and what to retrieve* throughout the generation process, and are applicable to a variety of long-form generation tasks? We provide a generalized view of active retrieval augmented generation. Our hypothesis regarding *when to retrieve* is that LMs should retrieve information only when they lack the required knowledge to avoid unnecessary or inappropriate retrieval that occurs in passive retrieval augmented LMs [17, 101, 174, 216]. Given the observation that large LMs tend to be well-calibrated and low probability/confidence often indicates a lack of knowledge [96], we adopt an active retrieval strategy that only retrieves when LMs generate low-probability tokens. When deciding *what to retrieve*, it is important to consider what LMs intend to generate in the future, as the goal of active retrieval is to benefit future generations. Therefore, we propose anticipating the future by generating a temporary next sentence, using it as a query to retrieve relevant documents, and then regenerating the next sentence conditioning on the retrieved documents. Combining the two aspects, we propose **F**orward-**L**ooking **A**ctive **RE**trieval augmented generation (**FLARE**), as illustrated in Figure 7.1. FLARE iteratively generates *a temporary next sentence*, uses it as the query to retrieve relevant documents *if it contains low-probability tokens*, and regenerates the next sentence until reaches the end.

FLARE is applicable to any existing LMs at inference time without additional training. Considering the impressive performance achieved by GPT-3.5 [155] on a variety of tasks, we examine the effectiveness of our methods on `text-davinci-003`. We evaluate FLARE on 4 diverse tasks/datasets involving generating long outputs, including multihop QA (2WikiMultihopQA), commonsense reasoning (StrategyQA), long-form QA (ASQA), and open-domain summarization (WikiAsp) [58, 69, 73, 196]. Over all tasks, FLARE achieves superior or competitive performance compared to single-time and multi-time retrieval baselines, demonstrating the effectiveness and generalizability of our method.

7.2 Retrieval Augmented Generation

We formally define single-time retrieval augmented generation and propose the framework of active retrieval augmented generation.

7.2.1 Notations and Definitions

Given a user input $\boldsymbol{x}$ and a document corpus $\mathcal{D} = \{\boldsymbol{d}_i\}_{i=1}^{|\mathcal{D}|}$ (such as all Wikipedia articles), the goal of retrieval augmented LMs is to generate the answer $\boldsymbol{y} = [\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_m] = [w_1, w_2, ..., w_n]$ containing m sentences or n tokens leveraging information retrieved from the corpus.

In retrieval augmented LM, the LM typically pairs with a retriever that can retrieve a list of documents $\mathcal{D}_{\boldsymbol{q}} = \text{ret}(\boldsymbol{q})$ for a query $\boldsymbol{q}$; the LM conditions on both the user input $\boldsymbol{x}$ and retrieved documents $\mathcal{D}_{\boldsymbol{q}}$ to generate the answer. Since we focus on examining various methods of determining when and what to retrieve, we follow existing methods [174, 216] to prepend the retrieved documents before the user input to aid future generation for both baselines and our method for fair comparisons: $\boldsymbol{y} = \text{LM}([\mathcal{D}_{\boldsymbol{q}}, \boldsymbol{x}])$, where $[\cdot, \cdot]$ is concatenation following the specified order.

7.2.2 Single-time Retrieval Augmented Generation

The most common choice is to directly use the user input as the query for retrieval and generate the complete answer at once $\boldsymbol{y} = \text{LM}([\mathcal{D}_{\boldsymbol{x}}, \boldsymbol{x}])$.

7.2.3 Active Retrieval Augmented Generation

To aid long-form generation with retrieval, we propose active retrieval augmented generation. It is a generic framework that actively decides when and what to retrieve through the generation process, resulting in the interleaving of retrieval and generation. Formally, at step $t (t \geq 1)$, the retrieval query $\boldsymbol{q}_t$ is formulated based on both the user input $\boldsymbol{x}$ and previously generated output $\boldsymbol{y}_{<t} = [\boldsymbol{y}_0, ..., \boldsymbol{y}_{t-1}]$:

$$\boldsymbol{q}_t = \text{qry}(\boldsymbol{x}, \boldsymbol{y}_{<t}),$$

where $\text{qry}(\cdot)$ is the query formulation function. At the beginning ($t = 1$), the previous generation is empty ($\boldsymbol{y}_{<1} = \emptyset$), and the user input is used as the initial query ($\boldsymbol{q}_1 = \boldsymbol{x}$). Given retrieved documents $\mathcal{D}_{\boldsymbol{q}_t}$, LMs continually generate the answer until the next retrieval is triggered or reaches the end:

$$\boldsymbol{y}_t = \text{LM}([\mathcal{D}_{\boldsymbol{q}_t}, \boldsymbol{x}, \boldsymbol{y}_{<t}]),$$

where $\boldsymbol{y}_t$ represents the generated tokens at the current step t, and the input to LMs is the concatenation of the retrieved documents $\mathcal{D}_{\boldsymbol{q}_t}$, the user input $\boldsymbol{x}$, and the previous generation $\boldsymbol{y}_{<t}$. We discard previously retrieved documents $\cup_{t'<t}\mathcal{D}_{\boldsymbol{q}_{t'}}$ and only use the retrieved documents from the current step to condition the next generation to prevent reaching the input length limit of LMs.

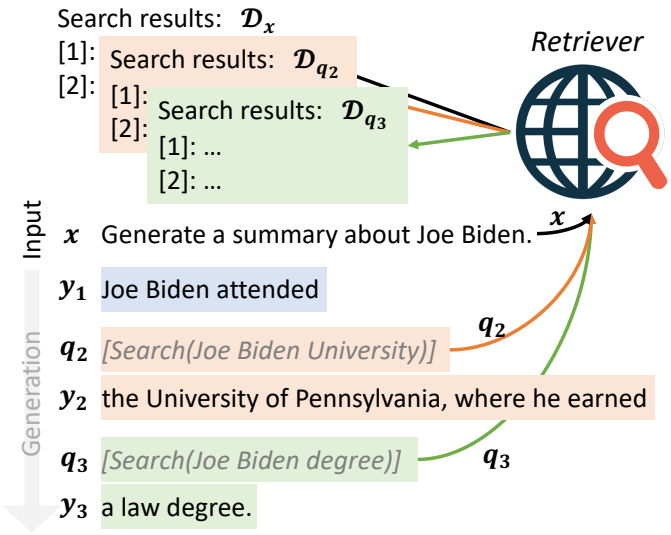


Figure 7.2: An illustration of forward-looking active retrieval augmented generation with retrieval instructions ($\text{FLARE}_{\text{instruct}}$). It iteratively generates search queries (shown in *gray italic*) to retrieve relevant information to aid future generations.

7.3 FLARE: Forward-Looking Active REtrieval Augmented Generation

Our intuition is that (1) LMs should only retrieve information when they do not have the necessary knowledge to avoid unnecessary or inappropriate retrieval, and (2) the retrieval queries should reflect the intents of future generations. We propose two forward-looking active retrieval augmented generation (FLARE) methods to implement the active retrieval augmented generation framework. The first method prompts the LM to generate retrieval queries when necessary while generating the answer using retrieval-encouraging instructions, denoted as $\text{FLARE}_{\text{instruct}}$. The second method directly uses the LM's generation as search queries, denoted as $\text{FLARE}_{\text{direct}}$, which iteratively generates the next sentence to gain insight into the future topic, and if uncertain tokens are present, retrieves relevant documents to regenerate the next sentence.

7.3.1 FLARE with Retrieval Instructions

Inspired by Toolformer [189], a straightforward way of expressing information needs for retrieval is to generate "[Search(query)]" when additional information is needed [189], e.g., "The colors on the flag of Ghana have the following meanings. Red is for [Search(Ghana flag red meaning)] the blood of martyrs, ..." When working with GPT-3.5 models that offer only API

access, we elicit such behavior by few-shot prompting [21].

Specifically, for a downstream task, we place the search-related instruction and exemplars at the beginning as skill 1, followed by the instruction and exemplars of the downstream task as skill 2. Given a test case, we ask LMs to combine skills 1 and 2 to generate search queries while performing the task. The structure of the prompt is shown in Prompt 2.

Prompt 2: retrieval instructions

Skill 1. An instruction to guide LMs to generate search queries.
Several search-related exemplars.

Skill 2. An instruction to guide LMs to perform a specific downstream task (e.g., multihop QA).
Several task-related exemplars.

An instruction to guide LMs to combine skills 1 and 2 for the test case.
The input of the test case.

As shown in Figure 7.2, when the LM generates "[Search(query)]" (shown in *gray italic*), we stop the generation and use the query terms to retrieve relevant documents, which are prepended before the user input to aid future generation until the next search query is generated or reaches the end.

7.3.2 Direct FLARE

Since we cannot fine-tune black-box LMs, we found queries generated by $\text{FLARE}_{\text{instruct}}$ through retrieval instructions might not be reliable. Therefore, we propose a more direct way of forward-looking active retrieval that uses the next sentence to decide when and what to retrieve.

Confidence-based Active Retrieval

As shown in Figure 7.1, at step t, we first generate a temporary next sentence $\hat{\boldsymbol{s}}_t = \text{LM}([\boldsymbol{x}, \boldsymbol{y}_{<t}])$ without conditioning on retrieved documents. Then we decide whether to trigger retrieval and formulate queries based on $\hat{\boldsymbol{s}}_t$. If the LM is confident about $\hat{\boldsymbol{s}}_t$, we accept it without retrieving additional information; if not, we use $\hat{\boldsymbol{s}}_t$ to formulate search queries $\boldsymbol{q}_t$ to retrieve relevant documents, and then regenerate the next sentence $\boldsymbol{s}_t$. The reason we utilize sentences as the basis of our iteration is due to their significance as semantic units that are neither too short nor too lengthy like phrases and paragraphs. However, our approach can also utilize phrases or paragraphs as the basis.

Since LMs tend to be well-calibrated so that low probability/confidence often indicates a lack of knowledge [91, 96, 218], we actively trigger retrieval if any token of $\hat{\boldsymbol{s}}_t$ has a probability

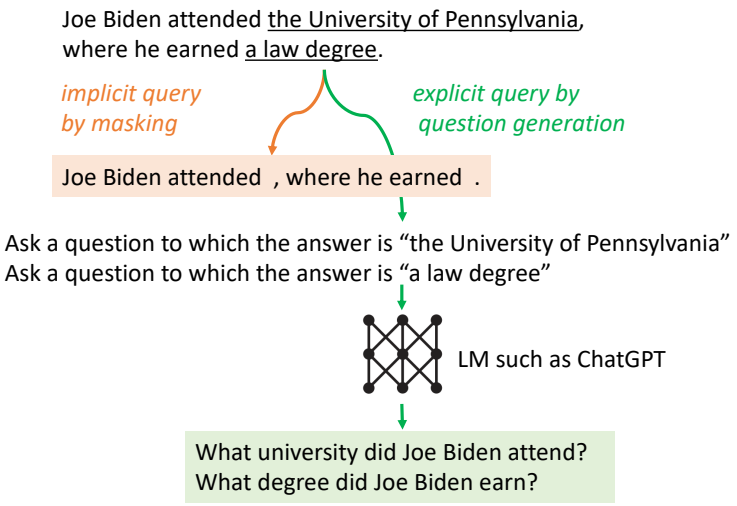


Figure 7.3: Implicit and explicit query formulation. Tokens with low probabilities are marked with underlines.

lower than a threshold $\theta \in [0, 1]$. $\theta = 0$ means retrieval is never triggered, while $\theta = 1$ triggers retrieval every sentence.

$$\boldsymbol{y}_t = \begin{cases} \hat{\boldsymbol{s}}_t & \text{if all tokens of } \hat{\boldsymbol{s}}_t \text{ have probs} \geq \theta \\ \boldsymbol{s}_t = \text{LM}([\mathcal{D}_{\boldsymbol{q}_t}, \boldsymbol{x}, \boldsymbol{y}_{<t}]) & \text{otherwise} \end{cases}$$

where the query $\boldsymbol{q}_t$ is formulated based on $\hat{\boldsymbol{s}}_t$.

Confidence-based Query Formulation

One way to perform retrieval is to directly use the next sentence $\hat{\boldsymbol{s}}_t$ as the query $\boldsymbol{q}_t$. This shares a similar spirit with methods that use generated hypothetical titles or paragraphs from LMs as retrieval queries or evidence [55, 140, 198, 245]. We generalize such techniques for long-form generation where active information access is essential.

We found retrieving with the next sentence achieves significantly better results than with the previous context, as shown later in subsection 7.6.2. However, it has a risk of perpetuating errors contained in it. For example, if the LM produces the sentence "Joe Biden attended the University of Pennsylvania" instead of the correct fact that he attended the University of Delaware, using this erroneous sentence as a query might retrieve misleading information. We propose two simple methods to overcome this issue as illustrated in Figure 7.3.

