Beyond Scaling: Frontiers of Retrieval-Augmented LMs

Akari Asai

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington
2025

Reading Committee:
Hannaneh Hajishirzi, Chair
Luke Zettlemoyer
Yulia Tsvetkov

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Beyond Scaling: Frontiers of Retrieval-Augmented Language Models

Akari Asai

Chair of the Supervisory Committee:
Hannaneh Hajishirzi
Computer Science and Engineering

Language Models (LMs) have made significant progress by scaling training data and model sizes. However, they still face key limitations, including hallucinations and outdated knowledge, which undermine their reliability especially in expert domains like scientific research and software development. In this thesis, I argue that overcoming these challenges requires moving beyond monolithic LMs toward Augmented LMs: systems that are designed, trained, and deployed alongside complementary modules to improve reliability and efficiency. Specifically, my work has pioneered the field of Retrieval-Augmented LMs, which precisely locate relevant knowledge from large-scale text data and incorporate them at inference time. I begin by analyzing the limitations of current LMs and demonstrate how retrieval augmentation provides a more reliable, adaptable, and efficient path forward. I then introduce our work on establishing new foundations for Retrieval-Augmented LMs, moving beyond simple post-hoc combinations of off-the-shelf models to tackle challenges driven by broader adoption. Finally, I highlight the real-world impact of Retrieval-Augmented LMs through applications in domains such as scientific literature synthesis. Our fully open OPENSCHOLAR system are now used by over 30,000 researchers. I conclude by outlining our vision for the future of Augmented LMs, including better handling of heterogeneous modalities, flexible integration with diverse components, and rigorous interdisciplinary evaluation.

Acknowledgments

I am deeply grateful to my advisor, Hannaneh Hajishirzi. When I applied to Ph.D. programs in the U.S., I had no major publications, only a strong desire to grow as a researcher. Hanna took a chance on me, and over the past six incredible years, she has been endlessly patient, supportive, and thoughtful. She taught me how to think both critically and creatively, how to pursue complex research ideas with care and clarity, how to teach and mentor junior students, and how to grow into an independent researcher. Despite her many responsibilities, she has always made time whenever I needed her support or advice. I truly couldn't have asked for a better advisor.

I would like to sincerely thank my committee members—Luke Zettlemoyer, Yulia Tsvetkov, Pang Wei Koh, and Lucy Lu Wang. Since my very first day at UW, Luke has been incredibly kind, thoughtful, and welcoming. From him, I've learned not only how to become a stronger researcher, but also how to be a better person. I feel incredibly fortunate to have closely collaborated with Yulia and Pang Wei over the past few years. Yulia's creativity and dedication to high-quality research have always inspired me, and her warmth and encouragement helped me stay grounded and resilient during difficult moments. Pang Wei's close mentorship has profoundly shaped my growth. He taught me how to tackle ambitious projects with persistence and to think deeply about building research that makes a real-world impact. I'm also deeply grateful to Lucy for serving on my committee. Her work has inspired me for years, and I feel truly honored to have her guidance at this important stage of my journey.

I would also like to thank the incredible mentors in the UW CSE community and beyond, with whom I've had the great fortune to work closely during my Ph.D. Every interaction I had with Noah Smith and Yejin Choi left a lasting impression. I learned not only about research, but also

about teaching, mentorship, and the importance of being a thoughtful and grounded person. I would like to thank Ranjay Krishna for his incredibly helpful feedback on my job talk and Magda Balazinska for providing all of necessary support. Wen-tau (Scott) Yih has provided unwavering mentorship and support throughout my time at Meta FAIR over the past couple of years and I can't thank enough for his guidance and support. Kazuma Hashimoto, Caiming Xiong, and Richard Socher guided me through my first major conference publication and taught me the fundamentals of becoming a strong researcher. I would also like to thank Matt Gardner for his thoughtful mentorship during my internship at AI2. I am deeply grateful to Eunsol Choi, Danqi Chen, and Graham Neubig for their outstanding mentorship. I've truly enjoyed collaborating with them across multiple research projects, workshops, and a tutorial. Their insight, support, and generosity have meant a great deal to me throughout my Ph.D. journey. I'm deeply grateful to Matei Zaharia for his generous support and encouragement throughout my job search. Finally, I want to thank my first advisors during my undergrad, Kiyoharu Aizawa and Yoshimasa Tsuruoka, at the University of Tokyo, and Alon Halevy. I am forever grateful for their early mentorship and their support of my Ph.D. applications, which set me on this path.

My Ph.D. has been the most fun and fulfilling time of my life, thanks to the amazing friends I've met from all over the world. When I started, I was nervous and unsure of myself, but the support I received from my friends in the CSE 318 office—Victor Zhong, Sewon Min, Tim Dettmers, Yizhong Wang, and Jungo Kasai—helped me find my footing. I'm deeply grateful that we've remained close friends ever since. I want to thank Victor, who has supported me from my Ph.D. application all the way to the job market. His friendship, encouragement, and generosity have meant the world to me. Sewon has always been my go-to person for everything, and her presence has been a constant source of grounding and comfort. Tim sat right behind me in the office, and we shared countless fun conversations over coffee, a tradition we've continued after both of us moving to the same city. I've also had so many fun and thoughtful conversations with Jungo and Yizhong, and truly enjoyed collaborating with them on multiple projects. I've learned so much from their creativity, insight, and generous perspectives. The UW NLP family is simply incredible. I thank a close collaborations and friendship with Weijia Shi, Rulin Shao, Jacqueline He, Stella Li, Tong

Chen, Niloofar Mireshghallah, Hamish Ivison, Victoria Graf, Zhiyuan Zeng, Scott Geng, Rui Xin, Ananya Harsh Jha, Joongwon Kim, Zeqiu Wu, Barghavi Paranjape, Jiacheng Liu, Reza Salehi, Yanai Elazar, Sarah Wiegreffe, Sachin Kumar, Rik Koncel-Kedziorski, Gabriel Ilharco, Ofir Press, Suchin Gururangan, Julian Michael, Mandar Joshi, Margaret Li, Hao Peng, Jesse Dodge, Wenya Wang, Ana Marasović, Tao Yu, Terra Blevins, Hila Gonen, Victoria Lin, Orevaoghene Ahia, Inna Wanyin Lin, Liwei Jiang, Alisa Liu, Peter West, Saadia Gabriel, Ben Newman, Ximing Lu, Jaehun Jung, Melanie Sclar, David Wadden, Vidhisha Balachandran, Prithviraj Ammanabrolu, Alane Suhr, Xiang Lorraine Li, Daniel Khashabi, Rajarshi Das, Maarten Sap, Ari Holtzman and Antoine Bosselut. I also thoroughly enjoyed working with my incredible undergraduate mentees, Velocity Yu, Go Kamoda, Abhika Mishra, and Alex Mallen, whose creativity, passion, and resilience have constantly inspired me. I had such a great time with Mohit Shridhar, Ivan Evtimov, and Sally Dong—from our pandemic hikes to fun nights at breweries. I'm deeply grateful to my friends from Japan who were part of my life during my time in U.S.—Yuka Ikarashi, Yudai Tanaka, Lisa Orii, Motoya Ohnishi, Kazuya Echigo, Yoshihide Arai, Akiko Eriguchi, Yosuke Tanigawa, Noah Wang, Yasutaka Tanaka, Ryo Suzuki, Kashu Yamazaki, Riku Arakawa, and Kimihiro Hasegawa—for all the fun times and constant support. The laughter, encouragement, and shared experiences we had together meant so much to me. I am sincerely thankful for the incredible support from the UW CSE grad advising team, especially Joe Eckert and Elise Dorough, and Chiemi Yamaoka. They were always available to answer my last-minute questions and provided unwavering support throughout my time in the program. I'm also deeply grateful to Sandy Kaplan, our amazing resident technical writer and editor, who generously proofread and suggested edits for many of my papers and statements.

During my Ph.D., I was extremely fortunate to closely work with many truly amazing researchers across institutions. I thank my fellow collaborators: Jonathan H Clark, Kenton Lee, Alon Talmor, Ori Yoran, Jonathan Berant, Ikuya Yamada, Matthew E Peters, Shayne Longpre, Rui Zhang, Trina Chatterjee, Junjie Hu, Zexuan Zhong, Keisuke Sakaguchi, James Grimmelmann, Machel Reid, Sebastian Ruder, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Odunayo Ogundepo, Tajuddeen R Gwadabe, Clara E Rivera, David Ifeoluwa Adelani, Bonaventure FP Dossou, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Dan Weld, Doug

Downey, Xiang Yue, Yueqi Song, Seungone Kim, Jean de Dieu Nyandwi, Simran Khanuja, Anjali Kantharuban, Lintang Sutawika, Sathyanarayanan Ramamoorthy, Zora Zhiruo Wang, Frank F Xu, Yiqing Xie, Daniel Fried and many others.

Last but certainly not least, I want to thank my family, who have loved, supported, and cheered me on throughout this journey. My mother, Noriko, has always been my role model. She inspired me to pursue my academic interests and to dream of becoming a professor. I can't imagine the hardships and sacrifices she has endured, and I am forever grateful to have such a strong, compassionate, and extraordinary woman as my mother. I've also been incredibly lucky to have the kindest and funniest older sibling, Chiaki, who's been there for me since day one. I'm grateful to my wonderful family in-laws, who welcomed me into the family with such warmth and open hearts. Your support has meant the world to us.

And finally, I cannot thank Sean enough for being by my side through the highest highs and the lowest lows, always giving me exactly what I needed, even before I realized it myself. Your unwavering support, unconditional love, and steady encouragement have made me a stronger, more grounded person. Beyond the joyful moments and adventures that sustained me throughout my Ph.D., there were times when I nearly gave up on the projects that mattered most or hesitated to pursue opportunities that could shape my future. You kept believing in me and encouraging me to keep going, and that made all the difference. Since our first conversation at the Burke Museum, you have brought so much joy, love, meaning, and color into my life. I'm endlessly grateful to have you by my side, and I can't wait to see what we'll build and discover together in this next chapter.

DEDICATION

To my family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

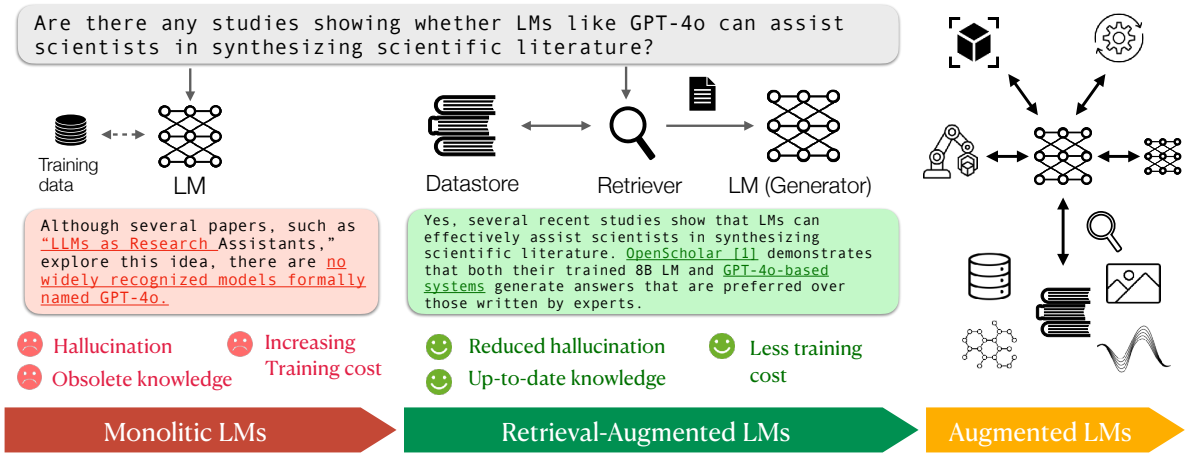


Figure 1.1: Roadmap from Monolithic LMs to Retrieval-Augmented LMs to Augmented LMs.

1.1 Motivation

In the field of Natural Language Processing (NLP), a wide range of techniques, spanning symbolic [Weizenbaum, 1966; Hutchins, 2004], statistical [Brown et al., 1993], and neural methods [Bahdanau et al., 2015], have historically been developed to address diverse tasks involving natural language understanding and generation. Traditionally, separate models were built for each specific task, often requiring extensive human effort, such as defining hand-crafted rules, architectural designs or substantial amounts of manually labeled data for each task.

The rapid advancement of pre-trained language models (LMs) has significantly reshaped the NLP landscape, demonstrating strong performance across a broad range of tasks [Brown et al., 2020], even without explicit task-specific training. These models are, at their core, neural networks—typically based on transformer architectures [Vaswani et al., 2017] with a massive number of trainable parameters, often referred to as parametric LMs. They are pre-trained on large-scale text corpora using a simple next-token prediction objective [Brown et al., 2020], and subsequently refined through extensive post-training procedures [Ouyang et al., 2022]. With continued scaling of both model size and training data [Hoffmann et al., 2022], modern LMs encapsulate extensive world knowledge and exhibit a remarkable ability to follow complex and novel instructions at inference time. Today, LM-based systems are widely deployed across society, influencing many aspects of our society.

However, these giant **monolithic LMs** (Figure 1.1 left) often encounter critical limitations in high-stakes real-world applications such as the synthesis of the scientific literature or expert decision-making domains. As illustrated in Figure 1.1, such models frequently generate factually incorrect statements such as making up non-existent papers [Mallen, Asai, et al., 2023; Asai et al., 2024a], or by relying on outdated parametric knowledge from their pretraining data e.g., incorrectly claims that "*there are no widely recognized models called GPT-4o*" despite that the model was released in 2024 [Kasai et al., 2022]. Moreover, although such errors are widespread, it remains challenging to trace model outputs to verifiable sources, making it difficult to explain or attribute the basis of these generations [Bohnet et al., 2022]. These challenges become particularly pronounced for knowledge-intensive tasks that require accurate and verifiable outputs, where scaling models alone fail to address fundamental deficiencies. Nevertheless, such models are widely deployed for these tasks, leading to issues such as the spread of misinformation [Metz and Weise, 2025].

We advocate for a paradigm shift from monolithic LMs to **Augmented LMs** [Asai et al., 2023a], a class of models that are designed, trained and deployed in the conjunction with complementary. Specifically, this thesis studies **Retrieval-Augmented LMs** (Figure 1.1 middle), which retrieve and incorporate external knowledge at inference time. Retrieval-Augmented LMs consist of three key components: a datastore, a retriever, and an LM. The datastore is a collection of documents,

potentially spanning billions to trillions of tokens. The retriever selects a small subset of documents relevant to a given input from this large corpus. The LM then conditions its generation on the retrieved content to produce informed, context-aware outputs. By retrieving and incorporating relevant knowledge at inference time, this framework improves factual accuracy, enables transparent attribution, and allows systems to remain up-to-date without the need for retraining. While Retrieval-Augmented LMs have a long history [Chen et al., 2017; Guu et al., 2020; Lewis et al., 2020b], earlier work primarily focused on training individual neural models for specific downstream tasks, particularly question answering (QA). However, their unique advantages have positioned Retrieval-Augmented LMs as a general framework for addressing key limitations of monolithic LMs.

We pioneered and advanced this paradigm by addressing three fundamental questions about Retrieval-Augmented LMs:

- *Why are Retrieval-Augmented LMs necessary, as opposed to simply scaling monolithic LMs?* (Chapter 3)
- *How can we build new foundations for Retrieval-Augmented LMs that make them broadly applicable across domains and tasks and fully unlock their potential?* (Chapter 4)
- *What real-world impacts can we achieve by leveraging these foundations?* (Chapter 5)

Together, our work has advanced the understanding, modeling, and application of Retrieval-Augmented LMs, demonstrating their effectiveness across tasks and domains. This has helped spark growing interest and rapid progress in both industry and academia. Retrieval-Augmented LMs are now widely adopted across our society including generative search engines like ChatGPT-Search[1] and Perplexity[2] that leverage these models to help users access information easily.

1.2 Thesis Outline

The following chapters are structured to address each of these directions in turn.

[1] https://openai.com/index/introducing-chatgpt-search/
[2] https://www.perplexity.ai/

Chapter 2: Overview of Retrieval-Augmented LMs. We begin with a brief overview of Retrieval-Augmented LMs, introducing key definitions and terminology. We then trace the evolution of the field over the past decade, highlighting a shift from building task-specific retrieval-augmented models to developing general-purpose frameworks built on top of pre-trained LMs. This shift has broadened the applicability of Retrieval-Augmented LMs, while also calling for the development of more versatile and scalable systems. As the field has progressed, it has led to a diverse range of architectures, training strategies, and applications to new areas. Finally, we review common evaluation protocols, datasets, and metrics that have evolved to better capture the capabilities of modern systems. This chapter provides a foundation for understanding how Retrieval-Augmented LMs are designed, trained, and assessed, setting the stage for the more detailed discussions that follow in this thesis.

Chapter 3: Establishing the Necessity of Retrieval-Augmented LMs. A prevailing belief in the NLP community has been that scaling LMs by increasing the size of training data and model parameters would eventually overcome many of their limitations [Roberts et al., 2020; Brown et al., 2020; Chowdhery et al., 2022]. However, our work challenges this assumption by demonstrating that many of these issues arise from the monolithic nature of LMs. We show that Retrieval-Augmented LMs offer a more effective and efficient path forward, improving both performance and training-time efficiency. Specifically, Retrieval-Augmented LMs can significantly reduce hallucinations, which refer to factual inaccuracies in model outputs, particularly in long-tail knowledge scenarios, which are underrepresented during pre-training [Mallen, Asai, et al., 2023; Mishra et al., 2024]. We conducted one of the earliest large-scale studies on LM hallucinations, revealing that even models with 175 billion parameters (e.g., GPT-3; Brown et al. 2020) trained on massive pre-training data struggle to reliably memorize long-tail facts. These findings indicate that scaling alone is insufficient to resolve hallucination. In contrast, Retrieval-Augmented LMs mitigate this issue by incorporating non-parametric memory through large-scale retrieval, enabling accurate and up-to-date knowledge access. Retrieval-Augmented LMs also improves their training efficiency by reducing the need for frequent parameter updates [Kasai et al., 2022] and eliminating the need to encode all knowledge in the model parameters [Shao et al., 2024].

Chapter 4: New Foundations of Retrieval-Augmented LMs. While Retrieval-Augmented LMs have gained significant public attention, early approaches often relied on off-the-shelf LMs and retrieval systems connected via a static two-stage pipeline. This rigid design, lacking training or model development specifically tailored for retrieval augmentation, exposes core limitations such as contradictions between generated content and citations, as well as brittleness to irrelevant context resulting from failed retrieval. We address these challenges by developing new retrieval systems and LMs for Retrieval-Augmented LMs. In particular, we focus on (1) training new LMs that can incorporate retrieval more effectively and efficiently through dynamic integration, and (2) developing versatile and efficient retrieval systems that support a wide range of use cases beyond simple QA. First, we introduce Self-Reflective Retrieval-Augmented Generation (SELF-RAG; Asai et al. 2024c), which trains LMs to decide when to retrieve, when to generate, and how to self-evaluate their outputs. By incorporating novel training objectives and inference-time strategies, SELF-RAG makes Retrieval-Augmented LMs more flexible, controllable, and reliable. This framework opens new directions for training LMs specifically for retrieval-augmented inference and have been widely adopted in both academic research and industry applications. Next, we demonstrate how our newly developed retrieval systems including the first instruction-following retrieval systems further enhance the effectiveness of Retrieval-Augmented LMs by efficiently identifying relevant context across diverse tasks from massive datastores [Asai et al., 2020, 2023b; Lin et al., 2023; Yamada et al., 2020, 2021; Shao et al., 2024]. Our new foundations for Retrieval-Augmented LMs have shaped today's state-of-the-art systems, opened up new research directions, and been widely adopted to build more versatile and reliable models.

Chapter 5: Real-World Impacts with Retrieval-Augmented LMs. Finally, we demonstrate how Retrieval-Augmented LMs can be deployed to address real-world challenges in three key domains: scientific research, code generation, and multilingual information access. In these settings, conventional LMs often struggle due to limited training data and the need for precise, up-to-date knowledge. We first introduce OPENSCHOLAR [Asai et al., 2024a], the first fully open Retrieval-Augmented LM designed to assist scientists with literature review tasks. To support rigorous evaluation, we also released SCHOLARQABENCH, a benchmark covering four scientific disciplines

Figure 1.2: OPENSCHOLAR public demo interface. Our demo is available at `https://openscholar.allen.ai/`. OPENSCHOLAR, built entirely from open components, provides comprehensive answers supported by extensive citations including the original paper [Asai et al., 2024a]. The paper was released alongside the demo, enabled by our up-to-date datastore.

with expert-curated, citation-rich long-form answers. OPENSCHOLAR leverages a massive, up-to-date scientific corpus and a self-reflective generation process to improve factuality and citation accuracy. Our experiments show that OPENSCHOLAR outperforms proprietary models and is even

preferred over expert-written answers. The system has been released as a public demo (shown in Figure 1.2) and has already been used by over 30,000 researchers and practitioners.

We further show that such holistic approaches are also effective for another expert domain. We introduce CODERAG-BENCH [Wang, Asai, et al., 2025], a new benchmark to enable rigorous testing of Retrieval-Augmented LMs for code generations across diverse code generation scenarios. CODERAG-BENCH highlights both the effectiveness and limitations of current Retrieval-Augmented LMs in incorporating diverse context such as code snippets and documentation to improve code generation. We then discuss our extensive efforts on addressing global information access disparities with Retrieval-Augmented LMs. Many world languages suffer from limited web presence, which restricts conventional LMs [Yu, Asai, et al., 2022; Asai et al., 2024b]. We pioneered the task of cross-lingual retrieval-augmented generation, where models retrieve and reason over documents written in different languages. We released XORQA [Asai et al., 2021a], the first large-scale benchmark in this space, which has inspired multiple shared tasks and follow-up datasets [Ogundepo et al., 2023; Shen et al., 2023; Asai et al., 2022b]. Finally, we introduced CORA [Asai et al., 2021b], the first end-to-end multilingual Retrieval-Augmented LM that performs retrieval and generation across languages using a single model. Together, we demonstrate that cross-lingual Retrieval-Augmented LMs can significantly improve information access, particularly in under-resourced languages such as many African languages.

Chapter 6: Conclusion and Future Work. We conclude this thesis by summarizing key findings and outlining future directions for generalizing Retrieval-Augmented LMs toward building **Augmented LMs** (Figure 1.1, right), systems that integrate diverse modules to jointly optimize overall performance and efficiency, while enabling new frontier applications.

1.3 Contributions

The contributions of this thesis are summarized as follows:

- We pioneered Retrieval-Augmented LMs, demonstrating their effectiveness in addressing fundamental limitations of scaling monolithic LMs. Our studies show that Retrieval-Augmented

LMs can significantly reduce hallucinations and alleviate the high training costs of conventional LMs, leading to their rapid adoption across academia and industry.

- We contributed key technical foundations for Retrieval-Augmented LMs, including the development of new generation models such as SELF-RAG, as well as novel retrieval components like instruction-following retrievers. These advances enable systems to flexibly adapt to diverse user needs, influencing today's widely used systems.
- Finally, we show how these technical foundations can unlock high-impact, real-world applications in domains where LMs typically struggle due to the long-tail nature of knowledge or the need for frequent updates. In particular, we developed OPENSCHOLAR, the first fully open Retrieval-Augmented LM for scientific literature synthesis, and further demonstrated its effectiveness in expert domains such as code generation and multilingual information access.

Prior Publications

The research presented in this dissertation is heavily based on the following jointly authored prior publications (* denotes equal contribution).

1. **Akari Asai**, Sewon Min, Zexuan Zhong, Danqi Chen. "Retrieval-based Language Models and Applications". In: Proceedings of ACL (Tutorial). 2023. [pdf] - In Chapter 2
3. Alex Mallen*, **Akari Asai***, Victor Zhong, Rajarshi Das, Daniel Khashabi, Hannaneh Hajishirzi. "When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories". In: Proceedings of ACL. 2023. [pdf] - In Chapter 3
4. **Akari Asai**, Zeqiu Wu, Yizhong Wang, Avirup Sil, Hannaneh Hajishirzi. "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection". In: Proceedings of ICLR. 2024. [pdf] - In Chapter 4
5. **Akari Asai**, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, Wen-tau Yih. "Task-aware Retrieval with Instructions". In: Proceedings of Findings of ACL. 2023. [pdf] - In Chapter 4
6. **Akari Asai**, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'arcy, David Wadden, Matt Latzke, Minyang Tian,

Pan Ji, Shengyan Liu, Hao Tong, Bohao Wu, Yanyu Xiong, Luke Zettlemoyer, Graham Neubig, Dan Weld, Doug Downey, Wen-tau Yih, Pang Wei Koh, Hannaneh Hajishirzi. "OpenScholar: Synthesizing Scientific Literature with Retrieval-augmented LMs". Under Review for Nature. 2025. [pdf] - In Chapter 5

The following jointly authored prior publications are also briefly mentioned in this dissertation.

