Designing accurate retrieval systems using language models

by

Devendra Singh Sachan
School of Computer Science
McGill University, Montreal

A thesis submitted to the School of Computer Science, McGill University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2024

© Devendra Singh Sachan, 2024.

Abstract

by

Devendra Singh Sachan

Submitted to the School of Computer Science, McGill University

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

The success of pre-trained language models (PLMs) in language understanding tasks has attracted attention to these models being applied as building blocks in information retrieval systems. Indeed, due to the application of these PLMs, the retrieval accuracy has improved considerably over traditional approaches. This improved retrieval has led to a big boost in the performance of downstream knowledge-intensive tasks such as question answering (QA). However, the training methods of these systems still suffer from a number of shortcomings, some being——reliance on thousands of aligned question-document pairs, dependence on the output of BM25 retriever to mine hard-negative examples, multi-stage training, etc. In this thesis, we propose several methods to address these drawbacks and further improve the underlying systems. Specifically, our contributions include: (i) a unifying approach of unsupervised pre-training of dense retrievers followed by supervised finetuning to improve retrieval accuracy, (ii) an end-to-end training approach to jointly train a system consisting of a language model and a dense retriever for QA tasks, (iii) using frozen large language models as unsupervised passage re-rankers to improve retrieval accuracy, and (iv) a simplified yet accurate approach to train a dense retriever without aligned question-document pairs and without using hard-negative examples. Our proposed approaches are more robust and have achieved new state-of-the-art results on multiple retrieval and QA benchmarks.

Résumé

par

Devendra Singh Sachan

Soumis au School of Computer Science, Université McGill

en satisfaction partielle des exigences de la degree of

DOCTOR OF PHILOSOPHY

Le succès des modèles linguistiques pré-entraînés (PLM) dans les tâches de compréhension du langage a attiré l'attention sur l'application de ces modèles comme éléments de base dans les systèmes de recherche d'informations. En effet, grâce à l'application de ces PLM, la précision de la récupération s'est considérablement améliorée par rapport aux approches traditionnelles. Cette récupération améliorée a conduit à une forte amélioration des performances des tâches en aval à forte intensité de connaissances, telles que la réponse aux questions. Cependant, les méthodes de formation de ces systèmes souffrent encore d'un certain nombre de lacunes, certaines étant : la dépendance à l'égard de milliers de paires question-document alignées, la dépendance à l'égard du résultat du récupérateur BM25 pour extraire des exemples fortement négatifs, la formation en plusieurs étapes, etc. Dans cette thèse, nous proposons plusieurs méthodes pour remédier à ces inconvénients et améliorer davantage les systèmes sous-jacents. Plus précisément, nos contributions comprennent : (i) une approche unificatrice de pré-entrainement non supervisée des récupérateurs denses suivie d'un réglage fin supervisé pour améliorer la précision de la récupération, (ii) une approche d'entrainement de bout en bout pour former conjointement un système composé d'un modèle de langage et d'un récupérateur dense pour les tâches d'assurance qualité, (iii) utiliser de grands modèles de langage figés comme reclasseurs de passages non supervisés pour améliorer la précision de la récupération, et (iv) une approche simplifiée mais précise pour former un récupérateur dense sans paires question-document alignées et sans utiliser d'exemples strictement négatifs.

Les approches proposées sont plus robustes et ont permis d'obtenir de nouveaux résultats de pointe sur plusieurs tests de récupération d'information et de question-réponse.

Contribution to Original Knowledge

This thesis makes several contributions toward improving passage rankings in open-domain retrieval tasks and improving the accuracy of question answering (QA) models. Our proposed approaches include better training mechanisms for dense retrievers, end-to-end training of QA models, and leveraging large language models to improve passage retrieval. Specifically, we make the following contributions:

(i) To train dense retrievers, we propose a unified approach of unsupervised pre-training followed by supervised finetuning. We explore two pre-training strategies and show that they impart substantial gains in zero-shot retrieval accuracy over masked language models which aids the finetuning process leading to improved overall results (Chapter 3).

(ii) We propose an end-to-end training method, EMDR^2 to jointly train a reader and retriever pipeline for the task of open-domain question answering. EMDR^2 trains the network in a single training cycle and requires just question-answer pairs without the requirement of intermediate document annotations (Chapter 4).

(iii) We introduce an unsupervised re-ranking approach, UPR, which leverages large language models as a black box to improve the rankings of a document list from a first-stage retriever. We conduct extensive experiments across diverse retrievers, language models, and datasets to empirically quantify UPR's performance (Chapter 5).

(iv) We propose ART, a denoising approach to train dense retrievers by distilling question reconstruction feedback from large language models. ART requires only questions and unaligned

document collections during the training step eliminating the requirement of hard-negative examples. We compare ART's performance against prior supervised and unsupervised retrieval models and also perform extensive ablation studies to understand its training process (Chapter 6).

Contribution of Authors

- Chapter 1 provides an introduction to the thesis contributions and Chapter 2 contains the technical background for the thesis. Both these chapters were written by me, and are loosely inspired by the thesis of Ryan Lowe.

- Chapter 3 is based on Sachan et al. (2021a), which is a paper published at the conference of the *Association for Computational Linguistics* (ACL) 2021. I came up with the original idea of pre-training retrievers, implemented all the approaches, performed model training, designed experiments, conducted the evaluation runs, and wrote the first version of the paper. Mostofa Patwary and Mohammad Shoeybi were the other core contributors who helped in idea brainstorming, planning, technical supervision, debugging code, and were involved in paper writing. Neel Kant helped in implementing an early version of the inverse cloze task training and the functionality to embed documents. Wei Ping assisted in paper writing and contributed to the experimental plan. William Hamilton and Bryan Catanzaro provided the overall technical supervision, set up the research direction, and contributed to paper writing.

- Chapter 4 is based on Sachan et al. (2021b), which is a paper published at the conference *Neural Information Processing Systems* (NeurIPS) 2021. I conceived the idea of end-to-end training of a retrieval-augmented system, implemented the EMDR^2 approach, and performed the experiments. Dani Yogatama was involved in every other aspect of the project, contributed to brainstorming, technical guidance, and project facilitation. Chris Dyer provided the overall technical supervision, contributed to idea refinement, and formulated the expectation-maximization interpretation of the end-to-end training. I, Dani, and Chris wrote

most of the paper. William Hamilton contributed to the idea formulation and both William and Siva Reddy were involved in writing the paper.

- Chapter 5 is based on Sachan et al. (2022), which is a paper published at the *Empirical Methods in Natural Language Processing* (EMNLP) 2022 conference. I implemented the underlying re-ranking approach, conceptualized and conducted the experiments, and was involved in writing the paper. Mike Lewis suggested the idea of zero-shot question generation, helped in implementation, code debugging, planning, execution, and writing the paper. Wen-tau Yih contributed to the experimental methodology, discussions, writing the paper, and drawing analogies with related approaches in statistical language modeling literature. Mandar Joshi contributed to the writing of the paper and Armen Aghajanyan helped in code debugging. Joelle Pineau contributed to the technical supervision, experimental plan, and writing the paper. Luke Zettlemoyer was involved in project initiation, overall supervision, discussions, designing experiments, and contributed to writing the paper.

- Chapter 6 is based on Sachan et al. (2023), which is a journal paper published in the *Transactions of the Association for Computational Linguistics*, Vol. 11, 2023. It was presented at the *Association for Computational Linguistics* (ACL) 2023 conference. I proposed the ART training objective, implemented the algorithm, conducted the experimental runs, and was involved in paper writing. Mike Lewis came up with the idea of distilling question generation scores from the language model to the retriever, contributed to experiments, code debugging, and helped in writing the paper. Dani Yogatama contributed to the idea refinement, autoencoding interpretation of the underlying training process, and writing the paper. Luke Zettlemoyer contributed to the experimental design and was involved in writing the paper. Manzil Zaheer was involved in project facilitation, devised the experimental plan, ablation studies, overall supervision, and helped in writing the paper. Joelle Pinneau was involved in project initiation, provided the technical guidance, overall supervision, and contributed to the writing of the paper.

Acknowledgments

First and foremost, I want to thank my supervisor, Joelle Pineau for all her support during the course of my Ph.D. studies. I have benefited tremendously from her advice throughout. She has been an extremely kind and patient advisor to me. My research style has evolved significantly as a result of her sharp insights and her ability to ask questions to set forth a clarifying perspective. I am grateful to her for giving me the freedom to pursue my research projects, foster collaborations, and learn, fail, and improve by trying.

It has been inspiring to see Joelle being a champion of advocating open-source in research especially as the NLP field drifts towards ever larger models. I also got the privilege to witness first-hand her unwavering commitment to the advancement of fundamental research and making research opportunities accessible to young researchers. I only got to know Joelle towards the beginning of my third year as a graduate student. I just wish that I should have approached her somewhat earlier so that we could have worked on more projects and I could have had the opportunity to learn much more from her.

I wholeheartedly thank William Hamilton for all his support and guidance during the first couple of years in the Ph.D. program. In my first year of the program, I worked with William on two projects related to applying graph neural networks to language tasks. Although these projects are not a part of this thesis I am still proud of all the work we did. I am also very grateful to Dani Yogatama for providing me the opportunity to work with him and being extremely supportive. Dani has a very optimistic, energetic, and friendly demeanour and is a real pleasure to work with. I am very proud of both the projects which have resulted from our collaboration. I also want to

thank my Ph.D. proposal exam committee members Jackie Cheung, Dzmitry Bahdanau, and Oana Balmau for their time and for providing valuable feedback.

Most of my time as a Ph.D. student was spent at the Mila-Quebec AI Institute, a place that became my second home in Montreal. Mila has a very vibrant ecosystem with some of the best in-class facilities provided to students. I want to thank the academic staff of Mila for providing such an enriching atmosphere to students. My special appreciation goes to the administrative staff of both Mila and McGill University for all their support during the COVID-19 pandemic.

During my Ph.D., I had the fortune to do a few internships that helped me gain research experience in industrial labs. I am thankful to my mentors who provided me the generous opportunity to be a part of their team: Bryan Catanzaro, Mostofa Patwary, Mohammad Shoeybi from Nvidia; Luke Zettlemoyer and Mike Lewis from FAIR-Meta AI; and Manzil Zaheer and Rob Fergus from DeepMind. I am also thankful to Marcello Federico from Amazon AWS and to Yuan Cao and Jeffrey Zhao from Google Brain for hosting me as a student researcher in their team.

Among the joys of being a graduate student is being able to collaborate with other researchers when working on a project. I have really enjoyed working together and learning from my co-authors: Joelle Pineau, Mike Lewis, Yuhao Zhang, Peng Qi, Dani Yogatama, Manzil Zaheer, Scott Yih, William Hamilton, Jeffrey Zhao, Yuan Cao, Luke Zettlemoyer, Chris Dyer, Mostofa Patwary, Mohammad Shoeybi, Prashant Mathur and many others in both my school and the companies I have interned at. I am also grateful to Mrinmaya Sachan for his mentorship and friendship over the course of all the years that we have known each other. He has always been very sweet in helping and guiding me throughout.

Life in a graduate school can be quite challenging at times but I feel privileged to have friends who have kept me upbeat and made my time much more cherishable: Harsh Satija, Clara Lacroce, Ladislav Rampasek, Breandan Considine, Shehzaad Dhuliawala, Anirban Laha, Ankesh Anand, Disha Shrivastava, Komal Teru, Kushal Arora, Jules, Yue Dong, Motahareh Sohrabi amongst many more in McGill and Mila. My friends from Pittsburgh and India have a special place in my heart: Avinash Bukittu, Shreya Kadambi, Suhaila Shakiah, Mrigesh, Atif Ahmed, Swapnil Singhavi,

Irfan Khan, Taranveer Singh, and the entire Petuum gang. A special shoutout to my team in DeepMind, New York who hosted me for my first in-person internship and made my time so much memorable: John Schultz, Will Grawthwohl, Madison Gleissner, Surya Bhupatiraju, Sheila Babayan, Kavya Kopparapu, Kenneth Marino, Ishita Dasgupta, Manzil Zaheer, and Rob Fergus.

I feel blessed to have incredibly supportive family members throughout the course of my personal endeavors and professional journey. I want to express my deepest gratitude to my parents: Smt. Chandra Prabha and Shri. Mahesh Singh, my elder brother Deependra Singh, my sister-in-law Richa Sachan, and my two adorable little nieces: Saachi and Pari, for always being there for me, rooting for me, and believing in me. They have always been by my side no matter the circumstances even though I could only visit them a few times in the last seven years that I have been outside of India. Special mention to my elder brother for being a rock-solid pillar in my life, my inspiration, and for everything he has done. Needless to say, I wouldn't have reached so far in my educational pursuits if not for my family's relentless support. I love you all!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Textual information needs, such as seeking answers to *information-seeking questions*, are fundamental to our modern human society and hence tools or services that serve these needs are of critical importance. As digital data especially text becomes rapidly available online, it presents an attractive opportunity to provide answers to these questions by automated approaches. Indeed, services such as search engines offer this functionality and as a result, their application has become ubiquitous in our daily lives.

Central to the functionality of search engines are information retrieval algorithms. These retrieval algorithms are designed such that they are fast and accurate which allows them to search over large text collections and then return relevant documents. Often, top-ranked results are processed by an answer extraction module to produce short text snippets as the answer to the question. More formally, in the ad-hoc retrieval setting, given a question, the task is to retrieve a ranked list of relevant documents from a large collection of documents such as web pages. This is a challenging task because the queries are often imprecise and there can be millions of candidate documents. For a rich user experience, it is crucial to return the most relevant documents which in turn would also lead to producing the correct final answer.

Given the critical importance of ad-hoc retrieval methods in serving information-seeking queries, a lot of progress has been made in the sub-fields of information retrieval and question answering

(QA). Numerous approaches have been proposed over the course of the last few decades with the recent excitement being centered around pre-trained language models (PLMs) (Manning et al., 2008; Chen, 2018; Zhu et al., 2023). Despite impressive progress, prevailing algorithms still suffer from several limitations such as being dependent on thousands of annotated examples and complicated training processes. Addressing these limitations to improve the performance of retrieval systems lies at the core of this thesis. In the following sections, we first cover the traditional methods for information retrieval including QA and their evolution towards PLM-based methods (§1.1). Then, we expand on the limitations of the prevailing models (§1.2) following which we provide a preview of our research contributions that are included in this thesis (§1.3).

1.1 Setting the Stage

Early successful approaches for information retrieval represented documents using bag-of-word features where each entry denoted the occurrence of a *key-term* in the document. It was found that a more useful feature representation of a document is the product of its term frequency and inverse document frequency (tf-idf; Sparck Jones, 1988). These led to the vector space models for text, where the documents were ranked based on their cosine similarity with the query (Manning et al., 2008). Subsequent approaches such as Okapi BM25 additionally incorporated document lengths into the tf-idf ranking function which led to a better estimation of document relevance to a query (Robertson and Zaragoza, 2009).

Although BM25 has been very popular, it tends to fail on semantic matches, *i.e.*, it ignores relevant documents which do not have term overlap with the queries. To circumvent the problems arising due to reliance on lexical matching, approaches were proposed to represent documents as a collection of semantic topics (Deerwester et al., 1990; Wei and Croft, 2006). Another popular class of approaches for ad-hoc retrieval were based on the *learning to rank framework* that trained classifiers according to pairwise or listwise ranking losses given queries, documents, and relevance judgments as the training data (Burges, 2010).

While methods like BM25 and learning to rank helped to propel the field forward they were based on leveraging hand-crafted features. An alternative direction is to devise objectives to automatically learn input features from textual data. This was pioneered by neural network models that were trained to learn distributed word representations which is also known as *word embeddings* (Collobert et al., 2011; Mikolov et al., 2013). In the embedding space, semantically similar words lie closer to each other and vice versa. Word embedding approaches were also extended to learn embedding representations at the document level (Le and Mikolov, 2014; Dai et al., 2015). Document embeddings are useful as representing both the query and document with embeddings has been shown to lead to faster retrieval on compute accelerators (Johnson et al., 2021).

Building over word embeddings, the next series of approaches considered the sequential ordering of words in the text. Representative approaches of this kind to process sequences include recurrent neural networks such as LSTM networks (Melis et al., 2018; Merity et al., 2018), convolutional neural networks (Kim, 2014; Johnson and Zhang, 2015), and self-attentional models (Parikh et al., 2016; Vaswani et al., 2017). In parallel, there were also efforts to extend these sequential models of text for the question answering tasks (Lee et al., 2017; Chen et al., 2017).

Over the last few years, language models consisting of neural networks when trained using lots of data have brought significant advances in the natural language processing (NLP) field. These models when pre-trained using self-supervised learning objectives such as masked token prediction or next token prediction leads to improved text representations. Prototypical examples of these models include BERT (Devlin et al., 2019) and GPT (Radford et al., 2019). Often referred to as pre-trained language models (PLMs), they provide a powerful foundational or set of base weights that can be finetuned to improve performance on language understanding and generation tasks sometimes even rivaling human performance (Brown et al., 2020; Chowdhery et al., 2022).

Of particular importance is the fact that these PLMs can be assembled to provide useful modules that can be adapted for ad-hoc retrieval tasks. In the following sections, we describe how these PLMs have led to the emergence of modern retrievers and associated retrieval-augmented systems that in turn have shaped the contributions presented in this thesis.

1.2 Focus of this Thesis

This thesis focuses on building systems that can accurately answer information-seeking questions with the help of textual documents. Designing such a system naturally decomposes itself into two sub-tasks of *open-domain retrieval* and *question answering* (QA). In open-domain retrieval, the goal is to select a small subset of relevant documents from a large array of text data while in open-domain question answering, the task is to read these retrieved documents to answer the question. It is worth mentioning that for a system to achieve good performance, it requires both the retrieval and the QA models to be accurate.

Due to the importance of these tasks in both research and practical applications, they have been extensively studied and thus have a rich history in the NLP field (§1.1). In late 2018, the success of pre-trained language models (PLMs) in language understanding tasks motivated researchers to apply PLMs as building blocks in retrieval and QA models. Subsequently, these efforts led to PLMs forming the backbone of modern retrieval and QA approaches. In the following paragraphs, we elucidate the key ideas that led to advances in these tasks post the introduction of PLMs while also highlighting the limitations associated with them. Finally, based on these limitations, we outline the research questions that we delve into in this thesis (§1.2.3).

1.2.1 Open-Domain Retrieval

In open-domain retrieval, given a question, the task is to retrieve a set of relevant documents from a large collection of documents. This is a challenging task because the queries are often imprecise and search space typically consists of millions of candidate documents. Directly leveraging PLMs such as BERT for retrieval presents an appealing option because they can be used to represent both queries and evidence documents in a latent (or dense) embedding space. However, due to the rather poor discriminative ability of BERT representations, they have been shown to underperform traditional methods in large-scale retrieval tasks.

A successful recipe to improve document retrieval has been to adapt the PLM weights by fine-

tuning them on examples consisting of positive (and negative) question-document pairs. In this method, a dense retriever which is generally modeled according to a dual-encoder network (Bromley et al., 1994), is first initialized with BERT weights. Then, the retriever is finetuned using contrastive training to maximize the similarity score of the positive question-document pairs (Lee et al., 2019). Incorporating additional tricks such as using in-batch negative examples and hard-negative examples during training further improves the retrieval accuracy (Karpukhin et al., 2020). This training process has resulted in dense retrievers obtaining an improvement in document rankings compared to BM25 on benchmark datasets.

The approach of finetuning a PLM to train dense retrievers has been instrumental in accelerating the progress in open-domain retrieval tasks. However, it is unclear if initializing the retriever with the PLM weights (such as BERT) is an optimal strategy or not as by default BERT achieves a very low recall in text ranking tasks. This is also reflected in the training objective of BERT where it is required to predict the masked tokens among all the tokens in the vocabulary. This presents a very different learning challenge as compared to the retrieval task, where the model needs to distinguish the correct document against a large number of candidates. Consequently, we argue that using the PLM weights to initialize retrievers is a rather suboptimal choice. In this thesis, we systematically study the question of how to best initialize the retriever weights in order to improve the overall effectiveness of the finetuning step.

Finetuning-based approaches to train retrievers are dependent on the availability of labeled training examples, *i.e.*, positive question and document pairs. Specifically, to obtain peak performance, these retrieval models require thousands of human-annotated question and document pairs for training, which is expensive as well as time-consuming to obtain thus limiting their rapid adoption. Furthermore, retrieval models tend to have poor generalization properties and as a result, separate models need to be trained for each dataset (Thakur et al., 2021). In order to address these bottlenecks, it is desirable to have a robust approach that does not require any kind of training data or finetuning step and yet improves document rankings. To this end, in this thesis, we propose an unsupervised re-ranking method that only requires performing inference over large language

models without any finetuning.

Although our approach of using large language models as a re-ranker has been effective (Sachan et al., 2022), we found it to suffer from two main limitations. First, performing inference using large language models can be expensive. Second, the maximum accuracy of the re-ranker is dependent on the first-stage retriever. An ideal solution to these limitations should also be unsupervised in spirit, *i.e.*, it should not require either positive question-document pairs or hard-negative examples. This leads to our next contribution, in which we present a zero-shot approach to distill the *retrieval knowledge* contained in large language models into dense retrievers.

1.2.2 Open-Domain Question Answering

The task of open-domain question answering requires reading retrieved documents to output the correct answer. Reader models based on PLMs can be broadly categorized as either extractive or generative. *Extractive readers* such as the ones based on BERT identify a text span from one of the documents to output an answer (Devlin et al., 2019). They are trained to predict the highest scoring start and end answer tokens in a document (Alberti et al., 2019).

On the other hand, *generative readers* generate answer tokens autoregressively conditioned on the question and retrieved documents (Raffel et al., 2020). These models are typically based on PLMs consisting of encoder-decoder architecture, in which the decoder attends to the question and document representations from the encoder to produce an answer (Lewis et al., 2020c). Jointly attending to independently computed document representations has been shown to benefit answer generation (Izacard and Grave, 2021b). In combination with better retrieval, generative retrieval-augmented models have led to an overall improvement in producing correct answers and thus have been the method of choice for the question answering task.

Although the above methods have been successful, these models need to be trained in two separate stages via a pipelined approach making the training and inference process complicated. Training the retriever is prone to the challenges of obtaining positive documents and hard-negative examples. Furthermore, as a result of the pipelined nature of the system, erroneous signals prop-

agate from the retriever to the reader without the possibility of self-error correction. To address these limitations, we propose a novel training paradigm for retriever-augmented models that facilitates the end-to-end training of both the reader and retriever networks, requiring no intermediate document annotations.

1.2.3 Research Questions

Based on the above discussion, we outline the specific research questions that we investigate in this thesis:

- What is the optimal strategy to initialize the parameters of dense retrievers so that finetuning using supervised datasets results in a better performance? (Chapter 3)
- How to jointly train a system consisting of reader and retriever networks for the task of open-domain question answering? (Chapter 4)
- How can we improve passage rankings obtained from a first-stage retriever without using any training data or finetuning any model? (Chapter 5)
- How can we remove the dependency on aligned question-passage data to train dense retrievers? (Chapter 6)

1.3 Preview of Contributions and Results

At its core, this thesis contains four contributions in which we seek to further probe the above-mentioned research questions. These contributions include several approaches: a pre-train then finetune pipeline to train dense retrievers, an end-to-end training framework for retrieval-augmented models, an unsupervised approach to improve document rankings, and a method to train dense retrievers using unpaired question-document pairs. In the following paragraphs, we provide a preview of our research contributions alongside key results.

In Chapter 3, we focus on improving the training process of dense retrievers. We propose a unified approach of *unsupervised pre-training of retriever parameters followed by supervised finetuning*. For unsupervised pre-training, we explore two tasks that are based on using large-scale textual datasets such as Wikipedia. In the first task, we consider a sentence as the query and train the retriever to predict the neighboring context paragraph. In the second task, we pre-train the retriever by predicting the masked salient entities in a sentence using a language model conditioned on retrieved documents. Such pre-training results in drastic improvements in zero-shot recall when evaluated on popular question answering datasets. When the retriever is adapted to retrieval tasks by supervised finetuning using question-document pairs, we obtain state-of-the-art results in retrieval accuracy.

In Chapter 4, we present an *end-to-end differentiable training* method for retrieval-augmented question answering systems that combine information from multiple retrieved documents when generating answers. We model retrieval decisions as latent variables over sets of relevant documents. Since marginalizing over sets of retrieved documents is computationally hard, we approximate this using an expectation-maximization algorithm. We iteratively estimate the value of our latent variable (the set of relevant documents for a given question) and then use this estimate to update the retriever and reader parameters. We hypothesize that such end-to-end training allows training signals to flow to the reader and then to the retriever better than performing stage-wise training. This results in a retriever that is able to select more relevant documents for a question and a reader that is trained on more accurate documents to generate an answer. We evaluate our proposed training method on three benchmark datasets for open-domain question answering and show that it leads to new state-of-the-art results.

In Chapter 5, we propose an *unsupervised re-ranking* method for improving passage retrieval in open question answering. The re-ranker re-scores retrieved passages with a zero-shot question generation model, which uses a pre-trained language model to compute the probability of the input question conditioned on a retrieved passage. This approach can be applied on top of any retrieval method (*e.g.*, neural or keyword-based), does not require any domain- or task-specific training,

and provides rich cross-attention between query and passage (*i.e.*, it must explain every token in the question). When evaluated on a number of retrieval datasets, our re-ranker improves both unsupervised and supervised retrievers.

In Chapter 6, we introduce a new *corpus-level autoencoding approach* called ART for training dense retrieval models that does not require any labeled training data. Dense retrieval is a central challenge for open-domain tasks, where state-of-the-art methods typically require large supervised datasets with custom hard-negative mining and denoising of positive examples. ART, in contrast, only requires access to unpaired inputs and outputs (*e.g.*, questions and potential answer passages). It uses a new passage-retrieval autoencoding scheme, where (1) an input question is used to retrieve a set of evidence passages, and (2) the passages are then used to compute the probability of reconstructing the original question. Training for retrieval based on question reconstruction enables effective unsupervised learning of both the passage and question encoders. In our extensive experiments, we demonstrate that ART obtains state-of-the-art results on multiple question answering retrieval benchmarks with only generic initialization from a pre-trained language model.

In order to better understand the above contributions, we also present technical background in Chapter 2. Finally, in Chapter 7, we summarize our key contributions and results including the impact of our work within the field. We also highlight limitations arising as a result of specific choices that were made in these contributions. We conclude by proposing several future research directions on how to make retrieval systems efficient, widely applicable, and integrate additional capabilities.

Chapter 2

Background

In this chapter, we present a technical background that is tailored according to the contributions presented in this thesis. First, we cover language models (§2.1), followed by a description of recurrent neural networks (§2.2) and transformer architectures (§2.3), and pre-trained language models (§2.4). We then introduce traditional retrievers and common evaluation metrics used in information retrieval (§2.5).

2.1 Language Models

2.1.1 N-gram Language Model

The task of autoregressive or forward language modeling consists of predicting the next word conditioned on the previous sequence of words. In *n-gram* language modeling, $n-1$ previous words are used to predict the next word (Shannon, 1948). N-gram language models are probabilistic models where a large text corpus is used to estimate the prediction values by applying the principle of maximum likelihood estimation (Manning and Schütze, 1999). This translates to estimating the empirical probability of the sequence with n words which requires calculating its *frequency* in the corpus. These counts are then normalized to ensure a valid probability distribution.

N-gram language models were successfully applied to tasks like speech recognition (Jelinek

et al., 1990), grammatical error correction (Bergsma et al., 2009), and grouping data into word clusters (Brown et al., 1992). These word clusters were very useful in improving the performance on end tasks such as named entity tagging (Miller et al., 2004). Reliable estimations can be obtained by using a higher order n-gram conditioning, but it exponentially increases the number of parameters to be estimated, an issue that is further compounded by data sparsity. Techniques such as linear interpolation and data smoothing have been proposed to minimize the impact of data sparsity (Church and Gale, 1991; Kneser and Ney, 1995).

2.1.2 Neural Network Language Model

Neural network language models (NNLMs) predict the next word based on a parametric probability distribution function, which typically consists of a word embedding layer, multiple hidden layers with non-linear activation, and a word prediction layer (Bengio et al., 2000). The word embedding layer represents each word of the vocabulary using a learnable vector, which is also known as "*distributed representation of words*". The whole network is trained using backpropagation (Rumelhart et al., 1988) by minimizing the next word prediction loss and teacher-forcing (Williams and Zipser, 1989). To condition the prediction of the next word on the previous words, concatenating the previous word embeddings was found to be an effective approach. Representing words using word embeddings also helps to alleviate the data sparsity issue of the n-gram language models.

NNLMs were also proposed to learn a good embedding representation of words. Collobert and Weston (2008) pre-train the parameters of a neural network model by predicting if the middle word in a text segment is related to the context words or not. This training revealed that in the embedding space, semantically similar words lie closer to each other. Further finetuning the model in a multi-task training setup by sharing the embedding parameters led to better performance on downstream tasks. Later, it was shown by Mikolov et al. (2013) that high-quality embedding representations can also be learned by training simple log-linear models (Ratnaparkhi, 1996) to predict the surrounding (context) words conditioned on a (current) word.

Building over word embeddings, powerful neural network architectures for processing sequen-

tial data such as LSTM (Hochreiter and Schmidhuber, 1997; Greff et al., 2017) and self-attentional transformer models (Vaswani et al., 2017) have been proposed. These networks have led to a lot of progress both in language modeling (Melis et al., 2018; Dai et al., 2019) as well as end tasks in NLP (Wu et al., 2016; Peters et al., 2018; Shaw et al., 2018). In the following sections, we present details of LSTM and transformer networks.

2.2 Recurrent Neural Network

Recurrent neural networks (Werbos, 1988) such as LSTM (Hochreiter and Schmidhuber, 1997) have been widely used because they can model the long-range dependencies present in the language structure with their memory cells and explicit gating mechanism.

The dynamics of an LSTM cell are controlled by an input vector ($\boldsymbol{x}_t$), a forget gate ($\boldsymbol{f}_t$), an input gate ($\boldsymbol{i}_t$), an output gate ($\boldsymbol{o}_t$), a cell state ($\boldsymbol{c}_t$), and a hidden state ($\boldsymbol{h}_t$), which are computed as:

$$
\begin{aligned}
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_i * [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + b_i) \\
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_f * [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + b_f) \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_o * [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + b_o) \\
\boldsymbol{g}_t &= \tanh(\boldsymbol{W}_g * [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + b_g) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t),
\end{aligned}
$$

where $\boldsymbol{c}_{t-1}$ and $\boldsymbol{h}_{t-1}$ are the cell state and hidden state, respectively, from the previous time step, σ is the sigmoid function ($\frac{1}{1+e^{-x}}$), $\tanh$ is the hyperbolic tangent function ($\frac{e^x - e^{-x}}{e^x + e^{-x}}$), $\odot$ denotes element-wise multiplication, and $\boldsymbol{W}_*$ are trainable weight parameters. The hidden state $\boldsymbol{h}_t$ is passed as input to the subsequent layers for further processing. The recurrence is inherent in the LSTM as the computation for the next timestep is dependent on the cell state and hidden state of the previous timestep. The parameters of the LSTM are shared for all the time steps.

Another popular variant of the LSTM is the bidirectional LSTM network where the sequence encoder consists of a forward LSTM and a backward LSTM (Schuster and Paliwal, 1997). The input to the backward LSTM cell is the reversed order of words in the sequence. When applied to end tasks, the input to the LSTM typically consists of the word embeddings and character-level features. These LSTM networks have been successfully applied to a wide range of NLP tasks such as text classification (Dai and Le, 2015; Johnson and Zhang, 2016), information extraction (Ma and Hovy, 2016; Zhang et al., 2017), and machine translation (Luong et al., 2015; Johnson et al., 2017).

2.3 Transformer Network

The transformer network was originally proposed for the task of machine translation (Vaswani et al., 2017). It avoided having recurrent layers for sequence processing in favor of attention modules that can process the sequence in parallel. Since their inception, transformer models have enjoyed wide popularity in a range of NLP tasks and have acted as a catalyst to the current large language model revolution.

The transformer model consists of an embedding layer, multiple encoder-decoder layers, and an output layer to generate tokens. Each encoder layer consists of two sublayers: self-attention and feed-forward networks. Each decoder layer consists of three sublayers: masked self-attention, encoder-decoder attention, and feed-forward networks. These layers primarily consist of different trainable weights to compute affine transformations of their inputs. Given input sequences to the encoder and decoder, first, an embedding layer obtains their word vectors. To capture the relative ordering of words in the input sequence, fixed position encodings are added to the word embeddings.

The first sublayer of the encoder performs multi-head self-attention. For the single-head case, let the input to the sublayer be $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$ and the output be $\boldsymbol{z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_T)$, where $\boldsymbol{x}_i, \boldsymbol{z}_i \in \mathbb{R}^d$. The input is linearly transformed to obtain key ($\boldsymbol{k}_i$), value ($\boldsymbol{v}_i$), and query ($\boldsymbol{q}_i$) vectors

as

$$\boldsymbol{k}_i = \boldsymbol{x}_i \boldsymbol{W}_K, \boldsymbol{v}_i = \boldsymbol{x}_i \boldsymbol{W}_V, \boldsymbol{q}_i = \boldsymbol{x}_i \boldsymbol{W}_Q.$$

In the next step, similarity scores are computed by performing a scaled dot-product between the query and key vectors

$$e_{ij} = \frac{1}{\sqrt{d}} \boldsymbol{q}_i \boldsymbol{k}_j^T.$$

Attention coefficients are then computed by applying softmax over these similarity values

$$\alpha_{ij} = \frac{\exp e_{ij}}{\sum_{l=1}^{T} \exp e_{il}}.$$

Self-attention output ($\boldsymbol{z}_i$) is computed by the weighted combination of attention weights with value vectors as

$$\boldsymbol{z}_i = (\sum_{j=1}^{T} \alpha_{ij} \boldsymbol{v}_j) \boldsymbol{W}_F.$$

In the above equations, $\boldsymbol{W}_K, \boldsymbol{W}_V, \boldsymbol{W}_Q, \boldsymbol{W}_F \in \mathbb{R}^{d \times d}$ are trainable parameters. Single-head attention can be extended to multi-head attention by chunking the key, value, and query vectors, and performing the attention computation in parallel for each of them followed by concatenating them. The second sublayer consists of a two-layer feed-forward network (FFN) with ReLU activation (Glorot et al., 2011).

$$\mathrm{FFN}(\boldsymbol{z}_i) = \max(0,\ \boldsymbol{z}_i \boldsymbol{W}_{\boldsymbol{L_1}} + b_1) \boldsymbol{W}_{\boldsymbol{L_2}} + b_2,$$

where $\boldsymbol{W}_{\boldsymbol{L_1}} \in \mathbb{R}^{d \times d_h}$, $\boldsymbol{W}_{\boldsymbol{L_2}} \in \mathbb{R}^{d_h \times d}$, b_1 and b_2 are biases, and d_h is the hidden size. The FFN sublayer outputs are subsequently given as input to the next layer.