Masked sentences as implicit queries. The first method masks out low-confidence tokens in $\hat{\boldsymbol{s}}_t$ with probabilities below a threshold $\beta \in [0, 1]$, where a higher β results in more aggressive masking. This removes potential distractions from the sentence to improve retrieval accuracy.

Generated questions as explicit queries. Another method is to generate explicit questions that target the low-confident span in $\hat{\boldsymbol{s}}_t$. For example, if the LM is uncertain about "the University of Pennsylvania", a question like "Which university did Joe Biden attend?" can help retrieve relevant information. Self-ask [164] achieved this by manually inserting follow-up questions into downstream task exemplars as shown later in Prompt 5, which requires task-specific annotation efforts. Instead, we developed a universal approach that generates questions for low-confidence spans without additional annotation. Specifically, we first extract all spans from $\hat{\boldsymbol{s}}_t$ with probabilities below β. For each extracted span $\boldsymbol{z}$, we prompt `gpt-3.5-turbo` to generate a question $\boldsymbol{q}_{t,\boldsymbol{z}}$ that can be answered with the span:

Prompt 3: zero-shot question generation

User input $\boldsymbol{x}$.
Generated output so far $\boldsymbol{y}_{<t}$.

Given the above passage, ask a question to which the answer is the term/entity/phrase "$\boldsymbol{z}$".

We retrieve using each generated question and interleave the returned documents into a single ranking list to aid future generations. In summary, queries $\boldsymbol{q}_t$ are formulated based on $\hat{\boldsymbol{s}}_t$ as follows:

$$\boldsymbol{q}_t = \begin{cases} \emptyset & \text{if all tokens of } \hat{\boldsymbol{s}}_t \text{ have probs} \geq \theta \\ \text{mask}(\hat{\boldsymbol{s}}_t) \text{ or qgen}(\hat{\boldsymbol{s}}_t) & \text{otherwise} \end{cases}$$

7.3.3 Implementation Details

Base LM We validate our method on one of the most advanced GPT-3.5 LMs `text-davinci-003` by iteratively querying their API.[2]

Document corpus and retrievers. Since we focus on the integration of retrieval and generation, we use off-the-shelf retrievers that take queries as inputs and return a list of relevant passages. The number of passages depends on the downstream task, which will be explained in section 7.5. For datasets that mainly rely on knowledge from Wikipedia, we use the Wikipedia dump from Karpukhin et al. [99] which chunks Wikipedia articles into passages, and employ BM25 [179] as the retriever. For datasets that rely on knowledge from the open web, we use the Bing search engine as our retriever.[3]

[2] `https://api.openai.com/v1/completions` April 23.
[3] `https://www.microsoft.com/en-us/bing/apis/bing-web-search-api`

Retrieved document formatting. Multiple retrieved documents are linearized according to their ranking and then added to the beginning of the user input using Prompt 4.

Prompt 4: document formatting

Search results:
[1] Document 1
[2] Document 2
...
The user input $\boldsymbol{x}$

7.4 Multi-time Retrieval Baselines

Existing passive multi-time retrieval augmented LMs can also be formulated using our framework (subsection 7.2.3). In this section, we formally introduce three baseline categories based on when and what to retrieve. These baselines are not exact reproductions of the corresponding paper because many design choices differ which makes direct comparisons impossible. We implemented them using the same settings, with the only variation being when and what to retrieve.

Previous-window approaches trigger retrieval every l tokens, where l represents the window size. Generated tokens from the previous window are used as the query:

$$\begin{aligned}\boldsymbol{q}_t &= \boldsymbol{y}_{t-1} \quad (t \geq 2),\\ \boldsymbol{y}_t &= [w_{(t-1)l+1}, ..., w_{tl}].\end{aligned}$$

Some existing methods in this category are RETRO [17], IC-RALM [174], which retrieve every few tokens, and KNN-LM [101], which retrieves every token.[4] We follow Ram et al. [174] to use a window size of $l = 16$.

Previous-sentence approaches trigger retrieval every sentence and use the previous sentence as the query, and IRCoT [216] belongs to this category:

$$\begin{aligned}\boldsymbol{q}_t &= \boldsymbol{y}_{t-1} \quad (t \geq 2),\\ \boldsymbol{y}_t &= \boldsymbol{s}_t.\end{aligned}$$

[4]Since KNN-LM uses the contextualized representation corresponding to the current decoding position to retrieve relevant information which encodes all previous tokens. Strictly speaking, $\boldsymbol{q}_t$ should be $\boldsymbol{y}_{<t}$.

Settings	2WikiMultihopQA [73]	StrategyQA [58]	ASQA [196]	WikiAsp [69]
Dataset statistics				
Task	multihop QA	commonsense QA	long-form QA	open-domain summarization
#Examples	500	229	500	500
Evaluation settings				
Metrics	EM, F_1, Prec., Rec.	EM	EM, Disambig-F_1, ROUGE, DR	UniEval, entity-F_1, ROUGE
Retrieval settings				
Corpus	Wikipedia	Wikipedia	Wikipedia	open web
Retriever	BM25	BM25	BM25	Bing
Top-k	2	3	3	5
Prompt format				
#Exemplars	8	6	8	4
Ret. for exemplars	✓	✗	✗	✗

Table 7.1: Dataset statistics and experimental settings of different tasks.

Question decomposition approaches manually annotated task-specific exemplars to guide LMs to generate decomposed sub-questions while producing outputs. For example, self-ask [164], a method in this category, manually inserts sub-questions in exemplars using Prompt 5. For the test case, retrieval is triggered dynamically whenever the model generates a sub-question.

Prompt 5: multihop QA with self-ask

```
Question: Who lived longer, Theodor Haecker or Harry Vaughan Watkins?
Are follow up questions needed here: Yes.
Follow up: How old was Theodor Haecker when he died?
Intermediate answer: Theodor Haecker was 65 years old when he died.
Follow up: How old was Harry Vaughan Watkins when he died?
Intermediate answer: Harry Vaughan Watkins was 69 years old when he died.
So the final answer is: Harry Vaughan Watkins.
```

The aforementioned approaches can retrieve additional information while generating. However, they have notable drawbacks: (1) Using previously generated tokens as queries might not reflect what LMs intend to generate in the future. (2) Retrieving information at a fixed interval can be inefficient because it might occur at inappropriate points. (3) Question decomposition approaches require task-specific prompt engineering, which restricts their generalizability in new tasks.

Dataset	θ	β	Query formulation	Combine single- & multi-time retrieval
2WikiMultihopQA	0.8	0.4	implicit	✗
StrategyQA	0.4	0.4	implicit	✗
ASQA & ASQA-hint	0.8	0.4	explicit	✓
WikiAsp	0.8	0.4	explicit	✓

Table 7.2: Hyperparameters of FLARE on different datasets.

7.5 Experimental Setup

We evaluate the effectiveness of FLARE on 4 diverse knowledge-intensive tasks using few-shot in-context learning [21, 134, 168]. We follow previous works [216] to sub-sample at most 500 examples from each dataset due to the cost of running experiments. Datasets, metrics, and settings are summarized in Table 7.1. The hyperparameters of FLARE are selected based on the development set and listed in Table 7.2. FLARE refers to $\text{FLARE}_{\text{direct}}$ if not specifically stated.

Multihop QA The goal of multihop QA is to answer complex questions through information retrieval and reasoning. We use 2WikiMultihopQA [73] which contains 2-hop complex questions sourced from Wikipedia articles that require composition, comparison, or inference, e.g., "Why did the founder of Versus die?" We follow Wang et al. [229] to generate both the chain-of-thought and the final answer. For "Why did the founder of Versus die?", the output we aim to generate is "The founder of Versus was Gianni Versace. Gianni Versace was shot and killed on the steps of his Miami Beach mansion on July 15, 1997. So the answer is shot." We use 8 exemplars from Trivedi et al. [216] for in-context learning, BM25 as the retriever, and Wikipedia articles as the retrieval corpus. Similar to the observation in Trivedi et al. [216], we found incorporating retrieval results for exemplars improves the performance. We use the input $\boldsymbol{x}$ of each exemplar to retrieve several documents and then add them using the format in Prompt 4. We found increasing the number of retrieval documents often increases performance. Therefore, we use the maximum number of documents that can fit within the input length limit of `text-davinci-003`, which is 2 for 2WikiMultihopQA.

We use regular expressions to extract the final answer from the output and compare it with the reference answer using exact match (EM), and token-level F_1, precision, and recall.

Commonsense reasoning Commonsense reasoning requires world and commonsense knowledge to generate answers. We use StrategyQA [58] which is a collection of crowdsourced yes/no questions, e.g., "Would a pear sink in water?" We follow Wei et al. [233] to generate both the chain-of-thought and the final yes/no answer. For "Would a pear sink in water?", the output we

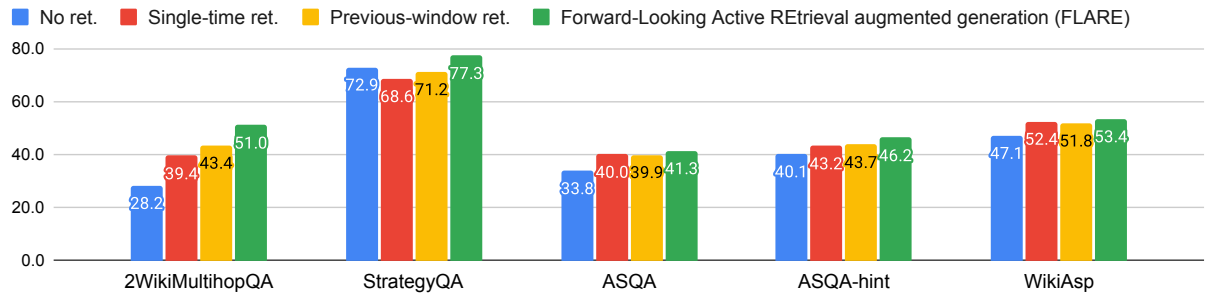


Figure 7.4: Comparision between FLARE and baselines across all tasks/datasets. We report the primary metric for each dataset: EM for 2WikiMultihopQA, StrategyQA, and ASQA, and UniEval for WikiAsp.

aim to generate is "The density of a pear is about 0.6g/cm^3, which is less than water. Objects less dense than water float. Thus, a pear would float. So the final answer is no." We use 6 exemplars from Wei et al. [233], BM25 on the Wikipedia corpus, and 3 retrieved documents to run experiments.

We extract the final answer and match it against the gold answer using exact match.

Long-form QA Long-form QA aims to generate comprehensive answers to questions seeking complex information [52, 196]. We use ASQA [196] as our testbed where inputs are ambiguous questions with multiple interpretations, and outputs should cover all of them. For example, "Where do the Philadelphia Eagles play their home games?" could be asking about the city, sports complex, or stadium. We found in many cases it is challenging even for humans to identify which aspect of the question is ambiguous. Therefore, we created another setting (ASQA-hint) where we provide a brief hint to guide LMs to stay on track when generating answers. The hint for the above case is "This question is ambiguous in terms of which specific location or venue is being referred to." For "Where do the Philadelphia Eagles play their home games?", the output we aim to generate is "We need to consider the different possible locations or venues that could be considered the home field of the Philadelphia Eagles. These include the city, the sports complex, or the stadium. Therefore, this question has 3 interpretations and the answers are: (1) The city is Philadelphia. (2) The sports complex is the South Philadelphia Sports Complex. (3) The stadium is the Lincoln Financial Field stadium." For both the original setting (ASQA) and the setting with hints (ASQA-hint), we manually annotate 8 exemplars, use BM25 on the Wikipedia corpus, and 3 retrieved documents to run experiments.

We use metrics from Stelmakh et al. [196], including EM, RoBERTa-based QA score (Disambig-F_1), ROUGE [128], and an overall score combining Disambig-F_1 and ROUGE (DR).

Open-domain summarization The goal of open-domain summarization is to generate a comprehensive summary of a topic by gathering information from open web [61]. We use WikiAsp [69] which aims to generate aspect-based summaries about entities from 20 domains in Wikipedia, e.g., "Generate a summary about Echo School (Oregon) including the following aspects: academics, history." The original WikiAsp dataset is designed for multi-document summarization and provides a list of references to systems. We converted it into the open-domain setting by removing the associated references and instead gathering information from the open web. For "Generate a summary about Echo School (Oregon) including the following aspects: academics, history.", the output we aim to generate is "# Academics. In 2008, 91% of the school's seniors received their high school diploma... # History. The class of 2008 was the 100th class in the school's history." where # is used to indicate aspects. We manually annotate 4 exemplars, and use the Bing search engine to retrieve 5 documents from the open web. To avoid leaking, we exclude several Wikipedia-related domains from Bing's search results, including `wikipedia.org`, `wikiwand.com`, `wiki2.org`, and `wikimedia.org`.

Metrics include ROUGE, named entity-based F_1, and UniEval [254] which measures factual consistency.

7.6 Experimental Results

We first report overall results across 4 tasks/datasets and compare the performance of FLARE with all the baselines introduced in section 7.4. We then run ablation experiments to study the efficacy of various design choices of our method.