7. **Akari Asai**, Zexuan Zhong, Danqi Chen, Pang Wei Koh, Luke Zettlemoyer, Hannaneh Hajishirzi, Wen-tau Yih. "Reliable, adaptable, and attributable language models with retrieval". 2024. [pdf] - In Chapter 2
8. Rulin Shao, Jacqueline He, **Akari Asai**, Weijia Shi, Tim Dettmers, Sewon Min, Luke Zettlemoyer, Pang Wei Koh. "Scaling Retrieval-Based Language Models with a Trillion-Token Datastore". In: Proceedings of NeurIPS. 2024. [pdf] - In Chapter 3
9. Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, **Akari Asai**, Xinyan Yu, Dragomir Radev, Noah A. Smith, Yejin Choi, Kentaro Inui. "RealTime QA: What's the Answer Right Now?". In: Proceedings of NeurIPS (Dataset and Benchmarks). 2023. [pdf] - In Chapter 3
10. Go Kamoda, **Akari Asai**, Ana Brassard, Keisuke Sakaguchi. "Quantifying the Influence of Evaluation Aspects on Long-Form Response Assessment". In: Proceedings of COLING. 2025. [pdf] - In Chapter 3
11. Tong Chen, **Akari Asai***, Niloofar Mireshghallah*, Sewon Min, James Grimmelmann, Yejin Choi, Hannaneh Hajishirzi, Luke Zettlemoyer, Pang Wei Koh. "CopyBench: Measuring Literal and Non-Literal Reproduction of Copyright-Protected Text in Language Model Generation". In: Proceedings of EMNLP. 2024. [pdf] - In Chapter 3
12. Abhika Mishra, **Akari Asai**, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, Hannaneh Hajishirzi. "Fine-grained Hallucination Detection and Editing for Language Models". In: Proceedings of COLM. 2024. [pdf] - In Chapter 3
13. **Akari Asai**, Matt Gardner, Hannaneh Hajishirzi. "Evidentiality-guided Generation for Knowledge-Intensive NLP Tasks". In: Proceedings of NAACL. 2022. [pdf] - In Chapter 4
14. **Akari Asai**, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, Caiming Xiong. "Learning to Retrieve Reasoning Paths over Wikipedia Graph for Question Answering". In: Proceed-

ings of ICLR. 2020. [pdf] - In Chapter 4

15. Sheng-Chieh Lin, **Akari Asai**, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, Xilun Chen. "How to Train Your DRAGON: Diverse Augmentation Towards Generalizable Dense Retrieval". In: Proceedings of Findings of EMNLP. 2023. [pdf] - In Chapter 4
16. Ikuya Yamada, **Akari Asai**, Hannaneh Hajishirzi. "Efficient Passage Retrieval with Hashing for Open-domain Question Answering". In: Proceedings of ACL. 2021. [pdf] - In Chapter 4
17. Ikuya Yamada, **Akari Asai**, Hiroyuki Shindo, Hideaki Takeda, Yuji Matsumoto. "LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention". In: Proceedings of EMNLP. 2020. [pdf] - In Chapter 4
18. Zora Zhiruo Wang*, **Akari Asai***, Xinyan Velocity Yu, Frank F. Xu, Yiqing Xie, Graham Neubig, Daniel Fried. "CodeRAG-Bench: Can Retrieval Augment Code Generation?". In: Proceedings of Findings of NAACL. 2025. [pdf] - In Chapter 5
19. **Akari Asai**, Jungo Kasai, Jonathan H. Clark, Kenton Lee, Eunsol Choi, Hannaneh Hajishirzi. "XOR QA: Cross-lingual Open-Retrieval Question Answering". In: Proceedings of NAACL. 2021. [pdf] - In Chapter 5
20. **Akari Asai**, Xinyan Yu, Jungo Kasai, Hannaneh Hajishirzi. "One Question Answering Model for Many Languages with Cross-lingual Dense Passage Retrieval". In: Proceedings of NeurIPS. 2021. [pdf] - In Chapter 5
21. **Akari Asai**, Sneha Kudugunta, Xinyan Velocity Yu, Terra Blevins, Hila Gonen, Machel Reid, Yulia Tsvetkov, Sebastian Ruder, Hannaneh Hajishirzi. "BUFFET: Benchmarking Large Language Models for Few-shot Cross-lingual Transfer". In: Proceedings of NeurIPS. 2021. [pdf] - In Chapter 5
22. Odunayo Ogundepo, Tajuddeen R. Gwadabe, Clara E. Rivera, Jonathan H. Clark, Sebastian Ruder, David Ifeoluwa Adelani, Bonaventure F. P. Dossou, Abdou Aziz DIOP, Claytone Sikasote, Gilles Hacheme, Happy Buzaaba, Ignatius Ezeani, Rooweither Mabuya, Salomey Osei, Chris Emezue, Albert Njoroge Kahira, Shamsuddeen H. Muhammad, Akintunde Oladipo, Abraham Toluwase Owodunni, Atnafu Lambebo Tonja, Iyanuoluwa Shode, **Akari Asai**, Tunde Oluwaseyi Ajayi, Clemencia Siro, Steven Arthur, Mofetoluwa Adeyemi, Orevaoghene Ahia, Anuoluwapo Aremu, Oyinkansola Awosan, Chiamaka Chukwuneke, Bernard Opoku, Awokoya

Ayodele, Verrah Otiende, Christine Mwase, Boyd Sinkala, Andre Niyongabo Rubungo, Daniel A. Ajisafe, Emeka Felix Onwuegbuzia, Habib Mbow, Emile Niyomutabazi, Eunice Mukonde, Falalu Ibrahim Lawan, Ibrahim Said Ahmad, Jesujoba O. Alabi, Martin Namukombo, Mbonu Chinedu, Mofya Phiri, Neo Putini, Ndumiso Mngoma, Priscilla A. Amuok, Ruqayya Nasir Iro, Sonia Adhiambo. "AfriQA: Cross-lingual Open-Retrieval Question Answering for African Languages". In: Proceedings of Findings of EMNLP. 2023. [pdf] - In Chapter 5

23. Xiaoyu Shen, **Akari Asai**, Bill Byrne, Adrià de Gispert. "xPQA: Cross-Lingual Product Question Answering across 12 Languages". In: Proceedings of ACL (Industry). 2023. [pdf] - In Chapter 5

Chapter 2

Overview of Retrieval-Augmented LMs

In this chapter, we provide a brief overview of Retrieval-Augmented LMs. We begin by introducing key definitions, terminology, and preliminaries (Section 2.1), followed by a discussion of how the field has evolved over the past decade (Section 2.2). Next, we present a high-level taxonomy to classify the diverse types of Retrieval-Augmented LMs (Section 2.3). In particular, we categorize these models based on their architectures (Section 2.3.1) and training strategies (Section 2.3.2). Finally, we briefly review common evaluation protocols including datasets and metrics (Section 2.3.3).

2.1 Definition and Preliminary

Definitions and Preliminary of LMs

LMs have a long history, ranging from early statistical approaches to modern neural approaches [Rosenfeld, 2000]. In this thesis, we focus on neural, or **parametric LMs**, that train a neural network to predict the continuation of a sequence given a prefix [Bengio et al., 2000]. Formally, given an LM with learnable parameters θ and an input x, the model predicts an output y according to a function $y = f_\theta(x)$. We refer to these models as **monolithic LMs** in this thesis because they encapsulate all knowledge and task capabilities within their parameters, without relying on any external modules.

There exists a variety of parametric LMs, differing in both their training objectives and architectures. **Masked LMs** predict missing or masked tokens in a sequence based on surrounding context

(e.g., BERT [Devlin et al., 2019], RoBERTa [Liu et al., 2019b]), while **autoregressive LMs** predict the next token(s) given a left-to-right prefix (e.g., GPT [Radford et al., 2019], GPT-3 [Brown et al., 2020]).

Architecturally, models are commonly categorized as **encoder-only** (e.g., BERT, RoBERTa), **encoder–decoder** (e.g., T5 [Raffel et al., 2020]), or **decoder-only** (e.g., GPT series). Many state-of-the-art models nowadays adopt the decoder-only approach based on Transformer architecture [Vaswani et al., 2017], are pre-trained on large corpora, and generate text in an autoregressive manner. Recently, alternative architectures such as state space models (e.g., Mamba [Gu and Dao, 2024]) have also been also explored.

Definitions and Preliminary of Retrieval-Augmented LMs

Unlike monolithic LMs, which are trained and used in a standalone manner, a **Retrieval-Augmented LM** (Figure 1.1 middle) typically comprises three key components: a **datastore** $\mathcal{D}$, a **retriever** $\mathcal{R}$, and a **parametric LM (generator)** θ. A datastore consists of a large number of documents. In this thesis, we assume a document d is represented as text. The retriever builds a search index $\mathcal{I}$ based on documents in the datastore $\mathcal{D}$.

Formally, at inference time, given an input sequence x, we first construct a retrieval query q. The query q may be identical to x or derived from it via query reformulation. The retriever then finds relevant documents z from the inference datastore, leveraging an index $\mathcal{I}$:

$$z = f_{\mathcal{R},\mathcal{I}}(q)$$

The retrieved documents are typically the top-k entries from the datastore $\mathcal{D}$ that are most similar to the query q. Formally, they are selected as: $\{d_1, \ldots, d_k\} = \mathrm{TopK}_{d \in \mathcal{D}} \mathrm{Sim}(q, d)$, where $\mathrm{Sim}(q, d)$ denotes the similarity between the query q and a document d. The retrieved content z can vary in granularity (e.g., text chunks, tokens, or phrases); see Section 2.3.1 for more details.

There are many different ways to construct the index. In term-based retrieval systems such as BM25 [Robertson and Zaragoza, 2009] that count the occurrences of words in documents in the datastore, the index $\mathcal{I}$ is a weighted bag-of-words vector, while in more recent trainable neural retrieval systems such as DPR [Karpukhin et al., 2020], the index is a collection of float embeddings

encoded by an encoder LM.

Subsequently, the LM θ uses both the original input x and the retrieved content z to generate the output y:

$$y = f_\theta(x, z).$$

How to design Retrieval-Augmented LMs to effectively incorporate retrieved information z, as well as how to train the retriever $\mathcal{R}$ and LM θ, has been a central focus of research in this area. We review a range of architectural designs and training strategies in Section 2.3.

2.2 Brief History

In this section, we briefly review how the field has evolved over the past decade. While the concept of retrieval augmentation has been explored across various machine learning domains [Tian et al., 2019], this thesis primarily focuses on scenarios where both the input x and the output y are textual. Figure 2.2 provides a brief summary of key model and dataset developments, along with timeline of widely adopted pre-trained LMs that have significantly influenced model design and related consumer products.

Earlier Development of Task-specific Models

Model developments for QA. In NLP, early efforts on retrieval augmentation focused on specific tasks such as open-domain QA. Open-domain QA involves answering questions covering diverse topics using external knowledge sources [Chen and Yih, 2020]. Although the task was originally designed to leverage a large corpus of unstructured documents (e.g., TREC QA; Voorhees et al. 1999), the development of large-scale knowledge bases (KBs) such as DBpedia [Auer et al., 2007] led to the rise of QA systems built over structured KBs [Yih et al., 2015; Yao and Van Durme, 2014]. However, inherent limitations of KBs such as sparsity of coverage and a fixed schema make them difficult to use as the sole knowledge source to answer diverse questions.

Motivated by these limitations and the success of neural reading comprehension (RC) systems [Seo et al., 2017], Chen et al. [2017] introduced **DrQA**, which combined a term-based informa-

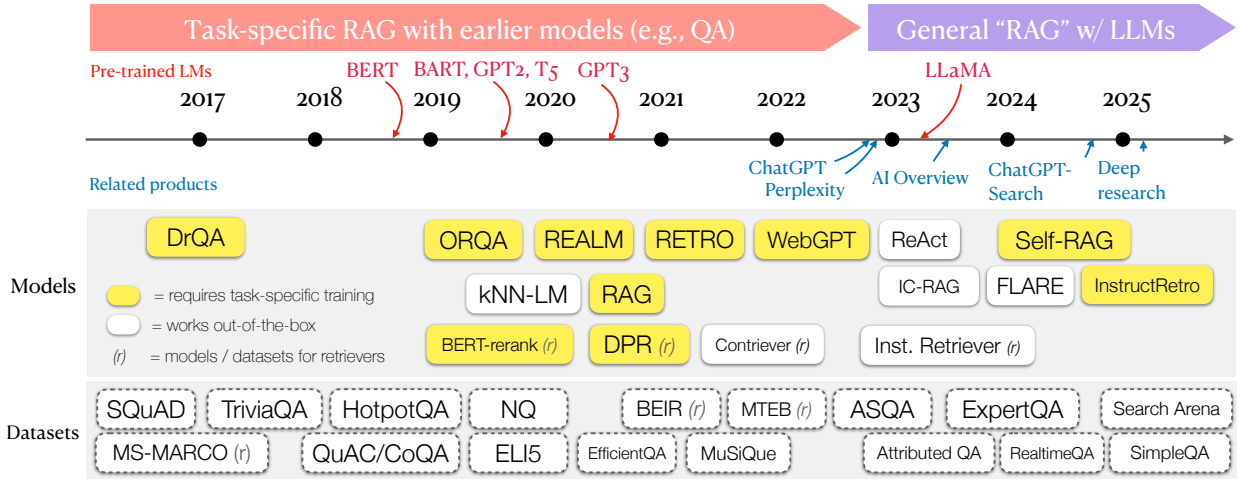


Figure 2.1: Brief history of the development of Retrieval-Augmented LMs. This timeline highlights key model and dataset developments in the past decade, with a focus on information-seeking and question answering. Entries marked with *(r)* are specifically related to retrieval, and yellow highlights indicate models that require downstream task-specific training.

tion retrieval (IR) system as the retriever $\mathcal{R}$, with a neural RC model as the generator θ.[1] This work spurred significant interest in developing neural retrievers [Wang et al., 2018; Asai et al., 2020], RC models [Clark and Gardner, 2018; Min et al., 2019], and their joint optimization [Das et al., 2019; Lin et al., 2018] for open-domain QA. The release of pre-trained BERT models [Devlin et al., 2019] further accelerated progress, as these models could be easily fine-tuned for a wide range of downstream tasks, including document and answer selection.

It is worth noting that early retrieval systems often combined a term-based retriever with a neural reranker, such as the **BERT-reranker** [Nogueira and Cho, 2019]. In contrast, **ORCA** [Lee et al., 2019] introduced unsupervised pre-training for neural retrieval models, enabling them to retrieve relevant passages directly from a datastore without relying on initial term-based retrieval. **Dense Passage Retrieval** (**DPR**; Karpukhin et al. 2020) proposed a dual-encoder architecture [Bromley et al., 1993] trained on question–answer pairs using a contrastive objective, thus avoiding the costly pre-training step required by ORCA. We will explore training strategies for retrieval systems in more detail later in this chapter.

[1] While early RC models were often extractive rather than generative like today's LMs, we refer to any model that processes retrieved text z and produces an output y as a generator throughout this thesis.

Pre-training LMs with retrieval and architectural development. While early work primarily focused on task-specific model development and was often evaluated only on open-domain QA, several studies began exploring more general pre-training of LMs with retrieval. **REALM** [Guu et al., 2020] augments LMs pre-training with a neural retriever, where both the retriever and a masked LM are trained jointly by maximizing the log-likelihood, with gradients back-propagated to the retriever. While effective for upstream language modeling and certain downstream tasks such as open-domain QA, where the final answer y can often be extracted directly from retrieved documents, these models frequently struggle to produce informative and fluent responses in more generation-oriented, non-extractive tasks.

Around 2019–2020, powerful pre-trained autoregressive decoder-only models like BART [Lewis et al., 2020a] and GPT-2 [Radford et al., 2019], as well as encoder–decoder models like T5 [Raffel et al., 2020], demonstrated strong performance across a range of tasks. **RAG** [Lewis et al., 2020b] builds on these advances by jointly training a pre-trained retriever (i.e., DPR) and a pre-trained autoregressive LM (i.e., BART), back-propagating the loss through the retriever in a manner similar to REALM. RAG proved effective not only for open-domain QA but also for generation-heavy tasks such as abstractive QA and question generation. **RETRO** [Borgeaud et al., 2022] similarly pre-trains a new autoregressive LM using a new architecture to efficiently incorporate retrieved documents (see Section 2.3 for more detailed discussions on architectural differences).

Follow-up work explores more effective and efficient architectures for encoder-decoder models [Izacard and Grave, 2021b; Asai et al., 2022a] and decoder-only LMs [de Jong et al., 2023] to improve information incorporation. These retrieval-augmented models have proven effective across various tasks—including long-form question answering [Asai et al., 2022a; Nakano et al., 2021], multi-turn dialogue [Shuster et al., 2022], and code generation [Parvez et al., 2021] when trained on task-specific, human-annotated data.

The Shift to General-Purpose Paradigms in LM Applications

The rapid rise of In-Context RAG. Most recently, there has been a shift in how Retrieval-Augmented LMs are viewed and developed. Rather than training retrieval or generation com-

ponents from scratch or fine-tune them for specific tasks, several studies [Ram et al., 2023; Shi et al., 2024; Mallen, Asai, et al., 2023] have proposed supplementing powerful existing parametric LMs (e.g., GPT-3; Black et al. 2022) with retrieval at inference time, without any additional model training. This approach is motivated by the strong ability of large LMs to adapt to diverse tasks via user instructions [Ouyang et al., 2022]. These methods, which we refer to as In-Context Retrieval-Augmented Generation (**IC RAG**), operate by concatenating the original input sequence x with retrieved documents z when prompting the model. Despite their simplicity, IC RAG has shown substantial performance gains on knowledge-intensive tasks. Note that this setting is often colloquially referred to as "Retrieval-Augmented Generation" or "RAG" today. However, we emphasize that such IC RAG is fundamentally different from the original RAG framework proposed by Lewis et al. [2021], which involves joint training of retrievers and generators. Building on this trend, many recent studies explore advanced prompting methods that integrate retrieval, such as **ReACT** [Yao et al., 2023] and **FLARE** [Jiang et al., 2023]. While these approaches often rely on proprietary models like GPT-3 or GPT-4, the release of strong open-source base LMs such as Llama2 [Touvron et al., 2023] and Llama3 [Dubey et al., 2024] has enabled further research on instruction-tuned Retrieval-Augmented LMs with retrieval. For example, **Self-RAG** (Chapter 4) aim to build general instruction-following LMs that more effectively incorporate retrieval during inference on top of such powerful LMs.

Broad adoption across society. Retrieval-Augmented LMs offer several unique advantages over purely parametric LMs, which we explore in detail in Chapter 3. These advantages, as well as the simplicity of IC RAG approaches have rapidly positioned them as a central approach in the development of LM-based systems, especially in areas requiring precision. As of February 2024, it is estimated that 60% of LLM applications incorporate some form of retrieval augmentation [Zaharia et al., 2024]. Many widely used customer-facing services now rely on Retrieval-Augmented LMs. One of the most successful applications to date is generative search engines or LM-based search [Liu et al., 2023a], where LMs summarize and generate long-form responses based on retrieved documents in response to users' diverse information-seeking queries. Notable examples

include Google's AI Overviews,[2] ChatGPT-Search,[3] and Deep Research.[4]

2.3 Current State of Retrieval-Augmented LMs

2.3.1 Architecture

Retrieval-Augmented LMs have diverse architectures. Our taxonomy defines architecture based on three axes (Table 2.1 left): what the unit of retrieved text z is (**granularity of** z), how z is incorporated (**incorporation of** z), and how often z is retrieved (**frequency of retrieval**). Here, we primarily classify approaches based on how they incorporate the retrieved text z (the Incorporation column in Table 2.1), and briefly discuss the other two aspects.

Incorporation of z

Essentially, Retrieval-Augmented LMs' architectures can be classified into the following three groups: 1) **input augmentation**, 2) **intermediate fusion**, and 3) **output interpolation**. Refer to Figure 2.2 for a taxonomy of these architectures. In essence, input augmentation and intermediate fusion typically involve retrieving text chunks and processing them with parametric LMs. On the other hand, output interpolation directly retrieves successive tokens or phrases.

Input augmentation. Input augmentation augments the original input x with retrieved results z in the input space of the LM θ and runs a standard LM inference. As in the pioneering work by Chen et al. [2017], input augmentation enables flexible plug-ins of different models for retrieval and LM components. Many widely adopted models, including recent IC RAG, mostly belong in this category [Yao et al., 2023; Mallen, Asai, et al., 2023; Shi et al., 2024; Ram et al., 2023]. One notable bottleneck of this approach is redundancy and inefficiency; encoding many documents together in the input space leads to context length window limitations and increases inference costs exponentially [Xu et al., 2024b]. While some work such as FiD [Izacard et al., 2022b] explores parallel encoding to overcome such inefficiencies, it still encodes repeatedly for each input x.

[2] `https://search.google/ways-to-search/ai-overviews/` (May 2024)
[3] `https://openai.com/index/introducing-chatgpt-search/` (October 2024)
[4] `https://openai.com/index/introducing-deep-research/` (February 2025)

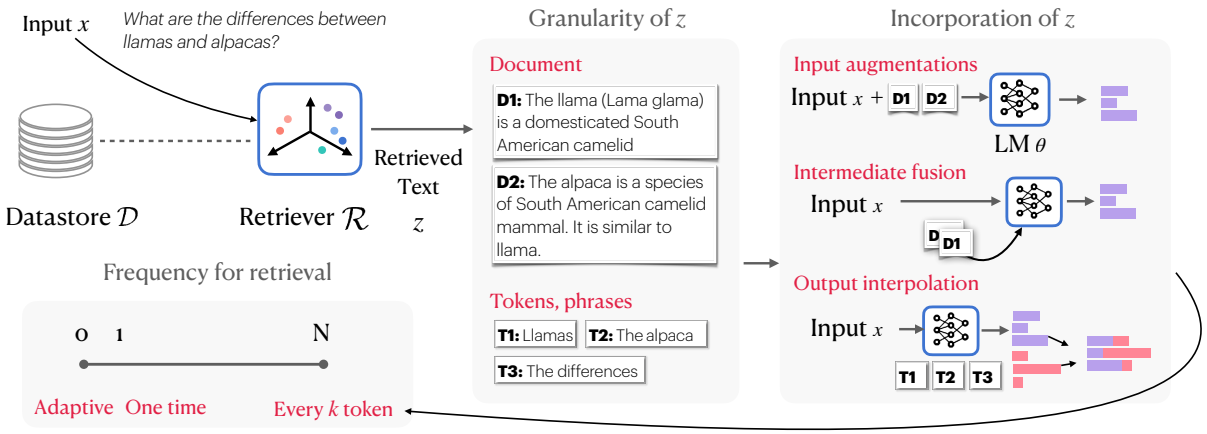


Figure 2.2: Taxonomy of Retrieval-Augmented LM Architectures. We categorize diverse Retrieval-Augmented LMs along three axes.

Intermediate fusion. RETRO [Borgeaud et al., 2022] introduces a new attention mechanism to integrate retrieved results in a more scalable manner, which takes many pre-encoded text chunks independent of query x and simultaneously incorporates them in intermediate spaces. RETRO++ [Wang et al., 2023a] and InstructRetro [Wang et al., 2024a] demonstrate the effectiveness of this method on top of larger, decoder-only LMs. However, a drawback of intermediate fusion is the need for extensive architecture modification and pre-training of LMs for the new encoding blocks, potentially limiting widespread adoption.

Output interpolation. Both input augmentation and intermediate fusion require the LM to generate continuations from their vocabularies. In contrast, kNN LM [Khandelwal et al., 2020] interpolates a parametric LM token distribution with a retrieved token distribution, without the need for additional training. Some work extends this direction by designing new training objectives [Zhong et al., 2022] or completely replacing parametric distributions with a non-parametric distribution over each phrase in the datastore [Min et al., 2023b; Lan et al., 2023]. While these approaches frequently demonstrate their efficacy compared to input augmentation in language modeling [Min et al., 2024], they require a considerably larger index than the other two architectures. This is due to the necessity of generating embeddings for all tokens in the datastore, presenting scalability challenges. Furthermore, recent studies show that kNN LMs and their variants may struggle in reasoning or open-ended text generation [Geng et al., 2025; Wang et al., 2023b].

Frequency of Retrieval

Another significant design choice in Retrieval-Augmented LMs is the frequency of retrieval. In essence, opting for more frequent retrieval tends to enhance performance, but comes at the expense of increased computational overhead. Retrieving once before generating given input x has been widely used such as REALM or DrQA, often in input space incorporation architectures. kNN LM, on the other hand, retrieves at every token, or some work retrieves every k token to maintain the relevance between the target sequence and retrieved context [Ram et al., 2023; Borgeaud et al., 2022]. Several recent papers introduce methods that make LMs adaptively decide when to retrieve [Jiang et al., 2023; Asai et al., 2024c].

Granularity of z and Datastores

We specify the retrieval granularity as follows: text chunks or smaller granularity such as tokens, phrases, or entities. While it has shown to be effective, text chunks often contain more information than necessary, resulting in redundancy.

Text chunks. The retrieval of text chunks, such as 100-word paragraphs, is a prevalent strategy in widely used Retrieval-Augmented LMs such as REALM, RAG, and RETRO. To implement this, a large-scale corpus $\mathcal{D}$ is segmented into text chunks based on the number of tokens or predefined structures like section headers or paragraphs. Retrieved chunks are typically integrated into input space or intermediate layers, as discussed above, while recent work shows that the choice of length significantly affects performance [Chen et al., 2024b; Wang, Asai, et al., 2025]. LMs are expected to predict output token probability distributions by jointly leveraging their original knowledge in parameters and retrieved text chunks.

Tokens and phrases. Several work explores much smaller units such as tokens [Khandelwal et al., 2020] or phrases [Min et al., 2023b]. Given the input prompt x, such token or phrase Retrieval-Augmented LMs directly search possible next tokens from the datastore by matching the input prompt and similar prefixes in the datastore, instead of making the LM read and generate from the

	Granularity	Incorporation	Frequency	Training	Data order
DrQA	Chunks	Input	One-time	Independent	$O(10^9)$
REALM, RAG, ATLAS	Chunks	Input	One-time	Joint	$O(10^9)$
RALM, REPLUG	Chunks	Input	Every k tokens, One-time	Independent*	$O(10^9)$
Active-Retriever, Self-RAG	Chunks	Input	Adaptive	Independent*, Sequential	$O(10^9)$
RETRO, InstructRetro	Chunks	Intermediate	Every k tokens	Sequential	$O(10^{12})$
kNN LM, TRIME	Tokens	Output	Every token	Independent*, Joint	$O(10^9)$
NPM, Copy Generator	Phrases	Output	Every phrase	Joint	$O(10^9)$
SPALM, Adaptive kNN	Tokens	Output	Adaptive	Independent*, Joint	$O(10^9)$

Table 2.1: Diverse Retrieval-Augmented LMs based on our architecture and training taxonomies. Full references of the papers are as follows: DrQA [Chen et al., 2017], REALM [Guu et al., 2020], RAG [Lewis et al., 2020b], ATLAS [Izacard et al., 2022b], RALM [Ram et al., 2023], REPLUG [Shi et al., 2024], Active Retriever [Jiang et al., 2023], Self-RAG [Asai et al., 2024c], RETRO [Borgeaud et al., 2022], InstructRetro [Wang et al., 2024a], kNN LM [Khandelwal et al., 2020], TRIME [Zhong et al., 2022], NPM [Min et al., 2023b], CopyGenerator [Lan et al., 2023], SPALM [Yogatama et al., 2021], Adaptive kNN [Drozdov et al., 2022]. * indicates that approaches combining off-the-shelf models without any task-specific training.

vocabulary. Token or phrase retrieval can often result in a much larger index size compared to text chunk retrieval given the same size of datastore (i.e., the number of embeddings is by default equal to the number of tokens in the datastore).