The transformer decoder layer consists of three sublayers. The first sublayer, similar to the

encoder, performs self-attention where masks are used to prevent positions from attending to future positions. The second sublayer performs encoder-decoder attention where the query vector is from the decoder layer while the key and value vectors are from the encoder's last layer. The third sublayer consists of a feed-forward network. To generate predictions for the next word, there is a linear layer on top of the decoder layer. Residual connections (He et al., 2016) and layer normalization (Ba et al., 2016) are applied on each sublayer for better training.

2.4 Pre-trained Language Models

From 2018 onwards, pre-trained language models (PLMs) have brought a revolution in the NLP field as they have led to consistent performance improvements on a number of end tasks (Howard and Ruder, 2018; Peters et al., 2018; Devlin et al., 2019; Brown et al., 2020). The pre-training process essentially involves training a transformer network using massive amounts of text data by optimizing self-supervised loss functions (Devlin et al., 2019; Lewis et al., 2020b; Radford et al., 2018). Axis of variations among PLMs include the type of self-supervised objective used for training and the model itself — either the transformer encoder, decoder, or both. After pre-training, these weights are directly finetuned using training examples for a particular task (such as entailment, reading comprehension, etc.) or are used as building blocks for more complex modules (such as dual-encoders).

In the following sections, we provide more details covering different aspects of PLMs tailored according to the contributions presented in this thesis.

2.4.1 Bidirectional Transformer Encoder Pre-training

BERT (Devlin et al., 2019) proposes to pre-train the parameters of the transformer encoder following the setup of the Cloze task (Taylor, 1953). The guiding principle behind BERT training is that it should be able to leverage the bidirectional context information that is essential to perform reasoning in tasks such as question answering. In addition, as many tasks also require reasoning

over two text segments such as textual entailment and coreference resolution, it is important to model the coherence relationship between pairs of text. Keeping this into consideration, BERT's training objective consists of masked language modeling and the next sentence prediction tasks.

In the masked language modeling task, a small fraction of tokens in the input text sequence is replaced by special "masked tokens". Then, the model is trained to predict the original tokens by conditioning on the remaining tokens in the sequence. This conditioning enables the model to use bidirectional context around the masked tokens during the pre-training process resulting in a powerful set of pre-trained weights. As the masked tokens are not present during the finetuning step, there is a chance of input distribution mismatch during the pre-training and finetuning steps. To reduce this mismatch, a small fraction of the masked tokens are replaced with random tokens sampled from the vocabulary, and some of the predicted tokens are also not masked.

In the next sentence prediction task, the model is fed with two sentences as the input and is required to predict if the second sentence is a continuation of the first sentence or not. The training data for this task is created in a balanced proportion with two potential target labels, *i.e.*, that of the next sentence being a continuation of the first sentence or being a random sentence from the training corpus. Binary cross-entropy loss is applied in order to train the model for this task.

2.4.2 Variants of BERT

Transformer encoder when finetuned after pre-training according to the BERT objective has shown impressive results across a variety of language understanding tasks (Wang et al., 2019a) and thus has been widely adopted. However, a potential drawback of BERT is that it under-utilizes its training data as the masked language modeling objective masks just 15% of the tokens in the input sequence and thus seems inefficient. Therefore, to improve upon the data efficiency of BERT, Yang et al. (2019) proposed XLNet that trains the model on all valid permutations arising from the factorization order of the input sequence tokens. Being an autoregressive model, XLNet also avoided the pre-training finetuning discrepancy of BERT.

Clark et al. (2020b) demonstrated that pre-training of the transformer encoder can be made

more data-efficient by first corrupting a sequence of tokens and then training the model to discriminate if a token in the corrupted sequence is original or corrupted. This objective was found to outperform original BERT training, especially at smaller model sizes. Furthermore, in their study, Liu et al. (2019) remarked that the next sentence prediction task may not be critical to achieving good performance as it can have adverse effects on the learning process. Instead, other factors such as scaling up the training process to use larger batch sizes and training data are more important. Finally, it was shown by Lan et al. (2020) that parameter sharing among transformer layers during pre-training tends to have a regularization effect leading to gains in performance.

2.4.3 Generative Pre-training of Transformers

Another approach is to train the transformer decoder using the forward language modeling objective, which is also known as generative pre-training (GPT). When training a large multi-billion parameter decoder using textual data consisting of billions of tokens, the GPT series of models (Brown et al., 2020; Chowdhery et al., 2022) have shown strong zero-shot and few-shot performance on NLP tasks. An appealing property of these GPT models is that they can be adapted to any task by providing task-specific instructions along with a few examples avoiding the need to perform finetuning of parameters.

Encoder-decoder pre-training: These approaches include text-to-text transformers (T5; Raffel et al., 2020) which pre-trains both the transformer encoder and decoder. In T5, spans of tokens are first replaced by sentinel tokens and are given as input to the encoder. The decoder is trained to autoregressively generate the original text spans by attending to the encoded sequence. Models trained at scale have been shown to perform quite well on generative tasks such as summarization and closed-book question answering (Roberts et al., 2020).

Instruction-tuned language models: This involves finetuning a pre-trained (generative) language model using training examples of unrelated tasks defined by natural language instructions and then evaluating the performance on a held-out task. For example, a model can be trained us-

ing examples from machine translation, and commonsense reasoning tasks, and then evaluated on entailment tasks. It is expected that such instruction-based finetuning would lead to a better transfer and thus improved zero-shot generalization than multi-task training. This approach has been shown to be effective for both encoder-decoder (Sanh et al., 2022) and decoder-only models (Wei et al., 2022).

2.5 Traditional Retrievers and Evaluation Metrics

Text retrieval is an essential component in applications such as information-seeking conversational agents and open-domain question answering systems. These systems primarily consist of a retriever module that intakes a question, compares it with millions of evidence passages (or documents), and returns relevant passages (Frakes and Baeza-Yates, 1992; Singhal, 2001; Manning et al., 2008). Traditional retrievers based on sparse (or non-trainable) bag-of-words representation of documents have been the workhorses of information retrieval systems and have remained a popular choice ever since. In the following paragraphs, we provide examples of two such retrievers.

2.5.1 Tf-idf

Tf-idf represents a document in a (sparse) vector space where each dimension denotes a keyword (or term). The ordering information of words in the document is ignored and hence this is also known as the *bag-of-words* representation. For each term in the document, its value at the vector dimension is the frequency of the respective term times the inverse document frequency (Sparck Jones, 1988). Factorizing the tf-idf representation of documents has been shown to produce (dense) semantic embeddings in a smaller-dimensional vector space (Deerwester et al., 1990).

2.5.2 BM25

BM25 is a widely used retrieval approach defined based on the bag-of-words representation of the terms present in the query and the evidence documents (Robertson and Zaragoza, 2009). A popular formulation of the ranking algorithm is to add the score of each unique term in the question, where the score is proportional to the product of the inverse document frequency of the term and a factor based on the term's frequency in the document.

More formally, given a question ($\boldsymbol{q} = \{q_1, \ldots, q_n\}$) consisting of n terms and a document ($\boldsymbol{d}$), the BM25 score is computed as:

$$\text{score}(\boldsymbol{q}, \boldsymbol{d}) = \sum_{i=1}^{n} \log \frac{N}{\mathit{df}(q_i)} \cdot \frac{f(q_i, \boldsymbol{d}) \cdot (k_1 + 1)}{f(q_i, \boldsymbol{d}) + k_1 \cdot (1 - b + b \cdot \frac{|\boldsymbol{d}|}{\mathit{avgdl}})},$$

where N is the text collection size, $\mathit{df}(q_i)$ is the number of documents in which the query term q_i occurs, $f(q_i, \boldsymbol{d})$ is the frequency of the term q_i in the document $\boldsymbol{d}$, $|\boldsymbol{d}|$ denotes the number of terms in the document $\boldsymbol{d}$, '*avgdl*' denotes the average document length in the text collection, k_1 and b are hyperparameters to control for the term frequency scaling and document length normalization, respectively.

Due to its reliance on the sparse representations of text, BM25 succeeds in cases involving lexical matching, *i.e.*, when the question and document terms overlap while it fails at semantic matching. Despite this limitation, the BM25 retriever is still a competitive baseline and has been used in the contributions included in this thesis.

2.5.3 Evaluation Metrics

We cover three evaluation metrics that are common in the information retrieval literature. These metrics have been used to measure the performance of retrieval systems presented in this thesis.

Top-K accuracy: It is defined as the fraction of queries for which one or more of the documents within the top-ranked K documents contain an answer to the query or is a relevant document. This

metric is sometimes also referred to as *Recall@*K.

Normalized discounted cumulative gain (nDCG): In contrast to the top-K accuracy metric, the nDCG metric considers the positions of the relevant documents in the retrieved list of documents (Järvelin and Kekäläinen, 2002; Wang et al., 2013). It penalizes a system if the relevant documents are ranked lower and vice versa. nDCG is based in turn on the discounted cumulative gain (DCG) measure which is defined as:

$$\text{DCG} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)}, \tag{2.1}$$

where i denotes the position of a document in the retrieved list consisting of p documents, rel_i denotes the graded relevance value of the document at position i in the retrieved list. nDCG is computed by normalizing the DCG score of the retrieved documents with the DCG score computed by assuming an ideal ranking of documents, *i.e.*, when the relevant documents are placed at the top of the document list.

Exact match (EM): This metric is used to measure the performance of question answering systems. A credit of one is assigned to the system if the model outputs an answer for a query that exactly matches one of the reference answers. Then, the exact match score is computed as the fraction of queries for which the system generated the correct answer.

Chapter 3

Pre-training Dense Embedders to Improve Passage Retrieval

The task of open-domain question answering (OpenQA) consists of finding *answers* to the information-seeking *questions* using a large knowledge source such as Wikipedia. This knowledge source is also referred to as *evidence* and it typically contains millions of documents. Most approaches for OpenQA consist of a two-stage pipeline (Chen, 2018). In the first stage, given a question, a *retriever* module identifies the most relevant documents (or passages), which is often a very small subset of the evidence. Traditionally, bag-of-words approaches based on sparse text representations such as BM25 (Robertson and Zaragoza, 2009) have been used as the retriever. In the second stage, these relevant documents are given as input to the *reader* module, which understands them and extracts the answer for the question (Figure 3.1).

Although successful, the main drawback of the BM25 method is that it is not trainable and hence it cannot be adapted to open-domain document retrieval tasks that require semantic matching. Recent work has addressed this limitation by building upon advances in self-supervised learning (Gillick et al., 2018). These approaches model the retriever using masked language models such as BERT (Devlin et al., 2019), allowing the retriever to be trained using task-specific datasets (Lee et al., 2019; Karpukhin et al., 2020). Typically, the retriever model consists of a *dual-*

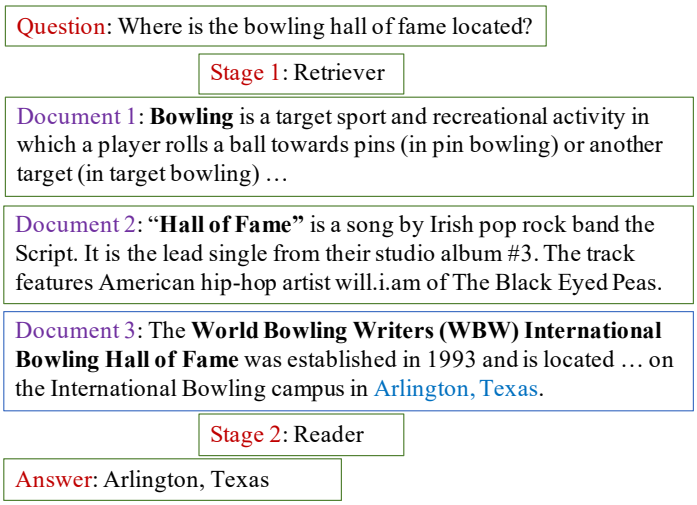

Figure 3.1: An example illustrating the two-stage OpenQA pipeline.

encoder architecture (Bromley et al., 1994), where each encoder computes dense embeddings of the question and evidence documents, respectively.

Prior work has investigated both unsupervised and supervised approaches to train the retriever. Unsupervised approaches include separately training the retriever with inverse cloze task (ICT; Lee et al., 2019) or jointly training the retriever and reader by predicting masked text spans using retrieval augmentations (Guu et al., 2020). Supervised approaches such as dense passage retrieval (DPR; Karpukhin et al., 2020) train the retriever using human-annotated sets of question and document pairs.

However, there is no study that investigates the comparative advantages of using these two styles of training when the retrieval task is challenging, *i.e.*, when the evidence contains millions of documents. It is unclear if the unsupervised approaches can further help to improve the performance of *strong* supervised approaches, and, if so, under what conditions. A core focus of this work is systematically studying these aspects of retriever training.

We propose a unified approach to train the retriever: unsupervised pre-training followed by supervised finetuning. We also investigate key design choices—such as retriever score scaling and longer training—and showcase their effectiveness. Our results demonstrate that the proposed approach obtains substantial accuracy gains when evaluated on benchmark QA datasets. Extensive

experiments also highlight the relative importance of different pre-training strategies, revealing important trade-offs when varying the amount of supervised data available to train the retriever.

Furthermore, motivated by recent work (Guu et al., 2020), we also explore supervised training of the language model and retriever for answer generation. The language model attends to each retrieved document separately and the retriever is trained based on the usefulness of these documents in answer generation. We show that this approach leads to an improved retrieval performance, which in turn results in an improved answer extraction. We outperform the previous best models to obtain new state-of-the-art results on retrieval accuracy and answer extraction. We also perform experiments by scaling the model size to a large configuration for both retriever and language models and observe consistent improvements, compared with smaller models.

The rest of this chapter is organized as follows. §3.1 presents the background on dense retrievers along with their training methodology and §3.2 details our proposed approach. §3.3-3.6 describe the experimental details with the results. §3.7 reviews the related work followed by discussion in §3.8.

3.1 Background: Dense Retriever

Given a question and a large collection of documents (or passages) in the evidence, the task of the retriever is to select a relevant subset of documents that answers the question.[1] To achieve this, the retriever matches the question with the evidence documents and outputs the top-ranked documents.

It is crucial for the retriever design to be efficient as it needs to search against tens of millions of documents. This flexibility is offered by dense retrievers, which are essentially neural networks, where both the question and document are represented by low-dimensional dense embeddings. These dense embeddings allow the document representations to be pre-computed and then cached in the accelerator's memory leading to faster inference.

Their scalable design combined with the recent advances in pre-trained text representations has motivated researchers to train improved dense retriever models. This has led to considerable

[1] We use the terms documents and passages interchangeably in this chapter.

progress with dense retrievers outperforming popular sparse models such as BM25 on many popular benchmarks.

In this section, we first review the retriever architecture and then discuss methods to train it, followed by our proposed approach.

3.1.1 Dual-Encoder Retriever

Let the question be denoted as $\boldsymbol{q}$ and evidence set by $\mathcal{D} = \{\boldsymbol{d}_1, \cdots, \boldsymbol{d}_M\}$, where M is the number of documents. For the retriever, we use the dual-encoder model (Bromley et al., 1994) which consists of two encoders, where

- one encoder computes the question embedding $f_q(\boldsymbol{q}; \Phi_q) : \mathcal{X} \mapsto \mathbb{R}^d$, and
- the other encoder computes the document embedding $f_d(\boldsymbol{d}; \Phi_d) : \mathcal{X} \mapsto \mathbb{R}^d$.

Here, $\mathcal{X} = \mathbb{V}^n$ denotes the universal set of text sequences, $\mathbb{V}$ denotes the vocabulary consisting of discrete tokens, and $\mathbb{R}^d$ denotes the (latent) embedding space. We assume that both the question and document embeddings lie in the same latent space. The *retrieval score* for a question-document pair $(\boldsymbol{q}, \boldsymbol{d})$ is then defined as the inner product between their respective embeddings,

$$s(\boldsymbol{q}, \boldsymbol{d}_i; \Phi) = f_q(\boldsymbol{q}; \Phi_q) \cdot f_d(\boldsymbol{d}_i; \Phi_d), \ \forall \boldsymbol{d}_i \in \mathcal{D}, \tag{3.1}$$

where $\Phi = [\Phi_q, \Phi_d]$ denotes the retriever parameters.

We use the transformer network (Vaswani et al., 2017) with BERT tokenization (Devlin et al., 2019) to model both encoders. To obtain the question or document embedding, we do a forward pass through the transformer and select the last layer hidden state corresponding to the `[CLS]` token. As the input document representation, we use both the document title and text separated by `[SEP]` token.

3.1.2 Training Process

Each training example consists of a tuple of question and its corresponding positive document $(\boldsymbol{q}_i, y_i)$, where $y_i \in [1, M]$ represents the document indexed in $\mathcal{D}$. Let the training set be denoted as $\mathcal{T} = \{(\boldsymbol{q}_1, y_1), \cdots, (\boldsymbol{q}_n, y_n)\}$, where n is the number of training examples. To train the retriever, we minimize the softmax cross-entropy loss of the positive question-document pairs

$$\mathcal{L}(\mathcal{T}, \mathcal{D}; \Phi) = -\frac{1}{|\mathcal{T}|} \sum_{i=1}^{n} \log \frac{\exp(s(\boldsymbol{q}_i, \boldsymbol{z}_{y_i}; \Phi)/\tau)}{\sum_{j=1}^{M} \exp(s(\boldsymbol{q}_i, \boldsymbol{z}_j; \Phi)/\tau)}, \tag{3.2}$$

where τ is a scaling parameter.[2] The above loss can also be understood from the viewpoint of *contrastive training* as it encourages the dual-encoder to learn improved representations by aligning together the embeddings of the question and positive document while pushing further apart the embeddings of unrelated documents.

Computing the partition function in the above expression requires a forward pass for all the evidence documents using the current retriever parameters. As the evidence consists of millions of documents, computing the partition function in each training step and then backpropagation would be prohibitively expensive. A more tractable option is to sample a small subset of documents ($\mathcal{S}$) from the evidence, *i.e.*, $\mathcal{S} \subset \mathcal{D}$, such as using uniform sampling (Jean et al., 2015). This modified loss can be expressed as

$$\mathcal{L}(\mathcal{T}, \mathcal{S}; \Phi) = -\frac{1}{|\mathcal{T}|} \sum_{i=1}^{n} \log \frac{\exp(s(\boldsymbol{q}_i, \boldsymbol{z}_{y_i}; \Phi)/\tau)}{\sum_{j=1}^{|\mathcal{S}|} \exp(s(\boldsymbol{q}_i, \boldsymbol{z}_j; \Phi)/\tau)}. \tag{3.3}$$

However, uniform samples often lead to poor estimates of the partition function and thus are ineffective in practice. Another preferred choice to approximate the partition function is to use positive documents of all other training examples in the batch as negative documents. This method is often referred to as *in-batch sampled softmax loss* and has shown to perform well in practice (Henderson et al., 2017; Gillick et al., 2018; Chen et al., 2020). It is also efficient as the document

[2] τ is also referred to as the temperature parameter.

embeddings only need to be computed once per batch.

3.2 Proposed Approach: Unsupervised Pre-training and Supervised Finetuning

Most of the previous retriever training methods initialized dual-encoder parameters using masked language model weights such as those of BERT. Although successful, it is not apparent if this is a good strategy as BERT suffers from poor discrimination ability when the evidence contains millions of documents. This is reflected in BERT's low recall scores on QA retrieval tasks (§3.5.1).

In light of these findings, we propose to *pre-train retriever* such that its zero-shot retrieval quality improves significantly as compared to BERT. We argue that finetuning a pre-trained retriever on QA datasets will lead to better training dynamics that will in turn improve retrieval accuracy, especially in low-resource settings. In order to pre-train the retriever, we propose two unsupervised tasks (§3.2.1). We then adapt the pre-trained retriever for each dataset by finetuning using dataset-specific training examples (§3.2.2).

3.2.1 Unsupervised Pre-training

For unsupervised training, we only use the text of evidence documents such as that of English Wikipedia. We propose two variants of existing approaches for unsupervised pre-training of the retriever that are described next.

Inverse Cloze Task (ICT)

In this task, we try to mimic the human-annotated question-document pairs using evidence documents. Specifically, we first extract paragraphs from the Wikipedia corpus. A randomly sampled sentence from a paragraph is considered as a *pseudo-question* while all remaining sentences in the paragraph are considered as its *context document*. The objective is to predict the context conditioned on the selected sentence. Training retrievers with this objective improves both the encoders

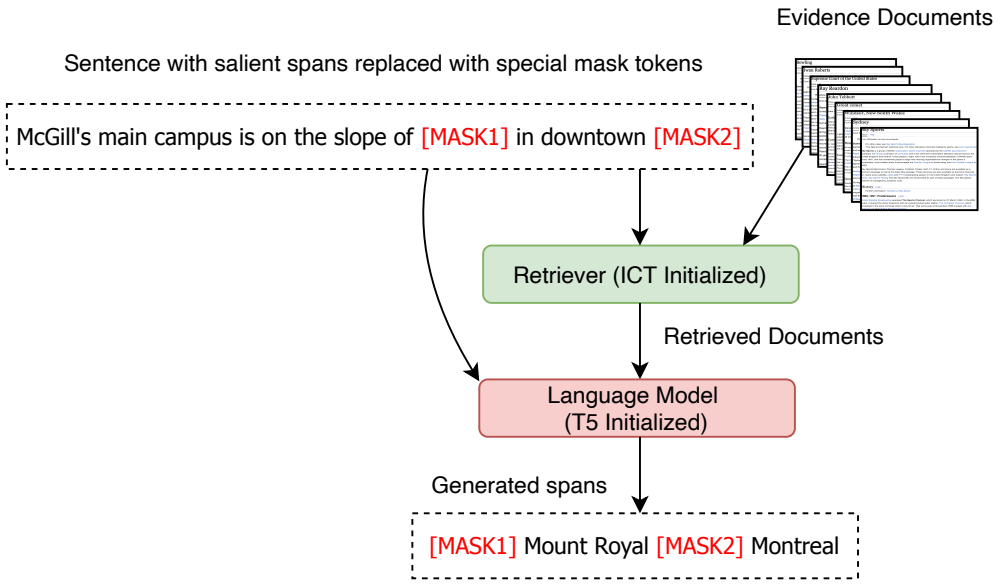

Figure 3.2: An example illustrating the masked salient spans (MSS) pre-training task.

as they need to learn both semantic and lexical matching abilities.

This task resembles the popular cloze task where the goal is to predict masked words in text while in ICT the objective is to predict the context from a sentence and hence the name. This approach to pre-train dual-encoders was originally proposed in Lee et al. (2019). We train the network by optimizing in-batch sampled softmax loss (§3.1.2). We train with a large batch size to increase the pool of negative examples which results in an improved performance.

Retriever score scaling: We initialize the parameters of both the question and document encoders using BERT weights as implemented in Megatron-LM (Shoeybi et al., 2019).[3] In our experiments, we set the scaling parameter as $\tau = \sqrt{d}$, where d is the model's hidden size. This is in contrast to previous work that used the setting of $\tau = 1$ (Karpukhin et al., 2020). A larger scaling factor helps in better optimization when the model size is large (§3.5.3).

Masked Salient Spans (MSS)

In this task, we seek to further improve the zero-shot performance of the ICT-initialized retriever by jointly training it with a language model. Masked salient spans (MSS) training is a variant of

[3]We also experimented with random initialization but it vastly underperformed BERT initialization.

the masked language modeling task where the language model predicts masked text spans with the help of retrieved documents (Guu et al., 2020).

To construct the training data for the MSS task, we segment sentences from English Wikipedia containing one or more named entities. We replace one or more named entities in the sentence with masked tokens and the task is to predict these masked entities, which is also depicted in Figure 3.2 for reference. Masked sentences can be considered as pseudo-questions while entity spans can be considered as the answers, resembling a typical open-domain QA setup. We next detail the MSS training method.

The trainable components consist of the retriever (Φ) and generative language model (Θ). We initialize the retriever with ICT weights and the language model with pre-trained T5 weights. The inputs to train the model are *masked sentences* ($\boldsymbol{x}$) and original *entity spans* ($\boldsymbol{e}$). Given a masked sentence, first the retriever selects top-K documents ($\mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K\}$) with maximum inner product scores according to Eq. 3.1.[4]

Each retrieved document ($\boldsymbol{z}_i$) along with the masked sentence is given as input to the T5 language model which autoregressively generates the masked entity spans, whose likelihood is defined as

$$p(\boldsymbol{e} \mid \boldsymbol{x}, \boldsymbol{z}_i; \Theta) = \prod_{t=1}^{T} p\left(\boldsymbol{e}_t \mid e_{<t}, \boldsymbol{x}, \boldsymbol{z}_i; \Theta\right), \tag{3.4}$$

where T denotes the number of tokens in entity spans. We jointly train the retriever and language model by maximizing the marginal log-likelihood as

$$\mathcal{L}(\boldsymbol{e} \mid \boldsymbol{x}; \Theta, \Phi) = \log \sum_{\boldsymbol{z}_i \in \mathcal{Z}} p(\boldsymbol{e} \mid \boldsymbol{x}, \boldsymbol{z}_i; \Theta) p(\boldsymbol{z}_i \mid \boldsymbol{x}, \mathcal{Z}; \Phi), \tag{3.5}$$

where $p(\boldsymbol{z}_i \mid \boldsymbol{x}, \mathcal{Z}; \Phi)$ is computed by applying softmax to Eq. 3.1. During training, we update all the parameters of the retriever (both the question and document encoders) and the language model. This joint training formulation rewards those retrievals that improve the likelihood of correct salient span prediction while penalizing incorrect retrievals. Such a dynamic feedback loop

[4] As the selection operation requires performing inner-product with millions of document embeddings, this can be efficiently performed on accelerators such as GPUs or TPUs.

from the language model forces retrieval quality to improve at every training step. Overall, the MSS training process leads to better pre-trained retriever weights.

3.2.2 Supervised Finetuning

In the supervised setting, *human-annotated* questions, short answers, and sometimes positive documents are provided. If the positive document is not included, then a common approach is to use distant supervision (Mintz et al., 2009) to obtain the positive document. Specifically, we select the top-ranked document using BM25 (Robertson and Zaragoza, 2009) from the evidence that contains the answer as the positive document. We also consider other top-ranked documents that do not contain the answer as *hard negative* examples (Gillick et al., 2019). These hard-negative examples in addition to in-batch negatives lead to a better approximation of the partition function thereby forcing the retriever to learn improved question-document representations (Karpukhin et al., 2020).

Let the training batch be denoted as $\mathcal{B}$. A training example in the batch can then be written as $(\boldsymbol{q}_i, y_i, \tilde{y}_i)$, where y_i and $\tilde{y}_i$ denotes the indices of the positive and hard-negative documents in the evidence, respectively. We finetune the retriever by minimizing the following sampled softmax loss

$$\mathcal{L}(\mathcal{B}, \mathcal{D}; \Phi) = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \log \frac{\exp(s(\boldsymbol{q}_i, \boldsymbol{z}_{y_i}; \Phi)/\tau)}{\sum_{j \in |\mathcal{B}|} \exp(s(\boldsymbol{q}_i, \boldsymbol{z}_{y_j}; \Phi)/\tau) + \exp(s(\boldsymbol{q}_i, \boldsymbol{z}_{\tilde{y}_j}; \Phi)/\tau)}, \tag{3.6}$$

where for each question $\boldsymbol{q}_i \in \mathcal{B}$, the partition function is computed using all the positive and hard-negative documents contained in the batch, *i.e.*, $\bigcup_{i \in \mathcal{B}} (\boldsymbol{z}_{y_i}, \boldsymbol{z}_{\tilde{y}_i})$.

End-to-End Supervised Training

We also explore *end-to-end finetuning* of the language model (also referred to as reader in the QA literature) and retriever. In this setup, the training data consists of question-answer $(\boldsymbol{q}, \boldsymbol{a})$ pairs. Similar to the MSS pre-training objective, the language model is tasked to predict the answer

conditioned on retrieved documents ($\mathcal{Z}$). Specifically, we maximize the following marginal log-likelihood of answer generation

$$\mathcal{L}(\boldsymbol{a} \mid \boldsymbol{q}; \Theta, \Phi) = \log \sum_{\boldsymbol{z}_i \in \mathcal{Z}} p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_i; \Theta) p(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{Z}; \Phi), \tag{3.7}$$

where the likelihood of answer generation conditioned on each retrieved document $p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_i; \Theta)$ is computed using a language model. This joint training setup iteratively improves the retriever quality using feedback from the language model and also adapts the language model to read the retrieved documents for answering the question. As this approach involves conditioning the language model on each retrieved document separately, we term it as *Individual Top-K*. During inference, we generate the answers using greedy search.

We note that the RAG model (Lewis et al., 2020c) also proposed a similar approach, but there are two main differences. The first is that while we update all the parameters of the retriever (both the question and document encoders), RAG just updates the question encoder. The second is that we use the pre-trained T5 model as the language model while RAG uses BART (Lewis et al., 2020b). These enhancements help us obtain substantial gains over the RAG model, which we will discuss in §3.6.

3.3 Experimental Setup

In this section, we describe the datasets and model settings.

3.3.1 Datasets

We perform experiments using two widely used question answering datasets whose details are provided below and their statistics are shown in Table 3.1.

Dataset	Train	Filtered Train	Dev	Test
NQ-Open	79,168	58,880	8,757	3,610
TriviaQA	78,785	60,413	8,837	11,313

Table 3.1: Dataset statistics. The training set is used for end-to-end training experiments (§3.5.2). The filtered train version is used for supervised retriever finetuning. The filtered set ignores those examples where the passages (or documents) retrieved from the evidence using BM25 does not contain the reference answer or align with the ground-truth passage.

Natural Questions (NQ-Open): This corpus consists of real anonymized questions issued to the Google search engine (Kwiatkowski et al., 2019).[5] Ground truth annotations are provided for these questions using the top-ranked Wikipedia pages. These annotations are in the form of long answers (such as a paragraph that answers the question) and short answers (such as one or more text spans including boolean '*yes*' or '*no*' answers). Human annotators can also mark a question to be unanswerable if the Wikipedia page does not contain sufficient information. The original dataset consists of 307,373 training examples, 7,830 development examples, and 7,842 test examples. In our experiments, following convention (Lee et al., 2019; Karpukhin et al., 2020), we use a subset of the short answer questions such that each question is answerable with the answer containing a maximum of 5 tokens. For the development set, 10% of the training examples are selected while for the test set, the short answers' subset of the original development set is selected.

TriviaQA: This corpus consists of a large collection of diverse trivia questions and their answers scraped from multiple sources in the Web (Joshi et al., 2017).[6] Originally introduced for the task of reading comprehension, the dataset also contains top-ranked documents mined from the Bing search engine. There are two versions of the dataset: (i) *filtered set* containing around 95,000 question-answer pairs and over 650,000 question-answer-document triples, and (ii) an *unfiltered set* of 110,495 question-answers and 740,000 evidence documents. In our work, we only consider the question-answer pairs from the unfiltered set and ignore its evidence documents.

[5] https://ai.google.com/research/NaturalQuestions/download
[6] http://nlp.cs.washington.edu/triviaqa/

Evidence data: As the information source containing world knowledge, we make use of English Wikipedia. Following Karpukhin et al. (2020), we use the Wikipedia database from December 2018 and extract its text-only portion.[7] During pre-processing, we discard semi-structured information such as those from tables, infoboxes, or images. Each article is greedily segmented into 100-word long non-overlapping sequences, resulting in a total of 21,015,324 passages.

3.3.2 Model Details

We use BERT and T5 models of two different configurations, *base* and *large*, for the experiments. The base configuration consists of 12 layers, 768 dimensional hidden size, and 12 attention heads. The BERT-base contains 110M parameters while the T5-base contains 220M parameters. The large configuration consists of 24 layers, 1024 dimensional hidden size, and 16 attention heads. The BERT-large contains 330M parameters while the T5-large contains 770M parameters.

3.4 Training Details

We provide training details for all the experiments below. We extend the open-source Megatron-LM toolkit (Shoeybi et al., 2019) to implement the models.[8] For both the base and large model configurations, we use the same training settings. To train the models, we employed mixed-precision training (Micikevicius et al., 2018) and leveraged distributed training features as implemented in the PyTorch framework (Li et al., 2020). All our experiments were performed on a cluster containing A100 GPUs.

3.4.1 Language Models Training

We train BERT (Devlin et al., 2019; Lan et al., 2020) and T5 (Raffel et al., 2020) language models from scratch closely following the settings from the original papers. The training hyperparameters for both the base and large configurations are detailed in Table 3.2. For training language models,

[7] Wikipedia dump is available at: https://archive.org/download/enwiki-20181220
[8] https://github.com/NVIDIA/Megatron-LM

Hyperparameter	BERT	T5
Dataset	Wikipedia, BookCorpus	Wikipedia, CC-Stories, RealNews, OpenWebText
Hidden Size	{768, 1024}	{768, 1024}
Attention Heads	{12, 16}	{12, 16}
Dropout	0.1	0.1
Attention Dropout	0.1	0.1
Optimizer	Adam	Adam
Training Steps	1M	1M
Warmup Steps	10k	10k
Peak Learning Rate	1e-4	1e-4
Weight Decay	1e-2	1e-2
Batch Size	256	2048
Learning Rate Decay	Linear	Linear
Gradient Clipping	1.0	1.0

Table 3.2: Hyperparameters for pre-training BERT and T5 models.

we used both data and model parallelism (Xu et al., 2021)—32 GPUs to train BERT and 128 GPUs to train T5.

3.4.2 Retriever Training

Inverse cloze task (ICT): We initialize the parameters of both the question and document encoders using BERT weights. We train the retriever using pre-processed English Wikipedia text where each example is a paragraph with a maximum length of 256 tokens. We also keep the query sentence in the context with a probability of 0.1. For training, we use a batch size of 4,096, a peak learning rate of 1e-4 with linear decay, and train for 100,000 steps using Adam optimizer (Kingma and Ba, 2014). This corresponds to training the retriever for roughly 20 epochs over the Wikipedia dataset. We set the weight decay rate to 1e-2 and the warmup ratio of the optimizer to 0.01. For scalable training and inference, up to 128 GPUs are used. In order to compute in-batch sampled softmax loss, we collect the paragraph embeddings computed on different GPUs by applying the all-gather communication collective in PyTorch distributed. However, applying an all-gather operation detaches the computation graph thus leading to incorrect loss gradients. To fix this, we multiply the loss in each GPU by the number of distributed processes used to compute embeddings (also known as world size).