7.6.1 Comparison with Baselines

Overall results. The overall performance of FLARE and baseline across all tasks/datasets are reported in Figure 7.4. FLARE outperforms all baseline on all tasks/datasets, indicating that FLARE is a generic method that can effectively retrieve additional information throughout the generation.

Among various tasks, multihop QA shows the most significant improvement. This is largely due to the task's clear definition and specific objective of producing the final answer through a 2-hop reasoning process, which makes it easier for LMs to generate on-topic output. In contrast, ASQA and WikiAsp are more open-ended, which increases the difficulty of both generation and evaluation. The improvement on ASQA-hint is larger than that of ASQA because identifying ambiguous aspects is challenging even for humans in many cases, and providing a generic hint helps LMs to stay on topic.

Methods	EM	F_1	Prec.	Rec.
No retrieval	28.2	36.8	36.5	38.6
Single-time retrieval	39.4	48.8	48.6	51.5
Multi-time retrieval				
Previous-window	43.2	52.3	51.7	54.5
Previous-sentence	39.0	49.2	48.9	51.8
Question decomposition	47.8	56.4	56.1	58.6
$\text{FLARE}_{\text{instruct}}$ (ours)	42.4	49.8	49.1	52.5
$\text{FLARE}_{\text{direct}}$ (ours)	**51.0**	**59.7**	**59.1**	**62.6**

Table 7.3: FLARE and baselines on 2WikiMultihopQA. Previous-window [17, 174], previous-sentence [216], and question decomposition [164, 244] methods are reimplemented for fair comparisons.

Thorough comparisons with baselines. The performance of all baselines on 2WikiMultihopQA are reported in Table 7.3. FLARE outperforms all baselines by a large margin, which confirms that forward-looking active retrieval is highly effective. Most multi-time retrieval augmented approaches outperform single-time retrieval but with different margins. The improvement of retrieving using the previous sentence is relatively small which we hypothesize is mainly because the previous sentence often describes entities or relations different from those in the next sentence in 2WikiMultihopQA. While the previous-window approach might use the first half of a sentence to retrieve information potentially helpful for generating the second half. Among all baselines, the question decomposition approach [164] achieves the best performance. which is not surprising since the in-context exemplars manually annotated with decomposed sub-questions (Prompt 5) guide LMs to generate sub-questions that align with the topic/intent of future generations. FLARE outperforms this baseline, indicating that manual exemplar annotation is not necessary for effective future-aware retrieval. The gap between $\text{FLARE}_{\text{instruct}}$ and question decomposition is large, indicating that teaching LMs to generate search queries using task-generic retrieval instructions and exemplars is challenging.

We report all metrics for the other datasets in Table 7.4. FLARE outperforms baselines with respect to all metrics. Retrieval using the previous window underperforms single-time retrieval on ASQA, which we hypothesize is because the previous window does not accurately reflect future intent. Since we focus on evaluating factuality, metrics with an emphasis on factual content (such as EM, Disambig-F_1, UniEval) are more reliable than metrics computed over all tokens (ROUGE-L).

Datasets	StrategyQA	ASQA				ASQA-hint				WikiAsp		
Metrics	EM	EM	D-F_1	R-L	DR	EM	D-F_1	R-L	DR	UniEval	E-F_1	R-L
No retrieval	72.9	33.8	24.2	33.3	28.4	40.1	32.5	36.4	34.4	47.1	14.1	26.4
Single-time retrieval	68.6	40.0	27.1	34.0	30.4	43.2	34.8	37.4	36.0	52.4	17.4	26.9
Multi-time retrieval												
Previous-window	71.2	39.9	27.0	**34.3**	30.4	43.7	35.7	37.5	36.6	51.8	18.1	27.3
Previous-sentence	71.0	39.9	27.9	**34.3**	30.9	44.7	35.9	37.5	36.7	52.6	17.8	27.2
FLARE (ours)	**77.3**	**41.3**	**28.2**	**34.3**	**31.1**	**46.2**	**36.7**	**37.7**	**37.2**	**53.4**	**18.9**	**27.6**

Table 7.4: Comparison between FLARE and baselines on StrategyQA, ASQA, ASQA-hint, and WikiAsp. D-F_1 is Disambig-F_1, R-L is ROUGE-L, and E-F_1 is named entity-based F_1.

	2WikiMultihopQA				ASQA-hint			
	EM	F_1	Prec.	Rec.	EM	D-F_1	R-L	DR
Previous	39.0	49.2	48.9	51.8	42.5	34.1	36.9	35.5
Next	48.8	57.6	57.1	60.5	45.9	35.7	37.5	36.6

Table 7.5: A head-to-head comparison between using the previous sentence and the next sentence for retrieval.

7.6.2 Ablation Study

Importance of forward-looking retrieval. We first validate that forward-looking retrieval is more effective than past-context-based retrieval. We run ablation experiments on 2WikiMultihopQA and ASQA-hint comparing retrieval using the previous versus the next sentence. Specifically, both methods retrieve every sentence and directly use the complete previous/next sentence as queries. As shown in Table 7.5, using the next sentence to retrieve is clearly better than using the previous sentence, confirming our hypothesis.

#Tokens	EM	F_1	Prec.	Rec.
16	43.2	52.3	51.7	54.5
32	43.6	52.4	52.0	55.0
48	40.0	49.3	49.0	52.0
All	39.0	48.5	48.2	51.1

Table 7.6: Previous-window approaches using different numbers of tokens as queries.

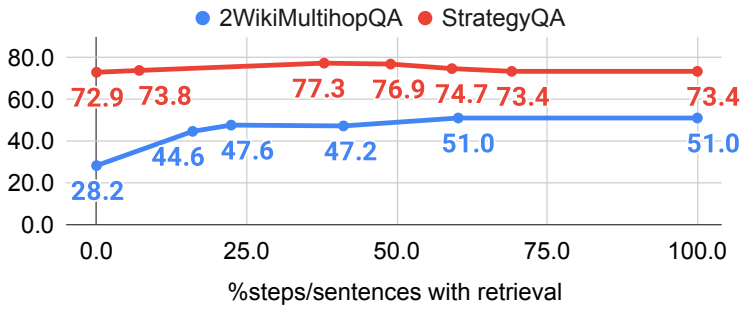


Figure 7.5: Performance (EM) of FLARE with respect to the percentage of steps/sentences with retrieval on 2WikiMultihopQA and StrategyQA.

β	EM	F_1	Prec.	Rec.
0.0	0.488	0.576	0.571	0.605
0.2	0.498	0.588	0.582	0.616
0.4	0.510	0.597	0.591	0.627
0.6	0.506	0.593	0.586	0.622

Table 7.7: Performance of FLARE with respect to the masking threshold β on 2WikiMultihopQA.

We also run previous-window approaches using different numbers of past tokens as queries. As shown in Table 7.6, using too many tokens (> 32) in the past hurts the performance, further confirming our hypothesis that previous context might not be relevant to the intent of future generations.

Importance of active retrieval. Next, we investigate how active retrieval threshold θ affects performance. To alter our method from not retrieving to retrieving every sentence, we adjust the confidence threshold θ that determines when to trigger retrieval from 0 to 1. We then calculate the proportion of steps/sentences where retrieval is activated, and present the performance based on it. As shown in Figure 7.5, on 2WikiMultihopQA, the performance plateaus when the retrieval percentage exceeds 60%, indicating that retrieval when LMs are confident is not necessary. On StrategyQA, the performance drops when the retrieval percentage exceeds 50%, indicating that unnecessary retrieval can introduce noise and impede the original generation process. We found triggering retrieval for 40%-80% of sentences usually leads to a good performance across tasks/datasets.

	ASQA-hint				WikiAsp		
	EM	D-F_1	R-L	DR	UniEval	E-F_1	R-L
Implicit	45.7	36.9	37.7	37.3	53.4	18.8	27.7
Explicit	46.2	36.7	37.7	37.2	53.4	18.9	27.6

Table 7.8: A comparison between implicit and explicit query formulation methods in FLARE.

Effectiveness of different query formulation methods We study implicit query formation by masking and explicit query formulation through question generation. In Table 7.7, we compare the performance of FLARE with different masking thresholds β. Retrieving directly with the complete sentence ($\beta = 0$) is worse than masking tokens with low probabilities, confirming our hypothesis that low-confidence erroneous tokens can distract retrievers. We compare implicit and explicit query formulation methods in Table 7.8. Performances of both methods are similar, indicating that both methods can effectively reflect information needs.

Efficiency As shown in Figure 7.5, on average retrieval is triggered for $30\% \sim 60\%$ of sentences depending on downstream tasks. In comparison, KNN-LM [101] retrieves every token, RETRO or IC-RALM [17, 174] retrievers every 4∼32 tokens, and IRCoT [216] retrieves every sentence. Compared to single-time retrieval, however, interleaving retrieval and generation with a naive implementation indeed increases overheads. LMs need to be activated multiple times (once for each retrieval) and a caching-free implementation also requires recomputing the previous activation each time after retrieval. This issue can be potentially alleviated with special architectural designs that encode the retrieved documents $\mathcal{D}_{\boldsymbol{q}_t}$ and the input/generation ($\boldsymbol{x}/\boldsymbol{y}_{<t}$) independently.

7.7 Related Work

We refer to subsection 7.2.2 and section 7.4 for extensive discussion on single-time and multi-time retrieval augmented LMs, which is the most relevant area to this chapter.

Iterative and adaptive retrieval Iterative retrieval and refinement has been studied in both text and code generation tasks [157, 246, 248, 249]. FLARE differs from these methods in the granularity of generation and retrieval strategies. Adaptive retrieval has been studied in single-time retrieval scenarios based on either question popularity or generation probabilities [126, 138], while we focus on long-form generation requiring active information access.

Browser-enhanced LMs WebGPT [147] and WebCPM [166] train LMs to interact with browser to enhance factuality using reinforcement learning or supervised training where multiple queries can be triggered before generation. FLARE is built on text-based retrievers but can be combined with a browser to potentially improve retrieval quality.

7.8 Conclusion

To aid long-form generation with retrieval augmentation, we propose an active retrieval augmented generation framework that decides when and what to retrieve during generation. We implement this framework with forward-looking active retrieval that iteratively uses the upcoming sentence to retrieve relevant information if it contains low-confidence tokens and regenerates the next sentence. Experimental results on 4 tasks/datasets demonstrate the effectiveness of our methods. Future directions include better strategies for active retrieval and developing efficient LM architectures for active information integration.

Chapter 8

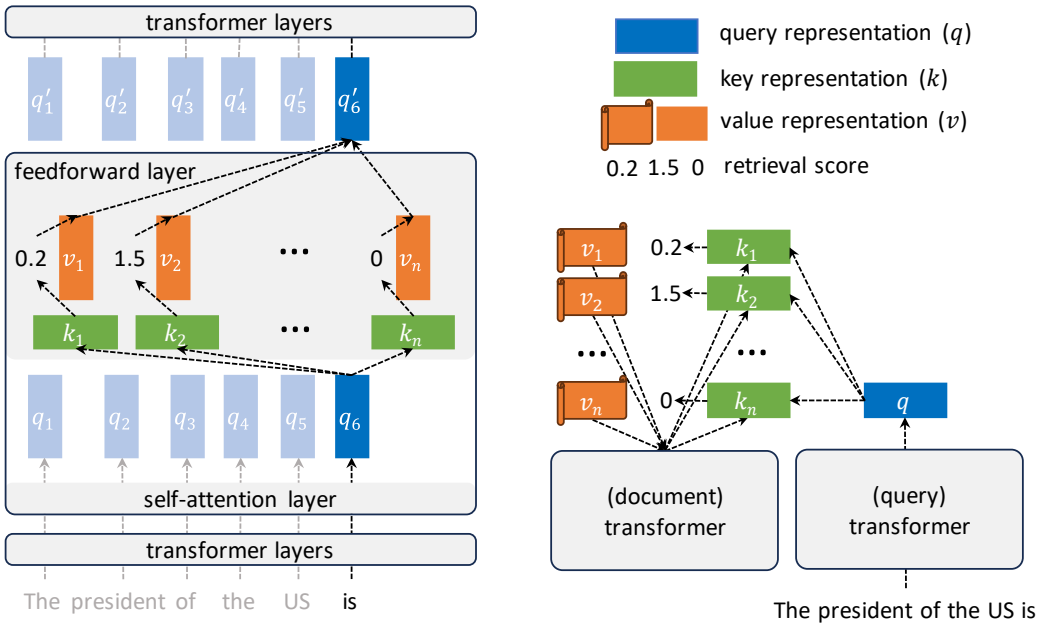
Conclusion

In this dissertation, we study two types of approaches to answer questions that require factual knowledge: parametric approaches and non-parametric approaches. Parametric approaches store knowledge within the parameters of LLMs and elicit this knowledge through prompting, while non-parametric approaches store knowledge in a non-parametric datastore such as a document corpus and utilize this knowledge using retrievers that are usually based on semantic dense representations. Managing factual knowledge involves two essential aspects: knowledge storage and knowledge retrieval. The query-key-value representation is a general framework to explain the knowledge retrieval process. In this framework, *queries* indicate information needs or search intents, which are matched with a set of *keys* to locate relevant information. Subsequently, the corresponding *values* of the matched items are returned to the user.