Datastore. Designing and building a reliable datastore is a key challenge of Retrieval-Augmented LMs. The inference datastore $\mathcal{D}$ may not be necessarily equivalent to the training datastore and is task-dependent. Some works, such as NPM [Min et al., 2023b], leverage the same corpus as the training data $\mathcal{D} = \mathcal{D}_{\text{train}}$ on more general tasks, while for certain downstream tasks, a smaller and general-domain corpus is often used (e.g., Wikipedia). Conversely, curating high-quality, domain-focused corpora is important for some tasks, e.g., code generation [Hayati et al., 2018; Zhou et al., 2023]. As Table 2.1 shows, most prior work use a datastore that is on the order of $O(10^9)$ tokens, with examples such as Wikipedia containing roughly a few billion tokens. In particular, Wang et al. [2024a]; Borgeaud et al. [2022] scaled the datastore to over one trillion tokens, demonstrating substantial perplexity reductions, although their datastores were not publicly released. In Chapter 3, we present our own studies that release a trillion-token datastore and demonstrate the importance of datastore scaling [Shao et al., 2024].

2.3.2 Training

Retrieval-Augmented LMs consist of three main components: the index $\mathcal{I}$, the retriever $\mathcal{R}$ (i.e., a model that generates encoding of input and documents), and the LM θ. How to efficiently and simultaneously update them to optimize the whole pipeline remains a challenging question. Currently, there are two paradigms: **independent training**, **sequential training** and **joint training** (Table 2.1 Training).

Independent training. Independent training involves the separate development of a retriever and LM with no direct interactions during training. This includes methods such as kNN LM, or recently, RAG applied to off-the-shelf LMs and retrieval systems. This allows practitioners to leverage existing training pipelines and objectives to enhance the individual components. There has been rich literature in the area of IR on how to build reliable and efficient IR systems. Classical term-based retrieval systems, such as TF-IDF or BM25 [Robertson and Zaragoza, 2009], have been widely used. More recently, neural retrieval systems, such as DPR [Karpukhin et al., 2020] or ColBERT [Khattab and Zaharia, 2020], have shown superior performance. Extensive pre-training of retrieval models further improved such models [Izacard et al., 2022a; Ni et al., 2022; Lin et al., 2023]. For a comprehensive review of independent training of retrieval systems, we direct readers to prior surveys [Zhao et al., 2023].

Sequential training. Independent training is often sub-optimal for the whole retrieval-augmented LM pipeline; for instance, LMs trained without retrieval could become easily distracted by irrelevant preceding context [Shi et al., 2023]. To alleviate this issue, sequential training trains either the retriever or LM first, and then trains the other subsequently using signals from the first trained component. Many studies train the LM component with a powerful pre-trained retriever e.g., DPR, search engines, or frozen pre-trained encoders [Izacard and Grave, 2021b; Nakano et al., 2021; Borgeaud et al., 2022], or conversely, train the retriever with signals from the LM [Shi et al., 2024; Izacard and Grave, 2021a].

Joint training. Joint training simultaneously trains the LM and retrieval components to further optimize their interactions and the end-to-end retrieval-augmented LM pipeline. A notable chal-

lenge in joint training is the substantial computational overhead incurred by updating both the retriever model and the resulting index during training. It is impractical to repeatedly generate embeddings for millions or billions of documents in the datastore at each time step. There are two approaches to achieve this under reasonable resource requirements: updating the datastore with updated parameters asynchronously or using an in-batch approximation to a full datastore. **Asynchronous updating** is a technique that allows the index to grow stale over a fixed number of training steps before the update, aiming to use the full corpus during training [Izacard et al., 2022b], as in inference time. There is a tradeoff between the update frequency and computational overhead [Guu et al., 2020]: to obtain better performance, the index should be updated more frequently. **In-batch approximation** builds a temporary index on the fly using training samples from the same mini-batch, which serves as an approximation to the full index during training [Zhong et al., 2022; de Jong et al., 2022; Min et al., 2023b; Lan et al., 2023]. Designing training batches that can provide strong training signals requires careful consideration.

2.3.3 Evaluations: Datasets and Metrics

Retrieval-Augmented LMs have demonstrated strong performance across a wide range of NLP tasks. Their impact is especially pronounced in knowledge-intensive tasks [Guu et al., 2020; Lewis et al., 2020a; Izacard et al., 2022b], including open-domain QA, fact verification, and knowledge-grounded dialogue. However, they have also proven effective in a wide range of other applications, such as summarization [Jiang et al., 2023], machine translation [Khandelwal et al., 2020; Gu et al., 2018], code generation [Wang, Asai, et al., 2025; Zhou et al., 2023], mathematical reasoning and proof generation [Welleck et al., 2022; Yang et al., 2023a], and general language understanding tasks [Min et al., 2023b; Shi et al., 2022; Yu et al., 2022], including sentiment analysis and commonsense reasoning.

In this section, we focus on datasets designed for information-seeking scenarios, such as open-domain QA. These datasets have not only driven rapid progress in model development, as discussed in the previous sections, but also continue to serve as the foundation for many evaluations of more recent systems. A list of these datasets and evaluations is shown at the bottom of Figure 2.1.

Datasets

Here, we review commonly used datasets focusing on their construction, adoption, and limitations, organized by expected answer formats. We then discuss several recent efforts that target evaluation beyond answer accuracy, highlighting emerging evaluation dimensions.

Short-form generation. Many widely-used open-domain QA datasets were originally developed for reading comprehension (RC), the task of comprehending and answering questions based on given reference context, where answers are often subspan in the given context [Liu et al., 2019a]. **SQuAD** [Rajpurkar et al., 2016], **TriviaQA** [Joshi et al., 2017] and **Natural Questions (NQ)** [Kwiatkowski et al., 2019] are widely used datasets in this categories. To repurpose them as open-domain QA datasets, the originally annotated gold documents are discarded and systems only have access to the input queries [Chen et al., 2017; Min et al., 2019]. Some KBQA datasets have been also used for evaluations, such as **WebQuestions** [Berant et al., 2013] or **ComplexWebQuestions** [Talmor and Berant, 2018]. These questions are often relatively simple, and the answers are typically short by design—either minimal text spans that answer the question [Rajpurkar et al., 2016] or predefined knowledge base entities [Berant et al., 2013]. For example, x = "Which NFL team represented the AFC at Super Bowl 50?" and y = "Denver Broncos." (from SQuAD v1).

Several studies have introduced datasets designed to evaluate models' ability to handle more complex queries, such as those requiring multi-hop reasoning or multi-turn dialogue understanding. **HotpotQA** [Yang et al., 2018] presents questions that require identifying and reasoning over two supporting documents to arrive at an answer. **MuSiQue** [Trivedi et al., 2022] constructs multi-hop questions that involve reasoning over two to four intermediate steps. For dialogue-based settings, **QuAC** [Choi et al., 2018a] and **CoQA** [Reddy et al., 2019] are conversational QA datasets in which a system answers a series of multi-turn questions grounded in a given Wikipedia article. More recently, new short-form QA datasets have been specifically designed to test factuality of LMs' responses, such as **SimpleQA** [Wei et al., 2024b] and **PopQA** [Mallen, Asai, et al., 2023] (see also Chapter 3).

As of 2025, short-form QA datasets remain widely used for evaluating state-of-the-art Retrieval-

Augmented LMs [Jin et al., 2025; Song et al., 2025]. However, they present significant limitations when applied to modern systems, with issues arising across the input queries, the relationship between input and answer (context), and the output format:

- **Question quality and realism:** Many of these datasets include simplistic or synthetic questions that do not reflect real-world information-seeking behavior, which typically involves complex, long-form queries and multi-step reasoning [Cao et al., 2025].
- **Context dependency:** Particularly for datasets originally designed for reading comprehension, questions and answers are often tightly coupled to a specific context. When decontextualized, questions may become ambiguous or ill-formed, and the expected answers may no longer align with the original intent [Asai and Choi, 2021].
- **Short-form answer bias:** These datasets were designed with short, often factoid-style answers in mind. In contrast, modern LMs are capable of producing more comprehensive, explanatory responses [Zhao et al., 2024], which are often penalized by evaluation metrics not aligned with longer-form generation [Kamalloo et al., 2023].

Long-form generation. Several datasets have been proposed to evaluate models' ability to handle questions that require long-form responses. **ELI5** [Fan et al., 2019] is a QA dataset comprising questions and answers sourced from Reddit, and has been used to train and evaluate models like WebGPT [Nakano et al., 2021]. However, several studies have pointed out issues with the quality of Reddit answers [Kamoda et al., 2025], as well as problems with the questions themselves: some questions contain false premises [Yu et al., 2023b] or are overly general, allowing for multiple plausible answers, which makes rigorous evaluation challenging [Krishna et al., 2021].

To address these issues, **ASQA** [Stelmakh et al., 2022] constructs factoid-style questions that require long-form answers, building on inherently ambiguous questions from **AmbigQA** [Min et al., 2020]. These long-form QA datasets have been increasingly used to evaluate recent RAG models [Asai et al., 2024c; Jiang et al., 2023], though the number of methods evaluated on them remains relatively small compared to short-form QA benchmarks, as noted in the dataset survey by Gao et al. [2023b]. Min et al. [2023a] introduce a biography generation task, where the system

generates a biography given a person's name, and the output is evaluated against the corresponding Wikipedia article.

Multiple-choice or classification. Conversely, datasets with a closed set of answer candidates—such as multiple-choice QA or classification tasks (e.g., yes/no, true/false) have also been widely adopted for evaluation, especially with the recent surge of interest in reasoning tasks. **MMLU** [Hendrycks et al., 2021] and **GPQA** [Rein et al., 2024] span a broad range of subjects, from high school to Ph.D.-level topics, and require substantial world knowledge and reasoning ability. Prior studies have shown that Retrieval-Augmented LMs can improve performance on such reasoning-intensive tasks, especially when the underlying language model is sufficiently capable [Shi et al., 2024; Shao et al., 2024]. Classification tasks such as fact verification (e.g., **FEVER** [Thorne et al., 2018]) are also commonly used to evaluate retrieval-based methods [Asai et al., 2022a; Izacard et al., 2022b].

New aspects: efficiency, temporal shift and attribution. As retrieval-augmented systems are increasingly deployed in real-world applications, new datasets have emerged to evaluate aspects beyond standard accuracy, focusing on practical and dynamic requirements.

- **Efficiency**: In real-world settings, systems often operate under resource constraints. EfficientQA [Min et al., 2021] is designed to evaluate models' ability to provide accurate answers while adhering to strict on-disk memory budgets, testing their efficiency under deployment-like conditions.
- **Temporal shift**: Real-world knowledge evolves over time. REALTIMEQA (Chapter 3; [Kasai et al., 2022]) establishes a dynamic evaluation framework in which new questions and answers are curated weekly, allowing for the assessment of LMs' ability to adapt to temporal knowledge shifts and remain up-to-date.
- **Attribution and citation accuracy**: In applications like generative search, providing accurate citations to support generated content is increasingly critical, especially in light of prevalent hallucination issues [Liu et al., 2023a]. AttributedQA [Bohnet et al., 2022] introduces a framework for evaluating citation accuracy and grounding model responses in retrieved evidence.

Retrieval datasets. Many retrieval datasets have driven rapid progress of recent retrieval systems. Those typically formulate the task as identifying the correct, labeled gold document(s) from a fixed datastore given a query. **MS MARCO** [Bajaj et al., 2016] is one of the most widely used benchmarks for evaluating retrieval performance. **BEIR** [Thakur et al., 2021] provides a unified benchmark comprising multiple retrieval datasets across diverse tasks, enabling evaluation of retrieval systems' out-of-domain robustness. **MTEB** [Muennighoff et al., 2023] is a benchmark suite for embedding-based tasks including retrieval, clustering, and classification, and is actively used to evaluate new embedding-based retrieval systems.

Metrics and Evaluation Protocol

Matching and ranking using gold references. Historically, evaluations for QA tasks have relied on string-matching metrics using annotated reference answers. In short-form QA datasets, systems are typically evaluated using F1 or Exact Match (EM) scores [Rajpurkar et al., 2016], based on one or more gold answers. For longer outputs, such as in long-form QA, ROUGE has also been widely used [Fan et al., 2019].

However, string-matching metrics have notable limitations. LMs often generate verbose responses that go beyond minimal answer spans [Dubois et al., 2024], and reference sets may not capture all valid answers [Min et al., 2021], leading F1 and EM to underestimate model performance. Additionally, ROUGE has been shown to correlate poorly with human judgments in long-form settings [Xu et al., 2023], raising concerns about its reliability. To address these issues, some recent work adopts Answer Match (AM) as a core metric [Mallen, Asai, et al., 2023], which checks whether any of the gold answers appear in the model output y.

For retrieval evaluations when gold documents are annotated, NDCG, Recall and Precision [Thakur et al., 2021] at top retrieved documents have been widely used. Another common evaluation approach assesses whether the retrieved passages contain the correct gold answer, even in the absence of gold document annotations. This is often referred to as Recall@k [Karpukhin et al., 2020].[5]

[5]This metric, which is based on the final answer string, is different from the previously mentioned recall computed using gold document annotations.

Model-based evaluation. To address the limitations of string-matching evaluations based on human-annotated gold reference answers, an increasing number of studies have adopted model-based evaluation approaches. These can be broadly categorized into two types: (1) evaluations using fine-tuned task-specific models (e.g., for QA or NLI), and (2) evaluations that prompt off-the-shelf LMs to assess response quality.

For (1), ASQA [Stelmakh et al., 2022] evaluates long-form answers by using a fine-tuned extractive RC model to answer disambiguation questions and measures how many of these questions can be correctly addressed based on the model-generated long-form answer. Other work evaluates citation accuracy by checking whether each sentence is entailed by its cited document [Bohnet et al., 2022; Gao et al., 2023a], or uses claim recall for long-form QA [Gao et al., 2023a], typically relying on a fine-tuned NLI model.

For (2), SimpleQA uses ChatGPT to assess whether the generated answer is correct based on the annotated gold answers. While this approach is conceptually similar to string matching, it replaces exact matching with a more flexible, model-based judgment, particularly helpful when the generated answer is verbose or phrased differently. However, recent studies show that model-based evaluations can diverge significantly from traditional string-matching metrics [Jiang et al., 2025; Ho et al., 2025], raising concerns about their consistency and highlighting the need for further investigation into the reliability of such evaluations.

Human evaluation. To address the limitations of both string-matching and model-based evaluations, human evaluations are often employed, although they tend to be more costly. Broadly, human evaluations can be categorized into two types: static and dynamic.

In static evaluations, questions and gold answers are pre-defined, and human annotators assess whether a model-generated answer matches any of the gold answers. This approach is especially useful when there is high answer variability or ambiguity [Min et al., 2021].

In dynamic evaluations, such as those in **Search Arena** [Miroyan et al., 2025], users pose new, open-ended questions to the system and evaluate the quality of responses without access to gold references. These are often conducted as pairwise comparisons between system outputs. While dynamic setups better reflect real-world information needs, evaluating long-form answers is

time-consuming and cognitively demanding even for humans.

Chapter 3

Establishing the Necessity of Retrieval-Augmented LMs

3.1 Overview

Recent progress in LMs has been largely driven by scale: expanding the size of training data and increasing the number of model parameters [Hoffmann et al., 2022; Brown et al., 2020]. This scaling strategy has led to remarkable gains across a wide range of tasks, revealing emerging capabilities such as strong instruction following [Chowdhery et al., 2022]. However, despite these advancements, real-world deployments of large LMs continue to face critical limitations, including widespread hallucinations (i.e., factual inaccuracies in LMs' outputs; Mishra et al. 2024), outdated knowledge [Kasai et al., 2022], and a lack of transparent attribution [Asai et al., 2023a]. These issues significantly hinder the safe and reliable application of LMs in high-stakes domains. Contrary to the dominant belief that scaling would resolve those issues eventually, our work demonstrates that these limitations are not merely the result of insufficient scale. Instead, they are rooted in the monolithic nature of LMs and cannot be resolved by scaling alone.

In the first half of this chapter (Section 3.2), we focus on hallucinations, one of the most pressing challenges in LMs, and show that scaling alone fails to mitigate hallucinations in the long tail, where knowledge is sparse or underrepresented during pre-training. Our results demonstrate

that small LMs (e.g., with 1B parameters), when augmented with large-scale external corpora, can significantly outperform much larger monolithic models such as GPT-3 (175B) on long-tail factuality, effectively reducing hallucinations. We present one of the first large-scale studies of IC RAG (RAG) ,[1] which combines powerful off-the-shelf LMs with retrieval mechanisms without any additional training, in contrast to the previous Retrieval-Augmented LMs [Lewis et al., 2020b; Guu et al., 2020]. This line of work has spurred a wave of research advancing retrieval-augmented systems on top of powerful, off-the-shelf LMs [Asai et al., 2024c; Lin et al., 2024; Yan et al., 2024] and deepening our understanding of LM hallucinations [Min et al., 2023a; Mishra et al., 2024].

In the second half (Section 3.3), we show how simple RAG systems also improve the efficiency of LM training by reducing compute costs. By decoupling factual knowledge from the parametric core, these systems enable flexible updates at inference time [Kasai et al., 2022] and allow for replacing large LMs with much smaller ones [Shao et al., 2024], significantly improving the adaptability and efficiency of the overall system.

Our comprehensive evaluation of the limitations of monolithic scaling and the effectiveness of Retrieval-Augmented LMs has inspired follow-up research and led to the widespread adoption of RAG systems across academia and industry. However, we also identify critical gaps: existing RAG systems often struggle with more complex tasks beyond standard QA. In Chapter 4, we introduce new foundations for Retrieval-Augmented LMs that generalize these systems into a broader and more powerful framework by redesigning and jointly training both retrievers and language models.

3.2 Evaluating and Mitigating LM Hallucinations

Large pre-trained LMs encode a significant amount of world knowledge in their parameters [Petroni et al., 2019; Roberts et al., 2020; Brown et al., 2020; Chowdhery et al., 2022]. However, even models with hundreds of billions of parameters, such as GPT-3 [Black et al., 2022], are prone to generating content that is fluent and plausible but factually incorrect, which is now widely referred to as hallucination [Huang et al., 2025]. While prior work has examined the memorization of substrings

[1] As discussed in Chapter 2, we use the term **RAG** to refer broadly to IC-RAG, the framework of incorporating retrieved documents into the input space of LMs, rather than limiting it to the specific model introduced by Lewis et al. [2020b].

from pre-training data [Carlini et al., 2022], our understanding of how LMs memorize factual knowledge, which is closely related to their tendency to hallucinate, remains limited.

We conduct a large-scale knowledge probing study to examine how LMs memorize factual knowledge, with the goal of understanding when we should and should *not* rely on their parametric knowledge. Specifically, we hypothesize that factual knowledge frequently discussed on the web is more likely to be memorized by LMs, whereas less frequently mentioned knowledge may not be well captured. To test this hypothesis, we construct a new dataset, POPQA(Section 3.2.1). We further investigate how Retrieval-Augmented LMs can mitigate this limitation by incorporating non-parametric memory (i.e., documents from an external datastore) at inference time. In particular, we address the following research questions:

- **Q1:** How much factual knowledge is memorized by LMs and what factors affect the memorization? (Section 3.2.2)
- **Q2:** To what extent can non-parametric memories alleviate the shortcomings of parametric memories of LMs? (Section 3.2.3)
- **Q3:** Can we build a system to adaptively combine non-parametric and parametric memories? (Section 3.2.4)

3.2.1 Probing Method

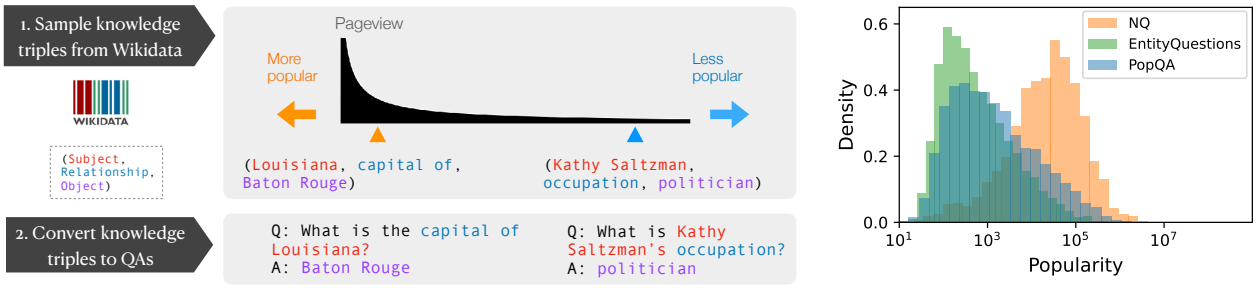
Evaluation Method

We evaluate LMs' ability to memorize factual knowledge through closed-book QA tasks with few-shot samples.

Focus: factual knowledge. Among diverse types of world knowledge, this work focuses on factual knowledge [Adams, 2015] of entities—knowledge about specific details of the target entities. We define factual knowledge as a triplet of (subject, relationship, object) as in Figure 3.1a left.

Task format: open-domain QA. We formulate the task as open-domain QA [Roberts et al., 2020]: given a question, a model predicts an answer without any pre-given ground-truth paragraph.[2]

[2]Some work conducts knowledge probing of encoder-only models by filling out `[MASK]` tokens [Petroni et al., 2019]. We use decoder-only models and thus do not use this `fill-in-the-blank` scheme, unlike Petroni et al. [2019].



(a) PopQA Construction

(b) Popularity distributions.

Figure 3.1: Overview of PopQA. (a) **PopQAconstructions:** PopQA is created by sampling knowledge triples from Wikidata and converting them to natural language questions, followed by popularity calculation. (b) **Popularity distributions:** Distribution of subject entity popularity for EntityQuestions, PopQA, and for NQ-open for reference.

As in Kandpal et al. [2023], we study few-shot settings and prompt LMs without any parameter updates, instead of fine-tuning them on QA datasets such as in Roberts et al. [2020].

Metrics: answer match. We mark a prediction as correct if any substring of the prediction is an exact match of any of the gold answers. As discussed in Chapter 2, LMs often generate correct but longer or more complete sentence-level answers to factoid questions. This behavior can lead commonly used QA metrics such as F1 and EM [Rajpurkar et al., 2016] to underestimate model performance.

Dimensions of Analysis

We hypothesize that factual knowledge that is less frequently discussed on the web may not be well-memorized by LMs. Previous research often uses the term frequency of object entities in pre-training corpora to understand memorization [Févry et al., 2020; Kandpal et al., 2023; Razeghi et al., 2022]. While such studies provide valuable insights, in real-world we often do not have access to the correct answer information. Instead, we investigate whether it's possible to predict memorization based on the input information only, and then apply the findings for modeling improvements, unlike prior analyses. Therefore, our work focuses on the other two variables in a factual knowledge triple: the subject entity and the relationship type.

Subject entity popularity. We use the popularity of the entities measured by Wikipedia monthly

page views as a proxy for how frequently the entities are likely to be discussed on the web, instead of using the occurrence of entities or strings in the pre-training corpus [Carlini et al., 2022; Kandpal et al., 2023; Razeghi et al., 2022]. Calculating frequencies over large pre-training corpora requires massive computations to link entities over billions of tokens, or can result in noisy estimations.[3] Our initial studies show that this is much cheaper[4] and aligns well with our intuition.

Relationship type. We also consider the relationship types as key factors for factual knowledge memorization. For example, even given the same combinations of the subject and object entities, model performance can depend on the relationship types; relationship types widely discussed can be easier to be memorized, while types that are less discussed may not be memorized much.

Benchmarks

PopQA. In our preliminary studies, we found that existing common open-domain QA datasets such as Natural Questions (NQ; Kwiatkowski et al. 2019) are often dominated by subject entities with high popularity, and it is often hard to identify relationship types due to diverse question surface forms. To enable a fine-grained analysis of memorization based on the aforementioned analysis dimensions, we construct PopQA, a new large-scale entity-centric open-domain QA dataset about entities with a wide variety of popularity, as shown in Figure 3.1b.

To construct PopQA, we first sample knowledge triples of 16 diverse relationship types from Wikidata (Step 1) and convert them into natural language questions, using a natural language template (Step 2) as depicted in Figure 3.1a. We verbalize a knowledge triple (S, R, O) into a question that involves substituting the subject S into a template manually written for the relationship type R. The set of acceptable answers to the question is the set of entities E such that (S, R, E) exists in the knowledge graph. We tried various templates and found that the results were fairly robust to the templates. Links to Wikidata entities allow for reliable analysis of popularity and relationship types.

[3]Moreover, several recent models like GPT-3 do not release their pre-training corpora, and it is an open question whether the frequencies in pre-training corpora reflect the frequencies in their private corpora.

[4]We can get page views by calling Wikipedia API.

EntityQuestions. We test on another popular open-domain QA dataset, EntityQuestions [Sciavolino et al., 2021], which also covers a long-tail entity distribution. They use Wikipedia hyperlink counts as a proxy of the frequency of entities and sample knowledge triples from WikiData, from the frequency distributions. Unlike PopQA, EntityQuestions doesn't provide entity annotations, so we only use 82% of the questions, where the mention of the subject entity has a unique match with a Wikidata entity.

3.2.2 Challenges of Scaling Monolithic LMs to Mitigate Hallucinations

We evaluate a range of LMs with varying numbers of parameters, to quantify how much factual knowledge they memorize and how different factors affect those memorization behaviors.

Experimental Setup

Models. We evaluate ten LMs with a varying scale of model size: OPT (Zhang et al. 2022; 1.3, 2.7, 6.7, and 13 billion), GPT-Neo (Black et al. 2022; 1.3, 2.7, 6, and 20 billion), and GPT-3 (Brown et al. 2020; `davinci-002`, `davinci-003`) on our benchmark without any fine-tuning.

Instructions and demonstrations. We use a simple template "`Q: <question> A:`" to format all of our questions for generative prediction. More sophisticated instructions were attempted in preliminary experiments but they did not improve upon the simple template significantly enough to merit using them, especially given that they may overfit to the model. While we use zero-shot prompting for GPT-3 to reduce API costs,[5] we use 15-shot prompting for all GPT-neo and OPT.

Results

Overall results. The top left column of Figure 3.2 (top) illustrates the overall performance on PopQA. As shown, even without using in-context examples, larger LMs exhibit reasonable performance: GPT-3 achieves 35% accuracy, and GPT-Neo 20B achieves 25% accuracy. This indicates that

[5]Using 15-shot prompts for GPT-3 would cost upwards of $3000 for the combination of vanilla, Contriever, BM25, and GenRead evaluations on `davinci-002` and `davinci-003`.