Masked salient spans (MSS): We initialize the retriever with ICT training and pre-train the T5 language model on an aggregated dataset from Shoeybi et al. (2019). We use the pre-trained models provided by the Stanza toolkit (Qi et al., 2020) to segment Wikipedia paragraphs into sentences and extract named entities.[9] The masked sentence is used as a query to retrieve evidence documents with the help of which the T5 generates the original salient spans. We train the model for 100,000 steps with Adam optimizer using a learning rate of 2e-5 and a warmup ratio of 0.05. To prevent pre-computed evidence embeddings from getting stale as the document encoder weights are updated, following Guu et al. (2020), we compute the fresh evidence embeddings asynchronously every 500 steps and then update the evidence index. We implement this by maintaining one process group for model training and a separate process group for evidence embedding computation. Training was performed on up to 240 GPUs.

Supervised finetuning: We use Adam optimizer, a batch size of 128, a learning rate of 2e-5 with a linear decay, and train for 80 epochs. The training was performed on 16 GPUs.

End-to-end supervised training: We initialize the retriever with finetuned ICT weights and use pre-trained T5 as the language model. For all experiments, we train for 10 epochs using a batch size of 64, a learning rate of 2e-5 with linear decay, and a weight regularization of 0.1. For the *Individual Top-K* approach, the evidence embeddings are re-computed after every 500 training steps. The number of retrieved documents is considered as a hyperparameter which is selected based on model performance over development set. During training, we uniformly sample the target answer from the list of reference answers. Training of Individual Top-K was performed on 240 GPUs. During training and inference, we retrieve the top-K documents from evidence ($\sim$21M documents) by performing an exact search.[10]

[9] We use the Stanza model trained on OntoNotes (Pradhan et al., 2012) to extract named entities for 10 NER categories.

[10] As matrix multiplication is highly optimized on GPUs we observed that performing exact search was efficient during training.

Retriever	NQ-Open				TriviaQA			
	Top-1	**Top-5**	**Top-20**	**Top-100**	**Top-1**	**Top-5**	**Top-20**	**Top-100**
Base Configuration								
BM25	–	–	59.1	73.7	–	–	66.9	76.7
BERT (zero-shot)	0.9	3.9	9.4	20.3	0.6	2.8	7.2	17.8
ICT (zero-shot)	12.6	32.3	50.6	66.8	19.2	40.2	57.5	73.6
MSS (zero-shot)	**20.0**	**41.7**	**59.8**	**74.9**	**31.7**	**53.3**	**68.2**	**79.4**

Table 3.3: Effect of unsupervised pre-training on retrieval accuracy when evaluated on test sets.

3.5 Results: Retriever Training

In this section, we compare different approaches to train the retriever. Retrieval accuracy is evaluated using the top-k accuracy metric ($k \in \{1, 5, 20, 100\}$).

3.5.1 Importance of Retriever Pre-training

To assess the usefulness of retriever pre-training, we evaluate zero-shot retrieval recall, *i.e.*, without supervised finetuning, on NQ-Open and TriviaQA datasets. We present the retriever's performance when its weights are initialized with BERT, ICT, and MSS pre-training in Table 3.3. We also provide the scores of the popular BM25 retriever (Robertson and Zaragoza, 2009).

We first note that BERT obtains a poor retrieval accuracy reaffirming the observation in Lee et al. (2019) that masked language models do not perform well on QA retrieval tasks. However, retrieval quality benefits greatly from ICT pre-training as it provides a substantial gain of 40-50% absolute points in top-20 accuracy over BERT. Furthermore, MSS pre-training offers an improvement over ICT by more than 8 absolute points. MSS also outperforms BM25 on TriviaQA and matches or exceeds its performance on NQ-Open showcasing the effectiveness of MSS pre-training as BM25 is a very strong baseline. These results demonstrate the importance of ICT and MSS pre-training as they are effective in bootstrapping the retriever quality (almost from scratch).

Setting	Top-1	Top-5	Top-20	Top-100
Base Configuration				
`[CLS]`, 40 epochs	32.6	60.1	76.4	85.9
+ score scaling	34.1	60.9	77.6	85.9
+ 80 epochs	36.7	62.2	77.4	**86.0**
+ 1 hard negative	**48.6**	**68.8**	**79.0**	85.8
DPR (Official)	–	67.1	78.4	85.4

Table 3.4: Exploring optimal training settings for supervised finetuning of the retriever when evaluated on the NQ-Open test set.

3.5.2 Effect of Retriever Initialization on Supervised Finetuning

Supervised Finetuning using BERT

We explore the best training settings for *supervised* finetuning of the retriever. To do so, we perform a series of experiments on the NQ-Open dataset starting with the training settings from the popular DPR model (Karpukhin et al., 2020) and then progressively improving it. DPR was initialized with BERT, trained for 40 epochs, with a scaling parameter of 1.0, and utilized `[CLS]` token embeddings from the retriever. Our result with this setting is shown in Table 3.4. We then observe that incorporating retriever score scaling and longer training till 80 epochs helps to improve the top-5 and top-20 accuracy by 1.5-2 points. These results also signify that the original DPR model was significantly undertrained and not fully optimized.

In addition to score scaling, we further include 1 additional hard-negative example (similar to DPR) for each question-document pair and train the model for 80 epochs. Our results, in sync with the results of DPR, obtain substantial additional gains in performance. *These findings highlight that retriever score scaling, longer training, and including a hard negative example are essential to improve the supervised retriever's accuracy*. These supervised training results can be considered as a very strong baseline. Hence, we employ these settings in subsequent experiments.

Retriever	NQ-Open				TriviaQA			
	Top-1	**Top-5**	**Top-20**	**Top-100**	**Top-1**	**Top-5**	**Top-20**	**Top-100**
Base Configuration								
BERT + Supervised	48.6	68.8	79.0	85.8	57.5	72.2	80.0	85.1
ICT + Supervised	48.4	**72.1**	81.8	**88.0**	58.4	73.9	81.7	86.3
MSS + Supervised	**50.3**	71.9	**82.1**	87.8	**60.6**	**74.8**	**81.8**	**86.6**
Large Configuration								
BERT + Supervised	51.4	71.0	81.0	87.2	60.4	74.5	81.4	86.0
ICT + Supervised	**52.4**	**72.7**	**82.6**	**88.3**	**61.9**	**76.2**	**82.9**	**87.1**

Table 3.5: Effect of retriever initialization on retrieval accuracy in the full supervised finetuning setting when evaluated on test sets.

Unsupervised Pre-training and Supervised Finetuning

We next empirically evaluate our proposed approach of unsupervised pre-training with ICT and MSS tasks followed by supervised finetuning on the dataset-specific examples. Results are presented in Table 3.5. We observe both ICT and MSS pre-training provide absolute improvements of 2-3 points over the already strong BERT-based finetuning results. In addition, these gains are consistent when scaling up the retriever size to a large configuration and also across top-1 to top-100 accuracy levels.

These results highlight that even after finetuning the retriever with thousands of labeled examples, it does not lead to catastrophic forgetting of the discriminative properties learned by the retriever during the unsupervised pre-training process. Another merit is that being unsupervised, large text collections can be leveraged to pre-train the retriever, a considerable advantage over data-augmentation methods that rely on the availability of human-annotated question-document pairs (Ma et al., 2021a). Furthermore, when comparing the impact of ICT and MSS retriever initialization in the full supervised setting, we note that their accuracy gains are roughly similar.

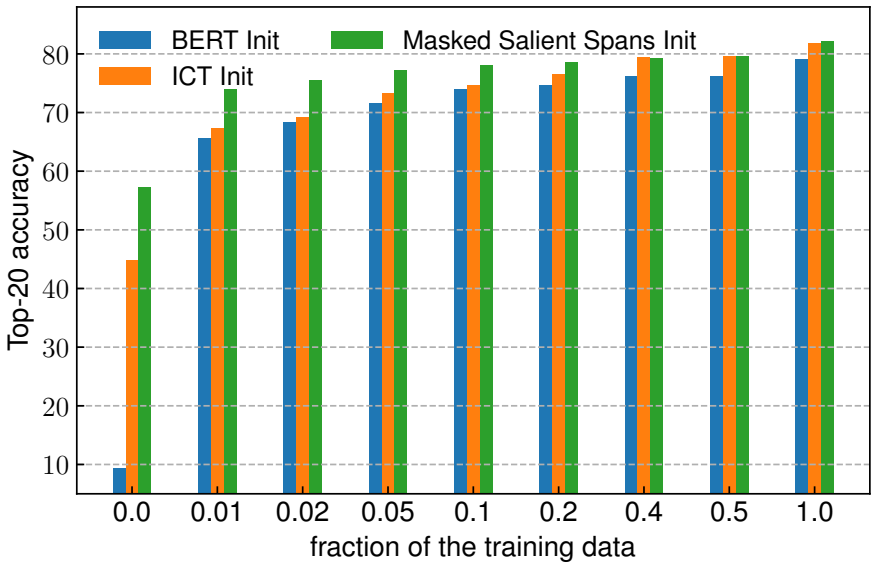

Figure 3.3: Sample efficiency analysis of different retriever initialization approaches. Evaluation is done on the NQ-Open test set.

Effect of the Amount of Training Data

We study the effect on accuracy when the retriever is initialized with BERT, ICT, or MSS and the amount of supervised training data is varied. We train the retriever with 1%, 2%, 5%, 10-50%, of NQ-Open's training examples and plot the top-20 accuracy in Figure 3.3. Results reveal that in the low-resource regime, *MSS pre-training is much more effective than ICT, consistently leading to large gains*. However, as the fraction of training data increases beyond 40% towards a high-resource setup, the gains from the MSS pre-training saturate to that of ICT. We believe that these findings will have important implications for future research—with only a few hundred supervised examples, performing MSS pre-training is beneficial while if the training data has thousands of examples, ICT pre-training is just as optimal as MSS pre-training.

End-to-End Retriever Training

For end-to-end training of the language model and retriever, we initialize the language model with T5 weights and the retriever with finetuned ICT weights. The number of retrieved evidence documents is considered as a hyperparameter and is selected using cross-validation on the development

Model			NQ-Open				TriviaQA			
	f_q	f_d	Top-1	Top-5	Top-20	Top-100	Top-1	Top-5	Top-20	Top-100
Base Configuration										
DPR (Official)			–	67.1	78.4	85.4	–	–	79.4	85.0
ICT + Supervised			48.4	72.1	81.8	88.0	58.4	73.9	81.7	86.3
Individual Top-k	✓	✗	54.5	73.7	83.2	88.6	61.4	75.6	82.1	86.7
Individual Top-k	✓	✓	**56.8**	**75.0**	**84.0**	**89.2**	**63.5**	**76.8**	**83.1**	**87.0**
Large Configuration										
ICT + Supervised			52.4	72.7	82.6	88.3	61.9	76.2	82.9	87.1
Individual Top-k	✓	✓	**57.5**	**76.2**	**84.8**	**89.8**	**66.4**	**78.7**	**84.1**	**87.8**

Table 3.6: Effect of end-to-end supervised training on retrieval accuracy. f_q and f_d denote if the question and the document encoder weights are finetuned or not during training, respectively.

set. In this section, we focus on analyzing the end-to-end training's effect on retrieval accuracy. We train the *Individual Top-K* model using question-answer pairs for each dataset following Eq. 3.7. We present the results for both base and large configurations of the model in Table 3.6 and compare them with previous approaches. We observe that for *Individual Top-K* when only the query encoder is finetuned, it improves accuracy uniformly across both datasets. In addition, finetuning the document encoder leads to additional gains. Specifically, accuracy improves to 75% at top-5 for NQ-Open, a big gain of 8 points over the previous best DPR retriever. Larger models further help to improve the scores leading to new state-of-the-art results (as of May 2021). These results highlight that when the retriever is pre-trained, the objective function of *Individual Top-K* is designed such that it improves the retriever.

3.5.3 Intuition for Retriever Score Scaling

Retrieval score scaling is used when computing the probability distribution of the retrieved documents according to Eq. 3.3, where the retrieval score is normalized by the scaling parameter (τ). To understand the effect of τ on the retrieval accuracy, we perform an ablation study with different values of τ on the NQ-Open retrieval task, whose results are presented in Table 3.7. We choose different values of τ as a multiple of $\sqrt{d}$, where d is the model hidden size. Our results indicate

$\times\sqrt{d}$	**Top-1**	**Top-5**	**Top-20**	**Top-100**	**Avg.**
			Base Configuration		
0.25	48.8	69.3	78.7	85.5	70.6
0.5	51.4	71.6	81.5	87.7	73.1
1	51.1	71.8	82.1	87.7	**73.2**
2	50.2	71.5	81.9	87.9	72.9
4	50.6	71.7	81.7	88.0	73.0

Table 3.7: Effect of score scaling parameter (τ) on the retrieval accuracy when evaluated on the NQ-Open test set. The first column denotes the multiple (m) that is multiplied by $\sqrt{d}$ to obtain τ, i.e., $\tau = m \times \sqrt{d}$ in Equation 3.3.

that the choice of $\tau = \sqrt{d}$ offers a sweet middle ground among other choices and works well in practice.

Here, we offer an intuitive explanation regarding the importance of the scaling parameter. In our preliminary experiments on retriever pre-training with $\tau = 1$, we observed that a few of the top-K document's similarity score with the query was very large. Such large scores skewed retriever probability distribution with most of the mass being centered over the top-1 or top-2 documents. A larger value of the scaling factor ensures a more even distribution of the probability mass over the top-K documents, which results in better training dynamics.

3.6 Results: Question Answering

In this section, we empirically quantify the impact of retrieval on the task of open-domain question answering. In this task, the reader (or language model) needs to output an answer given the question and the retrieved documents as inputs. We evaluate our previous best model *Individual Top-K* on NQ-Open and TriviaQA datasets, as it achieves the highest retrieval accuracy. During inference, the language model first greedily generates an answer for each retrieved document. We then score each answer using Eq. 3.7 and select the answer with the highest likelihood score. Results are reported using the conventional exact match (EM) metric.

We compare our results as presented in Table 3.8 with other recent approaches from the literature. For the base configuration on NQ-Open, our model outperforms both REALM (Guu et al.,

Model	NQ-Open	TriviaQA
Base Configuration		
ORQA (Lee et al., 2019)	33.3	45.0
REALM (Guu et al., 2020)	40.4	–
DPR (Karpukhin et al., 2020)	41.5	56.8
Individual Top-k	**45.9**	56.3
Large Configuration		
RAG (Lewis et al., 2020c)	44.5	56.8
Individual Top-k	**48.1**	**59.6**

Table 3.8: Question answering results with our proposed *Individual Top-K* approach. The grouping under base and large configurations is based on the sizes of the language model and retriever.

2020) and DPR by more than 4 points. For the large configuration, we compare it with the RAG model (Lewis et al., 2020c), where our approach outperforms it by 3.5+ points on NQ-Open and by 2.8 points on TriviaQA. We remark that compared to previous approaches, our strong results are due to a combination of several factors: (i) pre-trained retriever initialization, (ii) powerful T5 language model, and (iii) end-to-end retriever training updating both the question and document encoders.

3.7 Related Work

Early successful approaches to learning dense representations for questions and documents relied on using count-based estimates such as word frequency (Yih et al., 2011). Although successful in retrieval tasks, this style of approaches proved inefficient and hence not scalable. More recently, motivated by the success of masked language models such as BERT (Devlin et al., 2019) in language understanding tasks, several approaches have been proposed to learn dense passage representations by finetuning BERT using question-document pairs (Karpukhin et al., 2020). Although BERT initialization is critical to finetune retrievers, it suffers from poor discrimination ability in retrieval settings involving millions of documents. To improve this, unsupervised approaches have been proposed to further pre-train BERT using contrastive training (Lee et al., 2019).

Mining hard negative examples to train retrievers has proven to be quite effective in improving

retrieval accuracy as it results in a better approximation of the softmax partition function. One option to collect hard-negatives is by retrieving top documents with BM25 as they contain lexical similarities with the question (Karpukhin et al., 2020). Another promising alternative is to use the model parameters itself to mine hard negatives (Gillick et al., 2019; Xiong et al., 2021).

An emerging line of work investigates task-specific pre-training of the language model using retrieved documents. In the context of open-domain QA, Guu et al. (2020) predicts salient entities in order to pre-train the language model to generate an answer. Similarly, Lewis et al. (2020a) perform retrieval-augmented cross-lingual pre-training where the objective is to predict a text sequence conditioned on its retrieved paraphrases in different languages. This training objective demonstrated improved zero-shot performance in document translation tasks.

Our work is also related to the work of Chang et al. (2020). They explore several paragraph-level pre-training strategies in order to improve retrieval accuracy. However, our work differs in several ways. First, our retrieval setup is more challenging as the evidence consists of 21M passages while their evidence consists of 1M passages. Second, we pre-train with two strategies consisting of ICT and MSS and finetune using strong supervised methods leading to outperforming the strong DPR retriever. Finally, we further train the retriever with end-to-end supervised training which leads to new state-of-the-art results.

3.8 Discussion

In this work, we propose novel training approaches to improve the retrieval accuracy of dense embedder models for the task of open-domain question answering (OpenQA). We first propose two variants of existing approaches for unsupervised pre-training of the retriever: (a) inverse cloze task (ICT), and (b) predicting masked salient spans (MSS). We demonstrate the effectiveness of retriever initialization with ICT and MSS pre-training for supervised finetuning of the retriever. We then present an approach for supervised training of the retriever jointly with an encoder-decoder language model for the OpenQA task. These methods help achieve state-of-the-art results on both

retrieval benchmarks and the final answer extraction task.

In summary, the contributions of this work are:

- We demonstrate that our proposed method of *unsupervised pre-training* of the retriever with ICT followed by *supervised finetuning* leads to absolute gains of more than 2 points in the top-20 retrieval accuracy over the previous best result on NQ-Open and TriviaQA datasets.
- We show that *masked salient spans*-based pre-training of the retriever is more effective than ICT pre-training when the supervised dataset sizes are small.
- Our *end-to-end* supervised training approach obtains new state-of-the-art performance on retrieval accuracy. On the NQ-Open dataset, our top-20 accuracy is 84, which is a 5-point absolute gain over DPR results.
- We achieve competitive results on *answer extraction* with gains of more than 3 points over recent models such as REALM (Guu et al., 2020) and RAG (Lewis et al., 2020c).
- We *scale up* training to *large models* and show *consistent gains* in performance.

3.8.1 Limitations

Although our proposed approaches have been very effective in improving passage retrieval and QA accuracy, they suffer from several limitations that we elaborate on in this section.

Reliance on supervised datasets: Although our proposed approaches have been successful in achieving new state-of-the-art results on benchmark retrieval tasks, they still suffer from the same limitations as the previous approaches — existence of tens of thousands of labeled question answer or document pairs to finetune pre-trained retrievers. Collecting these human-annotated training examples is expensive as well as time-intensive. Lack of quality training data can otherwise hinder the application of finetuning-based methods in custom retrieval domains or tasks.

Dependence on external retrievers: Another critical factor underlying the success of dense retrievers is the usage of hard-negative examples in the training step. Typically, these hard-negatives are pre-computed using another retriever such as BM25. Such reliance on external modules for training increases the overall complexity of the training data pipeline, thereby limiting the rapid adoption of the approach.

Computationally expensive training: Unsupervised pre-training is computationally expensive as ICT and MSS pre-training requires large batch sizes necessitating a lot of GPUs for data parallelism. As such, this unsupervised pre-training may not be feasible in a resource-constrained setup. In addition, ICT pre-training requires performing an expensive all-gather collective operation at every step to aggregate embeddings from multiple GPUs. As such, using large batch sizes results in a slow forward pass during training.

Assessing out-of-domain generalization performance: In this work, we have used English Wikipedia as the pre-training dataset for both ICT and MSS tasks. Furthermore, our evaluations are performed on QA datasets that are primarily Wikipedia-based, *i.e.*, questions are supposed to be answered using Wikipedia. However, in many practical applications questions and documents are written in custom domain-specific terminology some examples being biomedical and finance domains. This presents a challenging setting as technical terms in these domains might be unseen in Wikipedia texts. It remains to be seen how well can retrievers trained using corpora such as Wikipedia transfer to to out-of-domain queries and texts. It will also be interesting to assess generalization after domain adaptation, e.g., finetuning retrievers on a small number of in-domain examples and also comparing with strong baselines such as BM25.

3.8.2 Follow-up Work

Our proposed idea of unsupervised retriever pre-training has been adopted as well as improved by several follow-up works. We list a few selected such works below:

- **Adversarial retriever-ranker**: Zhang et al. (2022) use a cross-encoder ranker to improve dense retriever finetuning. Their proposed method first initializes the retriever using unsupervised pre-training. Then, the retriever is finetuned using an adversarial training objective such that feedback from the ranker improves retriever quality in successive steps.

- **GPT embeddings:** To pre-train retrievers, Neelakantan et al. (2022) employed varying configurations of the GPT decoder (Radford et al., 2019; Brown et al., 2020) as the underlying embedder. They leverage web-scale text data crawled from the Internet and use neighboring sequences as positive examples. Similar to our work, they use sampled softmax loss with in-batch negatives for training but with large batch sizes.

- **Momentum contrastive retriever pre-training:** Izacard et al. (2022) pre-train dense retrievers using unsupervised text data. Similar to ICT (§3.2.1), it constructs training examples by sampling pseudo-queries and positive examples from a paragraph. The model is trained using momentum contrastive learning (He et al., 2020) by maintaining a large cache of negative documents. These learned embeddings have been shown to improve document rankings under the zero-shot setting.

- **Supervised retriever pre-training**: Oguz et al. (2022) pre-train retriever using a large set of question-passage pairs consisting of millions of examples that were obtained by either synthetically generated questions or using online Reddit conversations. This kind of training data is more closely aligned with the finetuning task and hence is better suited for pre-training retrievers.

3.8.3 Future Work

This contribution opens up several exciting directions for future work. Our work was one of the early efforts to pre-train retrievers in an unsupervised manner to boost their zero-shot accuracy. Follow-up works have made further progress but there remains a big gap between unsupervised and supervised retrievers performance. Future work can investigate better strategies for pre-training

such as the choice of input data, objective functions, etc. to further close this gap. Another option is to explore the idea of universal pre-training by using a combination of multiple pre-training strategies such that they are applicable to a range of end retrieval tasks.

Chapter 4

End-to-End Training of Multi-Document Reader and Retriever for Open-Domain Question Answering

Open-domain question answering (OpenQA) is a question answering task where the goal is to train a language model to produce an answer for a given question. In contrast to many question answering tasks, an OpenQA model is only provided with the question as its input without accompanying documents that contain the answer. One of the most promising approaches to OpenQA is based on augmenting the language model with an external knowledge source such as Wikipedia (often referred to as the evidence documents). In this approach, the model consists of two core components (Chen et al., 2017): (i) an information retrieval system to identify useful pieces of text from the knowledge source (the retriever), and (ii) a system to produce the answer given the retrieved documents and the question (the reader).

We can view such a model as a latent variable model, where the latent variables represent retrieved documents that are used to produce answers given questions (Lee et al., 2019). End-to-end (or joint) training of this model is challenging since we need to learn both to generate an answer given retrieved documents and what to retrieve. Previous work considers two potential solutions

(see Table 4.1 for a high-level summary). First, they adopt a stage-wise training, where the retriever is trained while freezing the reader and vice versa (Karpukhin et al., 2020; Izacard and Grave, 2021b,a). Another alternative is to constraint the reader to condition on each retrieved document individually[1] (Guu et al., 2020)—sometimes with extra supervision for the latent variables in the form of the relevant document for a question (Lewis et al., 2020c).

In this thesis, we consider a retrieval-augmented question answering model that combines information from multiple documents when generating answers. Expectation-maximization (Dempster et al., 1977) offers a principled template for learning this class of latent variable models. We present EMDR^2: **E**nd-to-end training of **M**ulti-**D**ocument **R**eader and **R**etriever (§4.1). EMDR^2 iteratively uses feedback from the model itself as "pseudo labels" of the latent variables for optimizing the retriever and reader parameters. We use two estimates of the latent variables: (i) prior scores for updating the reader parameters, and (ii) approximate posterior scores given all observed variables for the retriever parameters.

We evaluate our proposed method by experimenting with three commonly used OpenQA datasets: Natural Questions, TriviaQA, and WebQuestions (§4.2). EMDR^2 achieves new state-of-the-art results for models of comparable sizes on all datasets, outperforming recent approaches by 2-3 absolute exact match points. We also show that EMDR^2 is robust to retriever initialization. It achieves high accuracy with unsupervised initialization, suggesting that supervised training of the retriever may not be an essential component of the training process as suggested in prior work (Karpukhin et al., 2020).

In summary, our contributions are as follows: (i) we present an end-to-end training method (EMDR^2) for retrieval-augmented question answering systems, (ii) we demonstrate that EMDR^2 outperforms other existing approaches of comparable size without any kind of supervision on the latent variables, (iii) we provide ablation studies for a better understanding of the contributions of different components of our proposed method, and (iv) we release our code and checkpoints to

[1]This makes marginalization over the latent variables easier since we only need to consider one document at a time rather than multiple documents at once.

			Reader and Retriever Training			
Model	*Multi-doc Reader*	*Retriever Adaptation*	*Disjoint*	*End-to-end*	*Multi-step*	*Unsupervised Retriever*
REALM		✓		✓		✓
DPR			✓			
RAG		✓		✓		
FiD	✓		✓			
FiD-KD	✓	✓			✓	
EMDR^2 (Ours)	✓	✓		✓		✓

Table 4.1: Bird's-eye view of the recent OpenQA approaches. **Multi-doc reader** indicates whether the reader architecture uses multiple documents or a single document. **Retriever adaptation** shows whether the retriever gets feedback from the reader to update its parameters. **Disjoint** denotes that first the retriever is trained and then the reader is trained. **End-to-end** denotes that the reader and retriever are trained jointly in one cycle. **Multi-step** indicates that the reader and retriever are trained iteratively in multiple cycles. **Unsupervised retriever** indicates whether the retriever is initialized using unsupervised approaches or using supervised data. The references of above models are REALM (Guu et al., 2020), DPR (Karpukhin et al., 2020), RAG (Lewis et al., 2020c), FiD (Izacard and Grave, 2021b), FiD-KD (Izacard and Grave, 2021a).

facilitate future work and for reproducibility.[2]

EMDR^2 is a framework that can be used to train retrieval-augmented text generation models for any task. We believe that our estimation technique in EMDR^2 is also useful for learning similar latent variable models in other domains.

4.1 Model

Our proposed model EMDR^2 consists of two components: (i) a neural retriever, and (ii) a neural reader, which we train jointly in an end-to-end setting. Figure 4.1 shows an illustration of our model and training procedure. We discuss each component and our training objective in detail below.

[2]Our code is available at: https://github.com/DevSinghSachan/emdr2

4.1.1 Retriever: Dual-Encoder

Let the collection of evidence documents be denoted by $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_M\}$. Given a question $\boldsymbol{q}$, the goal of the retriever module is to select a subset of documents $\mathcal{Z} \subset \mathcal{D}$ to answer the question.[3] We model the retriever as a dual-encoder network (Bromley et al., 1994), where one encoder f_q encodes the question and another f_d encodes the evidence document (to a vector). The retrieval score is defined as the dot product between the two resulting vectors:

$$s(\boldsymbol{q}, \boldsymbol{d}_i; \Phi) = f_q(\boldsymbol{q}; \Phi_q)^\top f_d(\boldsymbol{d}_i; \Phi_d), \tag{4.1}$$

where $\Phi = [\Phi_q, \Phi_d]$ denotes the retriever parameters. We select top-K documents for the question $\boldsymbol{q}$ from $\mathcal{D}$ based on the retrieval scores. We denote the set of retrieved documents by $\mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K\}$.

We use transformer encoders (Vaswani et al., 2017) as our f_q and f_d. Our transformer architecture is similar to BERT with 12 layers and 768 hidden size (Devlin et al., 2019). We use the final representation of the first token (*i.e.*, the standard `[CLS]` token from BERT's tokenization) as our question (and similarly document) embedding. Initializing f_q and f_d with BERT weights has been shown to lead to poor retrieval accuracy (Lee et al., 2019; Sachan et al., 2021a). Therefore, we initialize the retriever with an unsupervised training procedure. We discuss our initialization technique in detail in §4.2.2.

4.1.2 Multi-Document Reader: Fusion-in-Decoder

The reader takes as input a question $\boldsymbol{q}$ and a set of retrieved documents (to be read) $\mathcal{Z}$ to generate an answer. Our reader is based on the Fusion-in-Decoder (FiD; Izacard and Grave, 2021b) model, which is built on top of T5 (Raffel et al., 2020). T5 is a pre-trained sequence-to-sequence transformer that consists of an encoder g_e and a decoder g_d (§2.4.3).

[3] This material is analogous to the dense retrievers introduced in §3.1.1, and is included here for ease of reading with additional notation.

In FiD, each retrieved document $\boldsymbol{z}_k$ is first appended with its title ($\boldsymbol{t}_{\boldsymbol{z}_k}$) and the question:

$$\boldsymbol{x}_k = \texttt{[CLS]}\,\boldsymbol{q}\,\texttt{[SEP]}\,\boldsymbol{t}_{\boldsymbol{z}_k}\,\texttt{[SEP]}\,\boldsymbol{z}_k\,\texttt{[SEP]},$$

where `[CLS]` is used to indicate the start of a document and `[SEP]` is used as a separator for the different parts of the document as well as the final token.

Each $\boldsymbol{x}_k$ is then independently given as an input to the T5 encoder g_e. The output representations corresponding to all of the retrieved documents are concatenated as:

$$\mathbf{X}_{\mathcal{Z}} = [g_e(\boldsymbol{x}_1); \ldots; g_e(\boldsymbol{x}_K)] \in \mathbb{R}^{(N \times K) \times H},$$

where N is the number of tokens in each $\boldsymbol{x}_k$[4] and H is the hidden size of the T5 encoder g_e. In this work, we use the T5-*base* configuration with $N = 512$ and $H = 768$.

$\mathbf{X}_{\mathcal{Z}}$ is then given as an input to the T5 decoder g_d. When generating an answer token, the decoder attends to both previously generated tokens (*i.e.*, causal attention) as well as the tokens encoded in $\mathbf{X}_{\mathcal{Z}}$ (*i.e.*, cross-attention). Since $\mathbf{X}_{\mathcal{Z}}$ contains information from multiple documents, the decoder has the ability to aggregate useful signals contained in multiple documents and jointly reason over them. We define the probability of the answer as:

$$p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}; \Theta) = \prod_{t=1}^{T} p\left(a_t \mid \boldsymbol{a}_{<t}, \boldsymbol{q}, \mathcal{Z}; \Theta\right), \tag{4.2}$$

where Θ denotes the reader parameters (*i.e.*, T5 encoder and decoder) and T is the number of answer tokens. We keep generating answer tokens until the decoder outputs a special `EOS` token or a pre-specified maximum answer length is reached.

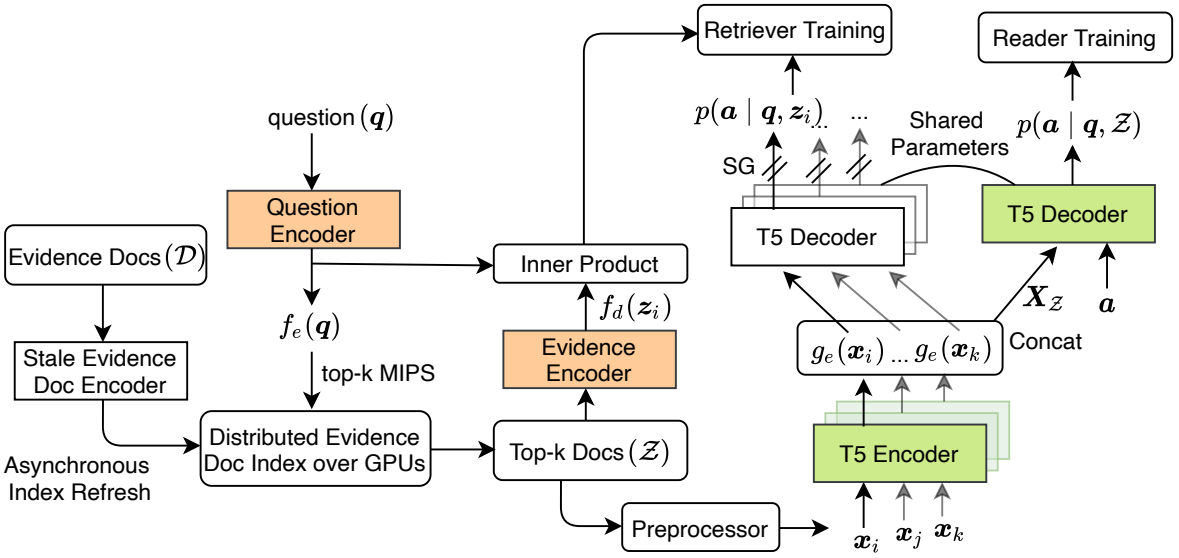

Figure 4.1: An illustration of the different components of EMDR^2. Colored blocks indicate components that contain trainable parameters.

4.1.3 End-to-End Training of Reader and Retriever

In contrast to previous work on generative question answering, we train both the reader and the retriever jointly in an end-to-end differentiable fashion.

Denoting our latent variable which represents a *set of retrieved documents* by Z and let $\mathcal{Z}$ be a possible value of Z. The marginal likelihood of an answer (marginalizing over all the possible values of Z) is: $p(\boldsymbol{a} \mid \boldsymbol{q}; \Theta, \Phi) = \sum_{Z=\mathcal{Z}} p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}; \Theta) p(\mathcal{Z} \mid \boldsymbol{q}; \Phi)$. The goal of our training procedure is to find Φ and Θ that would maximize the above objective. Exactly optimizing this equation is intractable as it is combinatorial in nature.[5] For one particular value $\mathcal{Z}$, the log-likelihood is simpler to compute: $\log p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}; \Theta) p(\mathcal{Z} \mid \boldsymbol{q}; \Phi) = \log p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}; \Theta) + \log p(\mathcal{Z} \mid \boldsymbol{q}; \Phi)$.

Expectation-maximization (EM) algorithm (Dempster et al., 1977) offers a solution to learning this latent variable model. In classical EM, we iteratively compute the posterior of Z given all observed variables and use it to update Θ and Φ.