The feedforward layer (FFW) of the transformer architecture is a position-wise function, processing each input token independently:

$$\text{FFW}(\boldsymbol{q}) = f(\boldsymbol{q} \cdot K^T) \cdot V, \tag{8.1}$$

where $\boldsymbol{q} \in \mathbb{R}^d$ is the input hidden state, $K, V \in \mathbb{R}^{n \times d}$ are two parameter matrices, and f is a non-linear activation function such as ReLU. As suggested by Geva et al. [59], we can view the feedforward layer as a key-value "datastore" with n slots, where each key-value slot corresponds to a row in K and V respectively. The input vector $\boldsymbol{q}$ is matched with all key vectors in K with dot-product to obtain a distribution over all slots, and the output is obtained through a weighted sum over all values in V. Factual knowledge can be encoded in any parameters of LLMs, not exclusively in the value matrix V. However, structuring the feedforward layer using the query-key-value framework can facilitate establishing connections with non-parametric counterparts. Embedding-based dense retrieval systems can also be understood within the query-key-value framework, where queries are embeddings of input queries, keys are document embeddings, and values are the corresponding documents. If we assume that fac-



Parametric "datastore" in an LLM *Non-parametric datastore of a dense retriever*

Dimensions	**Parametric "datastore"**	**Non-parametric datastore**
	knowledge storage	
key representation ($\boldsymbol{k}$)	key vector in feedforward (learned)	doc embedding (learned)
value representation ($\boldsymbol{v}$)	value vector in feedforward (learned)	doc (fixed)
storage mechanism	pre-training on docs	doc encoding (+ indexing)
# key-value slots	fixed	unlimited
	knowledge retrieval	
query representation ($\boldsymbol{q}$)	hidden states before feedforward (learned)	query embedding (learned)
retrieval mechanism	transformer forward pass	query encoding + dot product
retrieval training	instruction-tuning / fine-tuning	contrastive learning
query augmentation	diverse prompting	query formulation
retrieval granularity	per-layer, per-token	single-time

Figure 8.1: A conceptual head-to-head comparison between parametric and non-parametric datastores from both knowledge storage and retrieval perspectives, assuming that feedforward layers in the transformer architecture are key-value datastores [59]. Drawing from this comparison, each key-value slot in feedforward layers can be viewed as a virtual document. However, it is important to note that knowledge is not solely stored within the value vectors.

tual knowledge primarily resides in the feedforward layers, though this assumption may not always hold true, we can make a direct conceptual comparison between parametric and non-parametric approaches regarding both knowledge storage and retrieval. This comparison aids in understanding the context of existing works, including those discussed in this dissertation, and can motivate future research directions.

In Figure 8.1, we visually compare the query-key-value representations of parametric and non-parametric datastores, providing an explanation of the storage and retrieval mechanism in the accompanying table.

Knowledge storage Each key-value slot in the feedforward layer can be viewed as a virtual document optimized during pre-training to store knowledge. However, it is worth noting that knowledge can be encoded in any parameters of LLMs, not exclusively in the value vectors. Unlike non-parametric approaches, where keys and values are grounded on real documents, parametric key-value datastores are not linked to specific information like documents. While it offers the advantage of learning flexibility, it also raises the chance of hallucination. Parametric datastores have a fixed number of key-value slots, allowing them to capture compressed concepts, although this comes at the expense of limited storage capacity. They can be unpredictably overwritten during continued training on new documents, leading to the issue of catastrophic forgetting. In contrast, non-parametric datastores have unbounded capacity, and expanding them is straightforward–simply encode new documents and index the resulting dense embeddings.

Knowledge retrieval Retrieving knowledge from non-parametric datastores is straightforward, which involves encoding the input query and matching it against all document embeddings based on dot-product similarity. In the case of parametric datastores, knowledge is retrieved and accumulated across multiple layers and tokens during the forward pass of the transformer model. This provides greater flexibility, particularly for complex questions that require gathering multiple pieces of information. To improve retrieval accuracy, embedding models are fine-tuned using contrastive learning objectives, aiming to learn close representations for similar query-document pairs. Meanwhile, parametric approaches directly fine-tune LLMs on inputs and desired outputs, in the hope that necessary knowledge can be elicited from the parameters to guide the desired output.

Connection to this dissertation Within the query-key-value framework, chapter 2, chapter 3, and chapter 4 mainly focus on optimizing prompts so that the *query* vector corresponding to the last token of the prompt can effectively retrieve the correct key-value slot containing the answer or indicates when the question goes beyond the scope of knowledge. The FLARE

method proposed in chapter 7 performs retrieval in multiple decoding steps, which abandons the traditional design of RAG that only builds a single query vector based on the input question but instead mimics the multi-step retrieval mechanism of parametric approaches that formulate multiple *query* vectors throughout the generation process.

The above chapters only involve optimizing *query* vectors. chapter 5 focuses on jointly optimizing query vectors and key-value slots by interleaving training on documents and instruction-tuning on QA pairs. chapter 6 utilizes the query-key-value vectors in the self-attention module as a direct instantiation of the framework, and optimizes them jointly using learning signals obtained from downstream tasks.

We summarize contributions covered in this dissertation based on this conceptual comparison in section 8.1, and share insights into future directions in section 8.2.

8.1 Summary of Contributions

Overall, parametric approaches are end-to-end solutions since they require training only one giant transformer model to encompass all capabilities, whereas non-parametric approaches involve a pipeline structure with non-parametric components, enabling LLMs to engage with the external world and extending the range of applications for LLM-based agents.

Knowledge retrieval In chapter 2 and chapter 3, we focus on improving knowledge retrieval from parametric datastores through diverse prompting without additional training. Similar to query reformulation techniques used in information retrieval systems that use different wordings to increase the chance of hitting relevant pages, diversified prompting enlarges the chance of activating related memory within LLMs. Tuning prompts either manually or automatically has already become a standard practice to interact with LLMs. It serves not only as an inference-time heuristic to enhance performance but also offers insights into how to improve training data and objectives, such as instruction-tuning data collection and generation receipts [184, 231].

One major bottleneck of non-parametric knowledge retrieval is fixed-dimensional embeddings. Motivated by the working mechanism of self-attention that computes token similarity within the input context, we propose to extend the context and directly compute attention over the entire corpus for retrieval purposes in chapter 6, which (1) is more expressive than fixed-dimensional embeddings, and (2) offers a single-model solution without external retrievers. Compared to the parametric key-value datastore implemented in feedforward layers, retrieval as attention implements a key-value caching system over all tokens of a corpus in the self-attention layer. The current work only implements such retrieval as attention in one specific layer. Extending it to multiple attention heads and layers can further unleash its potential to

handle complicated information matching.

Another bottleneck of non-parametric knowledge retrieval is that it only performs retrieval once using the input query, which is not enough for complicated questions that require aggregating multiple pieces of information from different sources. Taking insights from the accumulative knowledge retrieval process in parametric datastores that retrieves across multiple tokens, we propose active retrieval-augmented generation in chapter 7, which advocates iterative retrieval during the generation process. Although we mainly focus on knowledge-intensive tasks in the study, the underlying methodology is also applicable to other scenarios where LLMs need to interact with the external environment in multiple steps, such as web pages, operating systems, and the physical world. Methods discussed in chapter 6 and chapter 7 can be further combined to perform retrieval using attention over the entire corpus across multiple tokens so that information can be gathered actively, which we will explain in detail in subsection 8.2.3.

Knowledge storage As shown in Figure 8.1, both parametric and non-parametric approaches require additional training to improve knowledge retrieval performance. The standard practice for training embedding models is to collect relevant query-document pairs and fine-tune the model using contrastive objectives. Document embeddings are generated and indexed *afterward* using the fine-tuned model. The pre-instruction-tuning method proposed in chapter 5 shares a similar training receipt, which fine-tunes LLMs on QA pairs and associated documents to improve the ability to match questions with associated knowledge covered in documents *before* applying them to absorbing knowledge from new documents.

Compared to contrastive learning that explicitly optimizes which documents should be retrieved, instruction-tuning on QA pairs only ultilizes final answers as training signals, missing intermediate supervision on which information to gather from the parametric datastore. Because parametric datastores are not grounded to real documents, directly applying loss over key-value slots in feedforward layers is impractical. Therefore, we suggest modifying the training data to generate not only the final answer but also the necessary supporting evidence, which will be explained in detail in subsection 8.2.1.

8.2 Future Directions

We anticipate that future systems will incorporate elements of both parametric and non-parametric components. As LLMs grow in size and capacity, they become better at condensing knowledge into their parameters and reliably accessing it. However, including non-parametric components is essential for dealing with long-tail knowledge and applications that need interaction with the external environment.

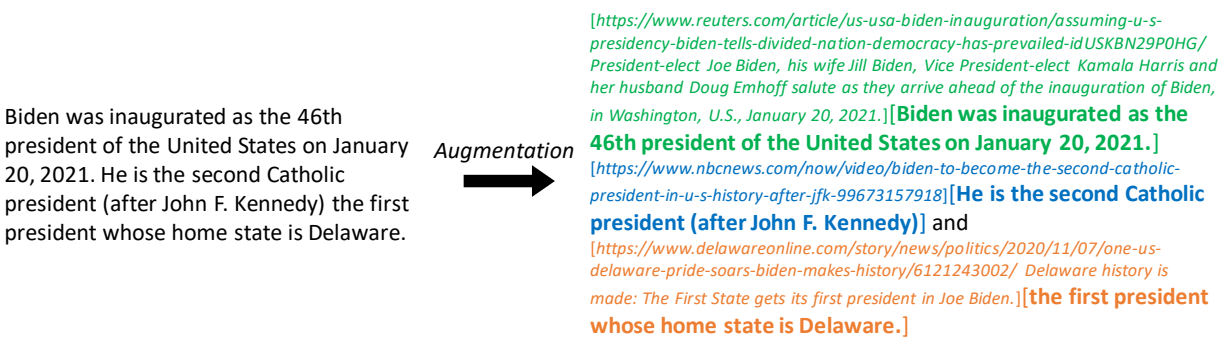


Figure 8.2: An example document from Wikipedia where each statement is preceded with a reference consisting of a URL and evidence text.

8.2.1 Robust Parametric Knowledge in LLMs

Considering the flexibility and scalability of parametric knowledge, we argue that more of knowledge could be encoded within the parameters of LLMs. Both data and model architecture play crucial roles in parametric knowledge. Inspired by the findings in chapter 5 that LLMs cannot implicitly recite memorized documents when answering questions, one future direction is to augment existing pre-training data of documents or supervised data of QA pairs with necessary references, and train models not only to produce the original output but also to explicitly recite the relevant references, as illustrated in Figure 8.2. It essentially optimizes an LLM to function as both a generative retriever and a text generator, merging existing works on generative retrieval [14, 205] with generic pre-training. This has three major benefits:

- **Improving the robustness of parametric knowledge retrieval**: compared to only using the original output as training signals, adding intermediate supervision on required references can explicitly optimize LLMs to retrieve desired information from the parametric datastore, which shares similarity with training methods used to optimize retrievers.
- **Providing supporting evidence for generated answers**: providing supporting evidence improves the credibility and verifiability of the generated answer, helping users evaluate the accuracy of the information provided.
- **Alleviating forgetting**: by forcing LLMs to recite relevant knowledge, forgetting can be potentially mitigated as long as the relevant knowledge is often recited.

Ensuring the accuracy of references poses a major challenge in data creation. One way to address this is by using statements extracted from the original output as queries to search for relevant documents with advanced dense retrievers. Then, we can sift through the top-K most similar documents using LLMs for more detailed checking, including verifying relevance and extracting supporting sections from the entire document. As shown in Figure 8.2, the aug-

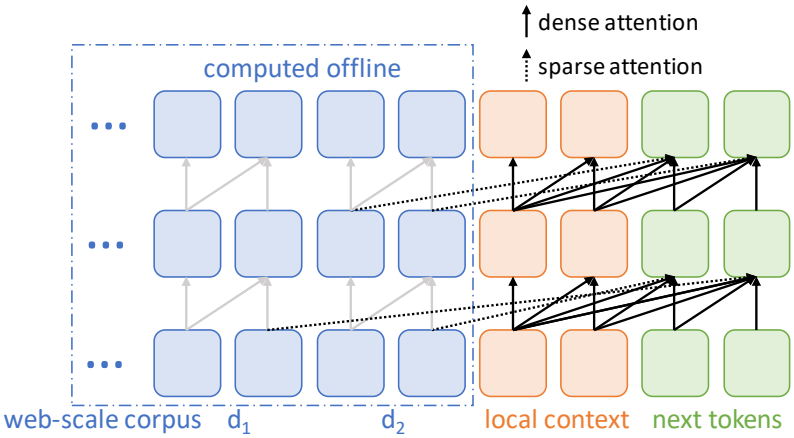


Figure 8.3: Illustration of transformer with sparse attention over the web-scale corpus. For simplicity, we only show two documents in the figure.

mented data includes only the identifier (e.g., URL) and the supporting section before the factual statement, which minimizes the increase in training cost. Since we use a special token to mark the beginning and end of the reference, we can skip the generation of references to reduce test-time latency by forbidding this special token if necessary. This approach can be also combined with a document corpus to force LLMs to generate valid substrings of real documents as supporting evidence, similar to the practice used by Bevilacqua et al. [14].