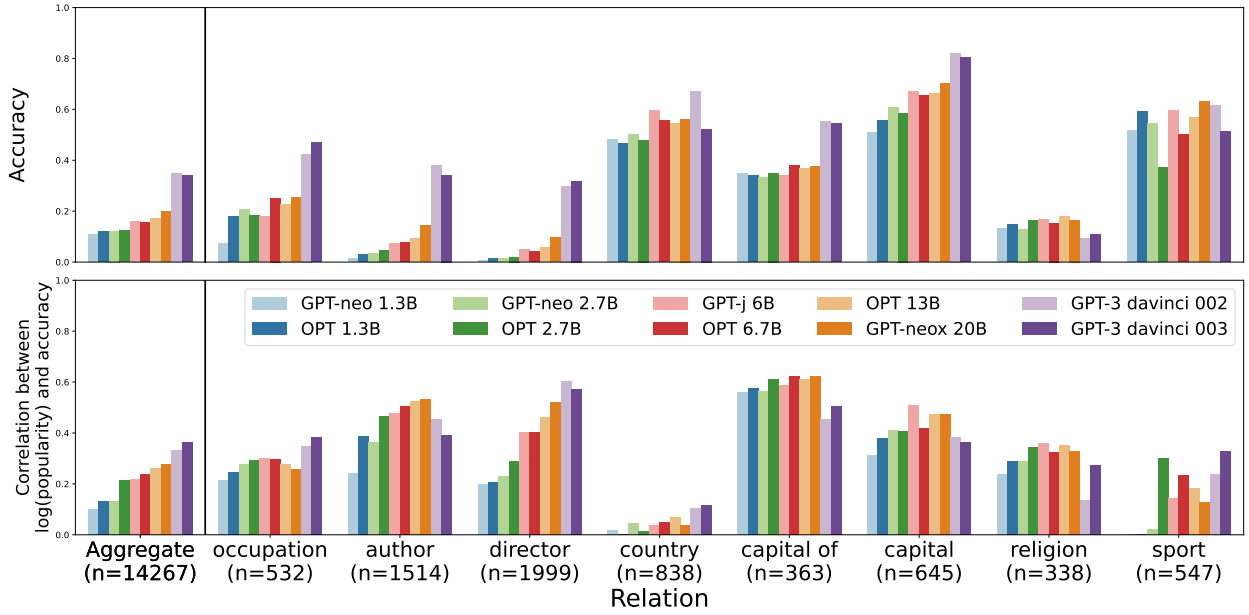


Figure 3.2: Per relationship type accuracy (top) and the correlation between accuracy and log popularity (bottom). n is the number of questions with the given relationship type. We report the correlation between the question's subject entity log-popularity and the binary indicator of whether the question was answered correctly. We see that **both subject entity popularity and relationship type are strong predictors of memorization across models.** The correlation with popularity **exists across relationship types and is stronger for larger LMs**.

powerful LMs memorize factual knowledge in their parameters to some extent. This section examines which types of knowledge are better memorized and what factors influence memorization.

Subject entity popularity predicts memorization. Figure 3.2 (bottom) shows that there is a positive correlation between subject entity popularity and models' accuracy for almost all relationship types. This supports our hypothesis that subject entity popularity can be a reliable indicator of LMs' factual knowledge memorization. In general, the correlations between subject entity popularity and accuracy are stronger for larger LMs; GPT-3 003 shows the highest positive correlation (roughly 0.4) while GPT-Neo-1.3B shows relatively weak positive correlations (approximately 0.1).

Relationship types affects memorization. We find that models have a higher average performance for some relationship types than for others. While this is evidence that factual knowledge of some relationship types are more easily memorized than others, we also observe that questions of certain relationship types can be easily *guessed* without memorizing the knowledge triple. Specifi-

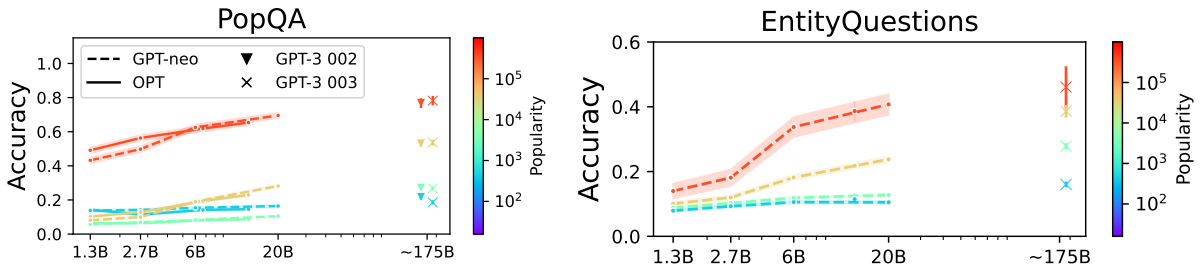


Figure 3.3: Per-popularity accuracy across different model sizes. Performance is broken down by question popularity level. x-axis shows the number of the parameters in LMs. **Scaling mostly improves memorization of more popular factual knowledge.** Error bars are 95% confidence intervals.

cally, certain relationship types (e.g., nationalities) allow models to exploit surface-level artifacts in subject entity names [Poerner et al., 2020; Cao et al., 2021]. Additionally, models often output the most dominant answer entities for questions about relationship types with fewer answer entities (e.g., red for the color relationship type). In Figure 3.2, relationships with lower correlation (e.g., country, sport) often shows higher accuracy, indicating that on those relationship types, models may exploit surface-level clues. On the other hand, for relationship types with relatively low accuracy (e.g., occupation, author, director), larger language models often exhibit stronger performance correlations. This suggests that such questions limit the usefulness of surface-level cues and instead require memorization of fine-grained knowledge, resulting in larger performance gaps between popular and less popular questions.

Scaling may not help with tail knowledge. As seen in the left column of Figure 3.2, there are clear overall performance improvements with scale on the PopQA dataset. However, Figure 3.3 shows that on both PopQA and EntityQuestions, most of scaling's positive effect on parametric knowledge comes from questions with high popularity. Specifically, for the questions about the entities whose $\log_{10}(\text{popularity})$ is larger than 4, there is an improvement in accuracy as model size increases (red and yellow lines), while performance on questions with lower popularity remains relatively constant (blue and green lines). For the 4,000 least popular questions, GPT-Neo 6B, 20B, and GPT-3 `davinci-003` have 15%, 16%, and 19% accuracy, respectively.

This somewhat dampens prior works' findings that scaling up models significantly improves

their factual knowledge memorization [Roberts et al., 2020; Kandpal et al., 2023]. We hypothesize that this is because their evaluations are often conducted on QA datasets with popular entities. In sum, scaling lowers the threshold of popularity for knowledge to be reliably memorized, but is not projected to move the threshold far into the long tail for practical model scales.

3.2.3 Strengths of Retrieval-Augmented LMs in Mitigating Hallucinations

Our analysis shows that even state-of-the-art LMs struggle with less popular subjects or specific relationship types, and simply increasing model size does not yield further performance gains. Motivated by this limitation, we extend our analysis to non-parametric sources of knowledge, specifically, evaluating the effectiveness of Retrieval-Augmented LMs.

In-context Retrieval-Augmented Generation

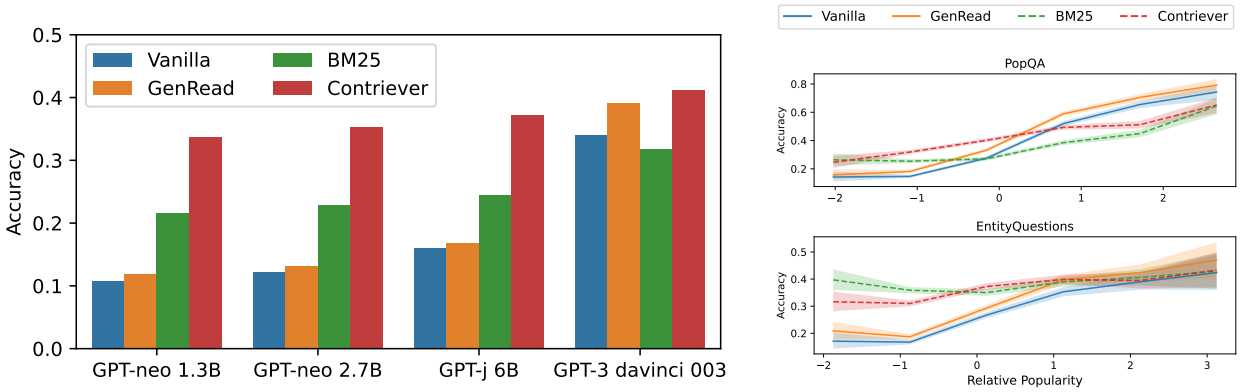
In this work, we tested a simple baseline, **In-context Retrieval-Augmented Generation (IC RAG)**, or **RAG** for simplicity. Unlike prior work that fine-tune task-specific models [Lewis et al., 2020b; Guu et al., 2020], we use an off-the-shelf, general-purpose LMs such as GPT-3 as the generator $\mathcal{G}$, and a standard retrieval model as the retriever $\mathcal{R}$.

Method. The inference pipeline operates as follows: given a user input x, the retriever $\mathcal{R}$ first identifies a small set of relevant documents from the datastore $\mathcal{D}$. Then these retrieved documents are then prepended to the original query x and passed together to the generator $\mathcal{G}$ to produce an output y. Since it operates solely by modifying the model's input, no additional training is required.

Experimental details. For $\mathcal{G}$, we use the same set of LMs as in Section 3.2.2. For $\mathcal{R}$, we test two widely-used retrieval systems: **BM25** [Robertson and Zaragoza, 2009] and **Contriever** [Izacard et al., 2022a]. BM25 is a static term-based retriever without training, while Contriever is pretrained on large unlabeled corpora, followed by fine-tuning on MS MARCO [Bajaj et al., 2016]. We retrieve top 10 documents from English Wikipedia relevant to a question,[6] We also experiment with a

[5] 30 POPQA and 26 EntityQuestions questions had popularity less than the smallest popularity bin, and are excluded to avoid showing results for small sample sizes.

[6] We use Wikipedia dump from December 2018.



(a) Overall Performance

(b) Performance breakdown

Figure 3.4: Results of IC-RAG baselines. (a) **Overall Performance:** POPQA accuracy of LMs augmented with BM25, Contriever, GenRead, and unassisted (vanilla). (b) **Performance Breakdown:** GPT-3 `davinci-003` accuracy versus relative popularity (how popular a question is relative to other questions of its relationship type). Relative popularity is defined as the log-popularity of a question, normalized by the mean and standard deviation of log-popularity for the question's relationship type (smaller for less popular entities). Error bars show Wilson 95% confidence intervals. Bins with less than 40 samples

parametric augmentation method, **GenRead** [Yu et al., 2023a], which prompts LMs to generate rather than retrieve a contextual document to answer a question. This results in 40 LMs and Retrieval-Augmented LMs.

Results

Retrieval largely improves performance. Figure 3.4a shows that augmenting LMs with non-parametric memories significantly outperforms unassisted vanilla LMs. A much smaller LM (e.g., GPT-Neo 2.7B) augmented by the Contriever retrieval results outperforms vanilla GPT-3. Large LMs such as GPT-3 also enjoy the benefits of non-parametric memories. Contriever gives 7% accuracy gains on top of GPT-3 `davinci-003`. GenRead shows little-to-no performance improvement over vanilla parametric knowledge for smaller models, while the technique shows sizeable gains for GPT-3, especially `davinci-003`. In addition to its limited effectiveness with smaller LMs, GenRead has potentially prohibitive inference time costs, with GPT-NeoX 20B taking 70 seconds per query.

	Contriever-augmented LM	
	succeeded	failed
LM succeeded	0.83 (24%)	0.14 (10%)
LM failed	0.88 (17%)	0.11 (49%)

Table 3.1: Relationship between retrieval results and final RAG results. The recall@1 of Contriever for questions that GPT-3 `davinci-003` answered correctly and incorrectly with and without retrieval on POPQA. The percent of questions falling in each category is shown in parentheses.

Naive RAG are effective for long-tail knowledge. How does retrieval augmentation lead to such significant improvements? Figure 3.4b shows the relationship between the entity popularity and models' QA performance. It can be seen that LMs guided by Contriever or BM25 have a clear advantage over unassisted vanilla LMs, especially on less popular entities, resulting in a significant performance gain. Overall, Contriever-guided LMs outperform BM25-based ones on POPQA, while the BM25-based models perform better on the least popular entities, consistent with the findings from Sciavolino et al. [2021]. On the other hand, for more popular entities, parametric knowledge shows equal or higher accuracy, indicating that the state-of-the-art LMs have already memorized the answers, and augmenting input with retrieved-context doesn't help much or even hurts the performance. Interestingly, GenRead generally outperforms vanilla LMs despite relying on LMs' parametric memory. This demonstrates the effectiveness of elicitive prompting [Wei et al., 2022b; Sun et al., 2023] as observed in prior work. However, like vanilla LMs, GenRead shows low performance on less popular entities.

Irrelevant retrieved context can mislead LMs. We conduct an in-depth analysis of why Retrieval-Augmented models suffer in more popular entities. We hypothesize that retrieval results may not always be correct or helpful, and can mislead LMs. To test this hypothesis, we group the questions based on two axes: whether unassisted GPT-3 `davinci-003` predict correctly or not, and whether Retrieval-Augmented predictions are correct or not. For each of the four categories, we calculate recall@1 (whether a gold answer is included in the top 1 document; Karpukhin et al. 2020).

Table 3.1 shows recall@1 for each group with percentages of the questions falling into each of the categories. For 10% of questions, retrieval-augmentation causes the LM to incorrectly answer a question it could otherwise answer correctly. We found that on those questions, recall@1 is

significantly lower than the overall recall@1 (0.14 vs 0.42 overall), indicating that failed retrieval can result in performance drops. Conversely, for the 17% of questions for which retrieval causes the LM to correctly answer a question it would otherwise have failed to answer, the recall@1 is 0.88.

3.2.4 Adaptive Retrieval: Using Retrieval Only Where It Helps

While incorporating non-parametric memories helps in long-tail distributions, powerful LMs have already memorized factual knowledge for popular entities, and retrieval augmentation can be harmful. Can we achieve the best of both worlds? We propose a simple-yet-effective method, Adaptive Retrieval, which decides when to retrieve documents only based on input query information and augments the input with retrieved non-parametric memories only when necessary.

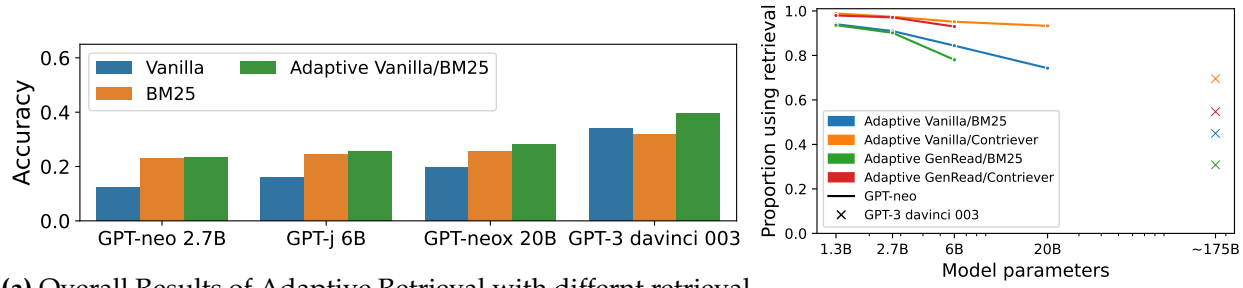
Method

Adaptive Retrieval is based on our findings: as the current best LMs have already memorized more popular knowledge, we can use retrieval only when they do not memorize the factual knowledge and thus need to find external non-parametric knowledge. In particular, we use retrieval for questions whose popularity is lower than a threshold (*popularity threshold*), and for more popular entities, do not use retrieval at all.

Using a development set, the threshold is chosen to maximize the adaptive accuracy, which we define as the accuracy attained by taking the predictions of the retrieval-augmented system for questions below the popularity threshold and the predictions based on parametric knowledge for the rest. We determine the popularity threshold independently for each relationship type.

Results

Adaptive Retrieval improves performance. Figure 3.5a shows the results when we adaptively retrieve non-parametric memories based on the per-relationship type thresholds. We can see that adaptively retrieving non-parametric memories is effective for larger models. The best performance on PopQA is using GPT-3 `davinci-003` adaptively with GenRead and Contriever, yielding 46.5% accuracy, 5.3% higher than any non-adaptive method.



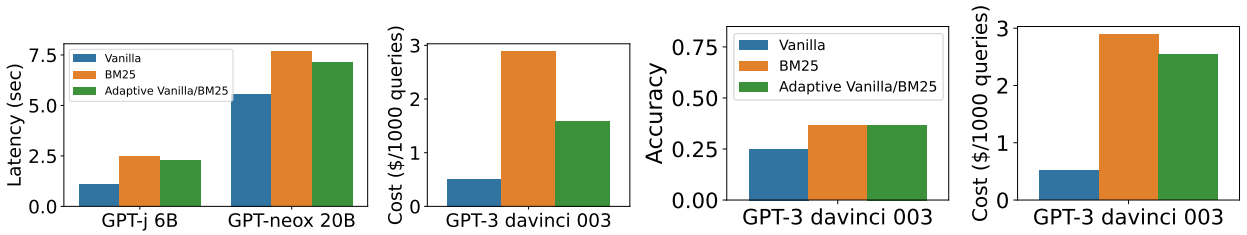
(a) Overall Results of Adaptive Retrieval with differnt retrieval systems and LMs.

(b) Threshold of Adaptive Retrieval.

Figure 3.5: Effectiveness of Adaptive Retrieval. (a) **Overall Performance:** PopQAperformance of GPT-neo models and GPT3 `davinci-003`, with different retrieval methods. (b) **Threshold of Adaptive Retrieval:** The proportion of questions for which various models use retrieval in the Adaptive Retrieval setup on PopQA.

The threshold shifts with LM scale. While Adaptive Retrieval shows performance gains for larger models, smaller models do not realize the same benefits; as shown in Figure 3.5a, the performance gain from Adaptive Retrieval is much smaller when we use models smaller than 10 billion. Why does this happen? Figure 3.5b shows that smaller LMs almost always retrieve, indicating that there are not many questions for which small LMs' parametric knowledge is more reliable than non-parametric memory. In contrast, large models typically retrieve much less. For example, GPT-3 `davinci-003` only retrieves for 40% of questions when paired with BM25, and even the much smaller GPT-neox 20B does not retrieve documents on more than 20% of the questions.

Adaptive Retrieval reduces inference-time costs. We also found that Adaptive Retrieval improves efficiency; if we know we do not need to retrieve documents, we can skip retrieval components and the input length becomes shorter, which improves latency in both retrieval and language model components. Figure 3.6a shows the inference latency of GPT-J 6B and GPT-neox 20B, and API costs of GPT-3. Especially for larger LMs, concatenating retrieved context results in significantly increased latency (e.g., for GPT-J 6B, the inference time latency almost doubles). Adaptive retrieval enables reducing inference time up to 9% from standard retrieval. We also observe cost reduction on EntityQuestions, as shown in Figure 3.6b.



(a) Latency and API Costs

(b) Results and Costs on EntityQuestions.

Figure 3.6: Cost Reductions by Adaptive Retrieval. (a) PopQAlatency for large GPT-neo models that were run on our machines, and API costs for GPT3. (b) Accuracy and cost savings of Adaptive Retrieval for EntityQuestions. Despite EntityQuestions's lack of popular entities (see Figure 3.1b), Adaptive Retrieval is able to reduce API costs by 15% while maintaining equivalent performance to retrieval only.

3.2.5 Summary and Limitations

We conducted large-scale knowledge probing to examine the effectiveness and limitations of relying on LMs' parameters to memorize factual knowledge and to understand what factors affect factual knowledge memorization. Our results showed that memorization has a strong correlation with entity popularity and that scaling up models on long-tail distributions may only provide marginal improvements. We demonstrated that Retrieval-Augmented LMs can greatly aid LMs on these long-tail distributions, but can also mislead LMs on questions about well-known entities, as powerful LMs have already memorized them in their parameters. Based on those findings, we devise simple-yet-effective Adaptive Retrieval, which only retrieves when necessary, using a heuristic based on entity popularity and relationship types. Our experimental results show that this method is not only more powerful than LMs or previous Retrieval-Augmented LMs but also more efficient.

Limitations. This work focuses on entity-centric factual knowledge and demonstrates that LMs' memorization is heavily affected by the popularity of the entities and the aspect of the entities being asked in the questions. It is important to emphasize that for running controlled experiments, we have relied on two synthetic datasets, and the extent to which our results apply to naturally occurring factual knowledge has not been firmly established. While we can be fairly confident about the relationship between scaling, retrieval, popularity, relationship type, and performance

for the kinds of knowledge studied here, the effectiveness of Adaptive Retrieval will depend on many details of the QA pipeline. Moreover, our work depends on a definition of popularity that is time-dependent and may not perfectly reflect how frequently entities are discussed on the web. Wikipedia page views are one possible definition of popularity for which we observe our results, and we invite others to improve upon it in future work.

3.3 Improving Efficiency with Retrieval-Augmented LMs

We have shown that Retrieval-Augmented LMs enable LMs to incorporate knowledge at inference time, offering a promising approach to mitigating hallucinations. This capability also helps reduce the expensive training costs associated with current monolithic LMs. In this section, two of our studies are briefly highlighted that illustrate these benefits. For full experimental details and results, please refer to the original papers [Shao et al., 2024] and [Kasai et al., 2022].

3.3.1 Enhanced Scaling Law with Retrieval-Augmented LMs

As shown in Section 3.2, a 1-billion-parameter LM can outperform much larger models by leveraging retrieved documents at inference time. This illustrates the potential of Retrieval-Augmented LMs to reduce training-time compute: because building a datastore index is significantly cheaper than training on the same amount of data, Retrieval-Augmented LMs allow for more scalable and efficient use of large corpora as retrieval sources. For example, models can be trained on a much smaller number of tokens, while relying on a substantially larger datastore at inference time to access external knowledge via retrieval.

However, prior work has primarily used Wikipedia as the sole datastore [Lewis et al., 2020b; Guu et al., 2020; Izacard and Grave, 2021b], which is limited to approximately 5 billion tokens, significantly smaller than the multi-trillion-token corpora used for LM pre-training (e.g., 3 trillion tokens in Soldaini et al. 2024). This raises an open question: *can scaling the datastore further improve the efficiency and performance of retrieval-augmented models?* To address this, we present the first systematic study on the scaling laws of Retrieval-Augmented LMs, and compare their performance to parametric scaling via model size and training data.

MASSIVEDS: A Trillion-token Datastore with a Diverse Domain Composition

While several prior work studies the effect of scaling datastores [Borgeaud et al., 2022], they often do not open source their datastores. To evaluate the effect of datastore scaling, we first construct MASSIVEDS, a massively multi-domain datastore containing 1.4 trillion tokens spanning both general web data and domain-specific sources. This datastore serves as the foundation for our scaling study. We also design an efficient construction pipeline that reduces computational cost by an order of magnitude while maintaining equivalence to the standard pipeline.

Domain	Datasets	Size (B)
BOOKS	RPJ Books	26.3
STEM	peS2o, RPJ ArXiv	97.7
ENCYCLOPEDIA	DPR 2018 Wiki, RPJ 2022 Wiki	31.9
FORUM (Q&A)	RPJ StackExchange	20.2
CODE	RPJ Github	52.8
MATH	OpenWebMath, NaturalProofs	14.1
BIOMEDICAL	PubMed	6.5
GENERAL WEB	RPJ CC (2019–2023), RPJ C4	1191.7
Total		1441.2

Table 3.2: The domain-wise data composition of MASSIVEDS. RPJ denotes REDPAJAMA V1 [Computer, 2023], CC denotes Common Crawl, Wiki denotes Wikipedia.

Experiments and Results

Experimental details. We evaluate a simple IC RAG approach as in Section 3.2. We use Contriever-MS MARCO [Izacard et al., 2022a] for a retrieval model and concatenate the top $k = 3$ documents in reverse order, so that higher-ranked documents are positioned closer to the query. We study datastore scaling performance with the Llama2 [Touvron et al., 2023], Llama3 [Grattafiori et al., 2024a] and Pythia [Biderman et al., 2023].

Effect of datastore scaling. Figure 3.7 (left) shows perplexity curves as a function of the size of the datastore on the general web and scientific papers, respectively. Retrieval is strictly beneficial for language modeling: the LM-only baselines (denoted by dashed lines) show the highest perplexity across all models and evaluation datasets. Scaling up the datastore reduces perplexity without signs of saturation, suggesting that further scaling is likely to yield additional improvements. Further,

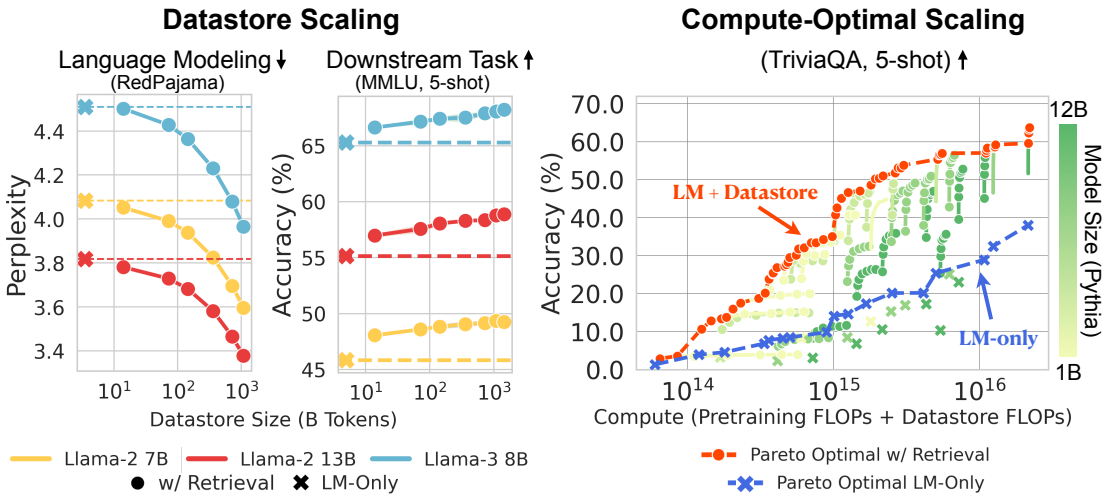


Figure 3.7: Results of RAG systems with MASSIVEDS. *Left:* Datastore scaling performance on language modeling task and a downstream task (MMLU) with Llama2 and Llama3 models. *Right:* Compute-optimal scaling of Retrieval-Based LMs vs. LM-only models with Pythia models. By considering the size of the datastore as an additional dimension of scaling, we can improve model performance at lower training cost.

datastore scaling enables small models to outperform their larger LM-only counterparts. Figure 3.7 (middle) shows the performance on MMLU [Hendrycks et al., 2021] as a function of datastore size. Datastore scaling monotonically improves performance across all model scales.