We propose using two estimates of Z—$\mathcal{Z}_{\text{reader}}$ and $\mathcal{Z}_{\text{retriever}}$—for updating the two components

[4]We truncate and pad as necessary such that every $\boldsymbol{x}_k$ has the same length N. See §4.2.2 for details.

[5]Contrast our objective with REALM (Guu et al., 2020), where the reader only conditions on one retrieved document $\boldsymbol{z}_k$ when generating an answer. In this case, the latent variable represents a document assignment instead of a set of retrieved documents.

of the model (reader parameters Θ and retriever parameters Φ):

$$\log \underbrace{p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}_{\text{reader}}; \Theta)}_{\text{reader}} + \log \underbrace{p(\mathcal{Z}_{\text{retriever}} \mid \boldsymbol{q}; \Phi)}_{\text{retriever}}. \tag{4.3}$$

In the first term, we set the value of the latent variable $Z = \mathcal{Z}_{\text{reader}}$ based on the prior scores. In the second term, we seek to maximize an approximate posterior of $Z = \mathcal{Z}_{\text{retriever}}$. We discuss them in more detail below.

Reader parameters Θ: For updating Θ (the first term of Eq. 4.3), we use the top-K documents with the highest individual scores (as computed by Eq. 4.1 based on the current value of Φ) to construct $\mathcal{Z}_{\text{reader}}$. This is equivalent to relying on the prior $p(Z \mid \boldsymbol{q}; \Phi)$ to estimate $\mathcal{Z}_{\text{reader}}$ (without using information from the answer $\boldsymbol{a}$). We choose to use the prior to train reader parameters since the prior scores are also used at evaluation time to obtain the top-K documents. As a result, there is no mismatch between training and test computations when computing $p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}; \Theta)$ (*i.e.*, $\mathcal{Z}$ that is used at test time is obtained in exactly the same way as $\mathcal{Z}_{\text{reader}} = \mathcal{Z}_{\text{top-}K}$).

Retriever parameters Φ: For updating Φ (the second term of Eq. 4.3), we propose to use the posterior estimate. In other words, we use additional information from $\boldsymbol{a}$ when evaluating $Z_{\text{retriever}}$ to train Φ. Using the posterior allows our retriever to learn from richer training signals as opposed to relying only on the prior.

We need to be able to compute $p(\mathcal{Z}_{\text{retriever}} \mid \boldsymbol{q}, \boldsymbol{a}; \Theta, \Phi)$ to maximize the retriever parameters. However, computing this quantity is difficult since it is a probability of a set.[6] Consider a set of K documents (e.g., $\mathcal{Z}_{\text{top-}K}$), where $\boldsymbol{z}_k$ denotes a document in the set. We approximate the maximization of the probability of the set by assuming that its probability is maximized if the sum of the probability of each document in the set is maximized.[7] With this approximation, we arrive

[6] This is true whether we choose to use the posterior probability or the prior probability.

[7] The intuition is that each element of the set contributes independently, which greatly simplifies the computation to find the maximum of the set.

at a simpler quantity: $\sum_{k=1}^{K} p(\boldsymbol{z}_k \mid \boldsymbol{q}, \boldsymbol{a}; \Theta, \Phi)$. Note that using Bayes rule, we can rewrite:[8]

$$p(\boldsymbol{z}_k \mid \boldsymbol{q}, \boldsymbol{a}; \Theta, \Phi) \propto p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_k; \Theta) p(\boldsymbol{z}_k \mid \boldsymbol{q}; \Phi). \tag{4.4}$$

The reader now only conditions on one document when computing the probability of an answer $p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_k; \Theta)$. This simpler reader uses the same parameters as the more sophisticated one Θ, but it only uses one document $\boldsymbol{z}_k$ instead of a set of documents.

To compute Eq. 4.4, we first obtain K documents with the highest scores as computed by Eq. 4.1 based on the current value of Φ. We compute the probability of document $\boldsymbol{z}_k \in \mathcal{Z}_{\text{top-}K}$ using sampled softmax as:

$$p(\boldsymbol{z}_k \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Phi) \approx \frac{\exp(s(\boldsymbol{q}, \boldsymbol{z}_k)/\tau; \Phi)}{\sum_{j=1}^{K} \exp(s(\boldsymbol{q}, \boldsymbol{z}_j)/\tau; \Phi)}, \tag{4.5}$$

where τ is a temperature (or scaling) hyperparameter and the approximation assumes that documents beyond the top-K contribute very small scores so we do not need to sum over all evidence documents M in the denominator (which is in the order of tens of millions in our experiments). We then compute $p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_k; \Theta)$ similarly to Eq. 4.2.

Overall training objective of EMDR^2: Combining the above derivations, our end-to-end training objective that we seek to maximize for a particular example becomes

$$\mathcal{L} = \underbrace{\log p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Theta)}_{\text{reader}} + \underbrace{\log \sum_{k=1}^{K} \mathbb{SG}\left(p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_k; \Theta)\right) p(\boldsymbol{z}_k \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Phi)}_{\text{retriever}}, \tag{4.6}$$

where $\mathbb{SG}$ is the stop-gradient operator so that the reader parameters Θ are not updated to also perform well given a single document $\boldsymbol{z}_k$. The stop-gradient operator in the second term of EMDR^2 has several benefits. First, the FiD reader is trained from the first term of the EMDR^2 objective in

[8] We choose not to normalize with $p(\boldsymbol{a} \mid \boldsymbol{q}; \Theta, \Phi)$ since computing this quantity would require summing over all evidence documents M. While this makes the resulting objective that we optimize not correspond to a proper probability distribution anymore, we observe that our training method still behaves well in practice.

which its likelihood is conditioned on all the retrieved documents, similar to how the reader is used at test time. Second, it also makes training faster since the backward pass which is computationally more expensive than the forward pass is not needed, which in turn reduces the usage of GPU RAM as intermediate activations need not be cached.

Given a training example, we update Θ and Φ by taking gradients of Eq. 4.6 with respect to Θ and Φ in an end-to-end fashion. Intuitively, we train the reader to generate the correct answer given K highest scoring documents $\mathcal{Z}_{\text{top-}K}$. For the retriever, we train it to select K documents which *collectively* has a high score of generating an answer (since the sum over K is inside the log in the second term) while taking into account feedback from the reader. Algorithm 1 summarizes our training algorithm.

Algorithm 1: End-to-end training of multi-document reader and retriever.

Input: Model parameters Θ and Φ, evidence documents $\mathcal{D}$.
while *not converged* **do**
- Compute $\mathcal{Z}_{\text{top-}K}$ using the current retriever parameters Φ. `// E-step`
- Compute $p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_k)$ for each $\boldsymbol{z}_k$ using the current reader parameters Θ. `// E-step`
- Update model parameters Θ and Φ to maximize the log-likelihood in Eq. 4.6. `// M-step`

end

4.2 Experiments

4.2.1 Datasets

We experiment with three commonly used open-domain question answering datasets:

- **Natural Questions (NQ-Open; Kwiatkowski et al., 2019):** NQ-Open contains questions asked by users of the Google search engine. Similar to Lee et al. (2019), we use the short answer subset. We refer the reader to §3.3.1 for further details.

- **TriviaQA (Joshi et al., 2017):** TriviaQA is a collection of trivia question-answer pairs that

Dataset	Train	Filtered Train	Dev	Test
WebQuestions (WebQ)	3,417	2,474	361	2,032
Natural Questions (NQ-Open)	79,168	58,880	8,757	3,610
TriviaQA	78,785	60,413	8,837	11,313

Table 4.2: QA dataset statistics. The training set is used for end-to-end training whereas the filtered training set is used for supervised retriever training. The filtered set ignores those examples where either the evidence document retrieved using BM25 (Robertson and Zaragoza, 2009) does not align with the original positive documents or the top-100 BM25 retrievals do not contain reference answers. We leverage the filtered training set as provided by Karpukhin et al. (2020).

were collected from multiple sources on the web (§3.3.1). For our experiments, following Izacard and Grave (2021a), we select human-annotated answers for training the QA model. We also filter out those questions whose answer length is more than 5 words. Overall, this filters out 2,362 examples from the training set.

- **WebQuestions (WebQ; Berant et al., 2013):** WebQ questions were collected using Google Suggest API and are focused around a single named entity.[9,10] The answers were annotated by crowdworkers using the Freebase knowledge graph. We use the version from Chen et al. (2017) where Freebase IDs in the answers are replaced by entity names.

Dataset statistics: For validation, we randomly select approximately 10% examples from the training set. For all the datasets, we use the dataset splits from Lee et al. (2019). We provide the size of the training, development, and test sets in Table 4.2.

Evidence documents $\mathcal{D}$: We use the preprocessed English Wikipedia dump from December 2018 released by Karpukhin et al. (2020) as our evidence documents. Each Wikipedia article is split into non-overlapping 100 words long segments. Each segment corresponds to a document in our case. There are a total of 21,015,324 documents in total.

[9] https://nlp.stanford.edu/software/sempre/

[10] https://github.com/google-research/language/tree/master/language/orqa#getting-the-data

4.2.2 Implementation Details

Hardware and library

We run all of our experiments on a machine with 96 CPUs, 1.3TB physical memory, and 16 A100 GPUs. We use PyTorch (Paszke et al., 2019) to implement our proposed model and relevant baselines.

Model configurations

For both the retriever and reader, we use the *base* configuration that consists of 12 layers, 768 dimensional hidden size, and 12 attention heads. In all experiments, we retrieve 50 documents, unless stated otherwise. We only use the base configuration in our experiments due to GPU memory constraints. However, we believe that our results would generalize to larger configurations as well.

Retrieval

To support fast retrieval, we pre-compute evidence document embeddings and store their shards over all the GPUs in a distributed fashion. We refer to these document embeddings as the document index. For each question, we retrieve documents in an online (on-the-fly) manner by performing the exact maximum inner product search (MIPS), implemented using asynchronous distributed matrix multiplication over the sharded document index.

Retrieved documents are tokenized to subwords according to BERT's tokenization and are given as input to the T5 reader. If a tokenized document is shorter than 512 tokens, it is padded using the tokens from the surrounding documents until the maximum token limit is reached. Such padding additionally helps to provide an extended context for answer generation.

Initialization and training details

We initialize the parameters of the model with unsupervised pre-training before supervised fine-tuning of the model using question-answer examples. Unsupervised pre-training is essential as it

Hyperparameter	ICT	MSS
Dataset	Wikipedia	Wikipedia
Number of Parameters	220M	440M
Hidden Size	768	768
Attention heads	12	12
Dropout	0.1	0.1
Optimizer	Adam	Adam
Batch Size	4096	64
Training Steps	100K	82K
Warmup Ratio	0.01	0.05
Peak Learning Rate	1e-4	2e-5
Weight Decay	1e-2	1e-1
Learning Rate Decay	Linear	Linear
Gradient Clipping (max L^2 norm)	1.0	1.0

Table 4.3: Hyperparameters for pre-training with ICT and MSS tasks.

helps to warm-start the retriever so that it outputs related documents for a given question.

We closely follow the unsupervised pre-training approach as described in §3.2.1 which is briefly described next. We first pre-train the retriever parameters with inverse cloze task (ICT) training for 100,000 steps (Sachan et al., 2021a). Then, we extract sentences containing named entities from the evidence documents. Next, we substitute 15% of these named entity spans with masked tokens. The masked sentence can be considered as the question to retrieve evidence documents and its salient spans (*i.e.*, named entities) can be considered as the answer to train the model with EMDR^2 (Eq. 4.6). During retrieval, we ignore the evidence document from which the masked sentence was derived. We train the model on these question-answer (masked sentence-named entities) pairs for 82,000 steps with a batch size of 64 using Adam (Kingma and Ba, 2014). We refer to this initialization method as unsupervised pre-training with *masked salient spans (MSS)*. We list the hyperparameters for ICT and MSS training in Table 4.3.

After MSS pre-training, we finetune the model on the dataset-specific QA training examples with EMDR^2. We perform training for 10 epochs on NQ-Open and TriviaQA with a batch size of 64, and for 20 epochs on WebQ with a batch size of 16. During training, we save a checkpoint every 500 steps and select the best checkpoint based on its performance on the development set. We list the training hyperparameters in Table 4.4. Apart from the number of epochs and batch size

Hyperparameter	NQ-Open	TriviaQA	WebQ
Number of Parameters	440M	440M	440M
Hidden Size	768	768	768
Attention heads	12	12	12
Dropout	0.1	0.1	0.1
Optimizer	Adam	Adam	Adam
Batch Size	64	64	16
Epochs	10	10	20
Warmup Ratio	0.01	0.01	0.01
Peak Learning Rate	2e-5	2e-5	2e-5
Weight Decay	1e-1	1e-1	1e-1
Learning Rate Decay	Linear	Linear	Linear
Gradient Clipping (max L^2 norm)	1.0	1.0	1.0
Temperate (τ)	27.7	27.7	27.7

Table 4.4: Hyperparameters for supervised finetuning on QA datasets.

in WebQ, we use the same hyperparameters for all the experiments. For the temperature parameter (τ) in Eq. 4.5, we follow Sachan et al. (2021a) and set it as the square root of the hidden size.

During end-to-end training, since the parameters of the document encoder ($f_d(\Phi_d)$) are also updated at every step, the pre-computed document embeddings become stale as training progresses. To prevent staleness, we use the most recent document encoder checkpoint to compute fresh document embeddings asynchronously with which the document index is updated after every 500 training steps. Asynchronous embedding updates are performed both during MSS pre-training and supervised finetuning.

Inference

For each question, we retrieve the top-K documents using MIPS and then feed them to the FiD reader. We use greedy decoding for answer generation at inference time.

4.2.3 Baselines

We compare our model to other approaches for OpenQA that can be categorized under the following two classes:

- **Closed-book QA models:** Large-scale language models capture a lot of world knowledge in their parameters derived from the corpus they have been trained on (Petroni et al., 2019). We compare with the work of Roberts et al. (2020) who show that larger T5 models—when finetuned with question-answer pairs—can perform remarkably well. We also compare with the few-shot results of GPT-3 (Brown et al., 2020).[11]

- **Open-book QA models:** Similar to this work, these models consist of retriever and reader components and adopt the retrieve then predict approach for answering questions given a collection of evidence documents. These models mainly differ in how the retriever is initialized (ORQA; Lee et al., 2019, DPR; Karpukhin et al., 2020), whether the reader processes a single document (ORQA, DPR, RAG; Lewis et al., 2020c) or multiple documents (FiD; Izacard and Grave, 2021b), or whether the reader and retriever are trained jointly or in a multistage process (REALM; Guu et al., 2020, FiD-KD; Izacard and Grave, 2021a).

4.3 Results

We follow standard conventions and report exact match (EM) scores using the reference answers included in each dataset. Table 4.5 presents our main results. We divide the table into three main sections: closed-book QA models, open-book QA models, and our implementation. The first two sections contain results from other papers, which we include for comparison. The last section includes results from our proposed model, as well as our reimplementation of relevant baselines to control for our experimental setup.

Our reimplementation of the T5-base provides strong baselines when the number of retrieved documents is set to 0 (no retrieval) and 1. From Table 4.5, we see that the setting of top-1 vastly improves performance over the setting with no retrieved documents, signifying the importance of retrieval for OpenQA tasks. When further increasing the top-K documents to 50, the performance of the FiD models substantially improves over the top-1 retrieval, verifying the observation

[11] We note that GPT-3 is not trained on the full training examples that we use, so the results are not directly comparable.

Model	top-K	**NQ-Open**		**TriviaQA**		**WebQ**		*# of*
		dev	test	dev	test	dev	test	*params*
Closed-Book QA Models								
T5-*base* (Roberts et al., 2020)	0	-	25.7	-	24.2	-	28.2	220M
T5-*large* (Roberts et al., 2020)	0	-	27.3	-	28.5	-	29.5	770M
T5-*XXL* (Roberts et al., 2020)	0	-	32.8	-	42.9	-	35.6	11B
GPT-3 (Brown et al., 2020)	0	-	29.9	-	-	-	41.5	175B
Open-Book QA Models								
BM25 + BERT (Lee et al., 2019)	5	24.8	26.5	47.2	47.1	27.1	21.3	220M
ORQA (Lee et al., 2019)	5	31.3	33.3	45.1	45.0	36.8	30.1	330M
REALM (Guu et al., 2020)	5	38.2	40.4	-	-	-	40.7	330M
DPR (Karpukhin et al., 2020)	25	-	41.5	-	56.8	-	34.6	330M
RECONSIDER (Iyer et al., 2021)†	30	-	43.1	-	59.3	-	44.4	440M
RAG-Sequence (Lewis et al., 2020c)†	50	44.0	44.5	55.8	56.8	44.9	45.2	626M
Individual Top-K (Sachan et al., 2021a)	-	-	45.9	-	56.3	-	-	440M
Joint Top-K (Sachan et al., 2021a)	50	-	49.2	-	64.8	-	-	440M
FiD (Izacard and Grave, 2021b)	100	-	48.2	-	65.0	-	-	440M
FiD-KD (Izacard and Grave, 2021a)	100	48.0	49.6	68.6	68.8	-	-	440M
Our Implementation (Base Configuration)								
FiD / T5-*base*	0	26.0	25.1	26.7	27.8	31.0	32.4	220M
FiD (DPR retriever, T5 reader)	1	37.3	38.4	50.8	50.4	40.2	38.3	440M
FiD (DPR retriever, T5 reader)	50	47.3	48.3	65.5	66.3	46.0	45.2	440M
FiD (MSS + DPR retriever, T5 reader)	50	48.8	50.4	68.0	68.8	43.5	46.8	440M
FiD (MSS retriever, MSS reader)	50	38.5	40.1	60.0	59.8	39.1	40.2	440M
EMDR^2 (MSS retriever, MSS reader)	50	**50.4**	**52.5**	**71.1**	**71.4**	**49.9**	**48.7**	440M

Table 4.5: Exact match scores on three QA datasets. Top-K denotes the number of retrieved documents that are used by the reader to produce an answer. To provide a fair comparison with our reimplementations, we show results from other papers with the base configuration, except for RAG-Sequence that uses BART-*large* (Lewis et al., 2020b). † indicates that their results on WebQ use NQ-Open training data to pre-train the model.

from Izacard and Grave (2021b) about the *importance of modeling the retrieved documents as a set*.

Comparing EMDR^2 with our reimplementation of FiD illustrates the benefit of our end-to-end training approach. The underlying model is similar in both cases, but the training method is different. FiD adopts a two-stage approach to first train the retriever and then the reader. We have three variants of FiD: (i) the reader and retriever are initialized with MSS training, (ii) the retriever is initialized with DPR training, which is the setting used in the original paper (Izacard and Grave,

2021b), and (iii) the retriever is initialized with MSS + DPR training from Sachan et al. (2021a), as it further improves DPR recall. EMDR^2 outperforms all the variants by large margins on all the datasets.

The current best approach for training multi-document reader and retriever is FiD-KD (Izacard and Grave, 2021a). FiD-KD is a complex training procedure that requires multiple training stages and performs knowledge distillation with inter-attention scores. We take the results from the original paper when comparing our model with FiD-KD. EMDR^2 outperforms the reported numbers of FiD-KD by more than 2.5 points on NQ-Open and TriviaQA to obtain new state-of-the-art results on these benchmarks.

In addition to better performance, EMDR^2 also has three other advantages compared to FiD-KD: (i) EMDR^2 is more efficient since it only uses 50 evidence documents, whereas FiD-KD leverages 100 documents; (ii) FiD-KD is based on a distillation approach which requires multiple cycles of retriever and reader training, while EMDR^2 only requires one cycle of end-to-end training; and (iii) FiD-KD relies on the supervised initialization of the retriever to achieve its best performance. EMDR^2 is more robust to the retriever initialization, as demonstrated by state-of-the-art results even with unsupervised initialization of the retriever.

For the WebQ dataset, the training set size is much smaller compared to the other datasets (Table 4.2). Previous approaches such as RAG rely on supervised transfer (i.e., they finetune a model pre-trained on NQ-Open) to obtain good results. In contrast, EMDR^2 improves over the results from this RAG model by 3.5 points *without the supervised transfer step*. This result demonstrates the applicability of our approach to the low-resource setting where we only have a limited number of training examples.

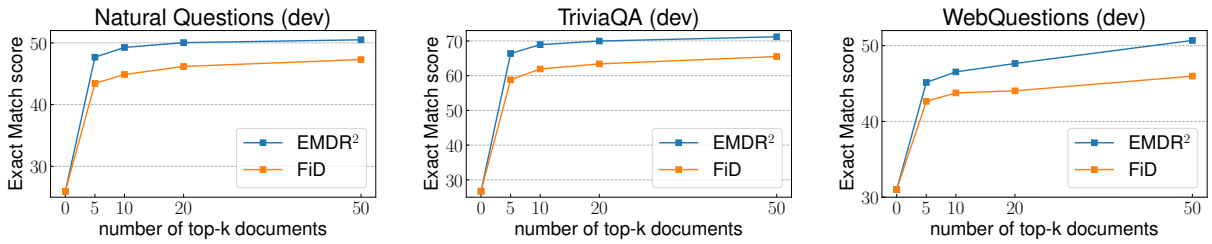

Figure 4.2: Effect on answer generation performance as the number of retrieved documents (top-K) is increased.

4.3.1 Ablation Studies

Number of retrieved documents

We investigate the performance of EMDR^2 and FiD as we vary the number of retrieved documents K in Figure 4.2. We observe that when the number of retrieved documents is increased, both EMDR^2 and FiD lead to an improvement in performance. When K is small, the gap between EMDR^2 and FiD is larger. This indicates the efficacy of EMDR^2 in a more constrained setting where we can only retrieve a small number of documents (e.g., due to memory limitations).

Retriever initialization

We explore the effect of different parameter initialization strategies when training with EMDR^2: (i) unsupervised MSS pre-training, (ii) supervised retriever training (DPR), and (iii) MSS pre-training followed by supervised retriever training (MSS + DPR; Sachan et al., 2021a). Table 4.6 presents our results. We can see that on NQ-Open, MSS pre-training being unsupervised leads to a lower initial retriever recall than DPR. After EMDR^2 training, the recall improves by 20% (highlighted in yellow cells). Training with DPR initialization leads to the same final recall as obtained by MSS pre-training, *suggesting that supervised initialization of the retriever may not be an essential component to obtain good performance in OpenQA tasks.* Similar trends are also observed on TriviaQA and WebQ. Similarly, MSS + DPR initialization has a better initial recall but leads to marginal or no improvements in answer extraction performance over MSS pre-training.

		NQ-Open (dev)			TriviaQA (dev)			WebQ (dev)		
Retriever Initialization	**Reader Initialization**	**R@50** B.T.	A.T.	**EM**	**R@50** B.T.	A.T.	**EM**	**R@50** B.T.	A.T.	**EM**
MSS pre-training	MSS pre-training	66.4	86.3	50.4	74.8	86.2	71.1	59.8	88.6	49.9
MSS pre-training	T5	66.4	86.3	50.3	74.8	86.3	70.9	59.8	88.6	47.7
DPR training	T5	82.3	86.3	50.0	83.2	86.2	70.5	84.2	88.6	49.0
MSS + DPR	MSS pre-training	84.5	86.3	50.5	85.3	86.3	71.2	85.0	88.6	49.9

Table 4.6: R@50 denotes the retrieval recall from the top-50 retrieved documents. B.T. and A.T. indicates R@50 score 'before training' and 'after training' the model, respectively.

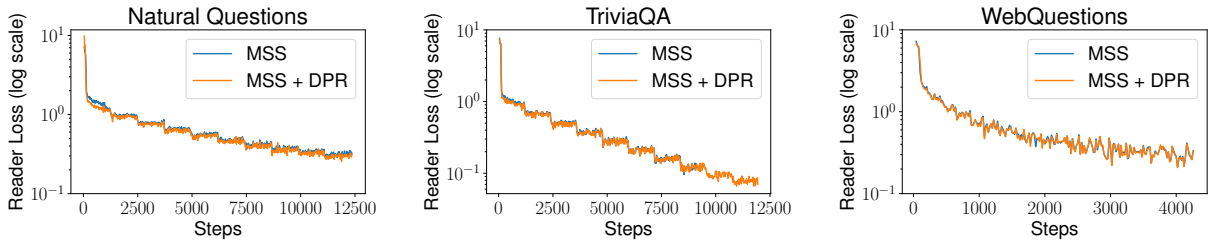

Figure 4.3: Comparison of reader training losses when the retriever is either initialized by MSS pre-training and by MSS followed by supervised DPR training (MSS + DPR).

We also observe that MSS pre-training also provides an improvement of 2 points in answer extraction on WebQ when compared to the T5 reader (shown in orange cells), highlighting the importance of a pre-trained reader in the low-resource setup.

Effect on reader training: We plot the reader's training loss when finetuned using QA pairs for two cases: (i) when the retriever is initialized with MSS pre-training, and (ii) when the retriever is initialized with MSS followed by DPR training (MSS + DPR). From the plots in Figure 4.3, we notice that retriever initialization has a marginal effect on the final answer generation performance. Specifically, for NQ-Open, MSS + DPR initialization leads to a lower training loss than MSS initialization for the first 1200 steps after which the difference between these two losses diminishes. Although surprising at first, this finding can be explained as MSS + DPR retriever being more accurate results in lower training loss initially. However, as training progresses, EMDR^2 updates the MSS retriever more aggressively than MSS + DPR eventually leading to both retrievers receiving

Question / Answer	Retriever	
	MSS Pre-training	**EMDR^2 Finetuning**
Question: what type of reaction occurs to form a dipeptide? **Answer**: peptide bond	*probability* = 0.39 . . . Bornyl diphosphate synthase In enzymology, bornyl diphosphate synthase (BPPS) () is an enzyme that catalyzes the chemical reaction Bornyl diphosphate synthase is involved in the biosynthesis of the cyclic monoterpenoid bornyl diphosphate. As seen from the reaction above, BPPS takes geranyl diphosphate as its only substrate and isomerizes into the product, (+)- bornyl diphosphate. This reaction comes from a general class of enzymes called terpene synthases that . . .	*probability* = 0.78 . . . Subsequent to this coupling reaction, the amine protecting group P and the ester are converted to the free amine and carboxylic acid, respectively. For many amino acids, the ancillary functional groups are protected. The condensation of the amine and the carboxylic acid to form the peptide bond generally employs coupling agents to activate the carboxylic acid. The Bergmann azlactone peptide synthesis is a classic organic synthesis for the preparation of dipeptides. . . .
Question: when was the japanese videogame company nintendo founded? **Answer**: 23 September 1889	*probability* = 0.37 . . . contributed to the development of the following games. Creatures (company) Ape, Inc. was founded in March 1989 and Shigesato Itoi became its chief executive officer. Nintendo president Hiroshi Yamauchi had wanted to support new talent in game design. Liking Itoiś work, he proposed the idea of the company to Itoi and invested in it. Apeś staff included Tsunekazu Ishihara, who later became the Pokémon Companyś CEO, and Ashura Benimaru Itoh, a renowned illustrator. They began work on "Mother", which released in July. Its music was composed by Hip Tanaka, who later became the second CEO of Creatures . . .	*probability* = 0.61 . . . Nintendo Co., Ltd. is a Japanese multinational consumer electronics and video game company headquartered in Kyoto. Nintendo is one of the world's largest video game companies by market capitalisation, creating some of the best-known and top-selling video game franchises, such as "Mario", "The Legend of Zelda", and "Pokémon". Founded on 23 September 1889 by Fusajiro Yamauchi, it originally produced handmade hanafuda playing cards. By 1963, the company had tried several small niche businesses, such as cab services and love hotels. Abandoning previous ventures in favour of toys in the 1960s . . .

Table 4.7: Examples of top-1 retrieved documents from the NQ-Open test set when the model is pre-trained with masked salient spans (MSS; first column) and then finetuned using NQ-Open data (second column). If the answer exists in the document it is highlighted in blue color, and the probability of the document (computed according to Eq. 4.5) is indicated in orange color.

similar gradient updates which reflects in both achieving similar training losses.[12]

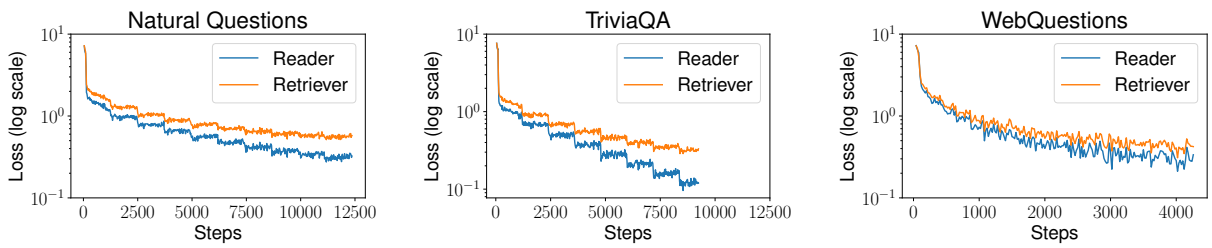

Figure 4.4: Reader and retriever training losses when the MSS pre-trained model is finetuned with EMDR^2 using dataset-specific examples.

4.3.2 Qualitative Analysis

In Table 4.7, we present some representative examples of the retriever output when it is initialized with MSS pre-training and when it is finetuned with EMDR^2 on NQ-Open. We observe that after MSS pre-training, the top-1 outputs are related to the question but are not relevant enough to answer them. However, when the retriever is finetuned with EMDR^2 on NQ-Open examples, the retrieval accuracy improves with the top-1 documents being much more relevant to answer the question. In addition, the retriever's likelihood of the top-1 document being relevant also improves.

Visualizing reader and retriever losses: In Figure 4.4, we show the trajectories of the reader and retriever training losses as the EMDR^2 training progresses.

4.3.3 Alternative End-to-End Training Objectives

We compare EMDR^2 objective (Eq. 4.6) to three alternative formulations for end-to-end training.

In the first alternative formulation, when training the retriever parameters Φ, we simply factorize $p(\mathcal{Z}_{\text{top-}K} \mid \boldsymbol{q}; \Phi) = \prod_{k=1}^{K} p(\boldsymbol{z}_k \mid \boldsymbol{q}; \Phi)$ to arrive at the following objective:

$$\mathcal{L}_{\text{alt-1}} = \log p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Theta) + \sum_{k=1}^{K} \log p(\boldsymbol{z}_k \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Phi).$$

The second term in this objective is maximized by a uniform retrieval, in other words, by *removing*

[12] As the MSS retriever is further away from the optimal state than MSS + DPR, it is more aggressively updated.

any discrimination between documents in the retriever. We include it to show the impact of an adversarial objective.

In the second formulation, for each retrieved document, we approximate its posterior under the assumption that we have a uniform prior over the set of retrieved documents: $\tilde{p}(\boldsymbol{z}_k \mid \boldsymbol{q}, \boldsymbol{a}, \mathcal{Z}_{\text{top-}K}; \Theta) \propto p(\boldsymbol{a} \mid \boldsymbol{q}, \boldsymbol{z}_k; \Theta) \times \frac{1}{K}$. We use this to train reader and retriever parameters as follows:

$$\mathcal{L}_{\text{alt-2}} = \log p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Theta) + \mathbb{KL}(\mathbb{SG}\left(\tilde{p}(\boldsymbol{z}_k \mid \boldsymbol{q}, \boldsymbol{a}, \mathcal{Z}_{\text{top-}K}; \Theta)\right) \;||\; p(\boldsymbol{z}_k \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Phi)).$$

Intuitively, we try to match the probability of retrieving a document $\boldsymbol{z}_k$ with the "contribution" of that document to the generated answer $\boldsymbol{a}$, regardless of whether the retriever is relatively more or less likely to retrieve the document *a priori*.

In the third formulation, similar to Lewis et al. (2020a), we infuse the retriever likelihood score within the FiD model. More specifically, we add retriever distribution (Eq. 4.5) to bias the encoder-decoder attention as it helps facilitate end-to-end training. Intuitively such biasing constraints the reader to attend strongly to the most relevant documents. The attention score during the encoder-decoder attention is computed as:

$$\text{attn}(\boldsymbol{q}, \boldsymbol{a}, \boldsymbol{z}_{1:K}) \propto Q(\boldsymbol{a})^\top K(\boldsymbol{z}_{1:K}, \boldsymbol{q}) + \lambda[p(\boldsymbol{z}_1 \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Phi), \ldots, p(\boldsymbol{z}_K \mid \boldsymbol{q}, \mathcal{Z}_{\text{top-}K}; \Phi)],$$

where Q is the query vector computed from the decoder's input, K is the key vector computed from the encoder's output, and λ is a trainable parameter. We train the model parameters by maximizing the likelihood of answer generation using teacher-forcing as:

$$\mathcal{L}_{\text{alt-3}} = \log p(\boldsymbol{a} \mid \boldsymbol{q}, \mathcal{Z}, p(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{Z}; \Phi); \Theta).$$

Table 4.8 presents our results on the development set of the QA datasets. We observe that training with the adversarial $\mathcal{L}_{\text{alt-1}}$ objective diverges, leading to poor performance, as expected.

Method	top-K	NQ-Open (dev)	TriviaQA (dev)	WebQ (dev)
FiD	50	47.3	65.5	46.0
EMDR^2	50	**50.4**	**71.1**	**49.9**
$\mathcal{L}_{\text{alt-1}}$	50	14.1	11.9	28.0
$\mathcal{L}_{\text{alt-2}}$	50	49.9	69.6	28.8
$\mathcal{L}_{\text{alt-3}}$	50	47.8	64.8	–

Table 4.8: Exact match scores on the development set for alternative end-to-end training objectives.

This shows that harming the retriever during training can significantly harm the performance of the QA system. In contrast, although it disregards the estimated prior, the $\mathcal{L}_{\text{alt-2}}$ objective still improves over the FiD baseline for NQ-Open and TriviaQA. However, it still lags behind EMDR^2. On WebQ, the $\mathcal{L}_{\text{alt-2}}$ objective diverges and leads to a poor performance. We leave further analysis on the convergence of $\mathcal{L}_{\text{alt-2}}$ objective as a part of future work. $\mathcal{L}_{\text{alt-3}}$ objective also leads to mixed results: it provides a small gain over the FiD baseline on NQ-Open but does not improve on TriviaQA. Overall, $\mathcal{L}_{\text{alt-3}}$ significantly lags behind EMDR^2, signifying that biasing the encoder-decoder attention with retriever distribution may not be an optimal strategy for end-to-end training.

4.4 Related Work

Our work is based on end-to-end training of neural readers and retrievers, which we have discussed in previous sections. Here we instead focus on discussing previous work related to standalone neural retrievers, neural readers, and their application in other natural language processing tasks.