From the model architecture perspective, mixture-of-expert (MoE) models offer a principled way to increase the knowledge capacity of the model without increasing the running cost. Future directions include advanced training strategies as well as efficient expert routing and expansion methods.

8.2.2 Continual Knowledge Update and Alignment

Continual knowledge update is essential to expand the parametric knowledge of LLMs. The standard method is continued pre-training on the new corpus, which is expensive and suffers from catastrophic forgetting. As shown in chapter 5, memorization of different information has different difficulties, a more efficient way is to (1) only absorb knowledge not encoded in the parameters of the current LLM by prompting the current LLM with new documents to extract the information it does not already know, or (2) develop optimization techniques that assign varying learning rates based on the level of information presence in the parameters. Given the evolving knowledge of LLMs, it is also crucial to develop alignment methods to adjust their scope of knowledge.

8.2.3 Transformer Architectures with Infinite Context Length

The context length of LLMs has been extended to millions of tokens by computing dense attention across multiple GPUs. In order to directly use LLMs for open-domain information retrieval without reliance on an external retriever, an important direction is to allow LLMs to consume the entire web, which necessitates efficient attention computation among billions of documents. chapter 6 provides an initial attempt in this direction by using a single attention head in a single transformer layer for retrieval, which is essentially sparse attention over the entire corpus. It is important to extend this sparse attention mechanism to multiple heads, layers, and decoding steps, which is illustrated in Figure 8.3. An overview of the high-level pipeline for a system featuring such efficient sparse attention is as follows:

- **Document encoding**: the representations of tokens in the web-scale corpus are computed offline by running the LLM over each document independently.
- **Vector indexing**: these token representations (i.e., the key and value vectors in the self-attention module) are saved and indexed using efficient vector search libraries such as the FAISS and ScaNN library [64, 94].
- **Local context encoding**: encode a test-time input by computing dense attention over all tokens within the local context.
- **Sparse attention over the corpus**: select top-K most similar key-value pairs from the index across multiple layers and decoding steps, and combine it with the dense attention over the local context.

A significant challenge in training these models is efficiently storing and updating token representations from billions of documents during training. This can be alleviated through token pruning, vector quantization, and asynchronous updates of the vector index. In contrast to dense attention, sparsity also raises challenges in training difficulty and stability. These issues can potentially be tackled through carefully designed training procedures such as warmup with dense attention computation, or advanced optimizers.

8.2.4 LLM-based Agents

Most real-world application scenarios require interaction between LLMs and the external environments using retrievers or other tools, potentially through multiple steps. An example system is proposed in chapter 7 that interleaves LLMs and retrievers, and the framework can be extended to other tools such as calendars, calculators, and code interpreters. Gathering data for such scenarios and developing efficient learning methods to teach LLMs in performing complex tasks are critical future directions. This requires advancements in algorithms such as improved

reinforcement learning methods, and scalable learning paradigms that leverage LLMs to provide feedback and annotations. One promising approach involves gathering human interaction data with websites and computers [147] to bootstrap the initial training phase. Afterwards, LLMs can gather trajectories in a real-world environment for further exploratory learning.

Bibliography

[1] Judit Ács. Exploring bert's vocabulary, 2019. URL http://juditacs.github.io/2019/02/19/bert-tokenization-stats.html. Accessed May 2020. 34

[2] Eugene Agichtein and Luis Gravano. *Snowball*: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries, June 2-7, 2000, San Antonio, TX, USA*, pages 85–94. ACM, 2000. doi: 10.1145/336597.336644. URL https://doi.org/10.1145/336597.336644. 16

[3] Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. A neural knowledge language model. *CoRR*, abs/1608.00318v2, 2016. URL http://arxiv.org/abs/1608.00318. 24

[4] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws, 2024. 2, 3

[5] Uri Alon, Frank F. Xu, Junxian He, Sudipta Sengupta, Dan Roth, and Graham Neubig. Neuro-symbolic language modeling with automaton-augmented retrieval. In *International Conference on Machine Learning*, 2022. 71

[6] Antonios Anastasopoulos and Graham Neubig. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1091. URL https://www.aclweb.org/anthology/D19-1091. 32

[7] Akari Asai, Timo Schick, Patrick S. H. Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. Task-aware retrieval with instructions. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 3650–3675. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.225. URL https://doi.org/10.18653/v1/

2023.findings-acl.225. 5

[8] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *CoRR*, abs/2310.11511, 2023. doi: 10.48550/ARXIV.2310.11511. URL https://doi.org/10.48550/arXiv.2310.11511. 71

[9] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1279. URL https://www.aclweb.org/anthology/P19-1279. 13

[10] Yonatan Belinkov and James R. Glass. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019. URL https://transacl.org/ojs/index.php/tacl/article/view/1570. 24

[11] Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1080. URL https://www.aclweb.org/anthology/P17-1080. 24

[12] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1533–1544. ACL, 2013. URL https://aclanthology.org/D13-1160/. 1

[13] Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: Llms trained on "a is b" fail to learn "b is a". *CoRR*, abs/2309.12288, 2023. doi: 10.48550/ARXIV.2309.12288. URL https://doi.org/10.48550/arXiv.2309.12288. 56

[14] Michele Bevilacqua, Giuseppe Ottaviano, Patrick S. H. Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. Autoregressive search engines: Generating substrings as document identifiers. In *NeurIPS*, 2022. 83, 114, 115

[15] Rahul Bhagat and Deepak Ravichandran. Large scale acquisition of paraphrases for learning surface patterns. In *Proceedings of ACL-08: HLT*, pages 674–682, Columbus, Ohio,

June 2008. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P08-1077. 17

[16] Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/6239. 45

[17] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR, 2022. URL https://proceedings.mlr.press/v162/borgeaud22a.html. 1, 71, 93, 99, 104, 107

[18] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. COMET: commonsense transformers for automatic knowledge graph construction. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4762–4779. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1470. URL https://doi.org/10.18653/v1/p19-1470. 43

[19] Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. Inducing relational knowledge from BERT. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, New York, USA, February 2020. URL https://arxiv.org/pdf/1911.12753.pdf. 24

[20] Samuel R. Bowman, Jeeyoon Hyun, Ethan Perez, Edwin Chen, Craig Pettit, Scott Heiner, Kamile Lukosiute, Amanda Askell, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Christopher Olah, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Jackson Kernion, Jamie Kerr, Jared Mueller, Jef-

frey Ladish, Joshua Landau, Kamal Ndousse, Liane Lovitt, Nelson Elhage, Nicholas Schiefer, Nicholas Joseph, Noemí Mercado, Nova DasSarma, Robin Larson, Sam McCandlish, Sandipan Kundu, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Ben Mann, and Jared Kaplan. Measuring progress on scalable oversight for large language models. *CoRR*, abs/2211.03540, 2022. doi: 10.48550/ARXIV.2211.03540. URL https://doi.org/10.48550/arXiv.2211.03540. 7

[21] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html. 1, 2, 41, 43, 45, 49, 56, 61, 96, 101

[22] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6491–6506. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EMNLP-MAIN.522. URL https://doi.org/10.18653/v1/2021.emnlp-main.522. 3

[23] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In Michael Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 2633–2650. USENIX Association, 2021. URL https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting. 1

[24] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr,

and Chiyuan Zhang. Quantifying memorization across neural language models. *CoRR*, abs/2202.07646, 2022. URL https://arxiv.org/abs/2202.07646. 1, 3, 70

[25] Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1):1:1–1:50, 2012. doi: 10.1145/2071389.2071390. URL https://doi.org/10.1145/2071389.2071390. 17

[26] José Cañete, Gabriel Chaperon, Rodrigo Fuentes, and Jorge Pérez. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*, 2020. URL https://users.dcc.uchile.cl/~jperez/papers/pml4dc2020.pdf. 34

[27] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1171. URL https://doi.org/10.18653/v1/P17-1171. 48, 71, 91

[28] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *CoRR*, abs/2306.15595, 2023. doi: 10.48550/ARXIV.2306.15595. URL https://doi.org/10.48550/arXiv.2306.15595. 6

[29] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785. 47, 50

[30] Daixuan Cheng, Shaohan Huang, and Furu Wei. Adapting large language models via reading comprehension. *CoRR*, abs/2309.09530, 2023. doi: 10.48550/ARXIV.2309.09530. URL https://doi.org/10.48550/arXiv.2309.09530. 70

[31] Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Unitedqa: A hybrid approach for open domain question answering. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3080–3090. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.240. URL https://doi.org/10.18653/v1/2021.acl-long.240. 84, 86

[32] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/. 70

[33] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *CoRR*, abs/2204.02311, 2022. doi: 10.48550/arXiv.2204.02311. URL https://doi.org/10.48550/arXiv.2204.02311. 2

[34] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416, 2022. doi: 10.48550/ARXIV.2210.11416. URL https://doi.org/10.48550/arXiv.2210.11416. 3

[35] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457, 2018. URL http://arxiv.org/abs/1803.05457. 45

[36] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. SPECTER: document-level representation learning using citation-informed transformers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, On-*

line, July 5-10, 2020, pages 2270–2282. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.207. URL https://doi.org/10.18653/v1/2020.acl-main.207. 86

[37] Nachshon Cohen, Oren Kalinsky, Yftah Ziser, and Alessandro Moschitti. Wikisum: Coherent summarization dataset for efficient human-evaluation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 212–219. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-short.28. URL https://doi.org/10.18653/v1/2021.acl-short.28. 92

[38] Alexis Conneau and Guillaume Lample. Cross-lingual language model pretraining. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 7057–7067, 2019. URL http://papers.nips.cc/paper/8928-cross-lingual-language-model-pretraining. 34

[39] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019. URL http://arxiv.org/abs/1911.02116. 34

[40] Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8493–8502. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.581. URL https://doi.org/10.18653/v1/2022.acl-long.581. 3

[41] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *NeurIPS*, 2022. 6

[42] Pradeep Dasigi, Nelson F. Liu, Ana Marasovic, Noah A. Smith, and Matt Gardner. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th Interna-*

tional Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 5924–5931. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1606. URL https://doi.org/10.18653/v1/D19-1606. 45

[43] Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. Bertje: A dutch BERT model. *CoRR*, abs/1912.09582, 2019. URL http://arxiv.org/abs/1912.09582. 34

[44] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *NeurIPS*, 2023. 5

[45] Lucio M. Dery, Paul Michel, Ameet Talwalkar, and Graham Neubig. Should we be pre-training? an argument for end-task aware training as an alternative. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=2bO2x8NAIMB. 70

[46] Shrey Desai and Greg Durrett. Calibration of pre-trained transformers. *CoRR*, abs/2003.07892, 2020. URL https://arxiv.org/abs/2003.07892. 42, 43, 46, 47, 52, 54

[47] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423. 1, 5, 13, 15, 19, 31, 32, 33, 34, 36, 43, 49

[48] Li Dong, Chris Quirk, and Mirella Lapata. Confidence modeling for neural semantic parsing. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 743–753. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-1069. URL https://www.aclweb.org/anthology/P18-1069/. 47

[49] Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard H. Hovy, Hinrich Schütze, and Yoav Goldberg. Measuring and improving consistency in pretrained language models. *Trans. Assoc. Comput. Linguistics*, 9:1012–1031, 2021. doi: 10.1162/tacl_a_00410. URL https://doi.org/10.1162/tacl_a_00410. 3

[50] Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*, 2018. URL http://www.lrec-conf.org/proceedings/lrec2018/summaries/632.html. 19, 29

[51] Martin Fajcik, Martin Docekal, Karel Ondrej, and Pavel Smrz. R2-D2: A modular baseline for open-domain question answering. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 854–870. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-emnlp.73. URL https://doi.org/10.18653/v1/2021.findings-emnlp.73. 84, 86

[52] Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. ELI5: long form question answering. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3558–3567. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1346. URL https://doi.org/10.18653/v1/p19-1346. 1, 92, 102

[53] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027, 2021. URL https://arxiv.org/abs/2101.00027. 3

[54] Luyu Gao and Jamie Callan. Unsupervised corpus aware language model pre-training for dense passage retrieval. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2843–2853. Association for Computational Linguistics, 2022. URL https://aclanthology.org/2022.acl-long.203. 84