Compute-optimal scaling with Retrieval-Augmented LMs. We show the scaling curves against computational cost based on training-time GPU FLOPs of Retrieval-Augmented LMs and LM-only performance on TriviaQA in Figure 3.7 (right). Specifically, we approximate models trained on different token amounts using intermediate checkpoints from Pythia, which share a fixed learning rate schedule and may slightly underperform given their compute. For compute analysis, we note that datastore construction is far cheaper than LM pre-training, requiring only forward passes with a small retriever.

The pareto-optimal points for retrieval-based and LM-only settings are highlighted in red and blue, respectively. With the same training-time compute, Retrieval-Augmented LMs achieves superior performance than LM-only models, indicating offloading FLOPs from pre-training to datastore construction can result in better performance. Therefore, we conjecture that storing factual knowledge in a datastore is more computationally efficient than memorizing factual knowledge in

model parameters at training time.

3.3.2 Enabling Knowledge Updates without Training

REALTIMEQA: Evaluating LMs on Emerging Information in Real Time

Parametric LMs require continuous retraining on newly collected data to stay up to date with the rapidly evolving world. In contrast, Retrieval-Augmented LMs allow for flexible knowledge updates simply by swapping the underlying datastore. This significantly reduces training costs by eliminating the need for frequent retraining.

While earlier studies have explored temporal knowledge shifts using synthetic or templated queries such as TempLAMA [Dhingra et al., 2022; Zhang and Choi, 2021], these benchmarks are typically limited to narrow patterns and coarse-grained, year-level changes. To address this limitation, we introduce REALTIMEQA, a benchmark designed to evaluate LMs' ability to incorporate real-time knowledge on a finer, weekly basis.

Data creation. REALTIMEQA is a dynamic benchmark that releases around 30 multiple-choice questions each week, drawn from current news quizzes on CNN, USA Today, and The WEEK. The authors first identify web pages containing "weekly quiz" content from three major news outlets: CNN (U.S.), USA Today (U.S.), and The WEEK (U.K.). An automated script is then used to extract multiple-choice questions from these pages. These questions are announced every Saturday, and participants are encouraged to answer them in real time using either their own knowledge sources or an optional set of retrieved documents provided via our retrieval systems. Each question is paired with the top-10 news articles retrieved at the time of release, parsed and stored with metadata. Participants can evaluate their systems in both multiple-choice and open-ended generation formats. All of the results over four years, from 2022 to 2025 are available at `https://realtimeqa.github.io/`.

Metrics. Since REALTIMEQA is a multiple-choice dataset, we report accuracy as the primary metric. We also evaluate a NOTA (none of the above) variant, where one original choice is

Real-time Baselines			Multi-Choice		Generation	
	Retrieve	Generate	Orig.	NOTA	EM	F1
Open	DPR	RAG	33.0	24.0	2.8	4.9
	DPR	GPT-3	45.8	34.1	8.4	14.6
	GCS	RAG	49.7	42.5	20.8	25.1
	GCS	GPT-3	**69.3**	**59.8**	**28.7**	**39.4**
Closed	—	T5	35.2	35.2	6.7	11.5
	—	GPT-3	39.7	31.3	7.3	15.2

Table 3.3: Results on REALTIMEQA. Results from the first six weeks (from June 17, 2022 through July 22, 2022) over a total of 179 questions.

randomly replaced with "none of the above" to reduce heuristic exploitation [Rajpurkar et al., 2018], following datasets like MCTest [Richardson et al., 2013] and RACE [Lai et al., 2017]. We also consider a generation setting without answer choices to better reflect real-world use. We evaluate using exact match and token-level F1, following standard QA practices [Rajpurkar et al., 2016].

Experiments and Results

Experimental settings. We evaluate in-context RAG baselines without any additional training, following the setup described in the earlier sections of this chapter. For retrieval, we use DPR [Karpukhin et al., 2020] and Google Custom Search (GCS), incorporating the top five Wikipedia documents retrieved by DPR and the top five news articles retrieved by GCS. For the generator components, we test RAG [Lewis et al., 2020b], T5 [Raffel et al., 2020], and GPT-3 [Brown et al., 2020].

Main results. Seen in Table 3.3 are the results from the six weeks. In all three settings (original/NOTA multiple choice and generation), GPT-3 with GCS retrieval achieves the best performance. In particular, GPT-3 with GCS substantially outperforms both closed-book GPT-3 and GPT-3 with DPR (from a December 2018 Wikipedia dump): e.g., 28.7 vs. 7.3/8.4 in generation exact matching. This suggests that GPT-3 is able to answer questions based on the given prompt, rather than relying on past information from pre-training. Nevertheless, we still see a large performance drop of all baselines from the original multiple-choice setting to NOTA ("none of the above"):

e.g., 59.8 vs. 69.3 for GPT-3 with GCS retrieval. Future work can further improve GPT-3's ability of reading comprehension, especially regarding answer uncertainty. We also note that the best baseline uses the black-box Google custom search API; we encourage participants to develop a competitive open-source system.

3.4 Conclusions and Discussion of Subsequent Work

This chapter provides an overview of our work on understanding the limitations of current LMs that cannot be resolved through scaling alone, and highlights the effectiveness of retrieval augmentation in addressing these challenges. We show that **Retrieval-Augmented LMs offer a more reliable, adaptable, and efficient path forward**: they reduce hallucinations in long-tail knowledge, enable dynamic knowledge updates without retraining, and lower training-time compute while supporting more compute-efficient scaling. I also led the first tutorial on Retrieval-Augmented LMs [Asai et al., 2023a],[7] which has been widely adopted in university courses and follow-up research.

Our work has inspired a broad range of subsequent studies, including advances in adaptive RAG strategies [Asai et al., 2024c; Jiang et al., 2023] and deeper investigations into LM hallucinations [Min et al., 2023a; Mishra et al., 2024]. Both POPQA and REALTIMEQA have become widely used benchmarks for evaluating LM capabilities [Lambert et al., 2024]. In this section, we provide a detailed discussion of this follow-up work.

Developments of adaptive RAG systems. Our prior work [Mallen, Asai, et al., 2023] provided early evidence of both the strengths and limitations of RAG systems. Notably, it was the first to introduce adaptive RAG, which selectively performs retrieval only when necessary. Numerous follow-up studies have proposed more sophisticated strategies, leveraging model-based decisions and richer features. **Self-RAG** [Asai et al., 2024c], discussed in detail in Chapter3, trains a single language model that jointly learns when to retrieve, generate, and self-evaluate. Other approaches, such as **CRAG** [Yan et al., 2024] and **Adaptive RAG** [Jeong et al., 2024b], employ

[7] https://acl2023-retrieval-lm.github.io/

separate lightweight classifiers to determine whether retrieval is necessary. **DRAGIN** [Su et al., 2024] estimates uncertainty based on the influence and semantic importance of each token, while **FLARE** [Jiang et al., 2023] triggers retrieval when low-confidence tokens are predicted. Finally, Marina et al. [2025] propose LM-independent adaptive retrieval strategies based on external signals, such as content popularity.

Understanding LM hallucinations. Our work also conducted one of the earliest large-scale investigations into LM hallucinations, demonstrating that knowledge popularity is a key factor influencing model memorization. While our analysis focused on short-form QA performance, follow-up research has extended this line of inquiry to long-form generation. **FactScore** [Min et al., 2023a] shows that factually inaccurate statements are more prevalent in long-form responses, particularly when LMs are prompted to generate biographies of rare entities. Our subsequent work, **FAVA** [Mishra et al., 2024], introduces a taxonomy of hallucinations and presents detailed human evaluations of multiple LMs on real-world user queries. The analysis reveals that even state-of-the-art models such as LLaMA 2 70B and ChatGPT hallucinate in over 50% of their responses, reinforcing the limitations of scaling parametric LMs alone.

Benchmarks using POPQA. POPQAhas been widely adopted as part of benchmarks testing LMs' factuality or reliability. **The Hallucinations Leaderboard** [Hong et al., 2024] unifies diverse benchmarks related to factuality to quantitatively measure and compare the tendency of each model to produce hallucinations. **FLASK** [Ye et al., 2024] is a fine-grained evaluation protocol for both human-based and model-based evaluation which decomposes coarse-level scoring to a skill set-level scoring for each instruction, including factuality. POPQAhas been used as one of the core benchmarks to test state-of-the-art LMs' performance, including **Tulu3** [Lambert et al., 2024].

Chapter 4

Building New Foundations for Retrieval-Augmented LMs

4.1 Overview

While Retrieval-Augmented LMs have been studied for years, earlier systems were typically designed and trained to perform relatively simple, single tasks such as open-domain QA (Chapter 2). Today, Retrieval-Augmented LMs, particularly IC RAG approaches that augment model inputs with retrieved context are widely adopted in real-world applications, powering systems such as ChatGPT Search and Google AI Overviews. However, as discussed in Chapter 3, such RAG systems often struggle with challenges such as sensitivity to irrelevant context, contradictions between generated outputs and cited evidence, and persistent hallucinations. These limitations have led to notable errors in widely deployed systems and have resulted in real-world consequences [Williams, 2024; Metz and Weise, 2025]. In this chapter, we introduce new foundations for building more versatile and reliable Retrieval-Augmented LMs, reflecting the evolving demands and use cases of these systems.

We begin in Section 4.2 by identifying key limitations of the common RAG pipelines. We argue that to enable broader applicability, we must design and train Retrieval-Augmented LMs that can more flexibly adapt to diverse inference scenarios. This requires moving beyond over-optimization

for narrow tasks or domains, toward models that are more scalable, general-purpose, and robust.

In Section 4.3, we present methods for designing and training LMs specifically for retrieval augmentation. In particular, we highlight SELF-RAG [Asai et al., 2024c], which equips arbitrary LMs with special tokens to decide when to retrieve, generate, and self-evaluate their outputs at inference time. SELF-RAG has demonstrated strong performance across six diverse tasks, including short-form generation, multiple-choice reasoning, and long-form synthesis, outperforming much larger models. It has inspired extensive follow-up research and has been widely adopted in industry. Our related work in this direction [Asai et al., 2022a; Mishra et al., 2024] further demonstrates the effectiveness of training LMs to more seamlessly integrate retrieval mechanisms.

However, the success of these new LMs also depends on powerful and reliable retrieval systems. In Section 4.4, we present our work on advancing neural retrieval systems to make them more **versatile** in handling diverse user intents, more **robust** to complex queries involving multi-hop reasoning and entity understanding, and more **efficient** at scaling to massive datastores. To this end, we introduce the first instruction-following retrieval and reranking systems, TART [Asai et al., 2023b], demonstrating their ability to adapt to novel user instructions. We also highlight a series of contributions, including novel retrieval architectures [Asai et al., 2020], pre-training techniques [Yamada et al., 2020] that enhance robustness to complex queries, and methods for improving retrieval efficiency [Yamada et al., 2021; Shao et al., 2024].

In Section 4.5, we discuss subsequent academic work that builds upon our foundational contributions to LMs and retrieval, as well as their adoption in industry. SELF-RAG has been rapidly adopted by both academia and industry, establishing new directions for advanced RAG. Our retrieval systems have been widely integrated into standard libraries and deployed in real-world applications, such as COVID-Search [Esteva et al., 2021]. They also form the backbone of today's state-of-the-art text retrieval systems [Lee et al., 2025] and multimodal retrieval systems [Wei et al., 2024a]. Many of our contributions have been incorporated into major software libraries with tens of millions of monthly installations, powering a broad range of industry applications.

4.2 Challenges of Common RAG Systems

In Chapter 3, we demonstrate the effectiveness of simple IC RAG (or RAG) systems that combine off-the-shelf LMs (e.g., GPT-3) with retrieval methods (e.g., BM25) at inference time for general-domain QA tasks, using a small, general-purpose datastore like Wikipedia. However, such straightforward pipelines often encounter limitations when applied to more complex scenarios across core components: LMs, retrievers, and datastores.

Limitations of off-the-shelf LMs and fixed pipelines. Using off-the-shelf LMs that are not trained with retrieval in mind presents several challenges when applied to retrieval-augmented use cases. First, as shown by Mallen, Asai, et al. [2023], state-of-the-art models such as GPT-3 are easily distracted by irrelevant context and may fail to answer correctly, even when they originally possess the correct knowledge. This issue has been widely confirmed in numerous follow-up studies [Yoran et al., 2024; Xu et al., 2024b], indicating its persistent and systemic nature. Conversely, LMs are also prone to generating outputs that are not fully supported by the cited evidence or faithful to the given context [Liu et al., 2023a; Goyal et al., 2025].

Furthermore, most current RAG pipelines rely on a fixed, two-stage architecture: all relevant documents are retrieved upfront based solely on the initial query x, and then the response is generated based on this static context. This rigid pipeline becomes problematic in interactive applications such as chatbots, where user queries vary widely in nature. Some queries (e.g., writing assistance; Zhao et al. 2024) require little to no external retrieval, while others demand multi-step retrieval to ensure coverage and factual accuracy [Asai et al., 2020].

An ideal system should flexibly integrate retrieval, selectively identify which pieces of evidence are relevant and trustworthy, and ensure that outputs are well-grounded in the retrieved context. However, current two-stage designs and the lack of retrieval awareness during language model training significantly limit the ability to meet these needs.

Limitations of using standard retrievers and datastores. Existing retrieval systems also face significant challenges, particularly in their generalizability, efficiency, and ability to handle complex

queries. These systems are often trained on specific retrieval tasks using datasets such as NQ or MS MARCO [Bajaj et al., 2016]. However, such supervised neural retrievers frequently struggle in out-of-domain scenarios, where models are evaluated on new domains or different types of retrieval tasks [Thakur et al., 2021]. Training separate retrievers for each task not only complicates the inference pipeline, but is also impractical given the wide variety of use cases and the limited availability of high-quality training data across domains. Furthermore, standard retrieval systems often fail to retrieve all necessary context when queries are complex, such as those requiring multi-step reasoning [Asai et al., 2020] or precise entity-level information [Chen et al., 2022].

Another common limitation is the reliance on narrow corpora like English Wikipedia [Mallen, Asai, et al., 2023; Shi et al., 2024; Jin et al., 2025] or black-box production search APIs [Kasai et al., 2022; Luo et al., 2023], which makes it difficult to build efficient and scalable retrieval systems that operate over much larger, more diverse datasets. In many real-world applications, critical documents are not publicly accessible, and systems must perform retrieval over proprietary or domain-specific data. Building retrieval models that are both effective and efficient in these settings remains a key open challenge.

4.3 Self-RAG: Learning to Retrieve, Generate and Self-Critique

To overcome the challenges in existing LMs and inflexible pipelines, we introduce **Self-Reflective Retrieval-Augmented Generation (SELF-RAG)** to improve an LM's generation quality, including its factual accuracy without hurting its versatility, via on-demand retrieval and self-reflection.

We train an arbitrary LM in an end-to-end manner to learn to reflect on its own generation process given a task input by generating both task output and intermittent special tokens (i.e., *reflection tokens*). Reflection tokens are categorized into *retrieval* and *critique* tokens to indicate the need for retrieval and its generation quality respectively (Figure 4.1 right). In particular, given an input prompt and preceding generations, SELF-RAG first determines if augmenting the continued generation with retrieved passages would be helpful. If so, it outputs a **retrieval** token that calls a retriever model on demand (Step 1). Subsequently, SELF-RAG concurrently processes multiple retrieved passages, evaluating their relevance and then **generating** corresponding task outputs

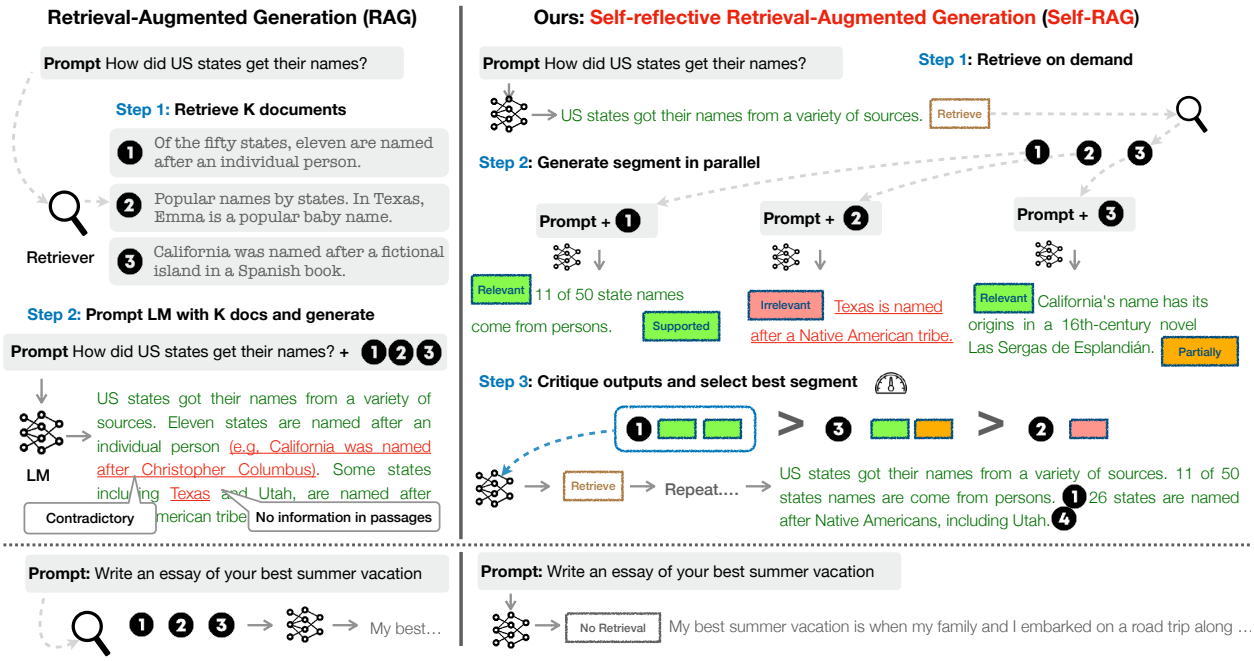


Figure 4.1: Overview of SELF-RAG. SELF-RAG learns to retrieve, critique, and generate text passages to enhance overall generation quality, factuality, and verifiability.

(Step 2). It then generates critique tokens to **criticize** its own output and choose best one (Step 3) in terms of factuality and overall quality. This process differs from conventional RAG (Figure 4.1 left), which consistently retrieves a fixed number of documents for generation regardless of the retrieval necessity (e.g., the bottom figure example does not require factual knowledge) and never second visits the generation quality. Moreover, SELF-RAG provides citations for each segment with its self-assessment of whether the output is supported by the passage.

SELF-RAG further enables a customizable decoding algorithm to satisfy hard or soft constraints, which are defined by reflection token predictions. In particular, our inference-time algorithm enables two key capabilities: (1) flexible adjustment of retrieval frequency to suit different downstream applications, and (2) customization of model behavior based on user preferences. The latter is achieved by leveraging reflection tokens through segment-level beam search, where each segment is scored using a weighted linear sum of the reflection token probabilities.

A key component of our approach is teaching LMs to generate reflection tokens with well-calibrated scores. To this end, we train our generator LM on a diverse corpus of text interleaved

Type	Input	Output	Definitions
Retrieve	x / x, y	{yes, no, continue}	Decides when to retrieve with $\mathcal{R}$
IsRel	x, d	{**relevant**, irrelevant}	d provides useful information to solve x.
IsSup	x, d, y	{**fully supported**, partially supported, no support}	All of the verification-worthy statement in y is supported by d.
IsUse	x, y	{**5**, 4, 3, 2, 1}	y is a useful response to x.

Table 4.1: Four types of reflection tokens used in SELF-RAG. Each type uses several tokens to represent its output values. The bottom three rows are three types of Critique tokens, and **the bold text** indicates the most desirable critique tokens. x, y, d indicate input, output, and a relevant passage, respectively.

with reflection tokens and retrieved passages. The reflection tokens, inspired by reward modeling in reinforcement learning [Ziegler et al., 2019; Ouyang et al., 2022], are inserted offline into the training corpus by a trained critic model. This design also removes the need to host a critic model during generator training, significantly reducing computational overhead. The critic model itself is partially supervised using a dataset consisting of input–output pairs and their corresponding reflection tokens, which were generated by a proprietary LM (i.e., GPT-4; OpenAI 2023).

Empirical results on six tasks, including reasoning and long-form generation, demonstrate that SELF-RAG significantly outperforms pre-trained and instruction-tuned LMs that have more parameters and widely adopted RAG approaches with higher citation accuracy. In particular, SELF-RAG outperforms retrieval-augmented ChatGPT on four tasks, Llama2-chat [Touvron et al., 2023] and Alpaca [Dubois et al., 2023] on all tasks. Our analysis demonstrates the effectiveness of training and inference with reflection tokens for overall performance improvements as well as test-time model customizations.

4.3.1 Method

SELF-RAG is a framework that enhances the quality and factuality of an LLM through retrieval and self-reflection, without sacrificing LLM's original creativity and versatility. Our end-to-end training lets an LM $\mathcal{M}$ **generate** text informed by **retrieved** passages, if needed, and **criticize** the output by learning to generate special tokens. These *reflection tokens* (Table 4.1) signal the need for retrieval or confirm the output's relevance, support, or completeness. In contrast, common RAG approaches

retrieve passages indiscriminately, without ensuring complete support from cited sources.

Problem Formalization and Overview

Formally, given input x, we train $\mathcal{M}$ to sequentially generate textual outputs y consisting of multiple segments $y = [y_1, \ldots, y_T]$, where y_t indicates a sequence of tokens for the t-th segment.[1] Generated tokens in y_t include text from the original vocabulary as well as the reflection tokens (Table 4.1).

Algorithm 1 SELF-RAG Inference

Require: Generator LM $\mathcal{M}$, Retriever $\mathcal{R}$, Large-scale passage collections $\{d_1, \ldots, d_N\}$
Input: input prompt x and preceding generation $y_{<t}$, **Output:** next output segment y_t
$\mathcal{M}$ predicts [Retrieve] given $(x, y_{<t})$ **if** [Retrieve] == Yes **then**
    Retrieve relevant text passages **D** using $\mathcal{R}$ given (x, y_{t-1}) ▷ Retrieve
    $\mathcal{M}$ predicts [IsRel] given x, d and y_t given $x, d, y_{<t}$ for each $d \in$ **D** ▷ Generate
    $\mathcal{M}$ predicts [IsSup] and [IsUse] given x, y_t, d for each $d \in$ **D** ▷ Critique
    Rank y_t based on [IsRel], [IsSup], [IsUse] ▷ Detailed in Section 4.3.1
end
$\mathcal{M}_{gen}$ predicts y_t given x ▷ Generate
$\mathcal{M}_{gen}$ predicts [IsUse] given x, y_t ▷ Critique

Inference overview. Figure 4.1 and Algorithm 1 present an overview of SELF-RAG at inference. For every x and preceding generation $y_{<t}$, the model decodes a retrieval token to evaluate the utility of retrieval. If retrieval is not required, the model predicts the next output segment, as it does in a standard LM. If retrieval is needed, the model generates: a critique token to evaluate the retrieved passage's relevance, the next response segment, and a critique token to evaluate if the information in the response segment is supported by the passage. Finally, a new critique token evaluates the overall utility of the response.[2] To generate each segment, SELF-RAG processes multiple passages in parallel and uses its own generated reflection tokens to enforce soft constraints (Section 4.3.1) or hard control (Algorithm 1) over the generated task output. For instance, in Figure 4.1 (right), the retrieved passages d_1 is selected at the first time step since d_2 does not provide direct evidence ([IsRel] is Irrelevant) and d_3 output is only partially supported while d_1 are fully supported.

[1] In this paper, we treat one sentence as a segment in our experiments, but our framework is applicable to any segment unit (i.e., sub-sentence).

[2] We follow Liu et al. [2023a] in using a "perceived" utility value that is independent of retrieved passages.

Training overview. SELF-RAG enables an arbitrary LM to generate text with reflection tokens by unifying them as next token predictions from the expanded model vocabulary (i.e., the original vocabulary plus reflection tokens). Specifically, we train the generator model $\mathcal{M}$ on a curated corpus with interleaving passages retrieved by a *retriever* $\mathcal{R}$ and reflection tokens predicted by a *critic* model $\mathcal{C}$. We train $\mathcal{C}$ to generate reflection tokens for evaluating retrieved passages and the quality of a given task output. Using the critic model, we update the training corpus by inserting reflection tokens into task outputs offline. Subsequently, we train the final generator model ($\mathcal{M}$) using the conventional LM objective to enable $\mathcal{M}$ to generate reflection tokens by itself without relying on the critic at inference time.

SELF-RAG Training

Here, we describe the supervised data collection and training of two models, the critic $\mathcal{C}$ and the generator $\mathcal{M}$.