Neural retrievers: There are two broad classes of neural retrievers based on the number of embeddings computed for a question and document: dual-encoders (Yih et al., 2011; Lee et al., 2019) and multi-vector encoders (Khattab and Zaharia, 2020; Luan et al., 2021). To perform retrieval, dual-encoders compute one embedding per evidence document while multi-vector encoders require multiple embeddings per document. Using multiple document encodings enables the multi-vector encoders to have deep query interactions and thus are more accurate but at the cost of being computationally expensive for large-scale settings. In this work, due to the large size

of the evidence, we use the more efficient dual-encoder retriever. Sachan et al. (2021a) show that the performance of dual-encoders can be improved by unsupervised pre-training of the retriever weights using contrastive approaches or by jointly training them with language models.

Neural readers: Neural readers output an answer for a question given context documents as its input. Neural readers can also be grouped under two categories: extractive and generative. *Extractive readers* extract a text span from a document to produce an answer (Chen, 2018). These are trained to predict the highest scoring start and end answer spans in a passage (Devlin et al., 2019). For the case of multiple passages, normalizing the answer span likelihood across them has shown to improve performance (Clark and Gardner, 2018; Wang et al., 2019b). On the other hand, for cases where multiple occurrences of an answer exist within a paragraph, training with a hard EM-based approach has proven to be effective (Min et al., 2019). *Generative readers* generate an answer conditioned on the question and context documents. These models are based on pre-trained encoder-decoder language models and are trained using teacher-forcing to autoregressively generate the answer tokens (Raffel et al., 2020; Lewis et al., 2020c). For the case of multiple retrieved documents for a question, jointly attending to independently computed document representations has been shown to benefit answer generation (Izacard and Grave, 2021b).

Other application areas: In addition to question answering, retrieval-augmented methods have been successfully applied to other natural language processing tasks. In causal language modeling, retrieving similar words from an external memory has been shown to improve perplexity (Khandelwal et al., 2020; Yogatama et al., 2021). In machine translation, retrieving domain-specific target language tokens has improved the performance of domain-specific texts (Khandelwal et al., 2021; Hoang et al., 2023). Finally, in dialog modeling, retrieving knowledge-informed text has helped improve factual correctness in the generated conversations (Fan et al., 2021; Shuster et al., 2021).

4.5 Discussion

In this chapter, we have presented EMDR^2, an end-to-end training method for retrieval-augmented question answering (QA) systems. We first defined the marginal likelihood formulation of the objective function by considering the set of retrieved documents to be a latent variable. As the exact optimization of marginal likelihood is intractable, we then showed how to arrive at our approximate training objective using the expectation-maximization algorithm. We demonstrated that our end-to-end training setup achieves substantially better answer extraction results than separately training retriever and multi-document reader. Below, we summarize our key contributions:

- We introduce a novel training framework, EMDR^2, to jointly train a multi-document reader and dense retriever system for the OpenQA task. The reader and the retriever are trained jointly in a single run of the training algorithm presenting a much simplified setting than the previous stage-wise trained models.

- When comparing similar-sized models, EMDR^2 obtains new state-of-the-art results on three benchmark datasets, outperforming the previous best method of FiD-KD by more than 2.5 absolute points.

- Our experiments signify that likelihood estimates of the answer conditioned on a single document are useful in training the retriever and the resulting improved retrieval contributes to more accurate answers.

- EMDR^2 only requires question-answer pairs to train a retrieval-augmented model. It removes the dependency on passage annotations that were previously presumed to be critical in training such a model.

- We conduct extensive ablation studies to analyze different aspects of the training procedure including proposing several alternate end-to-end training objectives and characterize their effectiveness on the OpenQA task.

4.5.1 Limitations

Although EMDR^2 successfully leverages language models to improve retrieval-augmented question answering systems, it shares a few limitations with other similar systems that we highlight below.

Potential negative societal impacts: While EMDR^2 has the potential to improve language models in the low-resource setting (as demonstrated by our results on WebQ in §4.3), it could exhibit typical biases that are associated with large language models. For example, our model does not have an explicit mechanism to generate answers that are calibrated for fairness across all spectra. Additionally, as a retrieval-augmented method, it also could be more prone to generating fake answers if an attacker manages to have access to and modify information in the collection of evidence documents.

Model performance evaluation: In this work, we have performed model evaluation using the exact match (EM) metric which matches the predicted answer with one or more manually annotated reference answers. In EM, if there is an answer equivalence, the predictions get full credit and vice versa. As the number of reference answers is finite and does not necessarily cover all possible variations, EM generally *underestimates* the performance of a QA system (Si et al., 2021). As a consequence, it is plausible that the *true* performance gains of EMDR^2 are higher than that reported in our results. In light of this aforementioned issue, attempts have been made to use token-level evaluation metrics such as F_1 score but it also suffers from issues like limited coverage. Recent efforts aimed towards addressing this propose model-based evaluation metrics such as BERT matching (BEM; Bulian et al., 2022) that uses BERT finetuned on human ratings to measure answer equivalence but we leave its application as a part of future work.

Requires language model finetuning: A key idea contributing to the success of methods like EMDR^2 is to finetune the language model (or reader) to predict the answer tokens conditioned on the question and retrieved documents. However, as the sizes of the language models continue to

scale up (Chowdhery et al., 2022), finetuning them would prove expensive. An even more detrimental consequence of finetuning is that of the language model losing its generality when applied to tasks other than QA. On the contrary, prompting the language models to generate the answer has so far resulted in sub-par performance when compared to finetuned language models (Lazaridou et al., 2022; Khattab et al., 2022). A common middle ground can be explored such as introducing additional task-specific parameters for finetuning while preserving the original language model to retain its general purpose usefulness (Li and Liang, 2021).

Computationally expensive: We store evidence documents in an uncompressed format, maintaining evidence indexes, and searching for relevant documents can be expensive (both in terms of compute and memory consumption), especially for web-scale datasets. However, it is worth noting that this limitation can be partly addressed by using efficient data structures (Monath et al., 2023) and approximate nearest neighbor search (Johnson et al., 2021; Guo et al., 2020; Chern et al., 2022). We also remark that our training procedure is relatively resource-heavy (requiring 16 A100 GPUs), potentially having environmental concerns. With our hardware setup, experiments on NQ-Open and TriviaQA took approximately 25 hours to complete. Before supervised training, we also performed a one-time unsupervised MSS pre-training that took roughly 1 week.

4.5.2 Follow-up Work

The ideas and models presented in this contribution have been referenced and improved by several follow-up works. We briefly describe a few of them as follows.

- **Few-shot finetuning of retrieval-augmented models:** Izacard et al. (2023) perform end-to-end training of a retrieval system similar to the one in this work by distilling the normalized answer prediction likelihood of an individual document to train the retriever. They demonstrate that large-scale pre-training followed by few-shot finetuning of a retrieval-augmented system using QA pairs outperforms larger parametric language models (Chowdhery et al., 2022).

- **Large language models vs task-specific models for QA:** In their study, Kamalloo et al. (2023) compares the performance of instruction-tuned large language models (OpenAI, 2022) and end-to-end trained systems such as EMDR^2 on QA tasks. Human evaluation conducted in their study reveals that large language models with few-shot prompting are becoming increasingly accurate in generating answers to information-seeking questions often matching task-specific models like EMDR^2.

- **End-to-end training of retrieval-augmented visual models:** Hu et al. (2023) propose an end-to-end training method for the task of answering visual queries. Their knowledge base consists of both multimodal and text documents, which are retrieved and attended to by the reader to answer questions. Retriever scores are used to bias the attention scores of the reader to achieve end-to-end training.

- **Synthetic data generation using large language model for input augmentation:** Recently, synthetic data generation has emerged as a popular method to leverage the knowledge contained within the parameters of a large language model to augment data for the end task (Lee et al., 2021). In particular, for the QA task, Yu et al. (2023) first generates contextual documents using a large language model by giving a question as the input. Augmented data obtained by combining generated documents with the documents retrieved from Wikipedia has been shown to improve answer prediction scores over pure retrieval-based approaches.

4.5.3 Future Work

This contribution presents several directions for future work. One direction is to extend the EMDR^2 algorithm to apply it to other important text generation tasks such as knowledge-grounded dialog generation. Another interesting direction would be to train retrieval-augmented language models (Borgeaud et al., 2022) in an end-to-end fashion. This would require designing adaptable retrievers for large-scale retrieval tasks such that the retrieved text segments improve the likelihood

of the next sequence prediction. Finally, it is worthwhile to recall that EMDR^2 is a tractable approximation to the marginal likelihood training objective. We feel that exploring tighter approximations to improve training dynamics can also be a useful direction to pursue.

Chapter 5

Unsupervised Passage Re-ranking

Text retrieval is a core sub-task in many NLP problems, for example, open-domain question answering where a document must be retrieved and then read to answer an input query. Queries and documents are typically embedded in a shared representation space to enable efficient search, before using a task-specific model to perform a deeper, token-level document analysis (e.g., a document reader that selects an answer span). We show that adding a zero-shot re-ranker to the retrieval stage of such models leads to large gains in performance, by doing deep token-level analysis with no task-specific data or tuning.

We focus on open-domain retrieval including question answering and introduce a re-ranker based on zero-shot question generation with a pre-trained language model. Our re-ranker, which we call *Unsupervised Passage Re-ranker* (UPR), re-scores the retrieved passages by computing the likelihood of the input question conditioned on a retrieved passage.[1] This simple method enables task-independent cross-attention between query and passage that can be applied on top of any retrieval method (e.g., neural or keyword-based) and is highly effective in practice (Figure 5.1).

In part, UPR is inspired by the traditional models of query scoring with count-based language models (Zhai and Lafferty, 2001). However, instead of estimating a language model from each passage, UPR uses pre-trained language models (PLMs). More recent work on re-rankers have

[1] In this chapter, we refer to the words documents and passages interchangeably. We consider the retrieval units as short passages and not entire documents.

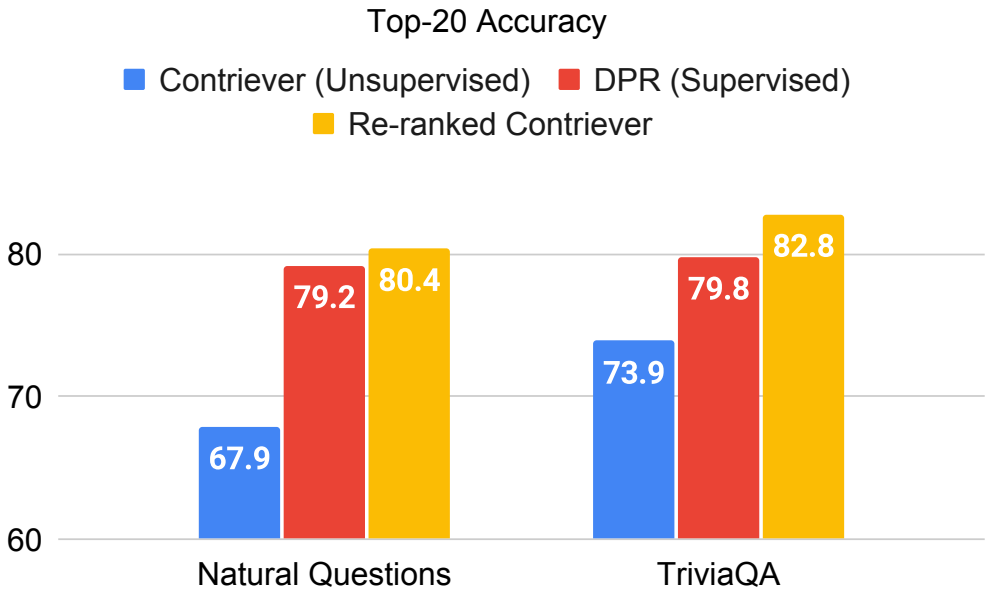

Figure 5.1: After UPR re-ranking of the unsupervised Contriever's (Izacard et al., 2022) top-1000 passages, we outperform strong supervised models like DPR (Karpukhin et al., 2020) on Natural Questions and TriviaQA datasets.

finetuned PLMs on question-passage pairs to generate relevance labels (Nogueira et al., 2020), sometimes to jointly generate question and relevance labels (Nogueira dos Santos et al., 2020; Ju et al., 2021). In contrast, UPR uses off-the-shelf PLMs, does not require any training data or finetuning, and still leads to strong performance gains (Figure 5.1).

Comprehensive experiments across a wide range of datasets, retrievers, and PLMs highlight the strengths of UPR in both improving the performance on retrieval and question answering tasks. To the best of our knowledge, this is the first work to show that a fully unsupervised pipeline (consisting of a retriever and re-ranker) can greatly outperform supervised dense retrieval models like DPR (Karpukhin et al., 2020). As language models continue to improve rapidly (Chowdhery et al., 2022; Touvron et al., 2023; Anil et al., 2023), the performance of UPR may see corresponding gains over time. UPR requires no annotated data and uses only generic pre-trained models, which means it may be easy to apply to a wide range of retrieval problems.

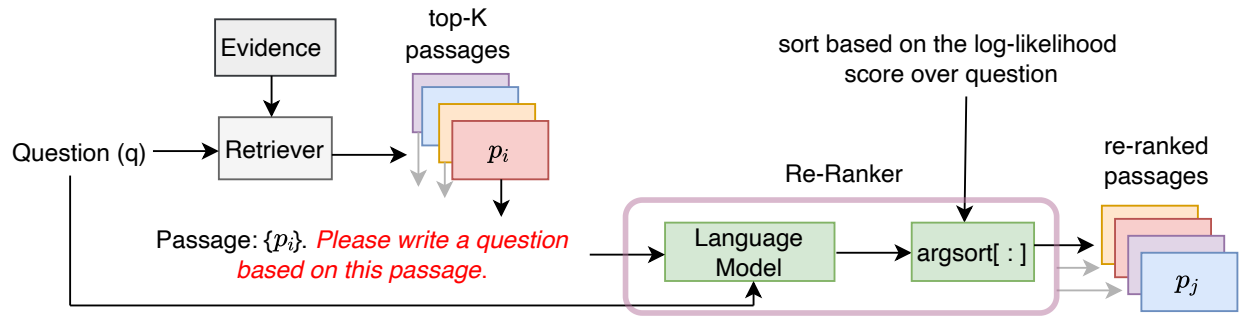

Figure 5.2: An illustration of the different components in UPR. For more details, please refer to text.

5.1 Method

Figure 5.2 presents an overview of our approach for open-domain retrieval, which introduces a new unsupervised re-ranker (§5.1.2) that can be applied to any existing text retriever (§5.1.1).

5.1.1 Retriever

Let $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_M\}$ be a collection of evidence passages (or documents). Given a question ($\boldsymbol{q}$), the retriever selects a subset of relevant passages $\mathcal{Z} \subset \mathcal{D}$, one or more of which will ideally contain the answer to $\boldsymbol{q}$. Our method will work with passages obtained from any retriever — either based on sparse representations like BM25 or dense representations like DPR. We only assume that the retriever provides the K most relevant passages. We denote this set of top-K passages as $\mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K\}$.

5.1.2 Unsupervised Passage Re-ranking (UPR)

Given the top-K retrieved passages, the goal of the re-ranker is to reorder them such that a passage with the correct answer is ranked as highly as possible. The ordering is computed with a *relevance score* $p(\boldsymbol{z}_i \mid \boldsymbol{q})$ for each passage $\boldsymbol{z}_i \in \mathcal{Z}$.

Our re-ranking approach is unsupervised, *i.e.*, it does not use any task-specific training examples. We refer to it as **UPR**, for *Unsupervised Passage Re-ranking*. UPR uses a pre-trained

language model to score the probability of generating the question $\boldsymbol{q}$ given the passage text $\boldsymbol{z}$, as described below. The question generation model is zero-shot, allowing for dataset-independent re-ranking, and also incorporates cross-attention between the question and passage tokens while forcing the model to explain every token in the input question. UPR is, therefore, more expressive than using dense retrievers alone, even if both methods fundamentally build on top of the same (or very similar) pre-trained models.

More specifically, we estimate $p(\boldsymbol{z}_i \mid \boldsymbol{q})$ by computing the likelihood of *question generation* conditioned on the passage, *i.e.*, the quantity $p(\boldsymbol{q} \mid \boldsymbol{z}_i)$. This also naturally emerges when applying Bayes' rule to $p(\boldsymbol{z}_i \mid \boldsymbol{q})$ as

$$\log p(\boldsymbol{z}_i \mid \boldsymbol{q}) = \log p(\boldsymbol{q} \mid \boldsymbol{z}_i) + \log p(\boldsymbol{z}_i) + c \,,$$

where $p(\boldsymbol{z}_i)$ is the prior on the retrieved passage and c is a common constant for all $\boldsymbol{z}_i$.

As a simplifying assumption, we assume that the passage prior $\log p(\boldsymbol{z}_i)$ is uniform, and can be ignored for re-ranking. With this, the above expression reduces to

$$\log p(\boldsymbol{z}_i \mid \boldsymbol{q}) \propto \log p(\boldsymbol{q} \mid \boldsymbol{z}_i), \; \forall \boldsymbol{z}_i \in \mathcal{Z} \,.$$

We estimate the quantity $\log p(\boldsymbol{q} \mid \boldsymbol{z}_i)$ using a pre-trained language model (PLM) by computing the average log-likelihood of the question tokens conditioned on the passage:

$$\log p(\boldsymbol{q} \mid \boldsymbol{z}_i) = \frac{1}{|\boldsymbol{q}|} \sum_t \log p(q_t \mid \boldsymbol{q}_{<t}, \boldsymbol{z}_i; \Theta) \,.$$

where Θ denotes the parameters of the PLM and $|\boldsymbol{q}|$ denotes the number of question tokens. We apply the PLM in a zero-shot fashion with no finetuning by simply appending the natural language instruction "*Please write a question based on this passage*" to the passage tokens as shown in Figure 5.2.

The initial passage ordering is then sorted based on $\log p(\boldsymbol{q} \mid \boldsymbol{z})$. This enables us to re-rank

the passages by just performing inference using off-the-shelf language models avoiding the need to label question-passage pairs for finetuning. Because the question generation model is applied zero-shot, this overall approach can be applied to improve the retrieval accuracy of any test collection, with no dataset-specific models or tuning data.

5.2 Experimental Setup

In this section, we describe the datasets, unsupervised and supervised retrievers, and language models used for our passage re-ranking experiments.

5.2.1 Open-Domain QA Datasets

Following previous work on passage retrieval, we use the popular datasets of SQuAD-Open (Rajpurkar et al., 2016), TriviaQA (Joshi et al., 2017), Natural Questions (NQ-Open; Kwiatkowski et al., 2019), and WebQuestions (WebQ; Berant et al., 2013). Their statistics are presented in Table 5.1. We refer the reader to §3.3.1 for more details on TriviaQA and NQ-Open and to §4.2.1 for WebQ. We briefly describe the SQuAD-Open dataset as follows.

SQuAD-Open: This corpus was primarily annotated for the reading comprehension task. Crowd-workers were assigned Wikipedia articles to frame questions from its paragraphs along with marking their answer text (Rajpurkar et al., 2016). Overall, SQuAD-Open contains more than 100,000 questions derived from 536 articles and was the largest dataset of its kind when it was released. For our retrieval experiments, we just consider the questions and their answers and ignore the associated paragraphs.

Evidence dataset $\mathcal{D}$: We use the preprocessed English Wikipedia dump from December 2018 as released by Karpukhin et al. (2020) as our evidence dataset. Each Wikipedia article is split into non-overlapping 100 word passages. There are over 21 million total passages.

Dataset	Train	Filtered Train	Dev	Test
WebQ	3,417	2,474	361	2,032
SQuAD-Open	78,713	70,096	8,886	10,570
TriviaQA	78,785	60,413	8,837	11,313
NQ-Open	79,168	58,880	8,757	3,610

Table 5.1: Number of question-answer pairs in QA datasets. The training set is used for open-domain QA experiments (§5.4). The filtered train version is used for supervised retriever training. The filtered set ignores those examples where the passages (or documents) retrieved from the evidence using BM25 does not contain the reference answer or align with the ground-truth passage.

5.2.2 Keyword-centric Datasets

To examine the robustness of UPR to keyword-centric datasets, we experiment with test collections where dense retrievers struggle and when the questions are from different domains.

- **Entity Questions** contains 22K short questions about named entities based on facts from Wikipedia. Previous work on this dataset has shown that dense retrievers struggle to retrieve relevant passages while sparse approaches like BM25 are more successful (Sciavolino et al., 2021).

- **BEIR** benchmark is a test suite for benchmarking retrieval algorithms and consists of multiple datasets, where each dataset consists of test set queries, evidence documents, and relevance document annotations (Thakur et al., 2021). These datasets contain different kinds of retrieval tasks like fact-checking, question answering, etc. and span diverse domains including news, technical, and Wikipedia making it a challenging benchmark.

5.2.3 Retrievers

In our re-ranking experiments, we retrieve passages from both unsupervised and supervised retrievers, as detailed below.

Unsupervised Retrievers

- **BM25** ranks based on the term-frequency and inverse document frequency of the keywords present in the question and passage (Robertson and Zaragoza, 2009). Prior work has shown that BM25 is a strong baseline for the datasets we consider (Ma et al., 2021b). We refer the readers to §2.5.2 for more details.

- **MSS** is a pre-training task to train dense retrievers by predicting masked salient spans like named entities with the help of a language model (Sachan et al., 2021a). MSS pre-training has also been shown to improve supervised retrieval performance. We refer the readers to §3.2.1 for further details.

- **Contriever** performs unsupervised training of the retriever by maintaining a very large pool of negative passages during training (Izacard et al., 2022). The passage embeddings from the previous training batches are cached and are re-used during contrastive training. In order to prevent a rapid change in the passage encoder weights, they are updated using exponential moving average of the previous weights which is also known as momentum contrastive training (He et al., 2020). Training with a large pool of negative examples has shown to improve the zero-shot ability of retrievers.

Supervised Retrievers

- **DPR** uses human-annotated question-document pairs and hard-negative examples to train a supervised dense retriever (Karpukhin et al., 2020). We refer the reader to §3.2.2 for further details on DPR training.

- **MSS-DPR** further improves DPR performance by first pre-training the dense retriever using MSS followed by DPR-style supervised finetuning (§3.5.2; Sachan et al., 2021a).

5.2.4 Pre-trained Language Models (PLMs)

To assess the relative strengths and weaknesses of pre-trained models in re-ranking, we experiment with a range of PLMs for computing our re-ranking relevance scores. These PLMs vary along multiple axes: architecture (encoder-decoder or decoder-only models), training style (prefix language modeling, autoregressive training, or instruction-tuning), and sizes.

- **T5 series** consists of encoder and decoder transformers pre-trained by denoising input text sequences (§2.4.3). We experiment with the T5 model (Raffel et al., 2020), its language model adapted version (T5-lm-adapt; Lester et al., 2021), and the T0 language model (Sanh et al., 2022). T0 was trained by finetuning T5-lm-adapt with multiple tasks defined by instructions. For more details on instruction-tuning, please refer to §2.4.3. Unless specified otherwise, we use the "xl" configuration that contain 3B parameters.

- **GPT** consists of a transformer decoder trained with the autoregressive language modeling objective (§2.4.3; Radford et al., 2018). We use the GPT-neo model with 2.7B parameters (Black et al., 2021).

5.2.5 Implementation Details

We run all the experiments on a cluster with V100-32GB GPUs. We use PyTorch (Paszke et al., 2019) to implement the UPR approach and relevant baselines. To get the top-K retrieved passages, we use the open-source implementations of the retrievers and their checkpoints. For BM25, we use the pre-computed top-K passages outputs from the pyserini toolkit (Lin et al., 2021a).[2] For MSS, DPR, and MSS-DPR retrievers, we use the open-source implementations from Sachan et al. (2021b).[3] For Contriever and PLMs, we use their open-source checkpoints (Wolf et al., 2020).

For the dense retriever experiments, we use the *base configuration*, which consists of 12 attention heads, 12 layers, and 768 model dimensions. To experiment with supervised retrievers, we

[2] https://github.com/castorini/pyserini/blob/master/docs/experiments-dpr.md
[3] https://github.com/DevSinghSachan/emdr2

Retriever	SQuAD-Open		TriviaQA		NQ-Open		WebQ	
	Top-20	Top-100	Top-20	Top-100	Top-20	Top-100	Top-20	Top-100
Unsupervised Retrievers								
MSS	51.3	68.4	67.2	79.1	60.0	75.6	49.2	68.4
MSS + UPR	75.7	80.8	81.3	85.0	77.3	81.5	71.8	80.4
BM25	71.1	81.8	76.4	83.2	62.9	78.3	62.4	75.5
BM25 + UPR	83.6	87.4	83.0	86.4	78.6	85.2	72.9	81.4
Contriever	63.4	78.2	73.9	82.9	67.9	80.6	65.7	80.1
Contriever + UPR	81.3	85.6	82.8	86.4	84.7	87.0	75.7	83.5
Supervised Retrievers								
DPR	59.4	74.5	79.8	85.1	79.2	85.7	74.6	81.6
DPR + UPR	80.7	85.4	84.3	87.2	83.4	88.6	77.7	84.1
MSS-DPR	73.1	84.5	81.9	86.6	81.4	88.1	76.9	84.6
MSS-DPR + UPR	**85.2**	**89.4**	**84.8**	**88.0**	**83.9**	**89.4**	**77.2**	**85.2**
E2E Supervised	-	-	84.1	87.8	84.8	89.8	79.1	85.2

Table 5.2: Top-{20, 100} retrieval accuracy on the test set of datasets before and after UPR re-ranking of the top-1000 retrieved passages with the T0-3B model. Best results of the unsupervised retriever are underlined while those of the supervised retriever are highlighted in bold. For reference, we also include the state-of-the art supervised results in the last row, which is obtained from end-to-end or joint training of the retriever and language model using question-answer pairs (Sachan et al., 2021a,b).

train DPR and MSS-DPR for 3 epochs on SQuAD-Open, 40 epochs on NQ-Open and TriviaQA, and 20 epochs on WebQ.[4] We train with Adam optimizer (Kingma and Ba, 2014), a batch size of 128, 1 hard negative example for each positive pair, a learning rate of 2e-5 with a linear decay, and a weight decay of 0.1. Model training was performed on 16 GPUs.

5.3 Results: Passage Retrieval

We evaluate the performance of our proposed Unsupervised Passage Re-ranker (UPR), conduct ablations to better understand the approach, evaluate robustness on challenging test collections, and discuss run-time efficiency.

Our goal is to improve the rankings of top-{20, 100} passages. Hence, in the first stage, a

[4]In contrast to previous work on SQuAD-Open, we train DPR and MSS-DPR for 3 epochs to prevent overfitting.

larger candidate list is fetched by retrieving the top-1000 passages. Then, in the second stage, these passages are re-ranked with the T0-3B PLM unless specified otherwise. To evaluate UPR performance, we compute the conventional *top-K retrieval accuracy* metric. It is defined as the fraction of questions for which at least one passage within the top-K passages contains a span that matches a reference answer to the question.

5.3.1 Main Task

We experiment with the four datasets and five retrievers as introduced in §5.2.1 and §5.2.3, respectively, and perform re-ranking with the T0-3B model. Table 5.2 reports the top-20 and top-100 retrieval accuracy before and after re-ranking. UPR provides consistent improvements across all the retrievers and datasets, improving unsupervised models by 6%-18% absolute and supervised models by up to 12% in top-20 accuracy.

Re-ranked Contriever outperforms DPR by an average of 7% in the top-20 and 4% in the top-100 when considering all the datasets. This shows that *a fully unsupervised pipeline of a retriever and re-ranker can outperform strong supervised models like DPR*. Sparse representations still remain competitive, with BM25 outperforming Contriever and MSS on SQuAD-Open re-ranking.

We also see that re-ranked MSS-DPR comes close to or matches the performance of state-of-the-art supervised retrievers (last row in Table 5.2). Because these supervised models are based on end-to-end training of the retriever and language model, they are memory-intensive and too expensive to train for very large models. As such, *UPR offers a viable alternative to expensive joint training*.

Intuition behind the performance gains obtained by UPR: The question generation step in the re-ranker involves expressive cross-attention with the passage tokens. As a result, each question token attends to all the passage tokens in each decoder layer before predicting the next question token. This results in an accurate estimation of the relevance (or log-likelihood) scores than the

original retriever scores, thus leading to an improved retrieval accuracy after re-ranking. This reasoning is further corroborated by our analysis presented next, where we present several examples where UPR improves over the incorrect BM25 retrievals.

Qualitative analysis: In Table 5.3, we include some examples of questions and their BM25 retrieved and UPR re-ranked top-1 passages. While BM25 retrieves passages with high lexical overlap, UPR owing to its cross-attention mechanism is able to better understand the relationships between tokens in the question and passage and thus leads to an improvement in passage rankings over the first-stage retriever. In the last example, we note that although the BM25 retrieved passage contains the ground-truth answer, it should be considered a false positive result. On the other hand, UPR leads to the correctly ranked passage but the exact match evaluation metric marks it as incorrect as it does not match the full ground-truth answer.

5.3.2 Ablation Studies

Importance of Question Generation

To understand the importance of re-ranking based on question generation $p(\boldsymbol{q} \mid \boldsymbol{z})$, we compare it with another unsupervised approach where re-ranking is based on passage generation conditioned on the question $p(\boldsymbol{z} \mid \boldsymbol{q})$. This quantity can be estimated by computing the average log-likelihood of generating the passage tokens using PLM and teacher-forcing as

$$\log p(\boldsymbol{z} \mid \boldsymbol{q}; \Theta) = \frac{1}{|\boldsymbol{z}|} \sum_{t} \log p(z_t \mid \boldsymbol{z}_{<t}, \boldsymbol{q}; \Theta) \ ,$$

where Θ denotes the parameters of the PLM and $|\boldsymbol{z}|$ denotes the number of passage tokens.

For this analysis, we work with the NQ-Open development set and obtain the union of top-1000 passages from the BM25 and MSS retrievers. These passages are re-ranked with two PLMs: T0-3B and GPT-2.7B. Our results in Figure 5.3 demonstrate that question generation obtains substantial improvements over the BM25 and MSS retrievers, highlighting its usefulness in passage re-ranking. On the other hand, re-ranking based on passage generation leads to a drop in retrieval

Question / Answer	BM25	BM25 + UPR
Question: when did beauty and the beast come out on dvd? **Answer**: June 6 , 2017	**Title**: Beauty and the Beast: The Enchanted Christmas **Passage**: these traits were incorporated into Forte, the pipe organ, who did not want the Beast to become human again. Unlike the other characters, Forte was animated entirely by computers. The film was first released on VHS on November 11, 1997. A bare-bones DVD was released on October 13, 1998. Both editions were quickly taken out of print and the film remained unavailable until Disney released the Special Edition DVD and VHS on November 12, 2002, just after the studio released the original filmś Special Edition DVD release. The new DVD featured a remake music video of the song "As Long'	**Title**: Beauty and the Beast (2017 film) **Passage**: Golden Globe Awards. Disney released the final trailer on January 30, 2017. "Beauty and the Beast" was released on Blu-ray, DVD and Digital HD on **June 6, 2017**. The film debuted at No. 1 on the NPD VideoScan overall disc sales chart, with all other titles in the top 20, collectively, selling only 40% as many units as "Beauty and the Beast". The movie regained the top spot on the national home video sales charts during its third week of release. The movie became available on Netflix on September 19, 2017. "Beauty and the Beast" grossed $504 million in the
Question: who sings i 'm back in the saddle again? **Answer**: Gene Autry	**Title**: Back in the Saddle (film) **Passage**: remember her as "an interesting actress" who had talent, but "seemed doomed to work in nothing but B-westerns." Remembering her work on "Back in the Saddle", Wells would later recall, "Actually, I didnt́ have much to do with Gene in the film, I had more scenes with Edward Norris." Regarding the singing sequences, she remembered, "Usually I was dubbed but, occasionally, if it wasnt́ something too difficult, I was allowed to do it. They prerecord the songs, then a huge machine comes onto the stage and you lip-sync to the recording." "Back in the Saddle" was filmed January 21 to	**Title**: Back in the Saddle Again **Passage**: "Gene Autryś Melody Ranch" which premiered over the CBS Radio Network on January 7, 1940 where the show ran until 1956. The song also became the title song for the Autry film "Back in the Saddle" (Republic Pictures, March 14, 1941). **Gene Autry** recorded "Back in the Saddle Again" for the first time on April 18, 1939 in Los Angeles for Columbia Record Corporation, matrix number LA 1865, which was originally issued on Vocalion 05080. LA 1865 also issued on the Conqueror, OKeh, and Columbia labels. Early Vocalion and Conqueror labels say "BACK TO THE SADDLE". Conqueror was a private.
Question: who won the big 10 football championship in 2016? **Answer**: Penn State Nittany Lions	**Title**: 2016 Big Ten Football Championship Game **Passage**: 2016 Big Ten Football Championship Game The 2016 Big Ten Football Championship Game was played December 3, 2016 at Lucas Oil Stadium in Indianapolis, Indiana. It was the sixth annual Big Ten Football Championship Game to determine the 2016 champion of the Big Ten Conference. The 2016 Big Ten Championship Game pitted the Wisconsin Badgers, champions of the West Division, who made its fourth appearance in six years in the conference title game, against the East Division champion **Penn State Nittany Lions**, who made their first-ever appearance in the conference championship game. Penn State and Ohio State had identical 8–1	**Title**: 2016 Big Ten Conference football season **Passage**: since the conference instituted divisions. Wisconsin won the West Division for the fourth time in the six years the division had existed. In the Big Ten Championship held on December 3, 2016 at Lucas Oil Stadium in Indianapolis, Indiana, ***Penn State*** defeated Wisconsin 38–31 to win the Big Ten. Several Big Ten teams changed head coaches in 2016. Tracy Claeys at Minnesota had the "interim" tag removed from his title and served as the permanent head coach. D. J. Durkin was the new head coach at Maryland taking over for Randy Edsall after having spent the previous year as the

Table 5.3: Selected examples from the NQ-Open development set of the top-1 retrieved passage from BM25 and the top passage obtained by UPR re-ranking of 1000 passages. If the answer exists in the passage it is highlighted in bold. UPR leverages powerful cross-attention between the question and passage tokens and hence is able to obtain improved passage rankings.

accuracy in comparison to the baseline retrievers, empirically confirming that this approach does not work well in practice.