[55] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. *CoRR*, abs/2212.10496, 2022. doi: 10.48550/arXiv.2212.10496. URL https://doi.org/10.48550/arXiv.2212.10496. 97

[56] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in*

Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 6894–6910. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.552. URL https://doi.org/10.18653/v1/2021.emnlp-main.552. 87

[57] Gemini Team. Gemini: A family of highly capable multimodal models, 2023. 1

[58] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021. 92, 93, 100, 101

[59] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5484–5495. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EMNLP-MAIN.446. URL https://doi.org/10.18653/v1/2021.emnlp-main.446. 3, 109, 110

[60] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6114–6123, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1633. URL https://www.aclweb.org/anthology/D19-1633. 32, 33, 34

[61] John M. Giorgi, Luca Soldaini, Bo Wang, Gary D. Bader, Kyle Lo, Lucy Lu Wang, and Arman Cohan. Exploring the challenges of open domain multi-document summarization. *CoRR*, abs/2212.10526, 2022. doi: 10.48550/arXiv.2212.10526. URL https://doi.org/10.48550/arXiv.2212.10526. 92, 103

[62] Yoav Goldberg. Assessing BERT's syntactic abilities. *CoRR*, abs/1901.05287v1, 2019. URL http://arxiv.org/abs/1901.05287. 24

[63] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017. URL http://proceedings.mlr.press/v70/guo17a.html. 45, 46, 47, 49, 54

[64] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 3887–3896. PMLR, 2020. 116

[65] Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8342–8360. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.ACL-MAIN.740. URL https://doi.org/10.18653/v1/2020.acl-main.740. 3

[66] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909, 2020. URL https://arxiv.org/abs/2002.08909. 1, 7, 45, 48, 71, 76, 83, 84, 86, 91

[67] Tianyu Han, Lisa C. Adams, Jens-Michalis Papaioannou, Paul Grundmann, Tom Oberhauser, Alexander Löser, Daniel Truhn, and Keno K. Bressem. Medalpaca - an open-source collection of medical conversational AI models and training data. *CoRR*, abs/2304.08247, 2023. doi: 10.48550/ARXIV.2304.08247. URL https://doi.org/10.48550/arXiv.2304.08247. 70

[68] Hiroaki Hayashi, Zecong Hu, Chenyan Xiong, and Graham Neubig. Latent relation language models. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, New York, USA, February 2020. URL https://arxiv.org/pdf/1908.07690.pdf. 24

[69] Hiroaki Hayashi, Prashant Budania, Peng Wang, Chris Ackerson, Raj Neervannan, and Graham Neubig. Wikiasp: A dataset for multi-domain aspect-based summarization. *Trans. Assoc. Comput. Linguistics*, 9:211–225, 2021. doi: 10.1162/tacl_a_00362. URL https://doi.org/10.1162/tacl_a_00362. 92, 93, 100, 103

[70] Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. Efficient nearest neighbor language models. In *Conference on Empirical Methods in Natural Language Processing*, 2021. 71

[71] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *CoRR*, abs/2009.03300, 2020. URL https://arxiv.org/abs/2009.03300. 2, 45, 49, 50, 92

[72] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word

representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4129–4138, 2019. URL https://www.aclweb.org/anthology/N19-1419/. 24, 54

[73] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing A multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Núria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6609–6625. International Committee on Computational Linguistics, 2020. doi: 10.18653/v1/2020.coling-main.580. URL https://doi.org/10.18653/v1/2020.coling-main.580. 92, 93, 100, 101

[74] Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. Cqadupstack: A benchmark data set for community question-answering research. In Laurence Anthony F. Park and Sarvnaz Karimi, editors, *Proceedings of the 20th Australasian Document Computing Symposium, ADCS 2015, Parramatta, NSW, Australia, December 8-9, 2015*, pages 3:1–3:8. ACM, 2015. doi: 10.1145/2838931.2838934. URL https://doi.org/10.1145/2838931.2838934. 86

[75] Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080, 2020. URL https://arxiv.org/abs/2003.11080. 36

[76] Nathan Hu, Eric Mitchell, Christopher D. Manning, and Chelsea Finn. Meta-learning online adaptation of language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 4418–4432. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.emnlp-main.268. 56, 61, 68, 70

[77] Robert L. Logan IV, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. Barack's wife Hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5962–5971, 2019. URL https://www.aclweb.org/anthology/P19-1598/. 24

[78] Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh

Hajishirzi. Camels in a changing climate: Enhancing LM adaptation with tulu 2. *CoRR*, abs/2311.10702, 2023. doi: 10.48550/ARXIV.2311.10702. URL https://doi.org/10.48550/arXiv.2311.10702. 70

[79] Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, Xian Li, Brian O'Horo, Gabriel Pereyra, Jeff Wang, Christopher Dewan, Asli Celikyilmaz, Luke Zettlemoyer, and Ves Stoyanov. OPT-IML: scaling language model instruction meta learning through the lens of generalization. *CoRR*, abs/2212.12017, 2022. doi: 10.48550/ARXIV.2212.12017. URL https://doi.org/10.48550/arXiv.2212.12017. 70

[80] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 874–880. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.eacl-main.74. URL https://doi.org/10.18653/v1/2021.eacl-main.74. 7, 77, 78, 81, 83, 84, 86, 91

[81] Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=NTEz-6wysdb. 77, 82, 84, 85, 86

[82] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Towards unsupervised dense information retrieval with contrastive learning. *CoRR*, abs/2112.09118, 2021. URL https://arxiv.org/abs/2112.09118. 5, 86, 87

[83] Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *CoRR*, abs/2208.03299, 2022. doi: 10.48550/arXiv.2208.03299. URL https://doi.org/10.48550/arXiv.2208.03299. 1, 7, 71, 76, 91

[84] Abhyuday Jagannatha and Hong Yu. Calibrating structured output predictors for natural language processing. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2078–2092. Association for Computational Linguistics, 2020. URL https://www.aclweb.org/anthology/2020.acl-main.188/. 42, 43, 44, 46, 47, 52, 54

[85] Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. Towards continual knowledge learning of language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=vfsRB5MImo9. 3, 56, 61, 70

[86] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT learn about the structure of language? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3651–3657, 2019. URL https://www.aclweb.org/anthology/P19-1356/. 24

[87] Robin Jia, Mike Lewis, and Luke Zettlemoyer. Question answering infused pre-training of general-purpose contextualized representations. In *ACL (Findings)*, pages 711–728. Association for Computational Linguistics, 2022. 70

[88] Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. Calibrating predictive model estimates to support personalized medicine. *J. Am. Medical Informatics Assoc.*, 19(2):263–274, 2012. doi: 10.1136/amiajnl-2011-000291. URL https://doi.org/10.1136/amiajnl-2011-000291. 54

[89] Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. X-FACTR: multilingual factual knowledge retrieval from pretrained language models. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5943–5959. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.479. URL https://doi.org/10.18653/v1/2020.emnlp-main.479. 4, 41

[90] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know. *Trans. Assoc. Comput. Linguistics*, 8:423–438, 2020. doi: 10.1162/tacl_a_00324. URL https://doi.org/10.1162/tacl_a_00324. 4, 27, 29, 32, 35, 48, 54, 61

[91] Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. How can we know *When* language models know? on the calibration of language models for question answering. *Trans. Assoc. Comput. Linguistics*, 9:962–977, 2021. doi: 10.1162/tacl_a_00407. URL https://doi.org/10.1162/tacl_a_00407. 4, 96

[92] Zhengbao Jiang, Luyu Gao, Jun Araki, Haibo Ding, Zhiruo Wang, Jamie Callan, and Graham Neubig. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. *CoRR*, abs/2212.02027, 2022. doi: 10.48550/arXiv.2212.02027. URL

https://doi.org/10.48550/arXiv.2212.02027. 7, 71, 91

[93] Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7969–7992. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.495. URL https://doi.org/10.18653/v1/2023.emnlp-main.495. 7, 71

[94] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547, 2021. doi: 10.1109/TBDATA.2019.2921572. URL https://doi.org/10.1109/TBDATA.2019.2921572. 80, 116

[95] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1147. URL https://doi.org/10.18653/v1/P17-1147. 1, 91

[96] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know. *CoRR*, abs/2207.05221, 2022. doi: 10.48550/arXiv.2207.05221. URL https://doi.org/10.48550/arXiv.2207.05221. 93, 96

[97] Amita Kamath, Robin Jia, and Percy Liang. Selective question answering under domain shift. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5684–5696. Association for Computational Linguistics, 2020. URL https://www.aclweb.org/anthology/2020.acl-main.503/. 42, 43, 54

[98] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. *CoRR*, abs/2211.08411, 2022. doi: 10.48550/arXiv.2211.08411. URL https://doi.org/10.48550/arXiv.

2211.08411. 2, 4

[99] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://doi.org/10.18653/v1/2020.emnlp-main.550. 5, 81, 84, 98

[100] Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A. Smith, Yejin Choi, and Kentaro Inui. Realtime QA: what's the answer right now? In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/9941624ef7f867a502732b5154d30cb7-Abstract-Datasets_and_Benchmarks.html. 1, 5

[101] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HklBjCEKvH. 6, 93, 99, 107

[102] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single QA system. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1896–1907. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.findings-emnlp.171. URL https://doi.org/10.18653/v1/2020.findings-emnlp.171. 42, 43, 45, 49, 50, 86

[103] Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 39–48. ACM, 2020. doi: 10.1145/

3397271.3401075. URL https://doi.org/10.1145/3397271.3401075. 5, 79, 80, 81, 84, 86

[104] Omar Khattab, Christopher Potts, and Matei Zaharia. Relevance-guided supervision for openqa with colbert. *CoRR*, abs/2007.00814, 2020. URL https://arxiv.org/abs/2007.00814. 77, 81, 84, 85

[105] Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *CoRR*, abs/2212.14024, 2022. doi: 10.48550/arXiv.2212.14024. URL https://doi.org/10.48550/arXiv.2212.14024. 93

[106] Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8082–8090. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/6319. 45

[107] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *CoRR*, abs/2210.02406, 2022. doi: 10.48550/arXiv.2210.02406. URL https://doi.org/10.48550/arXiv.2210.02406. 93

[108] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980. 21

[109] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. URL https://arxiv.org/abs/1612.00796. 36

[110] Vid Kocijan, Ana-Maria Cretu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. A surprisingly robust trick for the winograd schema challenge. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4837–4842. Association for Computational Linguis-

tics, 2019. doi: 10.18653/v1/p19-1478. URL https://doi.org/10.18653/v1/p19-1478. 54

[111] Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, and Chao Zhang. Calibrated language model fine-tuning for in- and out-of-distribution data. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1326–1340. Association for Computational Linguistics, 2020. URL https://www.aclweb.org/anthology/2020.emnlp-main.102/. 43, 54

[112] Andreas Kopf, Yannic Kilcher, Dimitri von Rutte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Rich'ard Nagyfi, ES Shahul, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. Openassistant conversations - democratizing large language model alignment. *ArXiv*, abs/2304.07327, 2023. URL https://api.semanticscholar.org/CorpusID:258179434. 70

[113] Yuri Kuratov and Mikhail Arkhipov. Adaptation of deep bidirectional multilingual transformers for russian language. *CoRR*, abs/1905.07213, 2019. URL http://arxiv.org/abs/1905.07213. 34

[114] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://doi.org/10.1162/tacl_a_00276. 1, 2, 60, 77, 91

[115] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. RACE: large-scale reading comprehension dataset from examinations. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 785–794. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1082. URL https://doi.org/10.18653/v1/d17-1082. 45

[116] Carolin Lawrence, Bhushan Kotnis, and Mathias Niepert. Attending to future tokens for bidirectional sequence generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1–10, Hong Kong, China, Novem-

ber 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1001. URL https://www.aclweb.org/anthology/D19-1001. 32

[117] Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. Internet-augmented language models through few-shot prompting for open-domain question answering. *CoRR*, abs/2203.05115, 2022. doi: 10.48550/arXiv.2203.05115. URL https://doi.org/10.48550/arXiv.2203.05115. 91

[118] Haejun Lee, Akhil Kedia, Jongwon Lee, Ashwin Paranjape, Christopher D. Manning, and Kyoung-Gu Woo. You only need one model for open-domain question answering. *CoRR*, abs/2112.07381, 2021. URL https://arxiv.org/abs/2112.07381. 71, 76, 84, 85, 86, 91

[119] Jinhyuk Lee, Alexander Wettig, and Danqi Chen. Phrase retrieval learns passage retrieval, too. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3661–3672. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.297. URL https://doi.org/10.18653/v1/2021.emnlp-main.297. 84

[120] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1612. URL https://doi.org/10.18653/v1/p19-1612. 76, 84, 86

[121] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=ryiAv2xAZ. 54

[122] Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. Pre-training via paraphrasing. *CoRR*, abs/2006.15020, 2020. URL https://arxiv.org/abs/2006.15020. 48

[123] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceed-*

ings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, pages 7871–7880. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.703. URL https://doi.org/10.18653/v1/2020.acl-main.703. 70