Data collection for critic model. Manual annotation of reflection tokens for each segment is expensive [Wu et al., 2023]. A state-of-the-art LLM like GPT-4 [OpenAI, 2023] can be effectively used to generate such feedback [Liu et al., 2023b]. However, depending on such proprietary LMs can raise API costs and diminish reproducibility [Chen et al., 2023]. Our method requires fine-grained evaluations on multiple passages as well as segments for each input-output instance from the training dataset, increasing the number of evaluations required to generate SELF-RAG training data exponentially. To overcome those issues, we create supervised data by prompting GPT-4 to generate reflection tokens and then distill their knowledge into an in-house $\mathcal{C}$. For each group of reflection tokens, we randomly sample instances from the original training data: $\{X^{sample}, Y^{sample}\} \sim \{X, Y\}$. As different reflection token groups have their definitions and input, as shown in Table 4.1, we use different instruction prompts for them. Here, we use Retrieve as an example. We prompt GPT-4 with a type-specific instruction ("Given an instruction, make a judgment on whether finding some external documents from the web helps to generate a better response.") followed by few-shot demonstrations I the original task input x and output y to predict an appropriate reflection token as text: $p(r|I, x, y)$. Manual assessment reveals that GPT-4 reflection

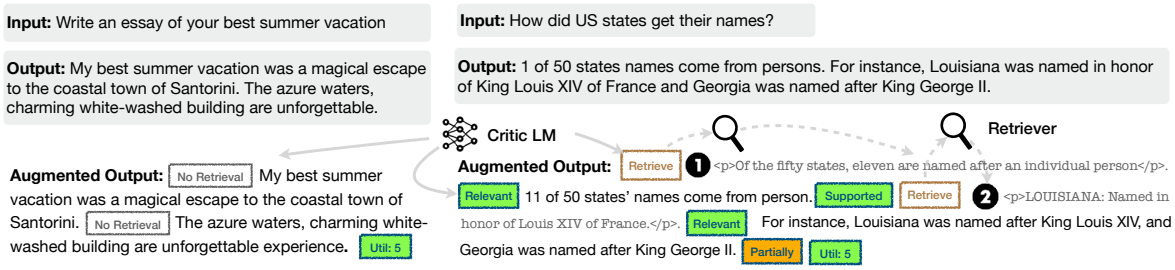


Figure 4.2: SELF-RAG training examples. The left example does not require retrieval while the right one requires retrieval; thus, passages are inserted.

token predictions show high agreement with human evaluations. We collect 4k-20k supervised training data for each type and combine them to form training data for $\mathcal{C}$.

Critic learning. After we collect training data $\mathcal{D}_{critic}$, we initialize $\mathcal{C}$ with a pre-trained LM and train it on $\mathcal{D}_{critic}$ using a standard conditional language modeling objective, maximizing likelihood:

$$\max_{\mathcal{C}} \mathbb{E}_{((x,y),r)\sim\mathcal{D}_{critic}} \log p_{\mathcal{C}}(r|x,y), \; r \text{ for reflection tokens.} \tag{4.1}$$

Though the initial model can be any pre-trained LM, we use the same one as the generator LM (i.e., Llama 2-7B; Touvron et al. 2023) for $\mathcal{C}$ initialization.

Data collection for generator. Given an input-output pair (x, y), we augment the original output y using the retrieval and critic models to create supervised data that precisely mimics the SELF-RAG inference-time process (Section 4.3.1). For each segment $y_t \in y$, we run $\mathcal{C}$ to assess whether additional passages could help to enhance generation. If retrieval is required, the retrieval special token Retrieve =`Yes` is added, and $\mathcal{R}$ retrieves the top K passages, $\mathbf{D}$. For each passage, $\mathcal{C}$ further evaluates whether the passage is relevant and predicts ISREL. If a passage is relevant, $\mathcal{C}$ further evaluates whether the passage supports the model generation and predicts ISSUP. Critique tokens ISREL and ISSUP are appended after the retrieved passage or generations. At the end of the output, y (or y_T), $\mathcal{C}$ predicts the overall utility token ISUSE, and an augmented output with reflection tokens and the original input pair is added to $\mathcal{D}_{gen}$. See the example training data in Figure 4.2.

Generator learning. We train the generator model $\mathcal{M}$ by training on the curated corpus augmented with reflection tokens $\mathcal{D}_{gen}$ using the standard next token objective:

$$\max_{\mathcal{M}} \mathbb{E}_{(x,y,r)\sim\mathcal{D}_{gen}} \log p_{\mathcal{M}}(y, r|x). \tag{4.2}$$

Unlike $\mathcal{C}$ training (Eq. 4.1), $\mathcal{M}$ learns to predict the target output as well as the reflection tokens. During training, we mask out the retrieved text chunks (surrounded by and in Figure 4.2) for loss calculation and expand the original vocabulary $\mathcal{V}$ with a set of reflection tokens {Critique, Retrieve}.

Connections to prior work on learning with critique. Recent work incorporates additional critique (feedback) during training, e.g., RLHF (Ouyang et al. 2022) via PPO. While PPO relies on separate reward models during training, we compute critique offline and directly insert them into the training corpus, where the generator LM is trained with a standard LM objective. This significantly reduces training costs compared to PPO. Our work also relates to prior work that incorporates special tokens to control generation [Keskar et al., 2019; Lu et al., 2022; Korbak et al., 2023]. Our SELF-RAG learns to generate special tokens *to evaluate its own prediction* after each generated segment, enabling the use of a soft re-ranking mechanism or hard constraints at inference (discussed next).

SELF-RAG Inference

Generating reflection tokens to self-evaluate its own output makes SELF-RAG controllable during the inference phase, enabling it to tailor its behavior to diverse task requirements. For tasks demanding factual accuracy [Min et al., 2023a], we aim for the model to retrieve passages more frequently to ensure that the output aligns closely with the available evidence. Conversely, in more open-ended tasks, like composing a personal experience essay, the emphasis shifts towards retrieving less and prioritizing the overall creativity or utility score. In this section, we describe approaches to enforce control to meet these distinct objectives during the inference process.

Adaptive retrieval with threshold. SELF-RAG dynamically decides when to retrieve text passages

by predicting Retrieve. Alternatively, our framework allows a threshold to be set. Specifically, if the probability of generating the Retrieve=`Yes` token normalized over all output tokens in Retrieve surpasses a designated threshold, we trigger retrieval.

Tree-decoding with critique tokens. At each segment step t, when retrieval is required, based either on hard or soft conditions, $\mathcal{R}$ retrieves K passages, and the generator $\mathcal{M}$ processes each passage in parallel and outputs K different continuation candidates. We conduct a segment-level beam search (with the beam size=B) to obtain the top-B segment continuations at each timestamp t, and return the best sequence at the end of generation. The score of each segment y_t with respect to passage d is updated with a critic score $\mathcal{S}$ that is the linear weighted sum of the normalized probability of each Critique token type. For each critique token group G (e.g., ISREL), we denote its score at timestamp t as s_t^G, and we compute a segment score as follows:

$$f(y_t, d, \text{Critique}) = p(y_t|x, d, y_{<t})) + \mathcal{S}(\text{Critique}), \text{where} \tag{4.3}$$

$$\mathcal{S}(\text{Critique}) = \sum_{G \in \mathcal{G}} w^G s_t^G \text{ for } \mathcal{G} = \{\text{ISREL}, \text{ISSUP}, \text{ISUSE}\}, \tag{4.4}$$

where $s_t^G = \frac{p_t(\hat{r})}{\sum_{i=1}^{N^G} p_t(r_i)}$ stands for the generation probability of the most desirable reflection token $\hat{r}$ (e.g., ISREL =`Relevant`) for the critique token type G with N^G distinct tokens (that represent different possible values for G). The weights w^G in Eq. 4.4 are hyperparameters that can be adjusted at inference time to enable customized behaviors at test time. For instance, to ensure that result y is mostly supported by evidence, we can set a weight term for the ISSUP score higher, while relatively lowering weights for other aspects. Alternatively, we could further enforce hard constraints during decoding using Critique e.g., filtering out a segment continuation when the model generates an undesirable token (e.g., ISSUP =`No support`).

4.3.2 Experiments

Tasks and Datasets

We conduct evaluations of our SELF-RAG and diverse baselines on a range of downstream tasks, holistically evaluating outputs with metrics designed to assess overall correctness, factuality, and

fluency. Throughout these experiments, we conduct zero-shot evaluations, where we provide instructions describing tasks without few-shot demonstrations [Wei et al., 2022a; Sanh et al., 2022].

Closed-set tasks include two datasets, i.e., a fact *verification dataset* about public health (**PubHealth**; Zhang et al. 2023a) and a *multiple-choice reasoning dataset* created from scientific exams (**ARC-Challenge**; Clark et al. 2018). We use accuracy as an evaluation metric and report on the test set. We aggregate the answer probabilities of target classes for both of these datasets.

Short-form generations tasks include two open-domain QA datasets, PopQA [Mallen, Asai, et al., 2023] and TriviaQA-unfiltered [Joshi et al., 2017], where systems need to answer arbitrary questions about factual knowledge. For PopQA, we use the long-tail subset, consisting of 1,399 rare entity queries whose monthly Wikipedia page views are less than 100. As the TriviaQA-unfiltered (open) test set is not publicly available, we follow prior work's validation and test split [Min et al., 2019; Guu et al., 2020], using 11,313 test queries for evaluation. We evaluate performance based on whether gold answers are included in the model generations instead of strictly requiring exact matching, following Mallen, Asai, et al. [2023]; Schick et al. [2023].

Long-form generation tasks include a biography generation task [Min et al., 2023a] and a long-form QA task **ALCE-ASQA** [Gao et al., 2023a; Stelmakh et al., 2022]. We use FactScore [Min et al., 2023a] to evaluate biographies, and we use official metrics of correctness (str-em), fluency based on MAUVE [Pillutla et al., 2021], and citation precision and recall [Gao et al., 2023a] for ASQA. [3]

Baselines

Baselines without retrievals. We evaluate strong publicly available pre-trained LMs, namely Llama2$_{7B,13B}$ [Touvron et al., 2023], instruction-tuned models, Alpaca$_{7B,13B}$ [Dubois et al., 2023] (our replication based on Llama2); and models trained and reinforced using private data, ChatGPT [Ouyang et al., 2022] and Llama2-chat$_{13B}$. For instruction-tuned LMs, we use the official system prompt or instruction format used during training if publicly available. We also compare our method to concurrent work, CoVE$_{65B}$ [Dhuliawala et al., 2023], which introduces iterative prompt engineering to improve the factuality of LM generations.

[3] https://github.com/princeton-nlp/ALCE

Baselines with retrievals. We evaluate models augmented with retrieval at test time or during training. The first category includes standard RAG baselines, where an LM (Llama2, Alpaca) generates output given the query prepended with the top retrieved documents using the same retriever as in our system. It also includes Llama2-FT, where Llama2 is fine-tuned on all training data we use without the reflection tokens or retrieved passages. We also report the result of retrieval-augmented baselines with LMs trained with private data: Ret-ChatGPT and Ret-Llama2-chat, which deploy the same augmentation technique above, as well as perplexity.ai, an InstructGPT-based production search system. The second category includes concurrent methods that are trained with retrieved text passages, i.e., SAIL [Luo et al., 2023] to instruction-tune an LM on the Alpaca instruction-tuning data with top retrieved documents inserted before instructions, and Toolformer [Schick et al., 2023] to pre-train an LM with API calls (e.g., Wikipedia APIs).[4]

Experimental Settings

Training data and settings. Our training data consists of diverse instruction-following input-output pairs. In particular, we sample instances from Open-Instruct processed data [Wang et al., 2023c] and knowledge-intensive datasets [Petroni et al., 2021; Stelmakh et al., 2022; Mihaylov et al., 2018]. In total, we use 150k instruction-output pairs. We use Llama2 7B and 13B [Touvron et al., 2023] as our generator base LM, and we use Llama2 7B as our base critic LM. For the retriever model $\mathcal{R}$, we use off-the-shelf Contriever-MS MARCO [Izacard et al., 2022a] by default and retrieve up to ten documents for each input.

Inference settings. As a default configuration, we assign the weight terms ISREL, ISSUP, ISUSE values of 1.0, 1.0 and 0.5, respectively. To encourage frequent retrieval, we set the retrieval threshold to 0.2 for most tasks and to 0 for ALCE [Gao et al., 2023a] due to citation requirements. We speed up inference using vllm [Kwon et al., 2023]. At each segment level, we adopt a beam width of 2. For a token-level generation, we use greedy decoding. By default, we use the top five documents from Contriever-MS MARCO [Izacard et al., 2022a]; for biographies and open-domain QA, we use additional top five documents retrieved by a web search engine, following Luo et al. [2023];

[4]We report numbers using the results reported in the paper as the implementations are not available.

LM	Short-form PopQA (acc)	TQA (acc)	Closed-set Pub (acc)	ARC (acc)	Long-form generations (with citations) Bio (FS)	Bio (em)	Bio (rg)	ASQA (mau)	ASQA (pre)	ASQA (rec)
LMs with proprietary data										
Llama2-c$_{13B}$	20.0	59.3	49.4	38.4	55.9	22.4	29.6	28.6	–	–
Ret-Llama2-c$_{13B}$	51.8	59.8	52.1	37.9	79.9	32.8	34.8	43.8	19.8	36.1
ChatGPT	29.3	**74.3**	70.1	**75.3**	71.8	35.3	36.2	68.8	–	–
Ret-ChatGPT	50.8	65.7	54.7	**75.3**	–	**40.7**	**39.9**	**79.7**	65.1	**76.6**
Perplexity.ai	–	–	–	–	71.2	–	–	–	–	–
Baselines without retrieval										
Llama2$_{7B}$	14.7	30.5	34.2	21.8	44.5	7.9	15.3	19.0	–	–
Alpaca$_{7B}$	23.6	54.5	49.8	45.0	45.8	18.8	29.4	61.7	–	–
Llama2$_{13B}$	14.7	38.5	29.4	29.4	53.4	7.2	12.4	16.0	–	–
Alpaca$_{13B}$	24.4	61.3	55.5	54.9	50.2	22.9	32.0	70.6	–	–
CoVE$_{65B}$ *	–	–	–	–	71.2	–	–	–	–	–
Baselines with retrieval										
Toolformer*$_{6B}$	–	48.8	–	–	–	–	–	–	–	–
Llama2$_{7B}$	38.2	42.5	30.0	48.0	78.0	15.2	22.1	32.0	2.9	4.0
Alpaca$_{7B}$	46.7	64.1	40.2	48.0	76.6	30.9	33.3	57.9	5.5	7.2
Llama2-FT$_{7B}$	48.7	57.3	64.3	65.8	78.2	31.0	35.8	51.2	5.0	7.5
SAIL*$_{7B}$	–	–	69.2	48.4	–	–	–	–	–	–
Llama2$_{13B}$	45.7	47.0	30.2	26.0	77.5	16.3	20.5	24.7	2.3	3.6
Alpaca$_{13B}$	46.1	66.9	51.1	57.6	77.7	**34.8**	36.7	56.6	2.0	3.8
Our SELF-RAG $_{7B}$	54.9	66.4	72.4	67.3	**81.2**	30.0	35.7	**74.3**	66.9	67.8
Our SELF-RAG $_{13B}$	**55.8**	**69.3**	**74.5**	**73.1**	80.2	31.7	**37.0**	71.6	**70.3**	**71.3**

Table 4.2: Overall experiment results on six tasks. Bold numbers indicate the best performance among non-proprietary models, and **gray-colored** bold text indicates the best proprietary model when they outperforms all non-proprietary models. * indicates concurrent or recent results reported by concurrent work. – indicates numbers that are not reported by the original papers or are not applicable. Models are sorted based on scale. FS, em, rg, mau, prec, rec denote FactScore (factuality); str-em, rouge (correctness); MAUVE (fluency); citation precision and recall, respectively.

for ASQA, we use the author-provided top 5 documents by GTR-XXL [Ni et al., 2022] across all baselines for a fair comparison.

4.3.3 Results and Analysis

Main Results

Comparison against baselines without retrieval. Table 4.2 (top) presents the baselines without retrieval. Our SELF-RAG (bottom two rows) demonstrates a substantial performance advantage over supervised fine-tuned LLMs in all tasks and even outperforms ChatGPT in PubHealth,

PopQA, biography generations, and ASQA (Rouge and MAUVE). Our approach also significantly outperforms a concurrent method that employs sophisticated prompt engineering; specifically, on the bio generation task, our 7B and 13B models outperform the concurrent CoVE [Dhuliawala et al., 2023], which iteratively prompts Llama2_{65B} to refine output.

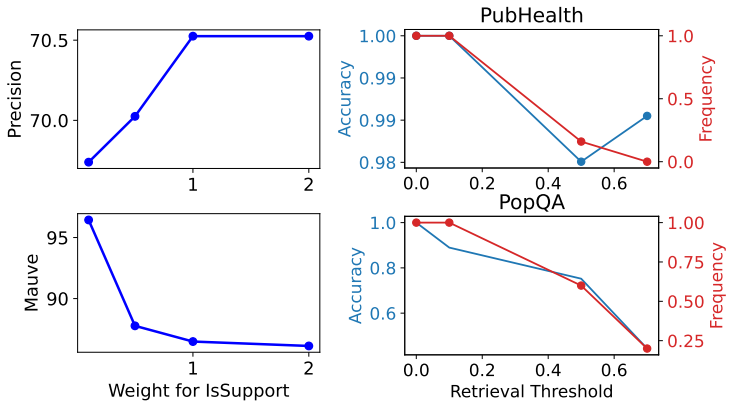
Comparison against baselines with retrieval. As shown in Tables 4.2 (bottom), our SELF-RAG also outperforms existing RAG in many tasks, obtaining the best performance among non-proprietary LM-based models on all tasks. Powerful instruction-tuned LMs with retrieval (e.g., LLama2-chat, Alpaca) show large gains from their non-retrieval baselines. However, we found that these baselines provide limited solutions for tasks where we cannot simply copy or extract sub-strings of retrieved passages. On PubHealth and ARC-Challenge, baselines with retrieval do not improve performance notably from their no-retrieval counterparts. We also observe that most baselines with retrieval struggle to improve citation accuracy. On ASQA, our model shows significantly higher citation precision and recall than all models except ChatGPT. Gao et al. [2023a] found that ChatGPT consistently exhibits superior efficacy in this particular task, surpassing smaller LMs. Our SELF-RAG bridges this performance gap, even outperforming ChatGPT in citation precision, which measures whether the model-generated claim is fully supported by cited evidence. Llama2-FT_{7B}, which is the baseline LM trained on the same instruction-output pairs as SELF-RAG without retrieval or self-reflection and is retrieval-augmented at test time only, lags behind SELF-RAG. This result indicates SELF-RAG gains are not solely from training data.

Analysis

Ablation studies. We conduct a set of ablations of our framework to identify which factors play key roles. We evaluate two model variants trained differently than our model: *No Retriever* trains an LM using the standard instruction-following method given instruction-output pairs, without retrieved passages; *No Critic* trains an LM trained with input-output pairs that are always augmented with the top one retrieved document without reflection tokens. This is similar to SAIL [Luo et al., 2023], and we use our instruction-output data instead of using the Alpaca dataset [Dubois et al., 2023], as in SAIL. We also conduct ablation on our inference-time algorithm, including *No retrieval* disables

	PQA (acc)	Med (acc)	AS (em)
SELF-RAG (50k)	45.5	73.5	32.1
Training			
No Retriever $\mathcal{R}$	43.6	67.8	31.0
No Critic $\mathcal{C}$	42.6	72.0	18.1
Test			
No retrieval	24.7	73.0	–
Hard constraints	28.3	72.6	–
Retrieve top1	41.8	73.1	28.6
Remove IsSup	44.1	73.2	30.6

(a) Ablation



(b) Customization **(c)** Retrieval

Figure 4.3: Analysis on SELF-RAG: (a) **Ablation studies** for key components of SELF-RAG training and inference based on our 7B model. (b) **Effects of soft weights** on ASQA citation precision and Mauve (fluency). (c) **Retrieval frequency** and *normalized* accuracy on PubHealth and PopQA.

retrieval during inference; *Hard constraints* indicates the model performance that retrieves when Retrieve=Yes instead of using the adaptive threshold; *Retrieve top 1* always retrieves and uses the top one document only, similar to standard RAG approaches; *Remove* IsSup indicates the model performance that removes IsSup score only during critique-guided beam search in Eq. 4.4. In this ablation experiment, we use a training instance size of 50k for a more efficient exploration of training variations. Later in this section, we conduct an analysis of the effect of training data size. We conduct the ablation studies on three datasets, PopQA, PubHealth, and ASQA. On ASQA, we evaluate models on sampled 150 instances and exclude ablations involving adaptive or no retrieval processes.

We show in Table 4.3a the ablation results. The top part of the table shows results for training ablations, and the bottom part is for inference ablations. We see that all components play important roles. We also observe a large performance gap between SELF-RAG and No Retriever or Critic baselines across tasks, indicating that training an LM with those models largely contributes to the performance gain of SELF-RAG. Using the top passages regardless of their relevance (Retrieve top 1) as in conventional RAG approaches causes a large drop in PopQA and ASQA, and removing IsSup during the beam search results hurts performance on ASQA. This demonstrates the effectiveness of SELF-RAG's capabilities of carefully selecting generations based on fine-grained multiple criteria,

instead of naively using all passages from the retrieval model or solely depending on relevance scores.

Effects of inference-time customization. One key benefit of our proposed framework is that it enables us to control how much each critique type affects the final generation sampling. We analyze the effects of different parameter weights on the top of our 7B model during inference time on ASQA, where multiple evaluation aspects are considered. Figure 4.3b shows the effects of changing the weighting term for ISSUP, which criticizes how supported the output is by the text passage. As the figure shows, increasing the weight leads to positive effects on the models' citation precision since this puts more emphasis on whether model generation is supported by the evidence. On the contrary, a larger weight results in lower MAUVE scores: when generation gets longer and more fluent, there are often more claims that are not fully supported by citations, consistent with findings by Liu et al. [2023a]. Our framework lets practitioners choose and customize models' behaviors at test time by adjusting such parameters without requiring additional training.

Efficiency and accuracy trade-off. Using our framework, practitioners can adjust how often retrieval occurs using the token probability of reward tokens. We evaluate how this adaptive threshold affects the overall accuracy and frequency of retrieval, and we evaluate the performance with varying numbers of threshold δ (larger δ results in less retrieval) on PubHealth and PopQA. Figure 4.3c shows that the model's retrieval frequencies dramatically change on both datasets. as δ varies. On one hand, performance deterioration by retrieving less is smaller on PubHealth but larger in PopQA.

4.3.4 Summary and Limitations

This work introduces SELF-RAG, a new framework to enhance the quality and factuality of LLMs through retrieval on demand and self-reflection. SELF-RAG trains an LM to learn to retrieve, generate, and critique text passages and its own generation by predicting the next tokens from its original vocabulary as well as newly added special tokens, called reflection tokens. SELF-RAG further enables the tailoring of LM behaviors at test time by leveraging reflection tokens. Our holistic evaluations on six tasks using multiple metrics demonstrate that SELF-RAG significantly

outperforms LLMs with more parameters or with conventional retrieval-augmented generation approaches.

Limitations. SELF-RAG has a few limitations, which we outline below. First, its effectiveness fundamentally relies on the quality of the underlying retrieval system. If the retrieved documents are entirely irrelevant, SELF-RAG, which is designed to select from and generate responses based on relevant information, will struggle to produce meaningful outputs. In the next section, we discuss how to build more reliable retrieval foundations to fully unlock the potential of models like SELF-RAG. Second, SELF-RAG requires training language models to generate a set of self-reflective tokens. While we briefly explore prompting-based approaches for self-reflective inference, we find that even state-of-the-art models often fail to reliably produce certain substrings, especially when multiple fine-grained aspects must be captured. We discuss follow-up work in this direction in Section 3.4. Lastly, our evaluation of SELF-RAG focuses on knowledge-intensive tasks using Wikipedia as the sole retrieval corpus. Assessing its effectiveness beyond general-domain knowledge-intensive settings is beyond the scope of this work.

4.4 Developing Versatile and Scalable Retrieval Systems

In this section, we introduce new retrieval foundations designed to address the limitations of standard neural retrieval systems outlined in Section 4.2. Our goal is to enhance the **versatility**, **robustness to complex queries**, and **efficiency** of current retrieval systems. We highlight TART (Section 4.4.1), our work on building the first instruction-following retriever to improve retrieval versatility. In Section 4.4.2, we briefly discuss additional contributions aimed at improving robustness and efficiency, including novel retrieval architectures and pre-training methodologies.

4.4.1 Task-aware Retriever: Improving Versatility of Retrievers

As discussed in Section 4.2, retrieval systems optimized for specific retrieval scenarios often struggle to identify relevant documents across diverse retrieval intents. Furthermore, even when the same query is given, the underlying intent can vary significantly depending on the task. For example,

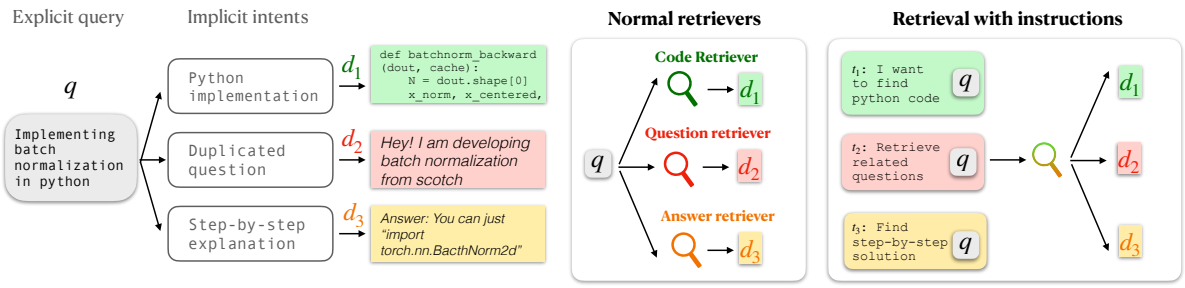


Figure 4.4: TART Overview. User intents are not fully captured in query $\boldsymbol{q}$ only (left). Conventional approaches (middle) take a query and retrieve documents from a closed corpus using a task-specific retriever. *Retrieval with instructions* (right) additionally takes explicit intent.

as illustrated in Figure 4.4, a query such as "implementing batch normalization in Python" may arise from different user goals, including finding an existing Python implementation, identifying duplicate answers in a forum, or seeking a step-by-step natural language explanation.

We study the problem of *retrieval with instructions*, where users provide explicit descriptions of their intent along with their queries to guide a retrieval system. Our solution is a general-purpose task-aware retrieval system, trained using multi-task instruction tuning and can follow human-written instructions to find relevant documents to a given query.