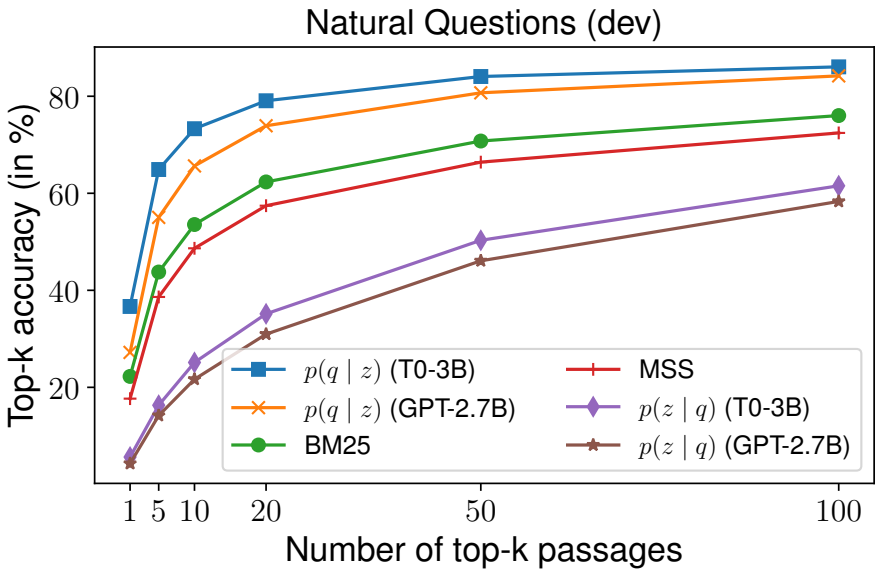

Figure 5.3: Comparison of two passage re-ranking approaches on the NQ-Open development set: (1) when generating question tokens conditioned on the passage $p(\boldsymbol{q} \mid \boldsymbol{z})$, and (2) when generating passage tokens conditioned on the question $p(\boldsymbol{z} \mid \boldsymbol{q})$. Results highlight the usefulness of question generation in UPR for re-ranking.

Impact of Pre-trained Language Models

To understand how much the choice of PLM contributes to top-K accuracy, we compare the performance of T5 (3B), T5-lm-adapt (different sizes), T0-{3B, 11B}, and GPT-neo (2.7 B) (as introduced in §5.2.4) on the NQ-Open development set. We obtain the union of top-1000 passages retrieved from BM25 and MSS and then re-rank them with UPR. Results in Table 5.4 reflect that all the PLMs obtain significant improvements over the baseline retrievers, with the T0 models achieving the best results. Scaling up the PLM size, especially the T5-lm-adapt models leads to consistent performance improvements.

When comparing across PLMs, we see that the performance of T5 suffers especially on top-{1, 5} accuracy levels. This might be because it was trained to predict corrupted spans, which is not ideal for text generation. On the other hand, autoregressive PLMs such as GPT-neo and T5-lm-adapt tend to be better re-rankers. Furthermore, T0 obtains large improvements on top-{1, 5, 20}, demonstrating that finetuning with instructions on unrelated tasks is also beneficial for re-ranking.

Retriever / Re-Ranker	NQ-Open (dev) Top-1	Top-5	Top-20	Top-100
BM25	22.3	43.8	62.3	76.0
MSS	17.7	38.6	57.4	72.4
T5 (3B)	22.0	50.5	71.4	84.0
GPT-neo (2.7B)	27.2	55.0	73.9	84.2
GPT-j (6B)	29.8	59.5	76.8	85.6
T5-lm-adapt (250M)	23.9	51.4	70.7	83.1
T5-lm-adapt (800M)	29.1	57.5	75.1	84.8
T5-lm-adapt (3B)	29.7	59.9	76.9	85.6
T5-lm-adapt (11B)	32.1	62.3	78.5	85.8
T0-3B	36.7	**64.9**	**79.1**	**86.1**
T0-11B	**37.4**	**64.9**	**79.1**	86.0

Table 5.4: Comparison of different pre-trained language models (PLMs) as re-rankers on the NQ-Open development set. We re-rank the union of BM25 + MSS retrieved passages with UPR. Results demonstrate that T0 PLMs achieves the best top-K accuracy among the compared PLMs.

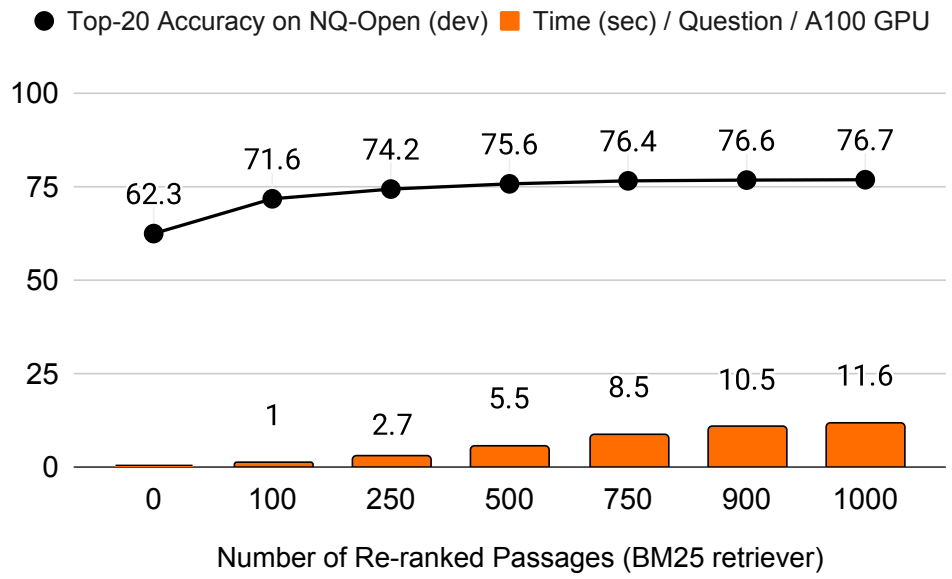

Figure 5.4: Effect of the number of passage candidates on top-20 accuracy and latency when re-ranked with T0-3B PLM. Evaluation is done on the NQ-Open development set using BM25 retrieved passages.

Passage Candidate Size vs Latency

We study the effect of the number of passage candidates to be re-ranked on the retrieval performance along with the time taken. For this, we consider the NQ-Open development set, re-rank

Retriever	Instruction Prompt	NQ-Open (dev)			
		Top-1	Top-5	Top-20	Top-100
BM25		22.3	43.8	62.3	76.0
+ UPR	''	28.3	56.1	73.2	82.4
+ UPR	"*Score the following question based on this passage.*"	35.3	62.6	76.4	83.0
+ UPR	"*A possible question based on this passage is.*"	33.8	61.6	76.2	**83.1**
+ UPR	"*This is a relevant document for the following question.*"	33.7	61.8	76.0	83.0
+ UPR	"*Please write a question based on this passage.*"	**36.1**	**62.8**	**76.8**	**83.1**

Table 5.5: Comparison of different instruction prompts when applied to the UPR framework and evaluated on the NQ-Open development set. Results highlight that UPR works better with simple instructions. Best results are highlighted in bold.

up to top-1000 passages obtained from BM25, and use top-20 accuracy as the evaluation criteria. Results in Figure 5.4 illustrate that a larger pool of passage candidates indeed helps to improve the performance. However, the gains tend to plateau as the number of passages is increased.

With more passages, the latency in re-ranking per question linearly increases reflecting the trade-off between accuracy and throughput. The higher latency can be partly alleviated with approaches like weight quantization, efficient implementations of the transformer kernel, model distillation, caching passage embeddings, and using data parallelism. However, we leave these explorations to future work.

Instruction Prompt Selection

We cross-validate using several prompts formulated as natural language instructions to aid in question generation. We re-rank the top-1000 passages of the NQ-Open development set obtained from BM25 using different instructions including the case with no instruction. Results in Table 5.5 reveal that when prompted via instructions, PLMs perform better than the case when not given any instructions. We also note that simple but effective instructions can lead to a higher top-1 accuracy. Due to its better accuracy, we have used the instruction "*Please write a question based on this passage*" for all the experiments in this chapter.

Retriever / Re-Ranker	NQ-Open (dev)			
	Top-1	Top-5	Top-20	Top-100
BM25	22.3	43.8	62.3	76.0
UPR (T0-3B)	36.1	62.8	76.8	83.1
monoT5 (250M)	39.1	62.4	75.6	82.6
monoT5 (800M)	43.5	66.1	77.5	83.3
monoT5 (3B)	**44.2**	**68.3**	**78.7**	**83.7**

Table 5.6: Zero-shot supervised transfer results on the NQ-Open development set. We use monoT5 (Nogueira et al., 2020) checkpoints of different sizes finetuned on the MS MARCO dataset to re-rank the top-1000 passages retrieved by BM25. We also include the results of UPR for comparison.

5.3.3 Zero-shot Supervised Transfer

To gain a better understanding of the relative strengths of UPR and supervised (or finetuned) re-rankers, we perform zero-shot supervised transfer experiments and compare the results with UPR. We adopt the approach of Nogueira et al. (2020), henceforth referred to as *monoT5*, who finetune the T5 PLMs on the MS MARCO (Bajaj et al., 2016) passage ranking dataset. To train, question and passage tokens are concatenated and fed to the T5 encoder. The decoder attends to the encoded sequence and the T5 PLM is finetuned to maximize the likelihood of the "true" label. To re-rank the passages during inference, the log-likelihood score of the "true" label is used as the relevance score.

We use the open-source checkpoints of monoT5 to re-rank the top-1000 passages retrieved by BM25 and report results on the NQ-Open development set (Table 5.6).[5] Interestingly, we see that supervised transfer improves the top-1 and top-5 retrieval accuracy by a large margin over UPR. However, when the set of retrieved passages increases, such as 20-100, the results of UPR come close to or match the results of monoT5. As end tasks such as open-domain question answering rely on a larger set of passages to achieve good results (as demonstrated in §5.4), *this highlights the importance of UPR over supervised models as it does not require collecting annotated data for finetuning*.

[5] https://github.com/castorini/pygaggle

Retriever	Entity Questions	
	Top-20	Top-100
Baselines		
MSS	51.2	66.3
DPR	51.1	63.8
MSS-DPR	60.6	73.7
Contriever	63.0	75.1
BM25	71.2	79.8
SPAR (Chen et al., 2022)	74.0	82.0
After Re-ranking with UPR (T0-3B PLM)		
MSS	71.3	76.7
DPR	65.4	72.0
MSS-DPR	73.9	80.1
Contriever	76.0	81.6
BM25	79.3	83.9
BM25 + Contriever	**80.2**	**85.4**

Table 5.7: Top-{20, 100} retrieval accuracy on the Entity Questions dataset before and after re-ranking. Following the original paper, we report macro-average scores.

5.3.4 Evaluation on Keyword-centric Datasets

Entity Questions

We re-rank the top-1000 passages from every retriever with UPR. As the training set is not provided in this dataset, we use the checkpoints of DPR and MSS-DPR trained on NQ-Open. Results are presented in Table 5.7. We observe that re-ranking leads to a gain of 8-20% absolute in top-20 accuracy and 4-10% in top-100 accuracy, with BM25 achieving the best results after re-ranking. It also substantially narrows the gap between BM25 and dense retrievers. Re-ranking the union of BM25 and Contriever outputs outperforms the current best results by 6% and 3% in the top-20 and top-100, respectively.

We also note that multi-vector approaches specially tailored towards the robust representation of textual entities (de Jong et al., 2022) are promising alternatives to dual-encoder retrievers as they offer improved retrieval accuracy although at the expense of increased memory and compute requirements. However, we defer the application of UPR to these retrievers as a part of future

Retriever	BEIR	
	nDCG@10	Recall@100
Baselines		
BERT (Devlin et al., 2019)	9.3	20.1
SimCSE (Gao et al., 2021)	27.4	48.1
REALM (Guu et al., 2020)	25.8	46.5
Contriever	36.0	60.1
BM25	41.6	63.6
After Re-ranking with UPR (T0-3B PLM)		
Contriever	44.6	66.3
BM25	**44.9**	**68.0**

Table 5.8: Macro-average nDCG@10 and Recall@100 scores on the BEIR benchmark. Performance numbers of the baseline models are from Izacard et al. (2022).

work.

BEIR Benchmark

We re-rank the top-1000 documents from the BM25 and Contriever retrievers with the T0-3B PLM and evaluate performance using nDCG@10 and Recall@100 metrics. Following convention, we report the macro average scores in Table 5.8 and compare them with previous baselines. The results on the individual datasets of BEIR are presented in Table 5.9. On both metrics, the average scores of BM25 are much higher than those of Contriever. After re-ranking, the BM25 retriever obtains improvements on 12 out of 15 datasets while Contriever obtains improvements on 13 out of 15 datasets. On average, nDCG@10 improves by 3-8% and Recall@100 improves by 5-6%. The performance gap between BM25 and Contriever also narrows down after re-ranking.

There is a considerable variation in the relative performance gains across the datasets in part owing to the diversity in queries and evidence documents. When re-ranking BM25 outputs, the highest relative gains are obtained on datasets containing information-seeking questions such as FIQA-2018, NQ, and MS MARCO. Similarly, for Contriever, the relative gains are much higher for Trec-Covid, NQ, and HotpotQA, where the queries are questions. On other datasets, the relative gains from re-ranking are moderate to little.

Dataset	nDCG@10				Recall@100			
	BM25		Contriever		BM25		Contriever	
	original	re-ranked	original	re-ranked	original	re-ranked	original	re-ranked
Scifact	66.5	70.3	64.9	69.6	90.8	94.2	92.6	94.3
Scidocs	15.8	17.0	14.9	17.3	35.6	39.0	36.0	39.0
Nfcorpus	32.5	34.8	31.7	33.3	25.0	28.0	29.0	31.3
FIQA-2018	23.6	44.4	24.5	45.0	53.9	67.7	56.2	72.8
Trec-covid	65.5	68.8	27.4	60.4	49.8	54.8	17.2	36.7
Touche-2020	36.8	20.6	19.3	21.3	53.8	45.7	22.5	42.4
NQ	32.9	45.4	25.4	44.2	76.0	87.7	77.1	88.4
MS MARCO	22.8	30.2	20.6	30.7	65.8	76.9	67.2	79.1
HotpotQA	60.3	73.3	48.1	72.2	74.0	82.5	70.4	80.8
ArguAna	31.5	37.2	37.9	50.3	94.2	98.2	90.1	97.5
CQADupStack	29.9	41.6	28.4	41.7	60.6	70.1	61.4	71.3
Quora	78.9	83.1	83.5	82.8	97.3	98.8	98.7	98.9
DBpedia	31.3	35.4	29.2	33.8	39.8	53.3	45.3	47.8
Fever	75.3	59.1	68.2	57.3	93.1	84.2	93.6	83.1
Climate-Fever	21.3	11.7	15.5	9.5	43.6	39.2	44.1	31.3
Average	41.6	**44.9**	36.0	44.6	63.6	**68.0**	60.1	66.3

Table 5.9: UPR re-ranking results on the BEIR benchmark (Thakur et al., 2021). Upon re-ranking the top-1000 documents with the T0-3B PLM, on average, the performance of both BM25 and Contriever improve on the NDCG@10 and Recall@100 metrics. We also observe a drop in scores on some datasets which is highlighted in red.

For both these retrievers, we also observe a drop in performance on the fact-verification datasets of Fever and Climate-fever (results highlighted in red color in Table 5.9). In addition, re-ranking BM25 also results in a drop in performance on the Touche-2020 dataset. We note that in these datasets, the queries are statements such as claims, which presents a challenging setting for re-rankers. We anticipate that results can be improved on these datasets by experimenting with different prompts such that they better suit the retrieval task and by cross-validating with the candidate pool size to be re-ranked.

5.4 Results: Question Answering

Finally, we show that UPR improves the performance of full open-domain QA systems.

5.4.1 Method

An open-domain QA system consists of a retriever and a reader component. The reader attends to the retrieved passages to produce a final answer to the question. We use the Fusion-in-Decoder (FiD; Izacard and Grave, 2021b) model as the reader. In FiD, each retrieved passage is concatenated with the question and is then passed as an input to the T5 encoder (Raffel et al., 2020). Then the encoded representations for all the passages are concatenated which the T5 decoder leverages for cross-attention.

We train the FiD reader using standard negative log-likelihood loss and teacher-forcing to generate an answer autoregressively. To understand the effect of UPR on answer generation, we then do inference with the previously trained reader and the re-ranked passages for each question.

5.4.2 Experiments and Results

For our experiments, we train the FiD base and large models using the top-100 retrieved passages from MSS, DPR, and MSS-DPR retrievers.[6] We conduct experiments on SQuAD-Open, TriviaQA, and NQ-Open datasets and train FiD models with a batch size of 64 using 64 GPUs. We use Adam optimizer, a learning rate of 2e-5 with a linear decay, a weight decay of 0.1, gradient clipping with a maximum value of 1.0, and train for 3 epochs on SQuAD-Open, 10 epochs on NQ-Open and TriviaQA.

We re-rank the top-1000 passages with UPR using the T0-3B PLM and then perform inference by feeding the top-100 re-ranked passages to FiD models. During inference, an answer is generated using greedy decoding. In contrast to previous work that makes use of the 2016 Wikipedia dump as evidence for SQuAD-Open, we use the same set of evidence passages for all the datasets. As our evidence set is larger and newer, some questions may be unanswerable, which renders a fair comparison difficult. However, to alleviate dataset-specific design choices, we adopt a common experimental setup.

[6]Base and large configurations of FiD are based on corresponding base and large T5 models.

Model	top-K	SQuAD-Open		TriviaQA		NQ-Open	
		dev	test	dev	test	dev	test
Baselines							
BM25 + BERT (Lee et al., 2019)	5	28.1	33.2	47.2	47.1	24.8	26.5
ORQA (Lee et al., 2019)	5	26.5	20.0	45.1	45.0	31.3	33.3
REALM (Guu et al., 2020)	5	-	-	-	-	38.2	40.4
DPR (Karpukhin et al., 2020)	25	-	38.1	-	56.8	-	41.5
RAG-Sequence (Lewis et al., 2020c)	50	-	-	55.8	56.8	44.0	44.5
Individual Top-K (large) (Sachan et al., 2021a)	-	-	-	-	59.6	-	48.1
Joint Top-K (large) (Sachan et al., 2021a)	50	-	-	-	68.3	-	51.4
FiD-base (Izacard and Grave, 2021b)	100	-	53.4	-	65.0	-	48.2
FiD-large (Izacard and Grave, 2021b)	100	-	56.7	-	67.6	-	51.4
FiD-KD-base (Izacard and Grave, 2021a)	100	-	-	68.6	68.8	48.0	49.6
FiD-KD-large (Izacard and Grave, 2021a)	100	-	-	71.9	72.1	51.9	53.7
EMDR^2-base (Sachan et al., 2021b)	50	46.8	51.1	71.1	71.4	50.4	52.5
Our Implementation							
FiD-base (MSS retriever, T5 reader)	100	36.2	39.6	60.9	60.3	43.7	44.5
+ Inference with UPR re-ranked passages		43.7	50.1	68.5	68.9	45.8	47.3
FiD-base (DPR retriever, T5 reader)	100	48.8	45.8	67.9	68.5	49.4	50.8
+ Inference with UPR re-ranked passages		51.5	54.0	70.1	71.2	49.8	51.3
FiD-base (MSS-DPR retriever, T5 reader)	100	50.1	52.2	69.9	70.2	49.7	50.8
+ Inference with UPR re-ranked passages		51.9	55.6	71.5	71.8	49.9	51.5
FiD-large (MSS-DPR retriever, T5 reader)	100	51.9	54.4	71.5	71.6	**51.8**	53.6
+ Inference with UPR re-ranked passages		**53.1**	**58.1**	**72.7**	**73.2**	51.5	**54.5**

Table 5.10: Exact match scores for the open-domain QA task. We train one FiD model for each retriever as indicated and then perform inference with its respective re-ranked outputs. Top-K denotes the number of retrieved passages that are used by the reader to produce an answer. We report the baseline performance numbers from the respective papers. The best performing models are highlighted in bold.

Our results are presented in Table 5.10 where we report the exact match (EM) scores for evaluation. More accurate passages after re-ranking improve the performance of the pre-trained FiD models for all the retrievers. Performing inference on the FiD-large model with re-ranked MSS-DPR passages achieves new state-of-the-art results, outperforming the pre-trained FiD model by 1-3 EM points. Overall, this provides a simple approach for obtaining performance gains without the need to iteratively re-train (Izacard and Grave, 2021a) or perform computationally expensive end-to-end training (Sachan et al., 2021b).

5.5 Related Work

Our work is based on re-ranking passages for open-domain retrieval using pre-trained language models (PLMs) which we have covered in earlier sections. Here, we instead focus on covering previous work related to generative pre-training, query likelihood for document ranking, and open-domain QA.

Generative pre-training and instruction tuning: Recently, there has been an increased adoption of the generative pre-trained transformer (GPT) series of models by the NLP community (Radford et al., 2019). Among the interesting properties of GPT models is their ability to understand task instructions specified in natural language and then perform well on tasks in a zero-shot or few-shot manner (Brown et al., 2020; Smith et al., 2022). The zero-shot performance of GPT models further improves when finetuning them on multiple different tasks using task-specific instructions, which is also known as instruction-tuning (Sanh et al., 2022; Wei et al., 2022; Min et al., 2022).

Document ranking based on query likelihood: In information retrieval, an appealing approach to rank documents is by utilizing language models to compute relevance scores for a query (Ponte and Croft, 1998). Prior approaches estimated a count-based language model for each document that was used to compute query likelihood scores for ranking (Zhai and Lafferty, 2001). However, these approaches suffer from issues such as data sparsity. More recent approaches utilize PLMs such as GPT or T5 to compute query likelihood (Nogueira dos Santos et al., 2020). To improve ranking accuracy, they perform supervised finetuning using query-document pairs (Ju et al., 2021). Our work also utilizes PLMs, but instead, we leverage a larger instruction-tuned language model and apply them in a zero-shot manner without finetuning.

Open-domain QA: This task involves producing answers to information-seeking questions from large document collections. Typical approaches consist of retriever and reader networks, where the retriever identifies a small number of documents to aid the reader in producing answers (Chen

et al., 2017). To be scalable, retrievers are often modeled using dual-encoders (Lee et al., 2019) as multi-vector encoders (Zhou and Devlin, 2021) are not scalable. To further improve retrieval accuracy, re-rankers are employed (Nogueira et al., 2020). Given retrieved documents, a reader is then trained to generate a short answer to the question (Izacard and Grave, 2021b; Lewis et al., 2020c).

5.6 Discussion

In this chapter, we propose UPR, an approach to perform *unsupervised passage re-ranking* to improve open-domain retrieval. To re-rank, UPR first computes a relevance score for each retrieved passage by obtaining a question generation likelihood estimate conditioned on the passage using pre-trained language models (PLM). Then, to obtain the re-ranked list, the original ordering of the passages is permuted based on the maximum relevance scores. We summarize our key contributions and findings as follows:

- When re-ranking unsupervised retriever outputs, UPR obtains substantial gains in the range of 6%-18% points absolute in top-20 accuracy across four popular passage retrieval datasets. Interestingly, our work is the first to demonstrate that an unsupervised pipeline consisting of a retriever and UPR greatly outperforms strong supervised retrievers like DPR.

- Our extensive experiments reveal that UPR achieves consistent improvements in retrieval accuracy for both unsupervised and supervised retrievers. These results present UPR as a promising alternative as it just uses off-the-shelf language models and is zero-shot, *i.e.*, it does not require any finetuning or data annotation.

- After re-ranking BM25 outputs, UPR also obtains large performance gains of 8%-14% on datasets such as Entity Questions and SQuAD-Open where dense retrievers struggle. These results highlight that UPR also generalizes well to sparse retrievers.

- We conduct an in-depth analysis of several factors to understand their importance in UPR

such as the effect of PLM, candidate size of input passages, how to best estimate the relevance of a passage, etc. Our experiments provide critical insights into which settings provide the best gains for passage re-ranking.

- On the open-domain QA task, by just performing inference using re-ranked passages and a pre-trained reader model, we achieve up to 3 EM points improvement over the existing best results for the large configuration.

5.6.1 Limitations

In the previous sections, we have shown that UPR has been extremely effective in improving open-domain passage retrieval. In addition, UPR also does not require any language model finetuning. Although broadly applicable and flexible, UPR suffers from several challenges that we discuss as follows.

Computationally expensive: In order to achieve good performance, using a large PLM and a large number of candidate passages (for instance, top-K=1000) is desirable (§5.3.2). However, it is worth mentioning that large PLMs lead to a slower inference procedure than their smaller counterparts. Moreover, the inference latency is linearly proportional to the candidate size $\mathcal{O}(K)$, *i.e.*, more candidates increase the number of PLM forward passes thus overall requiring more computation. Both these factors contribute to UPR being computationally expensive.

Upper bound on re-ranking performance: Generally, as the number of candidate passages are increased, the re-ranking accuracy improves as well. This trend is also evident from Figure 5.4. However, a fundamental limitation of any re-ranker including UPR is that the maximum achievable performance is upper bounded by the accuracy of the set of candidate passages from the top-K list of the first-stage retriever. This is consequential because if the recall of the top-K passages (such as top-1000) from the first-stage retriever is low, then the re-ranker's performance will be bottlenecked by this recall.

Sensitivity to input query types: In this chapter, we have applied UPR to a variety of benchmark QA datasets and shown its effectiveness in improving retrieval accuracy. However, on specific retrieval tasks such as that of retrieving supporting documents for factual claims (Thorne et al., 2018), UPR has shown to struggle (§5.3.4) as re-ranking adversarially affects the original retrieval accuracy. Another task where UPR leads to subpar results is that argument retrieval (Bondarenko et al., 2020). This drop in performance post re-ranking is agnostic to the retriever type as UPR underperforms for both sparse and dense retrievers. These findings illustrate that UPR is still not a one-model-fits-all framework and more work is needed to devise robust re-ranking solutions for ad-hoc retrieval tasks.

Requires access to language model logits: Our formulation of UPR is based on estimating the relevance score of a passage by computing the question generation likelihood using teacher-forcing. To obtain the log-likelihood of question tokens, logits are needed at every step for all the tokens in the vocabulary. This necessitates either access to the language model weights or to services that can score the target text given the input text. On the contrary, UPR will not work with platforms that only provide the language model generation endpoint.

5.6.2 Follow-up Work

- **Promptagator:** In this work (Dai et al., 2023), the authors emphasize the importance of training a custom retriever for specific types of queries differing in their search intent such as for the diverse datasets included in the BEIR benchmark. Their proposed approach consists of prompting a large language model such as PaLM (Chowdhery et al., 2022) using few-shot query-passage pairs and the task instruction to synthesize new questions from the evidence passages. These generated question-passage pairs are then used to train task-adapted retrievers which are more accurate than generic pre-trained retrievers.

- **Pairwise ranking prompting (PRP):** Similar in spirit to UPR, this is also an unsupervised re-ranking approach using large language models. However, instead of estimating the rele-

vance of each query-passage pair separately, PRP prompts the language model to compare the two input passages and output the passage identifier that is more relevant (Qin et al., 2023). The positions of the two passages are swapped based on the language model output and this process is repeated recursively to obtain the re-ranked list. Such pairwise contrasting and reasoning using language models have been shown to improve passage rankings.

- **Code re-ranking:** The re-ranking idea proposed in this chapter has also been successfully applied to improve the quality of generated programs by re-ranking the output of code generation systems (Zhang et al., 2023). More specifically, first, given a natural language instruction such as a code description, multiple programs are sampled after prompting a language model that is trained to generate code (Chen et al., 2021). In the next step, the generated program is given as input to the language model to score the language instruction. Finally, the scores from both the previous steps are added to re-rank the programs from the first step.

5.6.3 Future Work

UPR presents several interesting directions for future work. First, its applications to other retrieval tasks such as improving source-code retrieval based on textual queries can be explored. Second, another promising direction would be to tune instructions according to the nature of the retrieval tasks. For instance, when retrieving similar sentences in the BEIR benchmark, variations of the instruction prompt used in UPR can be explored. Finally, it would also be interesting to investigate the extent to which specialized language models such as the ones finetuned to generate questions using passage-questions data would further help in improving retrieval.

Chapter 6

Questions Are All You Need to Train a Dense Passage Retriever

Dense passage retrieval methods (Karpukhin et al., 2020; Xiong et al., 2021), initialized with encoders such as BERT (Devlin et al., 2019) and trained using supervised contrastive losses (Oord et al., 2018), have surpassed the performance achieved by previously popular keyword-based approaches like BM25 (Robertson and Zaragoza, 2009). Such retrievers are core components in models for open-domain tasks, such as open-domain QA, where state-of-the-art methods typically require large supervised datasets with custom hard-negative mining and denoising of positive examples. In this chapter, we introduce the first unsupervised retriever training method, based on a new corpus-level autoencoding approach, that can match or surpass strong supervised performance levels with no labeled training data or task-specific losses.

We propose ART: *Autoencoding-based Retriever Training* which only assumes access to sets of unpaired questions and passages. Given an input question, ART first retrieves a small set of possible evidence passages. It then *reconstructs the original question* by attending to these passages (see Figure 6.1 for an overview). The key idea in ART is to consider the retrieved passages as a noisy representation of the original question and question reconstruction probability as a way of denoising that provides *soft-labels* for how likely each passage is to have been the correct result.

To bootstrap the training of a strong retriever, it is important to both have a strong initial retrieval model and to be able to compute reliable initial estimates of question reconstruction probability when conditioned on a (retrieved) passage. Although passage representations from pre-trained encoders are known to be reasonable retrieval baselines, it is less clear how to do zero-shot question generation. We use a generative pre-trained language model (PLM) and prompt it with the passage as input to generate the question tokens using teacher-forcing. As finetuning of the question-generation PLM is not needed, only the retrieval model, ART can use large PLMs and obtain accurate soft-label estimates of which passages are likely to be the highest quality.

The retriever is trained to penalize the divergence of a passage likelihood from its soft-label score. For example, if the question is "*Where is the bowling hall of fame located?*" as shown in Figure 6.1, then the training process will boost the retrieval likelihood of the passage "*Bowling Hall of Fame is located in Arlington, ...*" as it is relevant and would lead to a higher question reconstruction likelihood, while the likelihood of the passage "*Hall of Fame is a song by ...*" would be penalized as it is irrelevant. In this manner, the training process encourages correct retrieval results and vice-versa, leading to an iterative improvement in passage retrieval.

Comprehensive experiments on five benchmark QA datasets demonstrate the usefulness of our proposed training approach. By simply using questions from the training set, ART outperforms models like DPR by an average of 5 points absolute in top-20 accuracy and 4 points absolute in top-100 accuracy. We also train using all the questions contained in the Natural Questions (NQ) dataset (Kwiatkowski et al., 2019) and find that even with a mix of answerable and unanswerable questions, ART achieves strong generalization on out-of-distribution datasets. Our analysis further reveals that ART is highly sample efficient, outperforming BM25 and DPR with just 100 and 1000 questions, respectively, on the NQ-Open dataset, and that scaling up to larger retriever models consistently improves performance.

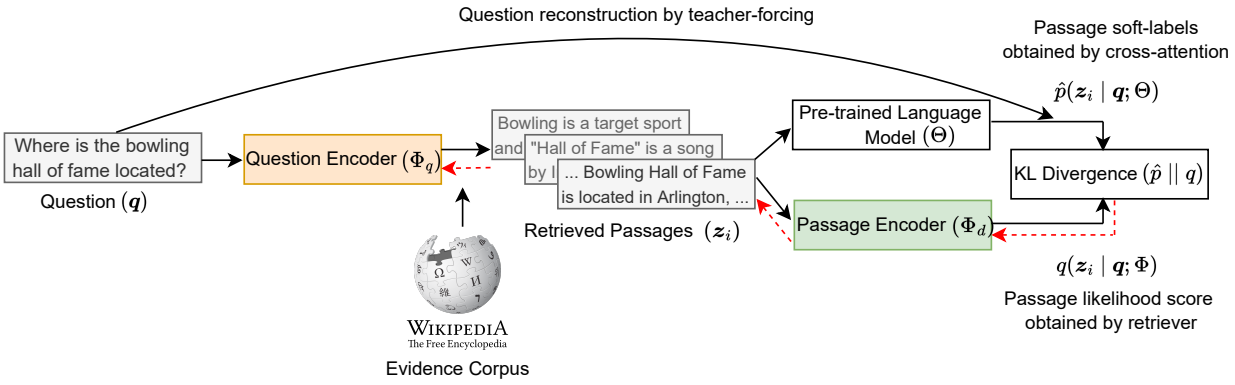

Figure 6.1: ART maximizes the retrieved passage likelihood computed from the dense retriever by considering the language model question reconstruction score conditioned on the passage as a *soft-label*. Colored blocks indicate trainable parameters. Red arrows show gradient flow during backpropagation.

6.1 Method

6.1.1 Problem Definition

We focus on open-domain retrieval, where given a question $\boldsymbol{q}$, the task is to select a small set of matching passages (*i.e.*, 20 or 100) from a large collection of evidence passages $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_M\}$. Our goal is to train a retriever in a *zero-shot manner*, *i.e.*, without using question-passage pairs, such that it retrieves relevant passages to answer the question. Our proposed approach consists of two core modeling components (§6.1.2, §6.1.3) and a novel training method (§6.1.4).

6.1.2 Dual-Encoder Retriever

For the retriever, we use the dual-encoder model (Bromley et al., 1994) which consists of two encoders, where

- one encoder computes the question embedding $f_q(\boldsymbol{q}; \Phi_q) : \mathcal{X} \mapsto \mathbb{R}^d$, and
- the other encoder computes the passage embedding $f_d(\boldsymbol{d}; \Phi_d) : \mathcal{X} \mapsto \mathbb{R}^d$.

Here, $\mathcal{X} = \mathbb{V}^n$ denotes the universal set of text sequences, $\mathbb{V}$ denotes the vocabulary consisting of

discrete tokens, and $\mathbb{R}^d$ denotes the (latent) embedding space.[1] We assume that both the question and passage embeddings lie in the same latent space. The *retrieval score* for a question-passage pair $(\boldsymbol{q}, \boldsymbol{d})$ is then defined as the inner product between their respective embeddings,

$$s(\boldsymbol{q}, \boldsymbol{d}_i; \Phi) = f_q(\boldsymbol{q}; \Phi_q) \cdot f_d(\boldsymbol{d}_i; \Phi_d), \ \forall \boldsymbol{d}_i \in \mathcal{D}, \tag{6.1}$$

where $\Phi = [\Phi_q, \Phi_d]$ denotes the retriever parameters. We select the top-K passages with maximum inner product scores and denote them as $\mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K\}$.[2]

We use the transformer network (Vaswani et al., 2017) with BERT tokenization (Devlin et al., 2019) to model both encoders. To obtain the question or passage embedding, we do a forward pass through the transformer and select the last layer hidden state corresponding to the `[CLS]` token. As the input passage representation, we use both the passage title and text separated by the `[SEP]` token.