[124] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html. 1, 7, 45, 48, 71, 76, 84, 86, 91

[125] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220v3, 2016. URL http://arxiv.org/abs/1612.08220. 24

[126] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jingyuan Wang, Jian-Yun Nie, and Ji-Rong Wen. The web can be your oyster for improving large language models. *CoRR*, abs/2305.10998, 2023. doi: 10.48550/arXiv.2305.10998. URL https://doi.org/10.48550/arXiv.2305.10998. 107

[127] Bill Y. Lin, Frank F. Xu, Kenny Q. Zhu, and Seung-won Hwang. Mining cross-cultural differences and similarities in social media. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 709–719. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-1066. URL https://www.aclweb.org/anthology/P18-1066/. 37

[128] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013. 102

[129] Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. Reasoning over paragraph effects in situations. In Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen, editors, *Proceedings of the 2nd Workshop on Machine Reading for Question Answering, MRQA@EMNLP 2019, Hong Kong, China, November 4, 2019*, pages 58–62. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-5808. URL

https://doi.org/10.18653/v1/D19-5808. 45

[130] Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 6385–6400. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.423. URL https://doi.org/10.18653/v1/2023.findings-emnlp.423. 5

[131] Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Scott Yih. RA-DIT: retrieval-augmented dual instruction tuning. *CoRR*, abs/2310.01352, 2023. doi: 10.48550/ARXIV.2310.01352. URL https://doi.org/10.48550/arXiv.2310.01352. 71

[132] Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016. URL https://transacl.org/ojs/index.php/tacl/article/view/972. 24

[133] Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. *CoRR*, abs/2310.01889, 2023. doi: 10.48550/ARXIV.2310.01889. URL https://doi.org/10.48550/arXiv.2310.01889. 6

[134] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9):195:1–195:35, 2023. doi: 10.1145/3560815. URL https://doi.org/10.1145/3560815. 101

[135] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692. 36

[136] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7. 60

[137] Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. Www'18 open challenge: Financial opinion mining and question answering. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas,

and Panagiotis G. Ipeirotis, editors, *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1941–1942. ACM, 2018. doi: 10.1145/3184558.3192301. URL https://doi.org/10.1145/3184558.3192301. 86

[138] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 9802–9822. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.546. URL https://doi.org/10.18653/v1/2023.acl-long.546. 4, 107

[139] Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/E17-1083. 17

[140] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. Generation-augmented retrieval for open-domain question answering. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4089–4100. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.316. URL https://doi.org/10.18653/v1/2021.acl-long.316. 97

[141] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. URL https://arxiv.org/abs/1911.03894. 34

[142] Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730v1, 2018. URL http://arxiv.org/abs/1806.08730. 13, 16

[143] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In Ellen

Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-1260. URL https://doi.org/10.18653/v1/d18-1260. 45

[144] Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3470–3487. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.244. URL https://doi.org/10.18653/v1/2022.acl-long.244. 70

[145] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=0DcZxeWfOPt. 3

[146] Kenton Murray and David Chiang. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6322. URL https://www.aclweb.org/anthology/W18-6322. 46

[147] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332, 2021. URL https://arxiv.org/abs/2112.09332. 7, 71, 91, 108, 117

[148] Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation, WMT 2019, Florence, Italy, August 1-2, 2019 - Volume 2: Shared Task Papers, Day 1*, pages 314–319, 2019. URL https://www.aclweb.org/anthology/W19-5333/. 21, 50

[149] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. In Tarek Richard Besold, Antoine Bordes, Artur S. d'Avila Garcez,

and Greg Wayne, editors, *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016. URL http://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf. 86

[150] Tuan Dung Nguyen, Yuan-Sen Ting, Ioana Ciuca, Charlie O'Neill, Ze-Chang Sun, Maja Jablonska, Sandor Kruk, Ernest Perkowski, Jack W. Miller, Jason Li, Josh Peek, Kartheik Iyer, Tomasz Rózanski, Pranav Khetarpal, Sharaf Zaman, David Brodrick, Sergio J. Rodríguez Méndez, Thang Bui, Alyssa Goodman, Alberto Accomazzi, Jill P. Naiman, Jesse Cranney, Kevin Schawinski, and UniverseTBD. Astrollama: Towards specialized foundation models in astronomy. *CoRR*, abs/2309.06126, 2023. doi: 10.48550/ARXIV.2309.06126. URL https://doi.org/10.48550/arXiv.2309.06126. 70

[151] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. From doc2query to doctttttquery. *Online preprint*, 6(2), 2019. 86

[152] Barlas Oguz, Kushal Lakhotia, Anchit Gupta, Patrick S. H. Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen-tau Yih, Sonal Gupta, and Yashar Mehdad. Domain-matched pre-training tasks for dense retrieval. *CoRR*, abs/2107.13602, 2021. URL https://arxiv.org/abs/2107.13602. 84

[153] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774. URL https://doi.org/10.48550/arXiv.2303.08774. 2

[154] Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. Analyzing uncertainty in neural machine translation. *arXiv preprint arXiv:1803.00047*, 2018. 46

[155] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *CoRR*, abs/2203.02155, 2022. doi: 10.48550/arXiv.2203.02155. URL https://doi.org/10.48550/arXiv.2203.02155. 1, 56, 68, 70, 93

[156] Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. Fine-tuning or retrieval? comparing knowledge injection in llms. *CoRR*, abs/2312.05934, 2023. doi: 10.48550/ARXIV.2312.05934. URL https://doi.org/10.48550/arXiv.2312.05934. 61, 70

[157] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, and Jianfeng Gao. Check your facts and try again: Improving large language models with external knowledge and automated

feedback. *CoRR*, abs/2302.12813, 2023. doi: 10.48550/arXiv.2302.12813. URL https://doi.org/10.48550/arXiv.2302.12813. 107

[158] Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1005. URL https://www.aclweb.org/anthology/D19-1005. 19, 24

[159] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. Language models as knowledge bases? In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1250. URL https://doi.org/10.18653/v1/D19-1250. 1, 2, 3, 13, 14, 15, 16, 19, 20, 24, 27, 28, 29, 30, 32, 35, 43, 54, 56, 61

[160] Fabio Petroni, Patrick S. H. Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. How context affects language models' factual predictions. *CoRR*, abs/2005.04611, 2020. URL https://arxiv.org/abs/2005.04611. 48

[161] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. Kilt: a benchmark for knowledge intensive language tasks. In *arXiv:2009.02252*, 2020. 50

[162] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999. 46

[163] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. E-bert: Efficient-yet-effective entity embeddings for bert. *CoRR*, abs/1911.03681, 2019. URL https://arxiv.org/abs/1911.03681. 19, 24, 27, 29, 41, 54

[164] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022. 6, 93, 98, 100, 104

[165] Hongjing Qian, Yutao Zhu, Zhicheng Dou, Haoqi Gu, Xinyu Zhang, Zheng Liu, Ruofei

Lai, Zhao Cao, Jian-Yun Nie, and Ji-Rong Wen. Webbrain: Learning to generate factually correct articles for queries by grounding on large web corpus. *CoRR*, abs/2304.04358, 2023. doi: 10.48550/arXiv.2304.04358. URL https://doi.org/10.48550/arXiv.2304.04358. 91

[166] Yujia Qin, Zihan Cai, Dian Jin, Lan Yan, Shihao Liang, Kunlun Zhu, Yankai Lin, Xu Han, Ning Ding, Huadong Wang, Ruobing Xie, Fanchao Qi, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Webcpm: Interactive web search for chinese long-form question answering. *CoRR*, abs/2305.06849, 2023. doi: 10.48550/arXiv.2305.06849. URL https://doi.org/10.48550/arXiv.2305.06849. 71, 108

[167] Katyanna Quach. Researchers made an OpenAI GPT-3 medical chatbot as an experiment. it told a mock patient to kill themselves. *The Register*, 2020. URL https://www.theregister.com/2020/10/28/gpt3_medical_chatbot_experiment/. 42

[168] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019. URL https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf. 13, 16, 32, 70, 101

[169] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *CoRR*, abs/2305.18290, 2023. doi: 10.48550/ARXIV.2305.18290. URL https://doi.org/10.48550/arXiv.2305.18290. 70

[170] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL http://jmlr.org/papers/v21/20-074.html. 1, 7, 41, 45, 49, 70, 76, 78

[171] Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! Leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1487. URL https://www.aclweb.org/anthology/P19-1487. 13

[172] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Lan-*

guage Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016, pages 2383–2392. The Association for Computational Linguistics, 2016. doi: 10.18653/v1/d16-1264. URL https://doi.org/10.18653/v1/d16-1264. 45

[173] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-2124. URL https://www.aclweb.org/anthology/P18-2124/. 45

[174] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *arXiv preprint arXiv:2302.00083*, 2023. 93, 94, 99, 104, 107

[175] Deepak Ravichandran and Eduard Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 41–47. Association for Computational Linguistics, 2002. URL https://www.aclweb.org/anthology/P02-1006.pdf. 14, 16

[176] Matthew Richardson, Christopher J. C. Burges, and Erin Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 193–203. ACL, 2013. URL https://www.aclweb.org/anthology/D13-1020/. 45

[177] Michael Ringgaard, Rahul Gupta, and Fernando C. N. Pereira. SLING: A framework for frame semantic parsing. *CoRR*, abs/1710.07032, 2017. URL http://arxiv.org/abs/1710.07032. 38

[178] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5418–5426. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.437. URL https://doi.org/10.18653/v1/2020.emnlp-main.437. 1, 2, 49, 56, 61

[179] Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019. 5, 81, 83, 98

[180] Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. In-

vestigating a generic paraphrase-based approach for relation extraction. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy, April 2006. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/E06-1052. 17

[181] Devendra Singh Sachan, Siva Reddy, William L. Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 25968–25981, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/da3fde159d754a2555eaa198d2d105b2-Abstract.html. 71, 76, 84, 86, 91

[182] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/6399. 45

[183] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Masked language model scoring. In *Proceedings of ACL 2020*, 2020. URL https://arxiv.org/abs/1910.14659. 32, 34

[184] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=9Vrb9D0WI4. 3, 56, 61, 70, 112

[185] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Za-

haria. Colbertv2: Effective and efficient retrieval via lightweight late interaction. *CoRR*, abs/2112.01488, 2021. URL https://arxiv.org/abs/2112.01488. 80, 84

[186] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035, 2019. URL https://arxiv.org/pdf/1811.00146.pdf. 13

[187] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4462–4472. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1454. URL https://doi.org/10.18653/v1/D19-1454. 45

[188] Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? In *NeurIPS*, 2023. 2

[189] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. 95

[190] Stefan Schweter. Berturk - bert models for turkish, April 2020. URL https://doi.org/10.5281/zenodo.3770924. 34

[191] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016. URL https://www.aclweb.org/anthology/P16-1009/. 17

[192] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. REPLUG: retrieval-augmented black-box language models. *CoRR*, abs/2301.12652, 2023. doi: 10.48550/arXiv.2301.12652. URL https://doi.org/10.48550/arXiv.2301.12652. 1, 71, 91

[193] Xing Shi, Inkit Padhi, and Kevin Knight. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1159. URL https://www.aclweb.org/anthology/D16-1159. 24

[194] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html. 6, 7

[195] Noah A. Smith, Chris Dyer, Miguel Ballesteros, Graham Neubig, Lingpeng Kong, and Adhiguna Kuncoro. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 1249–1258, 2017. URL https://www.aclweb.org/anthology/E17-1117/. 24

[196] Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. ASQA: factoid questions meet long-form answers. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8273–8288. Association for Computational Linguistics, 2022. URL https://aclanthology.org/2022.emnlp-main.566. 92, 93, 100, 102

[197] Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wentau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. One embedder, any task: Instruction-finetuned text embeddings. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1102–1121. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.71. URL https://doi.org/10.18653/v1/2023.findings-acl.71. 5

[198] Zhiqing Sun, Xuezhi Wang, Yi Tay, Yiming Yang, and Denny Zhou. Recitation-augmented language models. *CoRR*, abs/2210.01296, 2022. doi: 10.48550/arXiv.2210.01296. URL https://doi.org/10.48550/arXiv.2210.01296. 5, 97

[199] Zhiqing Sun, Yikang Shen, Hongxin Zhang, Qinhong Zhou, Zhenfang Chen, David D. Cox, Yiming Yang, and Chuang Gan. SALMON: self-alignment with principle-following reward models. *CoRR*, abs/2310.05910, 2023. doi: 10.48550/ARXIV.2310.05910. URL https://doi.org/10.48550/arXiv.2310.05910. 70

[200] Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David D.