Problem: Retrieval with Instructions

Problem formulation. This work introduces a new problem formulation, *retrieval with instructions* (Figure 4.4 right). We are given a large collection of N documents $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_N\}$, a search task instruction $\boldsymbol{t}$ and a query $\boldsymbol{q}$. The problem of retrieval with instructions is to find a document $\boldsymbol{d} \in \mathcal{D}$ that is relevant to $\boldsymbol{q}$ according to the instruction $\boldsymbol{t}$. Compared to the standard retrieval setting (e.g., Figure 4.4 middle), the difference is the explicit definition of *relevance* in the instruction $\boldsymbol{t}$ as additional input to the system and a retrieval system needs to be task-aware—changing their relevance measure by attending to the instruction. This new formulation brings both new research challenges and opportunities. For instance, a retriever is now required to modify its search behavior according to the instructions. On the plus side, different datasets can be naturally grouped to train a single retriever, yielding benefits from cross-task interdependence. Instructions provide extra

Dataset	Instruction
NQ	`Retrieve a Wikipedia paragraph that answers this question.`
QReCC	`Find a dialogue response from dialogue history to answer the user's question.`
Arguana	`Retrieve a paragraph from an argument website that argues against the following argument.`
SciFact	`Find a sentence from a scientific paper to check if the statement is correct or not.`
MultiLexSum	`I want to find the one-sentence summary of this legal case.`

Table 4.3: Example instructions for TART. Example instructions for Natural Questions (NQ; Kwiatkowski et al. 2019), QReCC [Anantha et al., 2021], Arguana [Wachsmuth et al., 2018], SciFact [Wadden et al., 2020] and MultiLexSum [Shen et al., 2022]. Each instruction defines *intent*, *domain* and *unit*.

flexibility and enable zero-shot transfer via natural language instructions, unlike training with fixed task tags [Maillard et al., 2021]. A single task-aware retriever obviates the need to host multiple task-specific retrievers.

Instruction schema for retrieval. We introduce a novel schema to define an informative instruction for retrieval tasks, which have not been studied in prior instruction-following literature. An instruction that sufficiently describes an arbitrary retrieval task should include: *intent*, *domain* and *unit*. Specifically, *intent* describes how the retrieved text relates to the query, such as whether the text answers a question in the query or paraphrases it. *Domain* is the expected source or type of retrieved text, such as Wikipedia or PubMed articles. *Unit* defines the text block to retrieve, such as a sentence or a paragraph. Table 4.3 shows examples of instructions.

Dataset: BERRI

Multi-task training with instructions has not been studied in the area of retrieval due to the lack of resources and dedicated models. To facilitate research on retrieval with instructions, we introduce BERRI (Figure 4.5), a large-scale retrieval benchmark with expert-written annotations in a unified format, and subsequently the multitask instruction-following retrievers.

Task format. Each task $\mathcal{T}$ in BERRI consists of a corpus $\mathcal{D}$, queries $\mathcal{Q} = \{\boldsymbol{q_1}, \ldots, \boldsymbol{q}_K\}$, and an instruction $\boldsymbol{t}$, where K is the number of queries included in the task. An instance of each task includes a query $\boldsymbol{q}$, gold (relevant) documents $\boldsymbol{d}^+$, and negative (irrelevant) documents $\boldsymbol{d}^-$. For

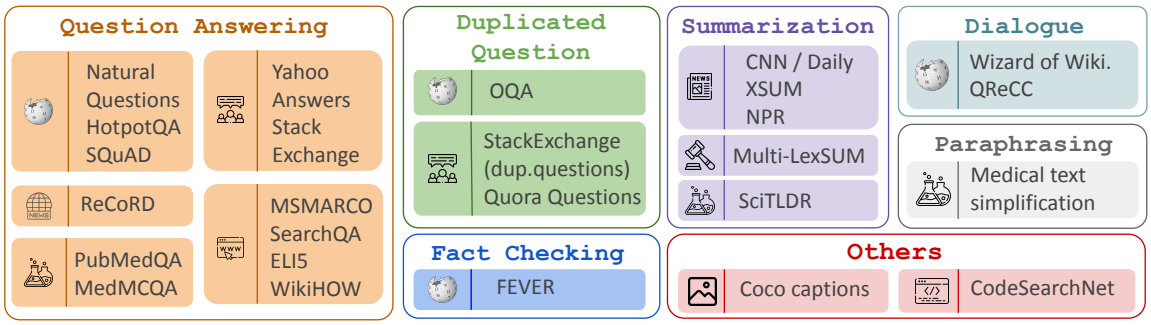


Figure 4.5: Overview of BERRI. This figure presents a subset of the datasets included in BERRI. For the complete list of datasets and data collections, please refer to Asai et al. [2023b].

each task, an explicit intent $\boldsymbol{t}$ is given.

Dataset selection and unification. We manually collect datasets from (1) KILT [Petroni et al., 2021], (2) the Sentence-Transformers Training Data for Text Embedding Models[5], and (3) manual searches in ACL anthologies and huggingface datasets[6] to cover diverse tasks and domains. Except for a few domains (e.g., Wikipedia) many domains do not have retrieval datasets, while there are a few datasets for other NLP tasks that can be cast as retrieval (e.g., sentence paraphrasing). Re-purposing those non-retrieval tasks as retrieval tasks enables the diversity of the domains as well as the instructions in BERRI. From initial collections of more than 60 datasets, we conduct manual dataset inspection and select 37 datasets (Figure 4.5) covering diverse domains (e.g., Wikipedia, scientific papers) and tasks (e.g., fact verification, dialogue response retrieval, QA).

Negative documents selection. Negative samples are crucial for training retrieval systems [Zhan et al., 2021; Qu et al., 2021]. In addition to randomly sampled negative samples (random negative documents), we introduce two types of challenging negative samples: denoised hard negative documents $\boldsymbol{d}^{\text{HD}}$ and instruction-unfollowing negative documents $\boldsymbol{d}^{\text{UF}}$. Figure 4.6 shows examples of gold documents and those negative samples.

For hard negatives $\boldsymbol{d}^{\text{HD}}$ we run Contriever [Izacard et al., 2022a] and then filter out false negative documents by running an off-the-shelf reranker[7] and keeping passages with low scores (smaller

[5] https://huggingface.co/datasets/sentence-transformers/embedding-training-data
[6] https://huggingface.co/docs/datasets/index
[7] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-12-v2

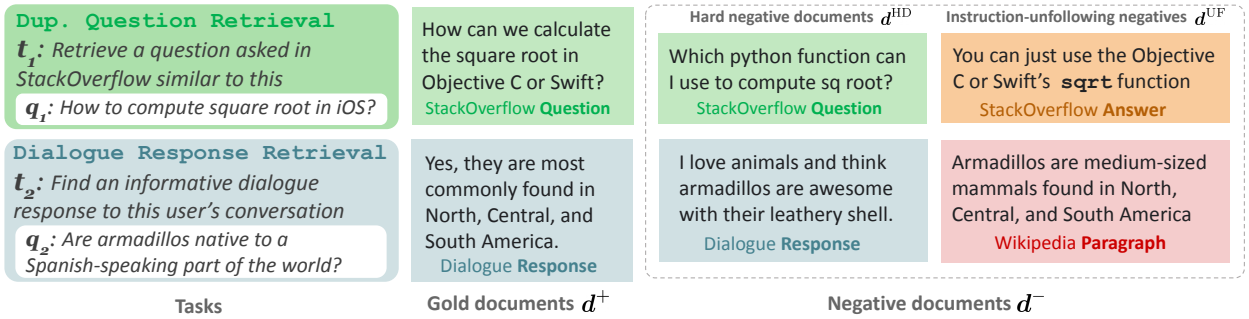


Figure 4.6: Examples of instruction-following training data. Examples of documents that are considered gold documents $\boldsymbol{d}^+$, and two types of negative documents $\boldsymbol{d}^-$: hard negatives $\boldsymbol{d}^{\text{HD}}$ and instruction-unfollowing negatives $\boldsymbol{d}^{\text{UF}}$ for two different query and instruction pairs.

than 0.1). We further introduce a new negative sampling strategy, instruction-*unfollowing* negative samples $\boldsymbol{d}^{\text{UF}}$, to make systems learn to retrieve documents that are well-suited to the instructions. As shown in Figure 4.6, given an instruction "find an informative dialogue response", a system should not retrieve a Wikipedia paragraph about armadillos, even though that is highly relevant to the query. To obtain such negative documents, we retrieve documents from a different task's target corpus using Contriever and consider all those documents to be negatives since they do not satisfy the instruction.

Models: TART-full and TART-dual

We now present TART (**TA**sk-aware **Re**Triever) trained on BERRI via multi-task instruction-tuning, leveraging our unified task-aware schema.

TART-dual inference. TART-dual adopts a dual-encoder architecture to independently encode queries with instructions and documents. It uses maximum inner product search (MIPS) over the embeddings [Karpukhin et al., 2020]. The similarity between a query $\boldsymbol{q}$ and a document $\boldsymbol{d}$, given an instruction $\boldsymbol{t}$, is calculated as follows:

$$\text{s}(\boldsymbol{t}, \boldsymbol{q}, \boldsymbol{d}) = \mathbf{E}([\boldsymbol{t}; \boldsymbol{q}])^T \mathbf{E}(\boldsymbol{d}), \tag{4.5}$$

where $\mathbf{E}(\cdot)$ is the embedding function[8] and $[\boldsymbol{t}; \boldsymbol{q}]$ is the concatenation of the instruction and query. For this model, document embeddings can be computed offline, improving inference efficiency at the cost of storage space [Yamada et al., 2021].

TART-full inference. The dual-encoder architecture is known to be less expressive due to its limited query-document interactions [Khattab and Zaharia, 2020]. To address this issue, we also explore a cross-encoder architecture [Nogueira and Cho, 2019], which computes the relevance between a query and each document by jointly encoding them with cross-attention. A cross-encoder model is often prohibitively expensive to scale up to millions of documents, so we first run a lightweight off-the-shelf dual-encoder retriever to retrieve the top documents. For each of these documents, TART-full computes the similarity score as:

$$\mathrm{s}(\boldsymbol{t}, \boldsymbol{q}, \boldsymbol{d}) = \mathrm{FFN}(\mathbf{E}([\boldsymbol{t}; \boldsymbol{q}; \boldsymbol{d}])), \tag{4.6}$$

where FFN represents an additional feed-forward network that predicts whether the document follows the instruction and is related to the query.

TART training. We train TART-dual and TART-full using the positive documents and three types of negative documents in BERRI with instructions (Figure 4.6). We initialize TART-full with encoders of T5-based instruction-following pretrained models, namely T0-3B [Sanh et al., 2022] and FLAN-T5-3B [Chung et al., 2024] for their empirical competitiveness, as found in prior work [Sachan et al., 2022]. We follow the EncT5 approach [Liu et al., 2021] and prepended each sequence with a start-of-sequence token. The token representation is then fed to a newly initialized feed-forward network. Unlike MonoT5 [Nogueira et al., 2020], we use their encoders only for parameter efficiency, reducing the number of the parameters to half.

We train TART-dual using annotated positive and negative documents in BERRI as well as in-batch negatives as follows:

$$\mathcal{L} = -\log \frac{e^{s(\boldsymbol{t}, \boldsymbol{q}, \boldsymbol{d}^+)}}{\sum_{\boldsymbol{d} \in \mathcal{B}} e^{s(\boldsymbol{t}, \boldsymbol{q}, \boldsymbol{d})}},$$

[8]We use a shared encoder since having separate encoders gave no additional gains in preliminary experiments.

Task	$\lvert q \rvert$	$\lvert \mathcal{C} \rvert$	Domain	Query	Gold documents
Ambig QA [Min et al., 2020]	1,172	18,809	Wikipedia	question	duplicated question
WIKIQA [Yang et al., 2015]	369	26,196	Wikipedia	question	answer sentence
SciFact [Wadden et al., 2020]	300	5183	Science	claim	scientific paper paragraph
GooAQ-Technical [Khashabi et al., 2021]	1,000	4,086	Technical	question	StackOverflow answer
LinkSo-Python [Liu et al., 2018]	1,000	485,413	Technical	question	StackOverflow question
CodeSearchNet-Python [Husain et al., 2019]	1,000	457,414	Code	comment	Python code

Table 4.4: The $\mathbb{X}^2$-Retrieval evaluation. In addition to the corpora listed above, we add the Natural Questions corpus data from BEIR [Thakur et al., 2021].

where $\mathcal{B}$ denotes all documents in the same mini-batch Karpukhin et al. [2020].

Following prior work [Nogueira and Cho, 2019], TART-full is trained with the cross entropy loss as:

$$\mathcal{L} = -\sum_{\boldsymbol{d} \in \boldsymbol{d}^+} \log s(\boldsymbol{t}, \boldsymbol{q}, \boldsymbol{d}) - \sum_{\boldsymbol{d} \in \boldsymbol{d}^-} \log(1 - s(\boldsymbol{t}, \boldsymbol{q}, \boldsymbol{d})).$$

Knowledge distillation from TART-full to TART-dual. The default hard negatives in BERRI rely on off-the-shelf models fine-tuned on MS MARCO; for some domains, the hard negative samples mined by those models can be less reliable. For a smaller dual-encoder model, those false positive and negative samples can diminish performance [Qu et al., 2021]. We apply hard knowledge distillation with TART-full [Qu et al., 2021]. We first train TART-full on the annotated gold documents and the negative documents in BERRI, and then update hard negative documents and positive documents with TART-full, with instructions.

Experimental Settings

We evaluate TART on zero-shot retrieval and our new evaluation setup, $\mathbb{X}^2$-Retrieval.

Zero-shot retrieval evaluations. We run experiments on two popular zero-shot retrieval benchmarks: **BEIR** [Thakur et al., 2021] and **LOTTE** [Santhanam et al., 2022]. None of the evaluation datasets overlap with BERRI. **BEIR** is a collection of diverse retrieval tasks in multiple domains where the retrieval target is restricted to the target corpus in a single domain. We used publicly

available datasets.[9] **LOTTE-Search** samples GooAQ [Khashabi et al., 2021] questions whose answers come from certain forums in StackExchange. We evaluate our model in the pooled setup, where documents come from forums in diverse domains (e.g., cooking, technical). GooAQ is not included in our training set. In LOTTE, our instructions specify which *forum* our system should retrieve evidence from (e.g., "Retrieve a *cooking* StackExchange forum post").

Following Thakur et al. [2021], for BEIR, we use NDCG@10 as our primary metric on BEIR. For LOTTE-pooled, we use Success@5 (= Recall@5) as our primary metric, as in the original paper Santhanam et al. [2022].

$\mathbb{X}^2$-Retrieval evaluation. Users' intents can be diverse, requiring searching in an open-domain environment [Piktus et al., 2021], which is currently under-explored. We introduce a more realistic evaluation setup, $\mathbb{X}^2$-Retrieval (Cross-task Cross-domain Retrieval), where several retrieval tasks with different intents are pooled to form a single retrieval target containing diverse documents. This requires a system not only to adapt to a new task in a zero-shot manner but also to model users' intents expressed in natural languages to meet their information needs. Below, we summarize the core design components of $\mathbb{X}^2$-Retrieval.

- **Tasks and queries:** Our $\mathbb{X}^2$-Retrieval evaluation covers six datasets across three domains, namely, Wikipedia, Science, and Technical (Table 4.4) domains. The key challenge here includes datasets with different search intents that may not always be obvious from the queries alone.
- **A pooled corpus:** For the primary *pooled* setup, we combine all documents from different tasks and the BEIR NQ Wikipedia corpus to form a single retrieval corpus, consisting of approximately 3.7 million documents. We also report the simplified *closed* setup performance as an oracle setup, where a system retrieves only from the original corpus.
- **Metrics**: We report NDCG@10 on both pooled and closed setups for each task. We also evaluate the performance gap between the closed and pooled setups and refer to it as *robustness*. A smaller gap means that the model is distracted less by the documents from undesirable corpora.

[9] Following Dai et al. [2023], we exclude Natural Questions, MS MARCO, HotpotQA, FEVER, and CQADupStack from our evaluation targets for fair comparison since they are included either in encoders' pretraining or in BERRI.

Baselines. We compare TART with various state-of-the-art methods. The first group is unsupervised models that are not trained or trained on unlabeled text; these include **Contriever** [Izacard et al., 2022a] and **BM25**. We also compare TART with **UPR** [Sachan et al., 2022], which reranks the Contriever results using a pretrained T0-3B. The second group trains retrievers and rerankers on MS MARCO or a few large-scale datasets and directly transfers them to new tasks with no adaptations, including **MonoT5** [Nogueira et al., 2020], **Contriever-MS MARCO** and **Contriever-MS MARCO + Cross Encoder (CE)**, **ColBERT v2** [Santhanam et al., 2022], and **SGPT-6.8B** [Muennighoff, 2022]. The final group of models is specialized retrievers trained for each task on automatically generated task data. **Promptagator** [Dai et al., 2023] generates large amount of in-domain data using FLAN [Wei et al., 2022a], and **GPL** [Wang et al., 2022] generates them using DocT5Query [Nogueira et al., 2019]. We also compare TART with their counterparts trained on BERRI and evaluated without instructions, **TART-dual w/o I** and **TART-full w/o I**.

Experimental settings. We initialize TART-full from the T0-3B Sanh et al. [2022] and FLAN-T5 encoder Chung et al. [2024]. We sample positive and negative passages with a 1:4 ratio. We initialize TART-dual from Contriever-MS MARCO [Izacard et al., 2022a], which is based on BERT-base.[10] Per-GPU batch size is 16, and for each positive document, we sample in total 5 negative passages, where 90% of them are randomly sampled from $\mathcal{D}$, and 10% are sampled from $\boldsymbol{d}^{\text{HD}}$ and $\boldsymbol{d}^{\text{UF}}$. We use top 100 Contriever-MS MARCO results as the TART-full initial candidates.[11]

Results

Zero-shot evaluation. As shown in Table 4.5, TART-full and TART-dual largely outperform their counterparts trained and tested without instructions, demonstrating the effectiveness of instruction-tuning for better zero-shot retrieval. TART-full significantly outperforms larger models and customized models trained on millions of synthetically generated in-domain data, advancing the state of the art on BEIR and LOTTE. Unlike prior methods that require additional data generation, TART

[10] We also tried larger models such as SGPT-1.3B [Muennighoff, 2022], but observed large performance drop on some datasets, resulting in lower average than Contriever-based TART-dual. Therefore, we use Contriever-based TART-dual.

[11] We found that combining TART-full with the original Contreiver performs better than combining TART-full with TART-dual, possibly because TART-full uses the hard negative samples retrieved by Contriever's top-retrieved results.

	model size & rerank			BEIR										LOTTE
	Ret.	Gen.	K	TREC	NFC	FQA	ARG	TOU	DBP	SCD	CLI	SCF	Avg.	Search-Pooled
BM 25	0	0	0	65.6	32.5	23.6	31.5	36.7	31.3	15.8	21.3	66.5	36.0	48.3
Contriever	110M	0	0	27.4	31.7	24.5	37.9	19.3	29.2	14.9	15.5	64.9	29.3	55.5
UPR†	3B	0	0	60.4	33.3	45.0	50.3	21.3	33.8	17.3	9.5	69.6	37.8	–
Contriever (MS)	110M	0	0	59.6	32.8	32.9	44.6	23.0	41.3	16.5	23.7	67.7	38.0	66.0
Contriever+CE†	133M	0	100	70.1	34.4	36.7	41.3	29.8	47.1	17.1	25.8	69.2	41.3	73.5
ColBERT-v2	110M	0	0	73.8	33.8	35.6	47.9	26.3	44.6	15.8	17.6	69.3	40.5	71.6
BM25 + MonoT5 (3B)†	3B	0	1000	**79.6**	**38.4**	**51.2**	28.8	20.0	**47.8**	18.4	28.9	**77.7**	43.4	–
SGPT-6.8B	6.8B	0	0	87.3	36.2	37.2	51.4	25.4	39.9	**19.7**	30.5	74.7	44.7	–
GPL	66M×9	220M	0	72.6	–	32.8	–	–	–	–	–	66.4	–	–
Promptagator	110M ×9	175B	0	72.7	33.4	40.4	53.8	26.6	36.4	16.3	21.4	62.3	40.4	–
Promptagator (rank)†	220M ×9	175B	200	76.0	36.0	45.9	53.1	27.8	41.3	19.1	22.6	73.6	43.9	–
TART-dual	110M	0	0	64.9	33.6	34.2	48.6	20.7	41.3	14.1	14.7	70.1	38.1	56.9
TART-full (T0-3B)†	1.5B	0	100	71.7	34.0	42.2	49.8	**31.2**	45.1	17.5	30.0	75.8	44.1	**75.7**
TART-full (FLAN-T5)†	1.5B	0	100	72.8	33.4	41.8	**51.5**	24.9	46.8	18.7	**35.4**	**77.7**	**44.8**	73.1
TART-dual w/o I	100M	0	0	46.3	32.7	28.6	44.7	12.3	33.0	12.8	15.7	67.4	32.6	56.7
TART-full† (T0-3b) w/o I	1.5B	0	100	57.2	37.1	41.3	50.4	18.3	41.3	18.3	32.5	73.2	41.1	71.2

Table 4.5: Zero-shot retrieval results on BEIR and LOTTE-Search. † indicates the models using cross-encoder-based reranking models. The first group of models use no labeled data during training. The second group uses MS MARCO at training time but has no customized task-specific data. The third group trains individual retrieval systems using automatically generated data. TREC, NFC, FQA, ARG, TOU, DBP, SCD, CLI, SCF indicates TREC-COVID, FIQA, NF Corpus, Arguana, Touche-2020, DBPedia, SciDocs, Climate- Fever, and SciFact, respectively. "×9" of GPL, Promptagator means that those models train customized models for each dataset.

only requires a single human-written instruction to adapt to a new task.

Compared to other methods using cross-encoder-based reranking models (e.g., BM25+MonoT5), TART-full uses a much smaller number of paragraphs to be re-ranked, which significantly reduces latency caused by reranking at test time. The large performance gain from Contriever (MS) to TART-dual on six out of the nine BEIR tasks (e.g., SciFact, Arguana) shows the effectiveness of instructions and knowledge distillations. However, for the other three datasets (e.g., Touche-2020), TART-dual shows large performance deterioration. We hypothesize that model capacity (i.e., BERT-base) and limited interactions between the query and document embeddings could be major bottlenecks. Prior work on instruction training in large LMs has shown that smaller models often do not get as much benefit as larger ones from instructions and increasing dataset size, possibly due to their limited model capacities [Chung et al., 2024]. Su et al. [2023] also observe more significant gain from instruction tuning when they use larger encoder models (i.e., GTR-base v.s. GTR-XL), reporting

	AMB		WQA		SCF		GAT		LSO		CSP		Avg.		Δ
	cl	pl	cl	pl	cl	pl	cl	pl	cl	pl	cl	pl	cl	pl	cl−pl
Contriever	**96.8**	93.8	80.9	54.1	67.7	57.4	73.2	59.8	28.0	26.7	36.7	36.1	63.9	54.6	9.3
Contriever+CE	96.6	47.4	78.2	58.4	69.1	61.7	75.4	66.0	**32.1**	**31.4**	42.0	40.2	65.5	50.9	13.4
TART-dual	96.3	**95.3**	80.2	**63.1**	70.1	66.2	75.0	65.0	23.0	23.4	31.3	31.3	60.5	53.6	**6.9**
TART-full (T0)†	91.1	90.5	82.1	52.5	74.7	66.2	**80.5**	**68.6**	25.1	24.9	51.4	**51.4**	67.5	**59.1**	8.4
TART-full (FLAN)†	94.0	89.6	**86.9**	55.9	**77.4**	**66.3**	78.3	66.7	18.1	18.4	**51.8**	50.1	**67.7**	57.8	9.9

Table 4.6: $\mathbb{X}^2$-Retrieval results. Δ shows the gap of the average performance in the pooled and closed settings. AMB, WQA, GAT, LSO, CSP denote AmbigQA, WikiQA, GooAQ-Technical, LinkSO, and CodeSearchNet-Python.

Query: how to calculate the distance between two points using longitude and latitude	
Instruction	**Top document**
Retrieve an answer post from StackOverflow to this question	SELECT getDistance(lat1,lng1,lat2,lng2) as distance FROM your_table. Here's a MySQL function that will take two latitude longitude pairs, and give you the distance in degrees between the two points. It uses the Haversine formula to calculate the distance.
Find a similar question asked in StackOverflow	tried implementing formula good two points testing yet code working distance returns.
Query: When did the kim family come to power?	
Instruction	**Top document**
find an answer sentence	Kim came to lead the Soviet-backed North's provisional government, becoming the first premier of its new government, the Democratic People's Republic of Korea (commonly known as North Korea), in 1948. He started the Korean War in 1950 with hopes to reunify the region. (Wikipedia)
Find a similar question	When did the kim family come to power in North Korea? (Ambig QA)
Query: 10% of sudden infant death syndrome (SIDS) deaths happen in newborns aged less than 6 months	
Instruction	**Top document**
retrieve a scientific paper paragraph to verify this	Despite declines in prevalence during the past two decades, sudden infant death syndrome (SIDS) continues to be the leading cause of death for infants aged between 1 month and 1 year in developed countries. Behavioral risk factors identified in epidemiological studies include prone and side positions for infant sleep, smoke exposure, soft bedding, and sleep surfaces, and overheating. (Scientific paper)
Find a Wikipedia paragraph to verify this	By definition, SIDS deaths occur under the age of one year, with the peak incidence occurring when the infant is at 2 to 4 months of age. (Wikipedia)

Table 4.7: TART Retrieval Results. Examples of the model's predictions given different instructions with the same query. The queries and documents are from $\mathbb{X}^2$-Retrieval.

performance deterioration in retrieval tasks when they instruction tune 335 million parameter base model. Future work can investigate efficient architectures that enable more rich interaction between queries with instructions and documents.

$\mathbb{X}^2$-Retrieval evaluation. Table 4.6 shows the models' $\mathbb{X}^2$-Retrieval performance. Contriever and Contriever+CE show competitive closed performance in the closed setup, as in BEIR, but they struggle in the pooled setup due to their inability to handle human instructions. Especially Contriever+CE shows a large performance drop on AmbigQA-pooled by retrieving documents instead of queries due to the biases from fine-tuning on a QA dataset (i.e., MS MARCO) only.

TART-full shows the best-closed performance and pooled performance, indicating its strong zero-shot adaptation and cross-task abilities. We found that a model can flexibly change its behavior based on the instructions, as shown in Table 4.7. TART-dual shows strong performance on the pooled setup, indicating that smaller models can be also guided by explicit instructions.

4.4.2 Improving Reliability and Efficiency of Retrievers

Robustness to Complex Queries

In real-world scenarios, it is crucial to handle complex search queries, such as those requiring multi-hop retrieval [Yang et al., 2018] or involving rare entities [Mallen, Asai, et al., 2023]. Our work focuses on improving the architectures and pre-training strategies to address these challenges. Below, we briefly outline the key ideas behind our approach. For a more comprehensive discussion of the methodologies and results, we refer readers to the original studies [Asai et al., 2020; Yamada et al., 2020, 2021].