6.1.3 Zero-shot Cross-Attention Scorer

We obtain an estimate of the *relevance score* for a question-(retrieved) passage pair $(\boldsymbol{q}, \boldsymbol{z})$ by using a pre-trained language model (PLM). In order to do this in a zero-shot manner, we use a large generative PLM to compute the likelihood score of a passage conditioned on the question $p(\boldsymbol{z} \mid \boldsymbol{q})$.

The quantity $p(\boldsymbol{z} \mid \boldsymbol{q})$ can be better approximated by the autoregressive generation of question tokens conditioned on the passage and teacher-forcing (Sachan et al., 2022). More formally, this can be written as

$$\log p(\boldsymbol{z} \mid \boldsymbol{q}; \Theta) = \log p(\boldsymbol{q} \mid \boldsymbol{z}; \Theta) + \log p(\boldsymbol{z}) + c \tag{6.2a}$$

$$\propto \frac{1}{|\boldsymbol{q}|} \sum_t \log p(q_t \mid \boldsymbol{q}_{<t}, \boldsymbol{z}; \Theta) \ , \tag{6.2b}$$

where Θ denotes the parameters of the PLM, c is a constant independent of the passage $\boldsymbol{z}$, and $|\boldsymbol{q}|$

[1] This material is analogous to the dense retrievers introduced in §3.1.1, and is included here for ease of reading with additional notation.

[2] As the selection operation requires performing inner-product with millions of passage embeddings, this can be efficiently performed on accelerators such as GPUs.

denotes the number of question tokens. Here, Eq. 6.2a follows from a simple application of Bayes' rule to $p(\boldsymbol{z} \mid \boldsymbol{q})$ and in Eq. 6.2b, we assume that the passage prior $p(\boldsymbol{z})$ is uniform for all $\boldsymbol{z} \in \mathcal{Z}$.

We hypothesize that estimating the relevance score using Eq. 6.2b would be accurate because it requires performing deep cross-attention involving all the question and passage tokens. In a large PLM, the cross-attention step is highly expressive, and in combination with teacher-forcing, requires the model to explain every token in the question resulting in a better estimation.

As the input passage representation, we concatenate the passage title and its text. In order to prompt the PLM for question generation, we follow Sachan et al. (2022) and append a simple natural language instruction "*Please write a question based on this passage.*" to the passage text.

6.1.4 Training Algorithm

For training the model, our only assumption is that a collection of questions ($\mathcal{T}$) and evidence passages ($\mathcal{D}$) are provided as input. During training, the weights of the retriever are updated while the PLM is not finetuned, *i.e.*, it is used in inference mode. Our training algorithm consists of five core steps. The first four steps are performed at every training iteration while the last step is performed every few hundred iterations. Figure 6.1 presents an illustration of our approach.

Step 1: Top-K passage retrieval: For fast retrieval, we pre-compute the evidence passage embeddings using the initial retriever parameters ($\hat{\Phi}_d$). Given a question $\boldsymbol{q}$, we compute its embedding using the current question encoder parameters (Φ_q) and then retrieve the top-K passages ($\mathcal{Z}$) according to Eq. 6.1. We then embed these top-K passages using the current passage encoder parameters (Φ_d) and compute *fresh* retriever scores as,

$$s(\boldsymbol{q}, \boldsymbol{z}_i) = f_q(\boldsymbol{q}; \Phi_q) \cdot f_d(\boldsymbol{z}_i; \Phi_d), \; \forall \boldsymbol{z}_i \in \mathcal{Z}.$$

Step 2: Retriever likelihood calculation: Computing the exact likelihood of the passage conditioned on the question requires normalizing over all the evidence passages

$$p(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{D}; \Phi) = \frac{\exp(s(\boldsymbol{q}, \boldsymbol{z}_i)/\tau)}{\sum_{j=1}^{M} \exp(s(\boldsymbol{q}, \boldsymbol{d}_j)/\tau)},$$

where τ is the temperature (or scaling) hyperparameter. Computing this term is intractable, as this would require re-embedding all the evidence passages using Φ_d. Hence, we define a new distribution to approximate the likelihood of $\boldsymbol{z}_i$ as

$$q(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{Z}; \Phi) = \frac{\exp(s(\boldsymbol{q}, \boldsymbol{z}_i)/\tau)}{\sum_{j=1}^{K} \exp(s(\boldsymbol{q}, \boldsymbol{z}_j)/\tau)}, \tag{6.3}$$

which we also refer to as the *student distribution*. We assume that passages beyond the top-K contribute a very small probability mass, so we only sum over all the retrieved passages $\mathcal{Z}$ in the denominator. While this approximation leads to a biased estimate of the retrieved passage likelihood, it works well in practice. Computing Eq. 6.3 is tractable as it requires embedding and backpropagating through a much smaller set of passages.

Step 3: PLM relevance score estimation: We compute the relevance score $\log p(\boldsymbol{z}_i \mid \boldsymbol{q})$ of all the passages in $\mathcal{Z}$ using a large PLM (Θ). This requires scoring the question tokens using teacher-forcing conditioned on a passage as described in §6.1.3. We then define a *teacher distribution* by applying softmax to the relevance scores

$$\hat{p}(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{Z}) = \frac{\exp(\log p(\boldsymbol{z}_i \mid \boldsymbol{q}; \Theta))}{\sum_{j=1}^{K} \exp\left(\log p(\boldsymbol{z}_j \mid \boldsymbol{q}; \Theta)\right)}.$$

Step 4: Loss calculation and optimization: We train the retriever (Φ) by minimizing the KL divergence loss between the teacher distribution (obtained by PLM) and the student distribution

(computed by retriever).

$$\mathcal{L}(\Phi) = \frac{1}{|\mathcal{T}|} \sum_{\boldsymbol{q} \in \mathcal{T}} \mathbb{KL}(\hat{p}(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{Z}) \mid\mid q(\boldsymbol{z}_i \mid \boldsymbol{q}, \mathcal{Z}; \Phi))$$

Intuitively, optimizing the KL divergence pushes the passage likelihood scores of the retriever to match the passage relevance scores from PLM by considering the relevance scores as soft-labels.

Step 5: Updating evidence embeddings: During training, we update the parameters of both the question encoder (Φ_q) and passage encoder (Φ_d). Due to this, the pre-computed evidence embeddings that were computed using initial retriever parameters ($\hat{\Phi}_d$) become stale, which may affect top-K passage retrieval. To prevent staleness, we re-compute the evidence passage embeddings using current passage encoder parameters (Φ_d) after every 500 training steps.

6.1.5 ART as an Autoencoder

Since our encoder takes as input question $\boldsymbol{q}$ and the PLM scores (or reconstructs) the same question when computing the relevance score, we can consider our training algorithm as an autoencoder with a retrieved passage as the latent variable.

In the generative process, we start with an observed variable $\mathcal{D}$ (the collection of evidence passages), which is the support set for our latent variable. Given an input $\boldsymbol{q}$, we generate an index i and retrieve the passage $\boldsymbol{z}_i$. This index generation and retrieval process is modeled by our dual-encoder architecture. Given $\boldsymbol{z}_i$, we decode it back into the question using our PLM.

Recall that our decoder (the PLM) is frozen and its parameters are not updated. However, the signal from the decoder output is used to train parameters of the dual-encoder such that the log-likelihood of reconstructing the question $\boldsymbol{q}$ is maximized. In practice, this improves the dual-encoder to select the best passage for a given question, since the only way to maximize the objective is by choosing the most relevant $\boldsymbol{z}_i$ given the input $\boldsymbol{q}$.

Dataset	Train Questions	Dev	Test
Question Answering Datasets			
WebQ	3,417	361	2,032
NQ-Open	79,168	8,757	3,610
SQuAD-Open	78,713	8,886	10,570
TriviaQA	78,785	8,837	11,313
Entity Questions	–	22,068	22,075
All Questions Datasets			
NQ-Full	307,373	–	–
MS MARCO	502,939	–	–

Table 6.1: Dataset statistics. During the training process, ART only uses the questions while evaluation is performed over the canonical development and test sets.

6.2 Experimental Setup

In this section, we describe the datasets, evaluation protocol, implementation details, and baseline methods for our passage retrieval experiments.

6.2.1 Datasets and Evaluation

Evidence passages: The evidence corpus includes the preprocessed English Wikipedia dump from December 2018 (Karpukhin et al., 2020). Following convention, we split an article into non-overlapping segments containing 100 words each resulting in over 21 million passages. The same evidence is used for both training and evaluation.

Question answering datasets: Following previous work, we use the open-retrieval version of Natural Questions (NQ-Open; Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), WebQuestions (WebQ; Berant et al., 2013), SQuAD-1.0 (SQuAD-Open; Rajpurkar et al., 2016), and Entity Questions (EQ; Sciavolino et al., 2021) datasets. For further details on these datasets, we refer the reader to §3.3.1 for NQ-Open and TriviaQA, §4.2.1 for WebQ, §5.2.1 for SQuAD-Open, and §5.2.2 for EQ. In Table 6.1, we list their training, development, and test set sizes.

All questions datasets: For our transfer learning experiments, we use all the questions from Natural Questions (henceforth referred to as NQ-Full) and MS MARCO passage ranking (Bajaj et al., 2016) datasets. Table 6.1 lists the number of questions. The questions in NQ-Full are information-seeking, as they were asked by real users. Its size is four times that of NQ-Open. NQ-Full consists of questions having just long-form of answers such as paragraphs, all the questions in NQ-Open (which have both long-form and short-form answers), questions having yes/no answers, and questions that do not contain the answer or are unanswerable. For MS MARCO, we use its provided passage collection (around 8.8 million passages in total) as the evidence corpus.

Evaluation: To evaluate retriever performance, we report the conventional top-K accuracy metric. It is the fraction of questions for which at least one passage among the top-K retrieved passages contains a text span that matches human-annotated answer(s) to the question.

6.2.2 Implementation Details

Model sizes: We use BERT base configuration (Devlin et al., 2019) for the retriever, which consists of 12 layers, 12 attention heads, and 768 embedding dimensions, leading to around 220M trainable parameters. For the teacher PLM, we use two configurations: (i) T5-XL configuration (Raffel et al., 2020) consisting of 24 layers, 32 attention heads, and 2048 embedding dimensions, leading to 3B parameters, and (ii) a larger T5-XXL configuration consisting of 11B parameters.

Model initialization: We initialize the retriever with unsupervised masked salient spans (MSS) pre-training (Sachan et al., 2021a) as it provides an improved zero-shot retrieval over BERT pre-training (§3.2.1).[3] We initialize the cross-attention (or teacher) PLM with the T5-lm-adapted (Lester et al., 2021) or instruction-tuned T0 (Sanh et al., 2022) language models, which have been shown to be effective zero-shot re-rankers for information retrieval tasks (Sachan et al., 2022).

[3]We use the open-source MSS retriever checkpoint from https://github.com/DevSinghSachan/emdr2.

Compute hardware: We perform training on instances containing 8 or 16 A100 GPUs, each containing 40 GB RAM.

Passage retrieval: To perform fast top-K passage retrieval at every training step, we pre-compute the embeddings of all the evidence passages. Computing embeddings of 21M passages takes roughly 10 minutes on 16 GPUs. The total size of these embeddings is around 30 GB (768-dimensional vectors in FP16 format). For scalable retrieval, we shard these embeddings across all the GPUs and perform exact maximum inner product search using distributed matrix multiplication.

Training details: When training with T0 (3B) PLM, for all the datasets except WebQ, we perform training for 10 epochs using Adam (Kingma and Ba, 2014) with a batch size of 64, 32 retrieved passages, dropout value of 0.1, peak learning rate of 2×10^{-5} with warmup and linear scheduling. Due to the smaller size of WebQ, we train for 20 epochs with a batch size of 16. When training with the T5-lm-adapted (11B) PLM, we use a batch size of 32 with 16 retrieved passages. We save the retriever checkpoint every 500 steps and perform model selection by evaluating it on the development set. We use mixed precision training to train the retriever and perform inference over the PLM using bfloat16 format (Micikevicius et al., 2018).[4] We set the value of the temperature hyperparameter (τ) using cross-validation.

6.2.3 Baselines

We compare ART to both unsupervised and supervised models. Unsupervised models train a single retriever using unlabeled text corpus from the Internet while supervised models train a separate retriever for each dataset. We report the performance numbers from the original papers when the results are available or run their open-source implementations in case the results are not available.

[4] https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus

Unsupervised models: These include the popular BM25 algorithm (Robertson and Zaragoza, 2009) that is based on the sparse bag-of-words representation of text. Dense models typically use Wikipedia paragraphs to create (pseudo-) query and context pairs to perform contrastive training of the retriever. These differ in how the negative examples are obtained during contrastive training: they can be from the same batch (ICT; Lee et al., 2019; Sachan et al., 2021a), or contexts passages from previous batches (Contriever; Izacard et al., 2022), or by using other passages in the same article (Spider; Ram et al., 2022). Context passages can also be sampled from articles connected via hyperlinks (HLP; Zhou et al., 2022).

Supervised models: These consist of approaches that use questions and positive passages to train the retriever. To obtain improved performance an additional set of hard-negative passages is often used (DPR; Karpukhin et al., 2020), iterative mining of negative passages is done using model weights (ANCE; Xiong et al., 2021), or the retriever is first initialized with ICT or MSS pre-training followed by DPR-style finetuning (ICT-DPR / MSS-DPR; Sachan et al., 2021a). The pre-trained retriever can be further trained by ANCE-style mining of hard-negative passages to further improve accuracy (coCondenser; Gao and Callan, 2022). Previous methods have also explored finetuning the cross-encoder PLM jointly with the retriever such that the cross-encoder provides more accurate training signals to improve retrieval accuracy. Among them include the approaches of end-to-end training of PLM and retriever which infuses supervision from the annotated answers to a question (EMDR^2; Sachan et al., 2021b), multi-stage mixed objective distillation approach to jointly train re-ranker (Nogueira and Cho, 2019) and retriever (RocketQAv2; Ren et al., 2021). A combination of adversarial and distillation-based training of re-ranker and retriever has been shown to obtain state-of-the-art performance (AR2; Zhang et al., 2022).

Retriever	Pre-trained Language Model	SQuAD-Open		TriviaQA		NQ-Open		WebQ	
		Top-20	Top-100	Top-20	Top-100	Top-20	Top-100	Top-20	Top-100
Unsupervised Approaches (trained using Wikipedia / Internet data)									
BERT		5.2	13.5	7.2	17.8	9.4	20.3	3.7	12.8
ICT		45.1	65.2	57.5	73.6	50.6	66.8	43.4	65.7
MSS	T5* (220M)	51.3	68.4	68.2	79.4	59.8	74.9	49.2	68.4
BM25		71.1	81.8	76.4	83.2	62.9	78.3	62.4	75.5
Contriever		63.4	78.2	74.2	83.2	67.8	82.1	74.9	80.1
Spider		61.0	76.0	75.8	83.5	68.3	81.2	65.9	79.7
cpt-text S†		–	–	75.1	81.7	65.5	77.2	–	–
HLP		–	–	76.9	84.0	70.2	82.0	66.9	80.8
Supervised Approaches (trained using question-passage aligned data)									
DPR		63.2	77.2	79.4	85.0	78.4	85.4	73.2	81.4
DPR-Multi‡		51.6	67.6	78.8	84.7	79.4	86.0	75.0	82.9
ANCE		–	–	80.3	85.3	81.9	87.5	–	–
ICT-DPR		–	–	81.7	86.3	81.8	88.0	72.5	82.3
MSS-DPR◇		73.1	84.5	81.8	86.6	82.1	87.8	76.9	84.6
coCondenser		–	–	83.2	87.3	84.3	89.0	–	–
RocketQAv2	ERNIE* (110M)	–	–	–	–	83.7	89.0	–	–
EMDR²○	T5* (220M)	–	–	83.4	87.3	85.3	89.7	**79.1**	**85.2**
AR2	ERNIE* (330M)	–	–	**84.4**	**87.9**	**86.0**	**90.1**	–	–
Our Approach (trained using questions and Wikipedia text)									
ART	T5-lm-adapt (11B)	74.2	84.3	82.5	86.6	80.2	88.4	74.4	82.7
ART-Multi	T5-lm-adapt (11B)	72.8	83.2	82.2	86.6	81.5	88.5	74.8	83.7
ART	T0 (3B)	75.3	85.0	82.9	87.1	81.6	89.0	75.7	84.3
ART-Multi	T0 (3B)	74.7	84.5	82.9	87.0	82.0	88.9	76.6	**85.0**

Table 6.2: Top-20 and top-100 retrieval accuracy on the test set of datasets. For more details regarding the unsupervised and supervised models, please see §6.2.3 in the text. Best supervised results are highlighted in bold while the best results from our proposed model (ART) are underlined. ART substantially outperforms previous unsupervised models and comes close to or matches the performance of supervised models by just using questions during training. * indicates that the cross-attention PLM is finetuned. † denotes that 'cpt-text S' model (Neelakantan et al., 2022) contains around 300M parameters. ‡ denotes that DPR-Multi was not trained on SQuAD-Open. ◇ indicates that the results on SQuAD-Open and WebQ are obtained by finetuning the open-source MSS checkpoint. ○ indicates that EMDR^2 results are obtained using their open-source checkpoints.

6.3 Experiments and Results

6.3.1 Zero-shot Passage Retrieval

For the passage retrieval task, we report results on SQuAD-Open, TriviaQA, NQ-Open, and WebQ and train ART under two settings. In the first setting, we train a separate retriever for each dataset using questions from their training set. In the second setting, to examine the robustness of ART training to different question types, we train a single retriever by combining the questions from all four datasets, which we refer to as ART-Multi. For both these settings, we train ART using T5-lm-adapted (11B) and T0 (3B) cross-attention PLM scorers. As our training process does not require annotated passages for a question, we refer to this as *zero-shot passage retrieval*.

Table 6.2 presents the top-20 and top-100 retrieval accuracy in these settings alongside recent baselines that train a similarly sized retriever (110M). All the variants of ART achieve substantially better performance than previous unsupervised approaches. For example, ART trained with T0 (3B) outperforms the recent Spider and Contriever models by an average of 9 points on top-20 and 6 points on top-100 accuracy. When compared to supervised models, despite using just questions, ART outperforms strong baselines like DPR and ANCE and is at par or slightly better than pre-trained retrievers like MSS-DPR. In addition, ART-Multi obtains comparable performance to its single dataset version, a considerable advantage in practical applications as a single retriever can be deployed rather than training a custom retriever for each use case.

ART's performance also comes close to the state-of-the-art supervised models like AR2 and EMDR^2, especially on the top-100 accuracy but lags behind in the top-20 accuracy. In addition to obtaining reasonable performance and not requiring aligned passages for training, ART's training process is much simpler than AR2. It also does not require cross-encoder finetuning and is thus faster to train. As generative language models continue to become more accurate (Chowdhery et al., 2022), we hypothesize that the performance gap between state-of-the-art supervised models and ART would further narrow down.

Our results showcase that both the PLM scorers, T5-lm-adapt (11B) and T0 (3B), achieve strong results on the QA retrieval tasks, with T0 achieving higher performance gains. This illustrates that the relevance score estimates of candidate passages obtained in the zero-shot cross-attention step are accurate enough to provide strong supervision for retriever training. We believe that this is a direct consequence of the knowledge stored in the PLM weights.[5]

While T5-lm-adapt's knowledge is obtained by training on unsupervised text corpora, T0 was further finetuned using instruction-prompted datasets of tasks such as summarization, QA, text classification, etc. However, T0 was not finetuned on the question generation task and was also not trained on any of the datasets we have used in this work. We refer the reader to the original paper for more training details. Hence, in addition to learning from instructions, the performance gains from T0 can be attributed to the knowledge infused in its weights by (indirect) supervision from these manually curated datasets. Instruction-based finetuning is especially helpful for smaller-sized datasets like WebQ in improving the performance on lower values of top-K accuracy (such as top-20).

Overall, our results suggest that an *accurate and robust passage retrieval can be achieved by training with questions alone*. This presents a considerably more favorable setting than the current approaches which require obtaining positive and hard-negative passages for such questions. Due to its better performance, we use the T0 (3B) PLM for subsequent experiments unless stated otherwise.

6.3.2 Sample Efficiency

To measure the sample efficiency of ART, we train the model by randomly selecting a varying number of questions from NQ-Open training questions and compute the top-K accuracy on its development set. These results are presented in Figure 6.2 and we also include the results of BM25 and DPR for comparison. We see that performance increases with the increase in questions until about 10,000 questions, after which the gains become less pronounced.

[5]We include more detailed comparisons of different PLMs as cross-attention scorers in §6.3.5.

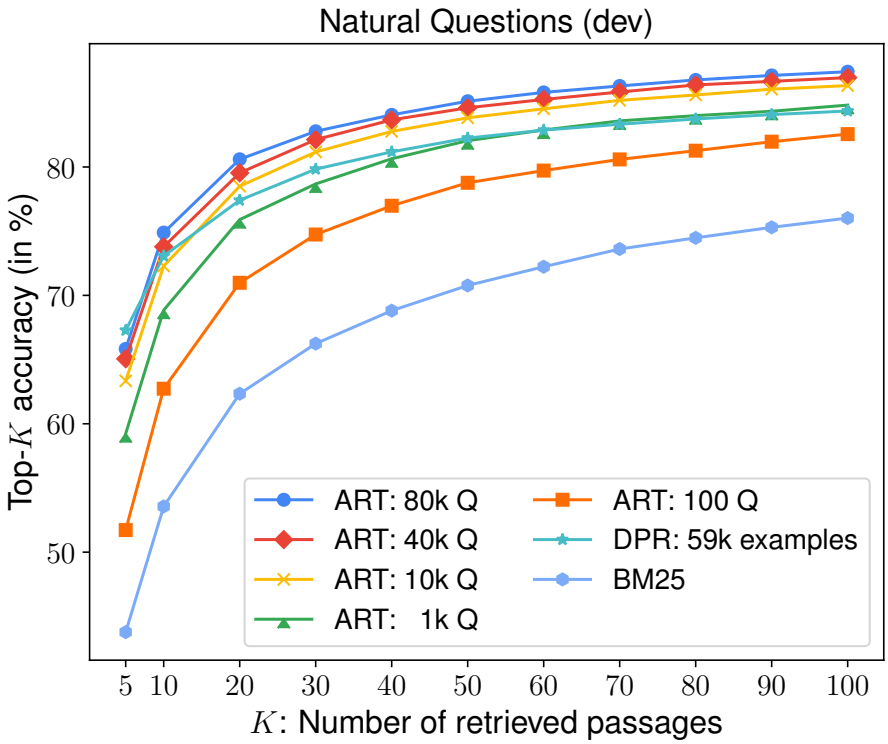

Figure 6.2: Top-K accuracy as the number of training questions (denoted as 'Q' in the legend) is varied. When trained with 100 questions, ART outperforms BM25 and when trained with 1,000 questions, it matches DPR's performance for top-$K > 50$ passages, illustrating that ART is highly sample efficient.

When trained with just 100 questions, ART significantly outperforms BM25 and when trained with 1,000 questions, it matches DPR performance levels for top-{50, ..., 100} accuracy. *This demonstrates that* ART *in addition to using just questions is also much more data efficient than* DPR, as it requires almost ten times fewer questions to reach a similar performance.

6.3.3 Zero-shot Out-of-Distribution Transfer

In the previous experiments, both the training and test sets contained questions that were sampled from the same underlying distribution, a setting that we refer to as *in-distribution training*. However, obtaining in-domain questions for training is not always feasible in practice. Instead, a model trained on an existing collection of questions must be evaluated on new datasets, a setting that we refer to as *out-of-distribution* (OOD) transfer.

Retriever	Training Dataset	SQuAD-Open		TriviaQA		WebQ		EQ	
		Top-20	Top-100	Top-20	Top-100	Top-20	Top-100	Top-20	Top-100
Training on answerable questions									
BM25	–	**71.1**	**81.8**	76.4	83.2	62.4	75.5	71.2	79.8
DPR†	NQ-Open	48.9	65.2	69.0	78.7	68.8	78.3	49.7	63.2
EMDR^2	NQ-Open	66.8	79.0	79.7	85.3	74.2	83.2	62.7	75.1
Spider†	NQ-Open	57.7	72.8	77.2	83.7	74.2	82.5	61.9	74.1
ART	NQ-Open	68.0	80.2	79.8	85.1	73.4	83.1	64.3	75.5
ART	MS MARCO	68.4	80.4	78.0	84.1	74.8	83.2	**75.3**	**81.9**
Training on a mix of answerable and unanswerable questions									
ART	NQ-Full	69.4	**81.1**	80.3	**85.7**	74.3	83.9	67.8	78.3
ART	MS MARCO + NQ-Full	69.6	**81.1**	**80.7**	**85.7**	**75.3**	**84.5**	69.2	79.1

Table 6.3: Top-20 and top-100 retrieval accuracy when evaluating zero-shot out-of-distribution (OOD) generalization of models on the test set of datasets. † denotes that these results are from Ram et al. (2022). ART generalizes better than supervised models on OOD evaluation even when trained on all the questions of the Natural Questions dataset (NQ-Full) which contains a mix of answerable and unanswerable questions.

We train ART using NQ-Open and NQ-Full questions and then evaluate its performance on SQuAD-Open, TriviaQA, WebQ, and EQ datasets. While it is desirable to train on answerable questions such as the ones included in NQ-Open this is not always possible, as real user questions are often imprecisely worded or ambiguous. Due to this, training on NQ-Full can be considered as a practical testbed for evaluating true OOD generalization as a majority of the questions (51%) were marked as unanswerable from Wikipedia by human annotators.[6]

Table 6.3 presents OOD generalization results on the four QA datasets including the results of DPR and Spider models trained on NQ-Open.[7] ART trained on NQ-Open always performs significantly better than both DPR and Spider, showcasing that it is better at generalization than supervised models. When trained using NQ-Full, ART performance further improves by 3 points on EQ and by 0.5-1 points on other datasets, over NQ-Open. This highlights that in addition to questions annotated as having short answers, *questions annotated with long answers also provide*

[6]The reasons for question unanswerability can be partly attributed to imprecise Wikipedia article retrieval during the annotation process, ambiguity in information-seeking questions, information required to answer not being localized to a single paragraph, etc (Kwiatkowski et al., 2019).

[7]We also include BM25 results for reference but do not directly compare with them because there is a high lexical overlap between question and passage tokens in the SQuAD-Open and EQ datasets which renders dense retrievers at a disadvantage over BM25, especially in the transfer setting.

	Retriever	NQ-Open		TriviaQA	
		Top-20	Top-100	Top-20	Top-100
Dev	ICT	44.2	61.0	58.8	74.4
	DPR	79.1	85.5	81.1	85.9
	ICT-DPR	81.4	87.4	82.8	86.9
	EMDR^2	<u>83.1</u>	<u>88.0</u>	<u>83.7</u>	<u>87.4</u>
	ART-base	80.6	87.4	83.6	87.4
	ART-large	**81.0**	**87.8**	**83.7**	**87.5**
Test	ICT	49.3	66.1	58.5	74.1
	DPR	81.0	87.2	81.4	86.0
	ICT-DPR	82.6	88.3	82.9	87.1
	EMDR^2	<u>85.3</u>	<u>89.7</u>	<u>83.4</u>	<u>87.3</u>
	ART-base	81.6	**89.0**	82.9	87.1
	ART-large	**82.1**	88.8	**83.6**	**87.6**

Table 6.4: Top-20 and top-100 accuracy when training large configuration retriever, which contains around 650M parameters. EMDR^2 (Sachan et al., 2021b) (base configuration) contains 440M parameters. Best supervised results are underlined while the best unsupervised results are highlighted in bold.

meaningful supervisory signals and *unanswerable questions do not necessarily degrade performance*.

We also train ART using MS MARCO questions and perform OOD evaluation. Due to the larger size of MS MARCO and a smaller number of evidence passages, we use a batch size of 512 and retrieve 8 passages for training. Quite surprisingly, it obtains much better performance than previous approaches including BM25 on EQ (more than 10 points gain on top-20 accuracy over training ART on NQ-Open). We suspect that this may be due to the similar nature of questions in MS MARCO and EQ. Further finetuning the pre-trained MS MARCO model on NQ-Full significantly improves performance on WebQ.

6.3.4 Scaling Model Size

We examine if scaling up the retriever parameters can offer further performance improvements. To this end, we train a retriever of BERT-large configuration (24 layers, 16 attention heads, 1024 embedding dimensions) containing around 650M parameters on NQ-Open and TriviaQA. Results are presented in Table 6.4 for both the development and test sets. We also include the results of

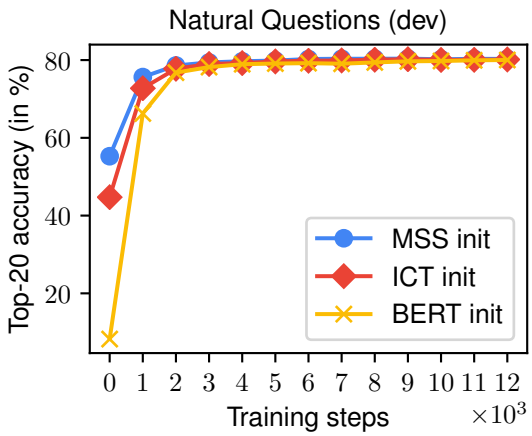

Figure 6.3: Effect of retriever initialization on ART training. The plot reveals that the training process is not sensitive to initial retriever parameters.

other relevant baselines containing a similar number of trainable parameters.

By scaling up the retriever size, we see small but consistent improvements in retrieval accuracy across both datasets. Especially on TriviaQA, ART matches or exceeds the performance of previous best models. On NQ-Open, it comes close to the performance of EMDR^2 (Sachan et al., 2021b), a supervised model trained using thousands of question-answer pairs.

We also attempted to use larger teacher PLMs such as T0 (11B). However, our initial experiments did not lead to any further improvements over the T0 (3B) PLM. We conjecture that this might be either specific to these QA datasets or that we need to increase the capacity of the teacher PLM even more to observe improvements. We leave an in-depth analysis of using larger teacher PLMs as part of future work.

6.3.5 Analysis

Sensitivity to retriever initialization: To examine how the convergence of ART training is affected by the initial retriever parameters, we initialize the retriever with (1) BERT weights, (2) ICT weights (as trained in Sachan et al., 2021a), and (3) MSS weights, and train using NQ-Open questions. Figure 6.3 displays the top-20 performance on the NQ-Open development set as the training progresses. It reveals that ART training is not sensitive to the initial retriever parameters

Number of Retrieved Passages	Top-1	Top-5	Top-20	Top-100
32	36.7	65.8	80.6	87.4
2	+2.4	+0.4	−0.9	−0.6
4	+1.9	+0.9	−0.6	−0.6
8	+0.8	+0.8	−0.5	−0.1
16	+0.9	+0.9	−0.3	−0.1
64	−0.5	−0.7	−0.3	0
128	−2.3	−1.7	−0.8	−0.2

Table 6.5: Effect of varying the number of retrieved passages during ART training as evaluated on the NQ-Open development set. For each case, we list the absolute gain or loss in top-K accuracy when compared to the setting utilizing 32 retrieved passages.

as all three initialization schemes converge to similar results. However, the convergence properties might be different under low-resource settings, an exploration of which we leave for future work.

Effect of the number of retrieved passages: Table 6.5 quantifies the effect of the number of retrieved passages used during training on performance. A smaller number of retrieved passages such as 2 or 4 leads to a somewhat better top-$\{1, 5\}$ accuracy, at the expense of a drop in top-$\{20, 100\}$ accuracy. Retrieving 32 passages offers a reasonable middle ground and beyond that, the top-K retrieval performance tends to drop.

A closer inspection of ART with supervised models: In order to have a better understanding of the tradeoff between supervised models and ART, we examine their top-1 and top-5 accuracy in addition to the commonly reported top-20 and top-100 scores. Table 6.6 presents these results for ART (large) along with supervised models of DPR (large) and EMDR^2. Supervised models achieve much better performance for top-$K \in \{1, \ldots, 5\}$ passages, *i.e.*, these models are more precise. This is likely because DPR is trained with hard-negative passages and EMDR^2 finetunes PLM using reference answers resulting in an accurate relevance feedback to the retriever. When considering top-$K \in \{20, \ldots, 100\}$ passages, ART comes close or matches the performance of EMDR^2. As top-performing models for knowledge-intensive tasks such as open-domain QA rely on a larger set of retrieved passages, such as top-K=100 (Sachan et al., 2022), this justifies the

Retriever	Top-1	Top-5	Top-20	Top-100
NQ-Open (dev)				
DPR	**50.1**	**69.6**	79.1	85.5
EMDR^2	**55.3**	**74.9**	83.1	88.0
ART	37.6	66.8	81.0	87.8
TriviaQA (dev)				
DPR	59.6	74.4	81.1	85.9
EMDR^2	**63.7**	78.0	83.7	87.4
ART	58.3	77.5	83.7	87.5

Table 6.6: Analysis reveals that ART (large) can even match the performance of end-to-end trained models like EMDR^2 when retrieving a larger number of passages. However, DPR (large) and EMDR^2 still outperform ART when retrieving a small number of passages such as top-$K \in \{1, \ldots, 5\}$ (highlighted in bold).

P	N	U	IB	Top-1	Top-5	Top-20	Top-100
0	0	32	✗	6.0	16.6	30.8	46.7
1	0	31	✗	31.8	58.9	74.8	84.4
1	1	30	✗	33.7	61.0	76.0	85.5
1	1	0	✓	32.6	59.5	75.1	84.9
Top-32 passages				**36.7**	**65.8**	**80.6**	**87.4**

Table 6.7: Effect of passage types on ART training when evaluated on the NQ-Open development set. P denotes a positive passage, N denotes a hard-negative passage (mined using BM25), U denotes that the passages are randomly sampled from the evidence, and IB denotes in-batch training.

argument to adopt zero-shot ART over supervised retrievers.