Cox, Yiming Yang, and Chuang Gan. Principle-driven self-alignment of language models from scratch with minimal human supervision. *CoRR*, abs/2305.03047, 2023. doi: 10.48550/ARXIV.2305.03047. URL https://doi.org/10.48550/arXiv.2305.03047. 70

[201] Martin Sundermeyer, Hermann Ney, and Ralf Schlüter. From feedforward to recurrent LSTM neural networks for language modeling. *IEEE ACM Trans. Audio Speech Lang. Process.*, 23(3):517–529, 2015. doi: 10.1109/TASLP.2015.2400218. URL https://doi.org/10.1109/TASLP.2015.2400218. 32

[202] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1421. URL https://doi.org/10.18653/v1/n19-1421. 45

[203] Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. olmpics - on what language model pre-training captures. *Trans. Assoc. Comput. Linguistics*, 8:743–758, 2020. URL https://transacl.org/ojs/index.php/tacl/article/view/2041. 41, 54

[204] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023. 70

[205] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Prakash Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. Transformer memory as a differentiable search index. In *NeurIPS*, 2022. 83, 114

[206] Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4593–4601, 2019. URL https://www.aclweb.org/anthology/P19-1452/. 24, 54

[207] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. What do you learn from context? Probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019,*

New Orleans, LA, USA, May 6-9, 2019, 2019. URL https://openreview.net/forum?id=SJzSgnRcKX. 24

[208] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/65b9eea6e1cc6bb9f0cd2a47751a186f-Abstract-round2.html. 77, 86

[209] Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D. Manning, and Chelsea Finn. Fine-tuning language models for factuality. *CoRR*, abs/2311.08401, 2023. doi: 10.48550/ARXIV.2311.08401. URL https://doi.org/10.48550/arXiv.2311.08401. 70

[210] Kushal Tirumala, Aram H. Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/fa0509f4dab6807e2cb465715bf2d249-Abstract-Conference.html. 3, 70

[211] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1499–1509, 2015. URL https://www.aclweb.org/anthology/D15-1174/. 16

[212] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023. doi: 10.48550/arXiv.2302.13971. URL https://doi.org/10.48550/arXiv.2302.13971. 1

[213] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel,

Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/ARXIV.2307.09288. URL https://doi.org/10.48550/arXiv.2307.09288. 56, 58, 70

[214] Trieu H. Trinh and Quoc V. Le. A simple method for commonsense reasoning. *CoRR*, abs/1806.02847, 2018. URL http://arxiv.org/abs/1806.02847. 13, 54

[215] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. In Phil Blunsom, Antoine Bordes, Kyunghyun Cho, Shay B. Cohen, Chris Dyer, Edward Grefenstette, Karl Moritz Hermann, Laura Rimell, Jason Weston, and Scott Yih, editors, *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017, Vancouver, Canada, August 3, 2017*, pages 191–200. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-2623. URL https://doi.org/10.18653/v1/w17-2623. 45

[216] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *CoRR*, abs/2212.10509, 2022. doi: 10.48550/arXiv.2212.10509. URL https://doi.org/10.48550/arXiv.2212.10509. 6, 93, 94, 99, 101, 104, 107

[217] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artières, Axel-Cyrille Ngonga Ngomo, Norman Heino, Éric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinform.*, 16:138:1–138:28, 2015. doi: 10.1186/s12859-015-0564-6. URL https://doi.org/10.1186/

s12859-015-0564-6. 86

[218] Neeraj Varshney, Man Luo, and Chitta Baral. Can open-domain QA reader utilize external knowledge efficiently like humans? *CoRR*, abs/2211.12707, 2022. doi: 10.48550/arXiv.2211.12707. URL https://doi.org/10.48550/arXiv.2211.12707. 96

[219] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html. 6, 49, 76

[220] Ellen M. Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. TREC-COVID: constructing a pandemic information retrieval test collection. *SIGIR Forum*, 54(1):1:1–1:12, 2020. doi: 10.1145/3451964.3451965. URL https://doi.org/10.1145/3451964.3451965. 86

[221] Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry W. Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc V. Le, and Thang Luong. Freshllms: Refreshing large language models with search engine augmentation. *CoRR*, abs/2310.03214, 2023. 5

[222] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7534–7550. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.609. URL https://doi.org/10.18653/v1/2020.emnlp-main.609. 86

[223] Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. Do NLP models know numbers? probing numeracy in embeddings. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5306–5314. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1534. URL https://doi.org/10.18653/v1/D19-1534. 41

[224] Alex Wang and Kyunghyun Cho. BERT has a mouth, and it must speak: BERT as a

markov random field language model. *CoRR*, abs/1902.04094, 2019. URL http://arxiv.org/abs/1902.04094. 32

[225] Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, Anima Anandkumar, and Bryan Catanzaro. Shall we pretrain autoregressive language models with retrieval? A comprehensive study. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7763–7786. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.emnlp-main.482. 71

[226] Cunxiang Wang, Pai Liu, and Yue Zhang. Can generative pre-trained language models serve as knowledge bases for closed-book qa? In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3241–3251. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.ACL-LONG.251. URL https://doi.org/10.18653/v1/2021.acl-long.251. 56, 61, 70

[227] Kexin Wang, Nils Reimers, and Iryna Gurevych. TSDAE: using transformer-based sequential denoising auto-encoderfor unsupervised sentence embedding learning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 671–688. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-emnlp.59. URL https://doi.org/10.18653/v1/2021.findings-emnlp.59. 87

[228] Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. GPL: generative pseudo labeling for unsupervised domain adaptation of dense retrieval. *CoRR*, abs/2112.07577, 2021. URL https://arxiv.org/abs/2112.07577. 87

[229] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *CoRR*, abs/2203.11171, 2022. doi: 10.48550/arXiv.2203.11171. URL https://doi.org/10.48550/arXiv.2203.11171. 101

[230] Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. How far can camels go? exploring the state of instruction tuning on open resources. *CoRR*, abs/2306.04751, 2023. doi: 10.48550/ARXIV.2306.04751. URL

https://doi.org/10.48550/arXiv.2306.04751. 70

[231] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR. 3, 56, 61, 70, 112

[232] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *CoRR*, abs/2206.07682, 2022. doi: 10.48550/arXiv.2206.07682. URL https://doi.org/10.48550/arXiv.2206.07682. 2

[233] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022. URL https://arxiv.org/abs/2201.11903. 6, 92, 101, 102

[234] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019. URL http://arxiv.org/abs/1910.03771. 34

[235] Chaoyi Wu, Weixiong Lin, Xiaoman Zhang, Ya Zhang, Yanfeng Wang, and Weidi Xie. Pmc-llama: Towards building open-source language models for medicine, 2023. 70

[236] Shijie Wu, Alexis Conneau, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. Emerging cross-lingual structure in pretrained language models. In *Proceedings of ACL 2020*, 2020. URL http://arxiv.org/abs/1911.01464. 38

[237] Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=TrjbxzRcnf-. 6

[238] Mengzhou Xia, Mikel Artetxe, Chunting Zhou, Xi Victoria Lin, Ramakanth Pasunuru, Danqi Chen, Luke Zettlemoyer, and Veselin Stoyanov. Training trajectories of language models across scales. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13711–13738. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.767. URL https://doi.org/10.18653/v1/2023.acl-long.

767. 70

[239] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=zeFrfgyZln. 81, 84, 86

[240] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. Effective long-context scaling of foundation models. *CoRR*, abs/2309.16039, 2023. doi: 10.48550/ARXIV.2309.16039. URL https://doi.org/10.48550/arXiv.2309.16039. 6

[241] Sohee Yang and Minjoon Seo. Is retriever merely an approximator of reader? *CoRR*, abs/2010.10999, 2020. URL https://arxiv.org/abs/2010.10999. 82

[242] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764, 2019. 33

[243] Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. Reference-aware language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1850–1859, 2017. URL https://www.aclweb.org/anthology/D17-1197/. 24

[244] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *CoRR*, abs/2210.03629, 2022. doi: 10.48550/arXiv.2210.03629. URL https://doi.org/10.48550/arXiv.2210.03629. 6, 7, 93, 104

[245] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators. *CoRR*, abs/2209.10063, 2022. doi: 10.48550/arXiv.2209.10063. URL https://doi.org/10.48550/arXiv.2209.10063. 5, 97

[246] Wenhao Yu, Zhihan Zhang, Zhenwen Liang, Meng Jiang, and Ashish Sabharwal. Improving language models via plug-and-play retrieval feedback. *CoRR*, abs/2305.14002, 2023. doi: 10.48550/arXiv.2305.14002. URL https://doi.org/10.48550/arXiv.

2305.14002. 107

[247] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In *NeurIPS*, 2020. 6

[248] Yury Zemlyanskiy, Michiel de Jong, Joshua Ainslie, Panupong Pasupat, Peter Shaw, Linlu Qiu, Sumit Sanghai, and Fei Sha. Generate-and-retrieve: Use your predictions to improve retrieval for semantic parsing. In Nicoletta Calzolari, Chu-Ren Huang, Hansaem Kim, James Pustejovsky, Leo Wanner, Key-Sun Choi, Pum-Mo Ryu, Hsin-Hsi Chen, Lucia Donatelli, Heng Ji, Sadao Kurohashi, Patrizia Paggio, Nianwen Xue, Seokhwan Kim, Younggyun Hahm, Zhong He, Tony Kyungil Lee, Enrico Santus, Francis Bond, and Seung-Hoon Na, editors, *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 4946–4951. International Committee on Computational Linguistics, 2022. URL https://aclanthology.org/2022.coling-1.438. 107

[249] Fengji Zhang, Bei Chen, Yue Zhang, Jin Liu, Daoguang Zan, Yi Mao, Jian-Guang Lou, and Weizhu Chen. Repocoder: Repository-level code completion through iterative retrieval and generation. *CoRR*, abs/2303.12570, 2023. doi: 10.48550/arXiv.2303.12570. URL https://doi.org/10.48550/arXiv.2303.12570. 107

[250] Ruohong Zhang, Luyu Gao, Chen Zheng, Zhen Fan, Guokun Lai, Zheng Zhang, Fangzhou Ai, Yiming Yang, and Hongxia Yang. A self-enhancement approach for domain-specific chatbot training via knowledge mining and digest. *CoRR*, abs/2311.10614, 2023. doi: 10.48550/ARXIV.2311.10614. URL https://doi.org/10.48550/arXiv.2311.10614. 70

[251] Yunfeng Zhang, Q. Vera Liao, and Rachel K. E. Bellamy. Effect of confidence and explanation on accuracy and trust calibration in ai-assisted decision making. In Mireille Hildebrandt, Carlos Castillo, Elisa Celis, Salvatore Ruggieri, Linnet Taylor, and Gabriela Zanfir-Fortuna, editors, *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*, pages 295–305. ACM, 2020. doi: 10.1145/3351095.3372852. URL https://doi.org/10.1145/3351095.3372852. 54

[252] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1441–1451, 2019. URL https://www.aclweb.org/anthology/P19-1139/. 19, 24

[253] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian

Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models. *CoRR*, abs/2303.18223, 2023. doi: 10.48550/arXiv.2303.18223. URL https://doi.org/10.48550/arXiv.2303.18223. 70

[254] Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. Towards a unified multi-dimensional evaluator for text generation. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2023–2038. Association for Computational Linguistics, 2022. URL https://aclanthology.org/2022.emnlp-main.131. 103

[255] Zexuan Zhong, Tao Lei, and Danqi Chen. Training language models with memory augmentation. *CoRR*, abs/2205.12674, 2022. doi: 10.48550/arXiv.2205.12674. URL https://doi.org/10.48550/arXiv.2205.12674. 6

[256] Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: less is more for alignment. *CoRR*, abs/2305.11206, 2023. doi: 10.48550/ARXIV.2305.11206. URL https://doi.org/10.48550/arXiv.2305.11206. 70

[257] Pei Zhou, Rahul Khanna, Bill Yuchen Lin, Daniel Ho, Xiang Ren, and Jay Pujara. Can BERT reason? logically equivalent probes for evaluating the inference capabilities of language models. *CoRR*, abs/2005.00782, 2020. URL https://arxiv.org/abs/2005.00782. 41

[258] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *CoRR*, abs/2307.13854, 2023. doi: 10.48550/ARXIV.2307.13854. URL https://doi.org/10.48550/arXiv.2307.13854. 1, 5

[259] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn RL. *CoRR*, abs/2402.19446, 2024. doi: 10.48550/ARXIV.2402.19446. URL https://doi.org/10.48550/arXiv.2402.19446. 7

[260] Zeyuan Allen Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. *CoRR*, abs/2309.14316, 2023. doi: 10.48550/ARXIV.2309.14316.

URL https://doi.org/10.48550/arXiv.2309.14316. 2, 3, 56, 61, 62, 65, 66, 69, 70

[261] Zeyuan Allen Zhu and Yuanzhi Li. Physics of language models: Part 3.2, knowledge manipulation. *CoRR*, abs/2309.14402, 2023. doi: 10.48550/ARXIV.2309.14402. URL https://doi.org/10.48550/arXiv.2309.14402. 70

[262] Geoffrey Zweig and Christopher JC Burges. The Microsoft Research sentence completion challenge. *Microsoft Research, Redmond, WA, USA, Tech. Rep. MSR-TR-2011-129*, 2011. URL https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR_SCCD.pdf. 13