Why training using passage retrieval? To assess the importance of passages in $\mathcal{Z}$ during the training process, we train the retriever under different settings by varying the passage types. Specifically, we train with a mix of positive, hard-negative, and uniformly sampled passages. We also perform in-batch training by defining $\mathcal{Z}$ to be the union of positive and hard-negative passages for all the questions in a batch. Results in Table 6.7 illustrate that when $\mathcal{Z}$ consists of uniformly sampled passages, it leads to poor performance. Including a (gold) positive passage in $\mathcal{Z}$ leads to good performance improvements. Results further improve with the inclusion of a hard-negative passage in $\mathcal{Z}$. However, in-batch training leads to a slight drop in performance. As the gold passages are not always available, our method of selecting the top passages from evidence at every training step

Language Model (Θ)	NQ-Open (dev) Top-1	Top-5	Top-20	Top-100
Models trained using Denoising Masked Spans				
T5-base (250M)	12.8	30.9	47.8	63.0
T5-xl (3B)	25.0	53.9	74.4	85.3
T5-xxl (11B)	29.5	59.8	77.8	86.3
Models trained using Language Modeling Objective				
T5-lm-adapt (250M)	29.4	56.6	74.4	84.7
T5-lm-adapt (800M)	30.9	59.1	76.5	85.9
T5-lm-adapt (3B)	31.8	61.0	77.9	86.5
T5-lm-adapt (11B)	32.7	62.6	78.6	87.0
Model trained using Natural Language Instructions				
T0 (3B)	**36.7**	**65.8**	**80.6**	**87.4**
T0 (11B)	34.3	64.5	79.8	87.2

Table 6.8: Comparison of different pre-trained language models (PLMs) when used as cross-attention scorers during training (§6.1.3). T0 (3B) PLM achieves the highest accuracy among the compared PLMs showcasing that training language models using instruction-tuning provides accurate relevance scores.

can be seen as an approximation to using the gold passages. With this, ART obtains even better results than the previous settings, an improvement by 4 points absolute in the top-20 accuracy.

Impact of language model training strategy: We examine which PLMs can provide accurate cross-attention scores during ART training. We compare across PLMs trained using three different objectives— (i) generative denoising of masked spans (T5 series; Raffel et al., 2020), (ii) further pre-training using autoregressive language modeling objective (T5-lm-adapt series; Lester et al., 2021), and (iii) finetuning T5-lm-adapt models on unrelated tasks using instructions (T0 series; Sanh et al., 2022). Our results in Table 6.8 highlight that PLM training methodology and model size can have a large effect on retrieval performance. The T5 base model leads to low scores possibly because pre-training using predicting masked spans is not ideal for question reconstruction. However, the accuracy improves with an increase in model size. T5-lm-adapt models are more stable and lead to improved performance with the best result achieved by the 11B model. Instruction finetuned T0 models outperform the T5-lm-adapt models. However, scaling up the size of T0 to 11B parameters does not result in meaningful improvements.

Dataset	#Q	#E	nDCG@10				Recall@100			
			DPR†	BM25†	Contriever	ART	DPR†	BM25†	Contriever	ART
Scifact	300	5K	31.8	**66.5**	64.9	55.2	72.7	90.8	**92.6**	88.0
Scidocs	1000	25K	7.7	**15.8**	14.9	14.4	21.9	35.6	**36.0**	32.4
Nfcorpus	323	3.5K	18.9	**32.5**	31.7	29.9	20.8	25.0	**29.0**	26.6
FIQA-2018	648	57K	11.2	23.6	24.5	**26.5**	34.2	53.9	**56.2**	55.4
Trec-covid	50	0.2M	33.2	**65.5**	27.4	50.3	21.2	**49.8**	17.2	36.9
Touche-2020	49	0.4M	13.1	**36.8**	19.3	16.2	30.1	**53.8**	22.5	44.7
NQ	3452	2.7M	**47.4**	32.9	25.4	40.5	88.0	76.0	77.1	**88.7**
MS-Marco	6980	8.8M	17.7	22.8	20.6	**32.6**	55.2	65.8	67.2	**81.7**
HotpotQA	7405	5.2M	39.1	60.3	48.1	**61.0**	59.1	**74.0**	70.4	73.9
ArguAna	1406	8.7K	17.5	31.5	**37.9**	32.2	75.1	94.2	90.1	**95.3**
CQADupStack	13145	0.5M	15.3	29.9	28.4	**33.5**	40.3	60.6	61.4	**62.6**
Quora	10000	0.5M	24.8	78.9	83.5	**84.2**	47.0	97.3	98.7	**98.8**
DBpedia	400	4.6M	26.3	31.3	29.2	**36.3**	34.9	39.8	45.3	**47.2**
Fever	6666	5.4M	56.2	**75.3**	68.2	72.4	84.0	93.1	**93.6**	93.1
Climate-Fever	1535	5.4M	14.8	21.3	15.5	**21.4**	39.0	43.6	44.1	**47.1**
Average Score			25.0	41.6	36.0	40.4	48.2	63.6	60.1	64.8

Table 6.9: Zero-shot results on the BEIR benchmark. #Q and #E denotes the size of the test set and evidence, respectively. Best scores for each dataset are highlighted in bold. ART is trained using MS MARCO questions. DPR is trained using NQ-Open. † denotes that these results are from Thakur et al. (2021).

Ad-hoc retrieval tasks: While the previous experiments were conducted on QA datasets, here we examine the robustness of the ART model trained using questions to different ad-hoc retrieval tasks. For this analysis, we evaluate the performance of ART on the BEIR benchmark (Thakur et al., 2021). It is a heterogeneous collection of many retrieval datasets, with each dataset consisting of test set queries, evidence documents, and gold document annotations. BEIR spans multiple domains and diverse retrieval tasks presenting a strong challenge suite, especially to the dense retrievers. We train ART using MS MARCO questions and report its nDCG@10 and Recall@100 scores on each dataset. For comparison, we include the results of three baselines: BM25, Contriever, and DPR trained using NQ-Open. Our results presented in Table 6.9 show strong generalization performance of ART as it outperforms DPR and Contriever results. ART also achieves at par results with the strong BM25 baseline outperforming BM25 on 8 out of the 15 datasets (according to nDCG@10 scores).

6.4 Related Work

Our work is based on training a dense retriever using pre-trained language models (PLMs), which we have covered in previous sections. Here, we instead focus on other related approaches.

A popular method to train the dual-encoder retriever is to optimize contrastive loss using in-batch negatives (Gillick et al., 2019) and hard-negatives (Karpukhin et al., 2020; Xiong et al., 2021). Alternatives to using hard-negatives such as sampling from cached evidence embeddings have also shown to work well in practice (Lindgren et al., 2021). Multi-vector encoders for questions and passages are more accurate than dual-encoders (Luan et al., 2021; Khattab and Zaharia, 2020; Humeau et al., 2020), although at the cost of increased latency and storage requirements.

PLMs have been shown to improve passage rankings as they can perform cross-attention between the question and the retrieved passages (Lin et al., 2021b). Supervised approaches to re-rank either finetune PLMs using question-passage pairs (Nogueira et al., 2020) or finetune PLMs to generate question conditioned on the passage (Nogueira dos Santos et al., 2020) while unsupervised re-rankers are based on zero-shot question scoring (Sachan et al., 2022). The re-ranking process is slow due to the cross-attention step and is bottlenecked by the accuracy of first-stage retrievers. To address these limitations, cross-attention distillation approaches from the PLM to the retriever have been proposed (Qu et al., 2021). Such distillation can be performed either in a single end-to-end training step (Guu et al., 2020; Sachan et al., 2021b) or in a multi-stage process (Khattab et al., 2021; Izacard and Grave, 2021a).

An alternative approach to using PLMs is to generate data that can aid retrieval. The data can be either the title or an answer that provides more information about the question (Mao et al., 2021). Generating new questions to augment the training data has also been shown to improve performance (Ma et al., 2021a; Bonifacio et al., 2022; Dai et al., 2022). In comparison, we do not generate new questions but train the retriever using existing questions and PLM feedback. Data augmentation is likely complementary, and can further improve accuracy.

6.5 Discussion

In this chapter, we introduced ART, a novel approach to train a dense passage retriever using only questions and a collection of evidence documents. ART does not require question-passage pairs or hard-negative examples for training and yet achieves state-of-the-art results. The key to making ART work is to optimize the retriever to select relevant passages such that conditioning on them, the question generation likelihood computed using a large pre-trained language model iteratively improves. We summarize our key contributions and results below:

- Using only questions as the primary training data, ART outperforms popular supervised methods like DPR (Karpukhin et al., 2020) when evaluated on multiple QA benchmarks. It also achieves better generalization performance on out-of-distribution evaluation sets than previous approaches.

- ART exemplifies better sample efficiency properties than DPR and BM25 retrievers as it outperforms them by training on a fraction of questions. Furthermore, ART is the first dense retriever training approach to outperform BM25 on the SQuAD-Open dataset which was previously assumed to be a challenging testbed for dense retrievers.[8]

- Another useful property of ART is that it can be trained on a mixture of answerable and unanswerable questions without any adverse effects on end performance. This presents a considerably favorable setting in real-world applications where the answerability of user questions is unknown apriori.

- Our experiments reveal that ART is not sensitive to retriever initialization and can achieve peak performance even when initialized with BERT weights. This finding presents an alternate mechanism to bootstrap retrievers for end-to-end supervised training.

[8] SQuAD-Open has been considered to be a challenging dataset because of two reasons (i) its training examples exhibit a high lexical overlap between questions and passages, and (ii) biased training distribution as the examples were collected from just 536 Wikipedia articles, representing a very small subset of full Wikipedia.

- On ad-hoc retrieval tasks such as the datasets included in the BEIR benchmark, the overall performance of ART is comparable to BM25. This is especially encouraging as previously dense retrievers had struggled to perform well on this benchmark.

6.5.1 Limitations

In the previous sections, we have covered the pros of using ART to train retrievers and discussed its strengths in detail. In this section, we identify and describe the potential shortcomings of this method.

Dependence on real information-seeking questions for training: To train, ART requires supervision from both questions and evidence passages. The questions that we used for training were relatively high quality in the sense that they were either written by crowdworkers or were posed by users seeking information through search engines. As such when bootstrapping retriever training, obtaining a database of human-written questions may not be readily feasible. An alternate option to using real questions is to generate questions by prompting a language model. However, more work is needed to quantify the relative tradeoffs of this choice as it is unclear if training with synthetic questions instead of real ones would cause performance degradation or not.

Less precise than supervised models for top-{1, 5} accuracy: ART despite being a simple training method performs on par with state-of-the-art supervised methods especially in the setting when retrieving 20-100 passages. However, when comparing the top-1 and top-5 accuracy, ART underperforms achieving lower scores than supervised methods, *i.e.*, it is less precise (Table 6.6) thus limiting its adoption to tasks requiring a larger set of ranked passages. The strong performance of supervised methods can be attributed to the use of hard-negative passages during their training. On the other hand, it is non-trivial to define hard-negatives for ART training as it does not rely on the existence of positive examples. It remains an open problem on how to improve the top-{1, 5} accuracy for ART training.

Pitfalls of relevance score estimation using PLM: During ART training, we calculate the relevance scores separately for each candidate passage, which is also referred to as pointwise estimation. However, these pointwise relevance scores may not be well calibrated to be used as a supervisory signal in retriever training as the question generation step is conditioned on a separate passage each time. This can in turn adversarially affect retriever training leading to suboptimal retrievals. An option to mitigate this is to explore alternate PLM prompting approaches such as pairwise prompting (Qin et al., 2023) in which the PLM jointly reasons among the candidate passages to output the most relevant passage.

Limited understanding of generalization on unanswerable questions: In this chapter, we have demonstrated that ART training is robust even when a mix of answerable and unanswerable questions are provided as input (§6.3.3, Table 6.3). However, it is not apparent what is the desired mixing ratio of the unanswerable questions which is required for the emergence of this robustness. It is also not clear if the unanswerable questions should be related to the evidence passages or not. As part of future work, systematic experiments are needed to further probe the generalization of ART training to unanswerable questions.

6.5.2 Follow-up Work

- **Retriever adaptation using frozen language models:** This work augments large language models with retrieved text in order to improve their predictions (Shi et al., 2023). Given a query text, matching passages are retrieved from the evidence and are independently given as input to the language model. Similar to ART, the retriever parameters are then finetuned at every step using feedback from the language model such that the retrieved text lowers the perplexity of the language model when generating the target tokens using teacher-forcing. During training, the language model parameters are kept frozen. Results showcased that adapted retrieval improves the original language model's predictions on tasks such as question answering, language modeling, etc.

- **Listwise permutation distillation:** Recent work has shown that large language models especially their instruction-following variants (Ouyang et al., 2022) are effective document re-ranking agents (Ma et al., 2023; Sun et al., 2023). These PLMs can jointly reason across an input query and retrieved list of documents and output a permuted list reflecting a more accurate ordering. However, as inference using large models is computationally expensive, efforts have been directed towards training smaller compute-efficient BERT re-rankers by applying ranking losses over the permutation list generated from larger models (Sun et al., 2023).

6.5.3 Future Work

ART presents several directions for future work. It would be interesting to apply this approach in low-resource retrieval including multi-lingual (Clark et al., 2020a) and cross-lingual question answering (Asai et al., 2021). Our training framework can also be extended to train cross-modality retrievers such as for image or code search (Li et al., 2022; Neelakantan et al., 2022) using textual queries. Finally, other directions worth exploring would be to make use of labeled data when available such as by finetuning PLM on passage-question aligned data and to train multi-vector retrievers (Luan et al., 2021) with ART.

Chapter 7

Conclusion

7.1 Summary of Contributions

In this thesis, we have introduced novel methods to improve open-domain retrieval especially focused towards passage retrieval for question answering tasks. We have also proposed approaches to perform well on the important task of answering factual questions. In the following paragraphs, we summarize the contributions of this thesis. We also assess the impact of our contributions as part of the broader scientific discourse.

In our first contribution (Chapter 3), we introduced the approach of unsupervised pre-training of dense retriever parameters such that it results in improved training dynamics during supervised finetuning. Our proposed pre-training tasks consist of predicting paragraph-level context using sentences and predicting salient text spans such as named entities using retrieved documents. We demonstrate that such unsupervised pre-training yields substantially better zero-shot recall than BERT-initialized retrievers. When finetuned using supervised datasets, pre-trained retrievers lead to significant gains in retrieval accuracy when compared to their non-pre-trained counterparts.

In our second contribution (Chapter 4), we proposed an end-to-end training method, EMDR^2, for retrieval-augmented question answering systems. EMDR^2 jointly trains a system consisting of a multi-document reader and a dense retriever. We first present a latent variable model of the system

and demonstrate that the exact optimization of the marginal likelihood is intractable. We then propose an approximate training objective using the expectation-maximization algorithm. When evaluated on three benchmark datasets for the task of open-domain question answering, EMDR^2 outperformed previous approaches by 2-3 exact match points. Contrary to the previous approaches that emphasized the need for intermediate document annotations and stage-wise training, EMDR^2 was the first method to showcase that state-of-the-art results can be attained with a single training run and using just question-answer pairs.

Both these contributions have already had a considerable impact within the NLP community, especially in terms of new techniques building upon them. Most notably, our work on unsupervised pre-training has contributed to the emergence of a new task where the objective is to attain the highest document rankings without supervised training, *i.e.*, zero-shot retrieval. Representative work proposing new pre-training methods include leveraging momentum contrastive learning (Izacard et al., 2022) and constructing positive query-document pairs using recurrent text spans (Ram et al., 2022) to learn robust document representations. EMDR^2 has been influential in spearheading the idea of end-to-end training of retrieval-augmented systems. An important follow-up work to EMDR^2 is ATLAS (Izacard et al., 2023) which showcased that end-to-end training of a retrieval-augmented system using a few exemplars outperforms a substantially large model without retrieval-augmentation.

Our next set of contributions addresses the training data dependency of state-of-the-art retrievers. Concerning this, in our third contribution (Chapter 5), we propose an unsupervised passage re-ranker (UPR) to improve the passage rankings of a first-stage retriever. UPR employs frozen language models as the underlying re-ranker. It re-ranks passages based on relevance score estimates obtained by computing the likelihood of question generation conditioned on the passage. Although a simple idea, UPR serves as a powerful re-ranker being the first unsupervised approach to convincingly outperform strong supervised retrievers. In our experiments, we show that the advantages of UPR are multifold: requires no training data, works with off-the-shelf language models, and obtains strong performance gains for both sparse and dense retrievers.

In our last contribution (Chapter 6), we introduce ART, a novel training method to distill large language model's retrieval knowledge into compact dense retrievers. For its training data, ART relies on questions and unpaired evidence documents as inputs. This presents a considerably more relaxed requirement than obtaining aligned question-document pairs and mining hard-negative documents. The retriever is trained by optimizing the retriever likelihood to mimic relevance scores estimated from the teacher language model. When evaluated on benchmark datasets, ART substantially outperforms unsupervised retrievers while coming close to the performance of state-of-the-art supervised methods. Training dense retrievers with ART also alleviates the higher inference cost associated with using large language models as re-rankers.

In terms of scientific impact, UPR has been influential in getting the community excited about using language models as a black box to improve passage retrieval. Several follow-up works have explored alternative techniques to prompt the language model such that it results in a better estimation of the passage relevance score (Ma et al., 2023; Sun et al., 2023; Qin et al., 2023). These techniques consist of either providing the passage list or pairwise inputs and prompting language models to iteratively generate the ranked ordering. Even though the ART method is fairly recent, its idea of distilling the relevance score from language models into smaller and more efficient models has already been applied by others. Among them include distilling knowledge into smaller re-rankers (Pradeep et al., 2023) and adapting retrievers by prompting language models to directly generate an answer for a question (Shi et al., 2023).

7.2 Limitations

In this section, we discuss limitations that are more broadly associated with our contributions and in some sense are fundamentally embedded into the recent retrieval-augmented models.

Our first point concerns the end-to-end training of retrieval-augmented systems. As mentioned previously, these systems have been extremely effective in providing big gains over vanilla language models, especially on question answering (QA) tasks in both full finetuning and few-shot

settings. However, these improvements have come at the expense of their increased training complexity. Specifically, these retrieval systems are composed of reader and retriever networks that need to be first pre-trained separately. For instance, the retrievers themselves require initialization using masked language models such as BERT. Then, they are further contrastively trained by predicting neighboring contexts from a sentence. The reader network also needs to be first trained by denoising masked text spans. Finally, the reader and retriever are then jointly pre-trained using a prefix language modeling task. We believe that such complicated training pipelines have hindered the wide adoption of retrieval-augmented models. Contrast this with autoregressive language models that need to be pre-trained once using generative training. A more promising direction would have been to explore simple pre-training methods for retrieval systems that are both easy to train and obtain empirically better performance than non-retrieval systems.

Our second point is specific to the training objective formulations that have been devised to distill from language model logits into the retriever. Specific cases of this distillation in this thesis are during end-to-end training of the language model and retriever in EMDR^2 and using language models as a black box to train retrievers in ART. In order to simplify the training process, almost all approaches consider the top-K matching documents from the evidence to calculate retriever distribution. This results in a limited exploration of the space of evidence documents and leads to biased retriever training. An alternative direction could have been to develop training objectives that enable more exploration by sampling documents from the evidence. Designing objectives that lead to a more principled training approach have the potential to further improve retrieval. A recent effort in this direction includes approximating marginal likelihood in end-to-end training by optimizing variational inference bounds (Liévin et al., 2023).

Our final point concerns the limited applicability of retrieval-augmented systems to a wider range of language understanding tasks. In this thesis, our proposed models and training approaches have been evaluated primarily on QA datasets. Apart from QA, success stories of retrieval-augmented systems are limited to a few end tasks such as dialog generation (Shuster et al., 2021) and machine translation (Khandelwal et al., 2021). Moreover, these systems are often task-specific

and may not be readily transferable. This is in stark contrast to language models in which the same model has been shown to perform remarkably well on a multitude of tasks (Chowdhery et al., 2022). Although it is well understood that retrieving from large evidence sets has also helped improve language modeling performance, these gains haven't yet translated into downstream tasks (Borgeaud et al., 2022). One reason might be that retrieval as a tool has inherently restricted usefulness or that there has been lack of sustained efforts to develop generic retrieval-augmented systems. Given the huge potential of retrieval-augmented models in scaling down parameter sizes, we feel that this is an opportunity. An interesting direction to pursue would be to develop unified retrieval-augmented systems applicable to a large array of downstream tasks.

7.3 Directions for Future Work

In this section, we layout a broad set of directions for future work based on the limitations of current retrieval systems and the additional capabilities that the next generation of retrieval-augmented systems ought to have in order to enable their wider adoption.

Most of the retrieval-augmented generation work in this thesis has been oriented towards systems that generate factually correct *short text span* as *answers*. However, from the viewpoint of user-facing systems they cater to a narrow subset of queries and thus overall have limited applicability. Much more useful are dialog agents that can handle diverse sets of queries including those that require generating long but factually correct text. However, even commercial dialog systems such as ChatGPT (OpenAI, 2022) despite being custom-trained are still prone to hallucinations leading to imprecise generations (Min et al., 2023). Current retrieval systems would offer limited help because they are trained to retrieve content based on the query but do not condition on the generated text. These limitations also present exciting future research directions on how to equip dialog systems with *adaptable retrieval models* such that they can retrieve accurate knowledge when conditioned on both the query and past generations. We believe that retrieval systems would play a key role in future conversational agents as they have the potential to reduce hallucinations

in the generated text while ensuring that factual entities are attributable to the retrieved knowledge source (Bohnet et al., 2022).

Our next point addresses the computational overhead of current retrieval-augmented systems. As a recap, these systems are designed to retrieve matching documents for *every input query* which are then fed to the language models. Despite being effective these systems are associated with additional computational costs. However, retrieval at every query may not be strictly necessary as large language models store a lot of factual knowledge directly in their parameters (Brown et al., 2020; Roberts et al., 2020). The next generation of retrieval models needs to be adaptive based on the input query and their internal state (Mallen et al., 2023). Going forward, developing principled approaches for identifying when to retrieve and when not to would be crucial in building computationally efficient systems.

So far, we have proposed research directions to improve the performance of a retrieval system based on textual queries. However, information-seeking queries can consist of multimodal inputs as well, *i.e.*, that of an image associated with a textual question (Mensink et al., 2023). For instance, given an image containing a picture of a landmark. An information-seeking query can be "*When was this landmark constructed?*" Designing retrieval systems that can answer multimodal queries will require innovation at multiple levels. First, multimodal retrievers are needed that can jointly embed both textual and visual inputs in a latent space. Then, the input query needs to be matched with the evidence documents such as Wikipedia articles that can also contain both text and visual content. Overall, the multimodal retrieval task is challenging as the retrievers need to encode four-way inputs—image and text data from both the query and evidence. Although prior work has proposed encoders such as CLIP (Radford et al., 2021), they often underperform in multimodal retrieval tasks probably because they encode image and text inputs separately (Chen et al., 2023c). We believe that more research in retriever architecture and their training are needed to make further progress in multimodal retrieval.

Our field has been witnessing the rise of large foundation models in the fields of both language understanding (Chowdhery et al., 2022; Ouyang et al., 2022) and visual-language under-

standing (Alayrac et al., 2022; Chen et al., 2023b). These models have demonstrated astonishing emergent capabilities on a myriad of evaluation benchmarks (Bommasani et al., 2021; Liang et al., 2022). At the same time, there is a lot of excitement around retrieval or semi-parametric approaches as they can often learn the data distribution more easily than fully parametric models (Izacard et al., 2023; Chen et al., 2023a). However, it is not clear how to effectively adapt foundation models so that they can attend to retrieved information and improve their predictions. It would be interesting to explore new training methods for foundation models so that they can reason using retrieved data consisting of both text and images. As such, we feel that developing *multimodal retrieval foundation models* can be the next frontier towards improving the capabilities of generic foundation models.

Bibliography

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*.

Chris Alberti, Kenton Lee, and Michael Collins. 2019. A bert baseline for the natural questions. *arXiv preprint arXiv:1901.08634*.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Akari Asai, Jungo Kasai, Jonathan Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021. XOR QA: Cross-lingual open-retrieval question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

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

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2009. Web-scale n-gram models for lexical disambiguation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Bernd Bohnet, Vinh Q Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, et al. 2022. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, et al. 2020. Overview of touché 2020: Argument retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Springer International Publishing.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems*.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based *n*-gram models of natural language. *Computational Linguistics*, 18(4).

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

Jannis Bulian, Christian Buck, Wojciech Gajewski, Benjamin Börschinger, and Tal Schuster. 2022. Tomayto, tomahto. beyond token-level answer equivalence for question answering evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Chris J.C. Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. Technical Report MSR-TR-2010-82.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations*.

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

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

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*.

Wenhu Chen, Hexiang Hu, Chitwan Saharia, and William W. Cohen. 2023a. Re-imagen: Retrieval-augmented text-to-image generator. In *The Eleventh International Conference on Learning Representations*.

Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. 2023b. PaLI: A jointly-scaled multilingual language-image model. In *The Eleventh International Conference on Learning Representations*.

Xilun Chen, Kushal Lakhotia, Barlas Oguz, Anchit Gupta, Patrick Lewis, Stan Peshterliev, Yashar Mehdad, Sonal Gupta, and Wen-tau Yih. 2022. Salient phrase aware dense retrieval: Can a dense retriever imitate a sparse one? In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. 2023c. Can pre-trained vision and language models answer visual information-seeking questions? *arXiv preprint arXiv:2302.11713*.

Felix Chern, Blake Hechtman, Andy Davis, Ruiqi Guo, David Majnemer, and Sanjiv Kumar. 2022. TPU-KNN: K nearest neighbor search at peak FLOP/s. In *Advances in Neural Information Processing Systems*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam

Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. PaLM: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Kenneth W. Church and William A. Gale. 1991. A comparison of the enhanced good-turing and deleted estimation methods for estimating probabilities of english bigrams. *Computer Speech and Language*, 5(1):19–54.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020a. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

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

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*.

Andrew M Dai, Christopher Olah, and Quoc V Le. 2015. Document embedding with paragraph vectors. *arXiv preprint arXiv:1507.07998*.

Zhuyun Dai, Arun Tejasvi Chaganty, Vincent Y Zhao, Aida Amini, Qazi Mamunur Rashid, Mike Green, and Kelvin Guu. 2022. Dialog inpainting: Turning documents into dialogs. In *Proceedings of the 39th International Conference on Machine Learning*.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith Hall, and Ming-Wei Chang. 2023. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Michiel de Jong, Yury Zemlyanskiy, Nicholas FitzGerald, Fei Sha, and William W. Cohen. 2022. Mention memory: incorporating textual knowledge into transformers through entity mention attention. In *International Conference on Learning Representations*.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

A.P. Dempster, N.M. Laird, and D.B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 1(39):1–38.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2021. Augmenting Transformers with KNN-Based Composite Memory for Dialog. *Transactions of the Association for Computational Linguistics*, 9.

William B. Frakes and Ricardo Baeza-Yates, editors. 1992. *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall, Inc.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*.

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *arXiv preprint arXiv:1811.08008*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*.

Klaus Greff, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. 2017. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar.

2020. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the 37th International Conference on Machine Learning*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *IEEE/CVF Conference on computer vision and pattern recognition (CVPR)*.

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

Cuong Hoang, Devendra Sachan, Prashant Mathur, Brian Thompson, and Marcello Federico. 2023. Improving retrieval augmented neural machine translation by controlling source and fuzzy-match interactions. In *Findings of the Association for Computational Linguistics: EACL 2023*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Ziniu Hu, Ahmet Iscen, Chen Sun, Zirui Wang, Kai-Wei Chang, Yizhou Sun, Cordelia Schmid, David A. Ross, and Alireza Fathi. 2023. Reveal: Retrieval-augmented visual-language pre-

training with multi-source multimodal knowledge memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Srinivasan Iyer, Sewon Min, Yashar Mehdad, and Wen-tau Yih. 2021. Reconsider: Re-ranking using span-focused cross-attention for open domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

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

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *International Conference on Learning Representations*.

Gautier Izacard and Edouard Grave. 2021b. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4).

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss I. 1990. Self-organized language modeling for speech recognition. In *Readings in Speech Recognition*.

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

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Rie Johnson and Tong Zhang. 2016. Supervised and semi-supervised text categorization using lstm for region embeddings. In *Proceedings of The 33rd International Conference on Machine Learning*, Proceedings of Machine Learning Research.

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

Jia-Huei Ju, Jheng-Hong Yang, and Chuan-Ju Wang. 2021. Text-to-text multi-view learning for passage re-ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

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

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Relevance-guided supervision for openqa with colbert. *Transactions of the Association for Computational Linguistics*, 9:929–944.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv preprint arXiv:2212.14024*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

R. Kneser and H. Ney. 1995. Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Kenton Lee, Kelvin Guu, Luheng He, Tim Dozat, and Hyung Won Chung. 2021. Neural data augmentation via example extrapolation. *arXiv preprint arXiv:2102.01335*.

Kenton Lee, Tom Kwiatkowksi, Ankur Parikh, and Dipanjan Das. 2017. Learning recurrent span representations for extractive question answering.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient

prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. In *Advances in Neural Information Processing Systems*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020b. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, F. Petroni, V. Karpukhin, Naman Goyal, Heinrich Kuttler, M. Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020c. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*.

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. 2020. Pytorch distributed: Experiences on accelerating data parallel training. *Proc. VLDB Endow.*

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. 2022. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga,

Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Valentin Liévin, Andreas Geert Motzfeldt, Ida Riis Jensen, and Ole Winther. 2023. Variational open-domain question answering. In *Proceedings of the 40th International Conference on Machine Learning*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021a. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2021b. Pretrained Transformers for Text Ranking: Bert and Beyond. *Synthesis Lectures on Human Language Technologies*, 14(4):1–325.

Erik Lindgren, Sashank J. Reddi, Ruiqi Guo, and Sanjiv Kumar. 2021. Efficient training of retrieval models using negative cache. In *Advances in Neural Information Processing Systems*.

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

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, Dense, and Attentional Representations for Text Retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021a. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. In *Proceedings of the 16th*

Conference of the European Chapter of the Association for Computational Linguistics: Main Volume.

Xueguang Ma, Kai Sun, Ronak Pradeep, and Jimmy Lin. 2021b. A replication study of dense passage retriever. *arXiv preprint arXiv:2104.05740*.

Xueguang Ma, Xinyu Crystina Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *ArXiv*, abs/2305.02156.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Gábor Melis, Chris Dyer, and Phil Blunsom. 2018. On the state of the art of evaluation in neural language models. In *International Conference on Learning Representations*.

Thomas Mensink, Jasper Uijlings, Lluis Castrejon, Arushi Goel, Felipe Cadar, Howard Zhou, Fei Sha, André Araujo, and Vittorio Ferrari. 2023. Encyclopedic vqa: Visual questions about detailed properties of fine-grained categories. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *International Conference on Learning Representations*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

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

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*.

Nicholas Monath, Manzil Zaheer, Kelsey Allen, and Andrew Mccallum. 2023. Improving dual-encoder training through dynamic indexes for negative mining. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research.

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

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Cicero Nogueira dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Beyond [CLS] through ranking by generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Barlas Oguz, Kushal Lakhotia, Anchit Gupta, Patrick Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Scott Yih, Sonal Gupta, and Yashar Mehdad. 2022. Domain-matched pre-training tasks for dense retrieval. In *Findings of the Association for Computational Linguistics: NAACL 2022*.

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

OpenAI. 2022. Chatgpt blog post. https://openai.com/blog/chatgpt.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, et al. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

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

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Jay M. Ponte and W. Bruce Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. Rankvicuna: Zero-shot listwise document reranking with open-source large language models.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2023. Large language models are effective text rankers with pairwise ranking prompting. *ArXiv*, abs/2306.17563.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. Learning to retrieve passages without supervision. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1988. *Learning Representations by Back-Propagating Errors*, page 696–699.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Devendra Singh Sachan, Mike Lewis, Dani Yogatama, Luke Zettlemoyer, Joelle Pineau, and Manzil Zaheer. 2023. Questions Are All You Need to Train a Dense Passage Retriever. *Transactions of the Association for Computational Linguistics*, 11:600–616.

Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L Hamilton, and Bryan Catanzaro. 2021a. End-to-end training of neural retrievers for open-domain question answering. In *Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Devendra Singh Sachan, Siva Reddy, William L. Hamilton, Chris Dyer, and Dani Yogatama. 2021b. End-to-end training of multi-document reader and retriever for open-domain question answering. In *Advances in Neural Information Processing Systems*.

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

Mike Schuster and Kuldip Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021. Simple entity-centric questions challenge dense retrievers. In *Empirical Methods in Natural Language Processing (EMNLP)*.

C. E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. 2023. Replug: Retrieval-augmented black-box language models. *ArXiv*, abs/2301.12652.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. *arXiv preprint arXiv:1909.08053*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Chenglei Si, Chen Zhao, and Jordan Boyd-Graber. 2021. What's in a name? answer equivalence for open-domain question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Amit Singhal. 2001. Modern information retrieval: a brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24:2001.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Karen Sparck Jones. 1988. *A Statistical Interpretation of Term Specificity and Its Application in Retrieval*, page 132–142. Taylor Graham Publishing, GBR.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agent. *ArXiv*, abs/2304.09542.

Wilson L. Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. 2013. A theoretical analysis of ndcg type ranking measures. In *Proceedings of the 26th Annual Conference on Learning Theory*, volume 30 of *Proceedings of Machine Learning Research*, pages 25–54, Princeton, NJ, USA. PMLR.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019b. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceed-*

ings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).

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

Xing Wei and W. Bruce Croft. 2006. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Paul Werbos. 1988. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

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

Yuanzhong Xu, HyoukJoong Lee, Dehao Chen, Blake Hechtman, Yanping Huang, Rahul Joshi, Maxim Krikun, Dmitry Lepikhin, Andy Ly, Marcello Maggioni, et al. 2021. Gspmd: general and scalable parallelization for ml computation graphs. *arXiv preprint arXiv:2105.04663*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32*.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive Semiparametric Language Models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023. Generate rather than retrieve: Large language models are strong context generators. In *The Eleventh International Conference on Learning Representations*.

Chengxiang Zhai and John Lafferty. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2022. Adversarial retriever-ranker for dense text retrieval. In *International Conference on Learning Representations*.

Tianyi Zhang, Tao Yu, Tatsunori Hashimoto, Mike Lewis, Wen-Tau Yih, Daniel Fried, and Sida Wang. 2023. Coder reviewer reranking for code generation. In *Proceedings of the 40th International Conference on Machine Learning*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Giulio Zhou and Jacob Devlin. 2021. Multi-vector attention models for deep re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Jiawei Zhou, Xiaoguang Li, Lifeng Shang, Lan Luo, Ke Zhan, Enrui Hu, Xinyu Zhang, Hao Jiang, Zhao Cao, Fan Yu, Xin Jiang, Qun Liu, and Lei Chen. 2022. Hyperlink-induced pre-training for passage retrieval in open-domain question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.