PATHRETRIEVER: learning to retrieve reasoning paths over graph. We developed PATHRETRIEVER [Asai et al., 2020], which combines neural sequence prediction models with a structured symbolic graph that connects all documents in the datastore. Rather than retrieving all relevant documents simultaneously [Karpukhin et al., 2020], PATHRETRIEVER searches for "reasoning paths" by predicting the next document node in the graph, enabling it to retrieve documents sequentially. We constructed this document graph using Wikipedia hyperlinks and trained an RNN to navigate it effectively. Notably, our system can dynamically determine the number of reasoning steps required for a given query, making it applicable to both complex multi-hop datasets like HotpotQA [Yang et al., 2018] and open-domain datasets like NQ [Kwiatkowski et al., 2019]. This adaptability stands

in contrast to prior methods [Ding et al., 2019], which require the number of retrieval steps to be fixed in advance, limiting generalization to diverse retrieval scenarios.

LUKE: entity-aware self-attention and pre-training. Prior studies have shown that neural retrieval systems tend to generalize well to common entities but often struggle to retrieve relevant documents for queries involving less frequent ones [Sciavolino et al., 2021; Chen et al., 2022; Mallen, Asai, et al., 2023]. While entity representations play a vital role in NLP [Yamada et al., 2021], conventional pre-training methods only learn them implicitly, splitting entities into subword tokens.

To address this, we introduced **LUKE**, a new encoder-only LM that jointly represents words and entities [Yamada et al., 2020]. LUKE introduces architectural modifications to Transformers [Vaswani et al., 2017], including entity-aware self-attention, and a novel pre-training objective that leverages Wikipedia's rich entity annotations. This design allows LUKE to treat entities as independent tokens and compute contextualized representations for both words and entities, enabling the model to directly capture complex entity relationship, similar to how pre-trained LMs capture word-level semantics [Clark et al., 2019].

After pre-training, LUKE can be fine-tuned for a wide range of downstream tasks. Our experiments show that LUKE achieves state-of-the-art performance on several benchmarks, including entity typing [Choi et al., 2018b], QA [Rajpurkar et al., 2016], and relation classification [Zhang et al., 2017]. Moreover, LUKE has been successfully extended to enhance retrieval systems [Tu and Padmanabhan, 2022], demonstrating the effectiveness of entity-aware representations in improving robustness to entity-centric queries.

Efficiency for Hosting Large-Scale Datastores

Scaling retrieval systems to massive datastores such as corpora containing billions of documents poses significant challenges due to the increased storage and memory demands at inference time [Shao et al., 2024]. This is primarily because large indexes must be kept in memory during retrieval [Yamada et al., 2021]. Reducing index size is therefore critical for deploying retrieval systems in real-world applications. We have developed new retrieval methods that significantly

improve inference-time efficiency, particularly with respect to memory usage.

BPR: Efficient Passage Retrieval with Hashing. Our method, **BPR** (Binary Passage Retriever) [Yamada et al., 2021], learns to hash continuous vectors into compact binary codes using a multi-task objective that jointly trains both the encoder and hash functions in an end-to-end fashion. To maintain high retrieval accuracy while improving search-time efficiency, BPR generates both binary codes and continuous embeddings for queries through multi-task learning over two objectives. We evaluated BPR on several standard QA benchmarks and found that it achieves performance comparable to DPR [Karpukhin et al., 2020], while dramatically reducing memory usage from 65GB to just 2GB.

4.5 Conclusions and Discussion of Subsequent Work

Our work establishes new foundations for Retrieval-Augmented LMs by addressing the core limitations of naive RAG systems. We introduced SELF-RAG to enable dynamic retrieval, generation, and self-evaluation, and developed robust retrievers through novel architectures, pre-training, and instruction tuning. These methods have shaped state-of-the-art systems, inspired new research directions, and been widely adopted in real-world applications.

Here, we summarize the subsequent research and real-world adoption of the foundational contributions presented in this chapter.

Advances in modeling of RAG. Since the release of SELF-RAG, there has been rapid progress in developing advanced RAG systems and training more effective LMs for retrieval, leading to improved performance and efficiency of SELF-RAG. Several work explores better methods for augmenting instructional tuning datasets with search results, enabling models to filter out irrelevant context and enhance citation accuracy [Chan et al., 2024; Nguyen et al., 2024; Huang et al., 2024]. While SELF-RAG investigates the effectiveness of using a single model to perform planning, generation, and self-evaluation, other studies have proposed using separate models for each of these functions to improve performance and efficiency [Wang et al., 2025b]. The idea of

leveraging special tokens for retrieval and enhanced reasoning has also been successfully applied to multimodal reasoning [Cocchi et al., 2025].

Advances in retriever modeling and dataset development. Our proposed retrieval systems have been widely adopted and extended by follow-up work. In particular, instruction-following retrieval has become a core component of state-of-the-art retrievers on popular benchmarks such as MTEB [Muennighoff et al., 2023].[12] Our instruction design and training recipes have been used in several top-performing models, including **GRIT** [Muennighoff et al., 2025] and **NV-Embed** [Lee et al., 2025]. Instruction-tuning has also been successfully extended to multimodal retrieval scenarios, as demonstrated by **UNIIR** [Wei et al., 2024a].

Several recent works have introduced new benchmarks aimed at evaluating how well LMs and retrievers follow complex user instructions, advancing the field of instruction-aware retrieval. While $\mathbb{X}^2$-Retrieval introduced a cross-task retrieval setup, its instruction diversity and dataset coverage remain relatively limited due to constraints in its source data. **INSTRUCTOR** [Oh et al., 2024] expands this direction by generating diverse, instruction-rich queries from MS MARCO, revising target texts to match the instructions, and applying systematic filtering using GPT-4o. **FollowIR** [Weller et al., 2025] builds on three TREC collections, incorporating instructions written for professional annotators and modifying them with additional constraints that narrow the definition of relevance. While results on both benchmarks suggest that TART maintains robustness to more complex instructions compared to other similarly sized or even larger retrieval systems [Su et al., 2023; Ni et al., 2022], there remains room for improvement in achieving strong instruction-following performance, especially in smaller and more efficient retrievers.

Applications in safety-critical domains. Our new methodologies for LMs and retrievers aim to make them more reliable and explainable, by generating reasoning chains or using explicit self-reflective tokens. These capabilities make them particularly well-suited for safety-critical domains such as biomedical applications and scientific research assistants. Below, we highlight several follow-up works that build on this line of research. **Bio-SelfRAG** [Jeong et al., 2024a] extends Self-

[12] `https://huggingface.co/spaces/mteb/leaderboard_legacy`

RAG to biomedical domains by constructing specialized training data using biomedical instruction sets such as Mol-Instructions [Fang et al., 2024], achieving state-of-the-art performance on multiple medical QA benchmarks. **ScholarCopilot** [Wang et al., 2025a] introduces a new RAG system designed to assist with academic writing. It dynamically determines when retrieval is necessary by generating special retrieval tokens, an approach inspired by SELF-RAG, to guide the model's decision-making process. **CO-Search** [Esteva et al., 2021] extends PATHRETRIEVER to create a search engine tailored for complex queries over COVID-19 literature. It adapts the retriever to the biomedical domain through further fine-tuning on PubMed, enabling more effective retrieval in a high-stakes scientific context.

Integration into widely-used libraries. Our work has been incorporated into some of the most widely adopted libraries that serve as the backbone of modern NLP and LLM applications. SELF-RAG has been officially integrated into both LangChain[13] and LlamaIndex[14], two of the most popular libraries for building LLM-based pipelines. Additionally, LUKE has been integrated into Hugging Face Transformers[15], and has been downloaded over 1.5 million times.

[13] https://langchain-ai.github.io/langgraph/tutorials/rag/langgraph_self_rag_local/
[14] https://docs.llamaindex.ai/en/stable/api_reference/packs/self_rag/
[15] https://huggingface.co/docs/transformers/en/model_doc/luke

Chapter 5

Making Real-World Impacts with Retrieval-Augmented LMs

5.1 Overview

In Chapter 4, we introduced new foundations for Retrieval-Augmented LMs, achieving state-of-the-art results on widely used academic benchmarks. As discussed in Chapter 2, Retrieval-Augmented LMs offer distinct advantages over conventional LLMs, such as mitigating hallucinations in long-tail scenarios, accessing up-to-date information, and avoiding reliance on non-permissively licensed corpora. These strengths are especially relevant in safety-critical applications that often involve sensitive or private data. Yet, the effectiveness of Retrieval-Augmented LMs in these domains remains underexplored, even as they are increasingly deployed in real-world settings. This raises a critical question: *Can they effectively support expert-domain tasks in high-stakes applications?*

In this chapter, we demonstrate the real-world impact of Retrieval-Augmented LMs in various domains, including **scientific research**, **software engineering**, and **multilingual information access**. A key challenge in these settings is the scarcity of training data, evaluation benchmarks, and even usable datastores. Addressing this requires a holistic approach spanning data collection, model development, and rigorous evaluation.

We first describe OPENSCHOLAR (Section 5.2), the first fully open Retrieval-Augmented LM de-

signed for scientific literature synthesis. Despite growing excitement around building LM assistants to accelerate scientific discovery [Si et al., 2025; Tian et al., 2024; Lu et al., 2024; Skarlinski et al., 2024], prior efforts have largely relied on proprietary, black-box LMs and extensive prompting strategies. Moreover, the absence of realistic benchmarks that capture real-world scientist–LM interactions has hindered rigorous progress evaluation. To address these gaps, we developed a realistic, multi-disciplinary benchmark, SCHOLARQABENCH, by recruiting dozens of expert annotators (Ph.D. holders and candidates) and designing a multi-dimensional evaluation framework tailored to literature synthesis. Building on the retrieval-augmented generation methods introduced in Chapter 4, we developed a novel self-feedback-guided generation approach that tightly integrates retrieval and generation to produce more well-structured and comprehensive responses—qualities especially important for scientific tasks. Using synthetically generated queries, we trained a 7B LM and associated retrievers. Our resulting model, OPENSCHOLAR-8B, outperformed proprietary systems such as GPT-4o, Perplexity Pro, and PaperQA2 [Skarlinski et al., 2024] on SCHOLARQABENCH. Expert annotators also preferred OPENSCHOLAR 's answers over expert-written responses in our systematic expert evaluations. We further released the first freely accessible public demo of its kind[1], which has already been tested by over 30,000 researchers and practitioners from a wide range of scientific fields.

In Section 5.3, we explore the effectiveness of Retrieval-Augmented LMs for code generation. Code generation presents many of the previously mentioned challenges associated with parametric LMs, such as the need for frequent knowledge updates and the dynamic incorporation of private or newly available information, making retrieval-augmented approaches particularly promising in this domain. We develop CODERAG-BENCH [Wang, Asai, et al., 2025], a new holistic benchmark designed to evaluate Retrieval-Augmented LMs across diverse software engineering scenarios. Our findings show that competitive LMs benefit significantly from incorporating a broad range of documents, such as library documentation, coding tutorials, and open-access repositories. This integration notably boosts performance in tasks involving long-tail knowledge and understanding of private repositories. However, both retrieval and language modeling still have considerable

[1] https://openscilm.allen.ai/

room for improvement to fully realize the potential of such context integration.

Finally, we introduce our pioneering work on cross-lingual Retrieval-Augmented LMs (Section 5.4) to enhance information access across languages. In such systems, retriever locates relevant documents across languages and the LM generates outputs in the target language conditioned on these multilingual contexts. This capability is especially important for addressing global disparities in information access, as many languages lack sufficient online resources. We first introduce a new task, Cross-lingual Open-Retrieval QA (XORQA), and developed the first large-scale benchmark for open-retrieval QA in seven languages, called XOR-TyDi [Asai et al., 2021a]. It inspired numerous follow-up datasets and shared tasks [Asai et al., 2022b; Zhang et al., 2023b], including the first open-domain QA benchmarks for African languages [Ogundepo et al., 2023] and product QA [Shen et al., 2023]. We also developed the first end-to-end multilingual Retrieval-Augmented LM, CORA, which achieved state-of-the-art results on both XORQA and MKQA [Longpre et al., 2021].

5.2 Science: OpenScholar

Synthesizing knowledge from scientific literature is essential for uncovering new research directions, refining methodologies, and supporting evidence-based decisions. However, the vast volume of papers published annually makes it increasingly difficult for researchers to stay informed. Effective synthesis requires precise retrieval, accurate attribution, and real-time access to current literature. While LMs show promise in assisting researchers, they face significant challenges, including hallucinations [Mallen, Asai, et al., 2023; Mishra et al., 2024], reliance on outdated pre-training data [Kasai et al., 2022], and a lack of transparent attribution. For instance, when tasked with citing up-to-date literature, GPT-4 fabricated citations in 78-90% of cases across fields like computer science and biomedicine in our experiments.

Retrieval-augmented LMs [Lewis et al., 2020b; Guu et al., 2020], on the other hand, can mitigate many of these issues by integrating retrieved external knowledge sources at inference time, driving the development of systems for literature search and synthesis. However, many such systems rely on black-box APIs or general-purpose LLMs that are neither optimized for literature synthesis nor paired with open, domain-specific retrieval datastores (i.e., a processed corpus and corresponding

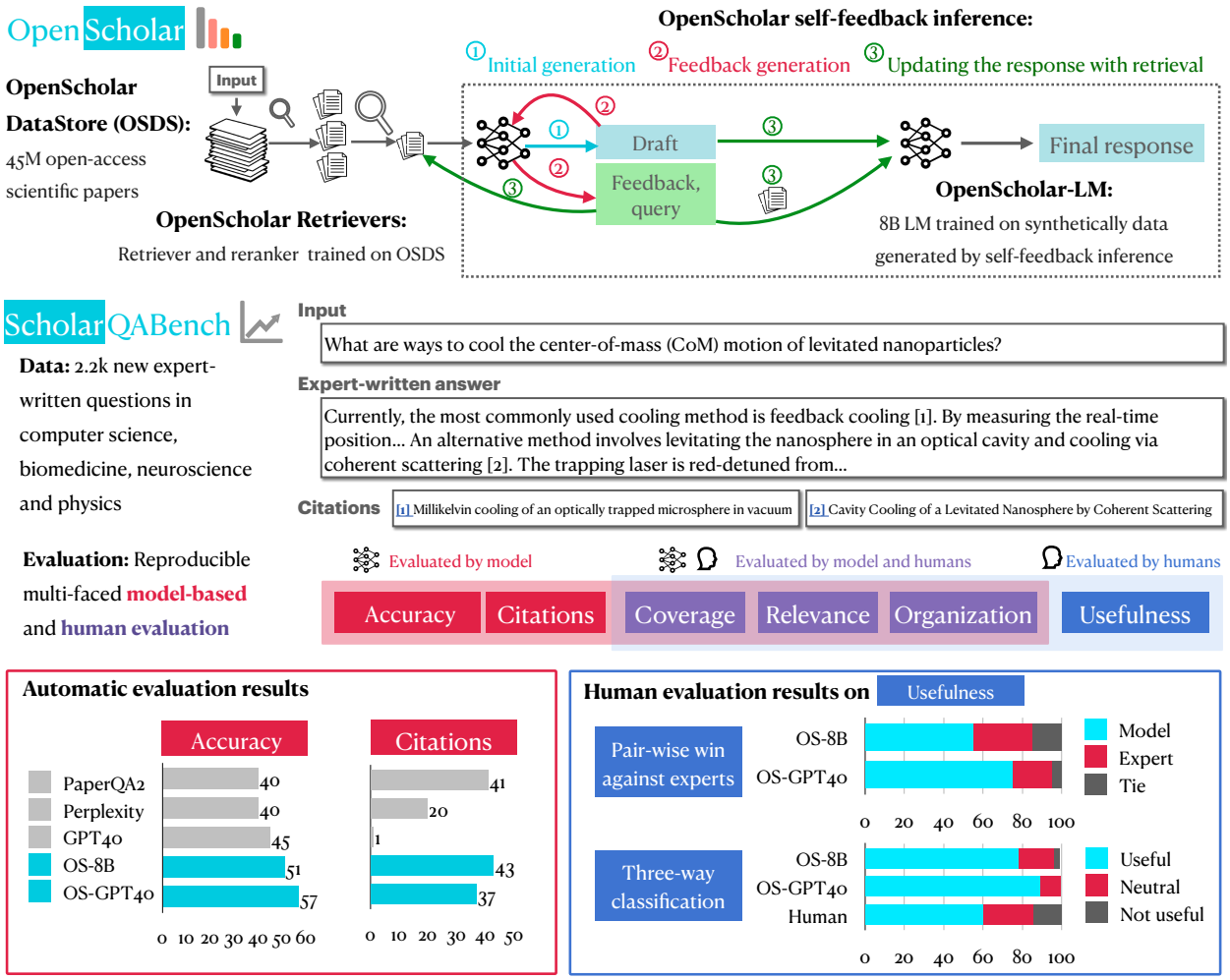


Figure 5.1: (Top) Overview of OPENSCHOLAR: OPENSCHOLAR consists of a specialized datastore, retrievers and LMs and iteratively improves responses using self-feedback inference with retrieval. **(Middle) Overview of SCHOLARQABENCH:** SCHOLARQABENCH consists of 2.2k expert-written questions across multiple scientific disciplines, and we introduce automatic and human evaluation protocols for SCHOLARQABENCH. **(Bottom) Automatic and Human Evaluation Results:** Experimental results show the effectiveness of SCHOLARQABENCH, and that OPENSCHOLAR with our trained 8B or GPT4o significantly outperforms other systems, and is preferred over experts over 50% of the time in human evaluations.

retrieval index) that are specifically suited for scientific domains. Moreover, evaluations in this area have been limited, using single-discipline and small-scale human evaluations [Agarwal et al., 2024; Zheng et al., 2024] or simplified tasks such as multiple-choice QA [Skarlinski et al., 2024].

To address these gaps, we present OPENSCHOLAR (Figure 5.1, top), a state-of-the-art retrieval-augmented LM with a specialized paper datastore and retrievers trained on scientific literature. At inference time, OPENSCHOLAR retrieves relevant passages and uses iterative self-feedback

generation to refine its own output. We further train a new, efficient 8B LM. To evaluate the effectiveness of OPENSCHOLAR, we introduce SCHOLARQABENCH (Figure 5.1, middle), a benchmark specifically designed to enable a realistic and reproducible evaluation of open-ended scientific question answering.

5.2.1 Background

Scientific LMs. Scientific LMs have spanned various domains, including biomedical [Phan et al., 2021; Yuan et al., 2022; Luo et al., 2022], medical [Singhal et al., 2023a; Gu et al., 2024; Wu et al., 2024; Singhal et al., 2023b], biomedical [Zhang et al., 2024c; Li et al., 2024a], geoscience [Feng et al., 2023], astronomy [Nguyen et al., 2023] and multidisciplinary science [Shaikh et al., 2023], with some models such as SciGLM [Zhang et al., 2024a] and UniSmart [Chi et al., 2024] that aim to cover diverse scientific domains in a single model. Recently, several works show that powerful general-purpose LLMs can also show strong capabilities in scientific tasks, such as medical question answering [AI4Science and Quantum, 2023; Singhal et al., 2023a], chemistry experimentation [Zheng et al., 2023b] and applied mechanics [Brodnik et al., 2023]. However, the language model's reliance on information memorized within its parameters leads to frequent hallucinations in its output [Li et al., 2024b].

LMs to assist scientists. Recent studies have also examined LLMs' capabilities to assist scientists in performing a range of scientific procedures, including generating novel research ideas [Baek et al., 2025; Yang et al., 2023b] and automating experimental code generation [Huang et al., 2023; Tian et al., 2024]. Our work, however, focuses specifically on benchmarking and developing methods for automating literature reviews and addressing questions related to up-to-date research—tasks that are crucial to, and particularly challenging, for scientific inquiry. Several concurrent studies have attempted to build retrieval-augmented pipelines using proprietary LLMs and external APIs (e.g., Semantic Scholar API) for scientific literature review agents [Agarwal et al., 2024; Skarlinski et al., 2024; Wang et al., 2024b]. While these studies and our research all explore the potential of retrieval-augmented LMs in automating literature synthesis, prior works often relied on proprietary, black-box systems and limited evaluations, which commonly entail small-scale human evaluation

or simplified setups such as multiple-choice QA. In contrast, our work introduces a comprehensive benchmark with automated metrics, involves user studies with experts across three scientific disciplines, and develops new methodologies to train specialized open models. OPENSCHOLAR significantly outperforms previously introduced systems and shows superiority over human experts in five domains.

5.2.2 OPENSCHOLAR: Open Retrieval-Augmented LM to Synthesizing Literature

OPENSCHOLAR (detailed in Figure 5.2) is a new retrieval-augmented LM designed to ensure reliable, high-quality responses to a range of information-seeking queries about scientific literature.

Task formulation. Given a scientific query x, the task is to identify relevant papers, synthesize their findings, and generate a response y that effectively addresses the query. This response should be accompanied by a set of citations, $\mathbf{C} = c_1, c_2, \ldots, c_K$, wherein each citation c_i corresponds to an existing scientific paper. Each c_i in $\mathbf{C}$ corresponds to specific passages from scientific literature, and should be provided as an in-line citation, linked to the relevant spans of text in y, following standard practice in scientific writing. These citations allow researchers to trace the output back to the original literature, ensuring transparency and verifiability.

Overview of OPENSCHOLAR. To ensure the retrieval of relevant papers and generate high-quality outputs, SCHOLARQABENCH consists of three key components: a datastore $\mathbf{D}$, a retriever $\mathcal{R}$, and a generator LM $\mathcal{G}$. In standard retrieval-augmented inference pipelines, the process begins with $\mathcal{R}$, which retrieves a set of passages $\mathbf{P} = \{p_1, p_2, \ldots, p_N\}$ from $\mathbf{D}$—a large-scale corpus of previously published scientific papers—based on semantic relevance to the input query x. These passages serve as context for the next step. The generator LM $\mathcal{G}$ then takes both the retrieved passages $\mathbf{P}$ and the input query x to produce the output y along with corresponding citations $\mathbf{C}$. Formally, this process can be represented as:

$$y, \mathbf{C} = \mathcal{G}(x, \mathcal{R}(x, \mathbf{D})),$$

where each c_i in $\mathbf{C}$ corresponds to a specific passage from $\mathbf{P}$. In OPENSCHOLAR (Figure 5.1), we leverage a suite of specialized components designed for scientific domains: the OPENSCHOLAR-

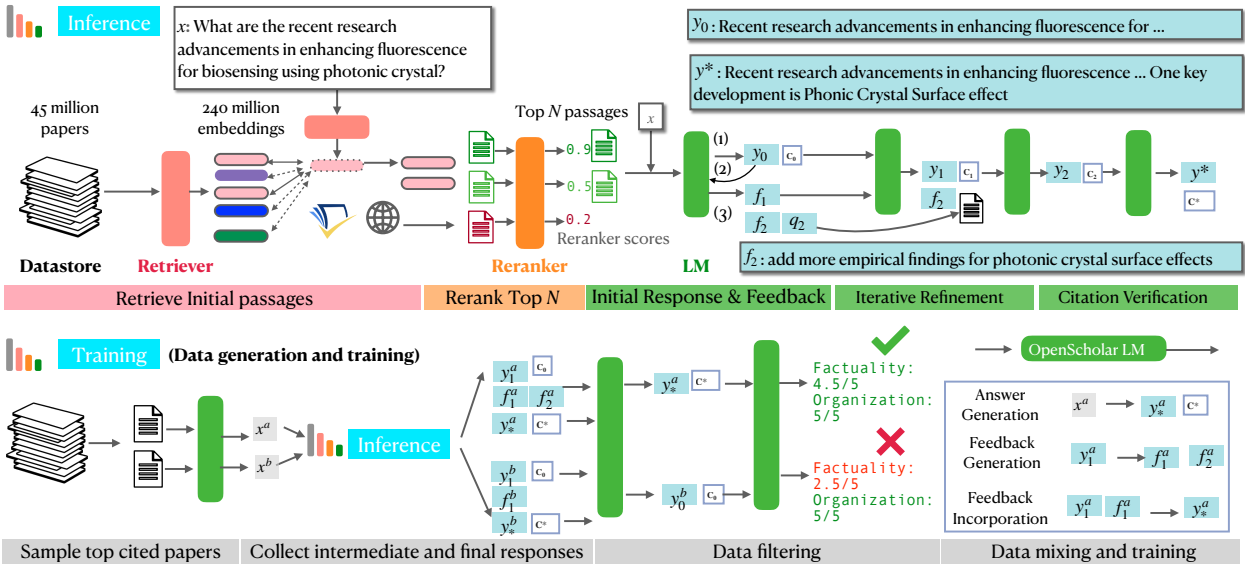


Figure 5.2: Detailed overview of OPENSCHOLAR inference (top) and training (bottom). At inference time, given an input x, OPENSCHOLAR first uses a retriever to identify relevant papers from a specialized datastore (OPENSCHOLAR-Datastore), and then uses a reranker to refine and identify the top N retrieved documents. The retrieved output is then passed to the LM, which generates both an (1) initial response y_0 and (2) self-feedback f_1. By incorporating its own feedback, the LM iteratively refines its output a pre-defined number of times. Subsequently, an LM (1) generates initial response y_0, (2) generates self-feedback on the initial output, and (3) incorporate feedback (f_i) to generates an updated response y_1. The LM repeats the process until all feedback is incorporated. To train a smaller yet competitive 8B LM, we generate high-quality training data using this inference-time pipeline followed by data filtering and mixing.

DATASTORE **D**, a OPENSCHOLAR-RETRIEVER/-RERANKER, and an LM, enabling flexible use of either off-the-shelf LMs (e.g., GPT4o) or our newly trained OPENSCHOLAR-LM. We develop self-feedback retrieval-augmented inference to improve reliability and citation accuracy.

OPENSCHOLAR-DATASTORE (OSDS) is a database of 45 million scientific papers, for which we build embeddings. We train OPENSCHOLAR-RETRIEVER and OPENSCHOLAR-RERANKER on scientific data, which passes the top N passages to the generator $\mathcal{G}$. Subsequently, we use iterative self-feedback inference with retrieval: the LM first generates an initial draft y_0 with $\mathcal{G}$, then iteratively enhances its output through retrieval-augmented self-feedback. We use this pipeline to generate high-quality training data, enabling the training of specialized LMs that produce higher-quality output and more accurate citations.

OPENSCHOLAR Retrieval Pipeline

Figure 5.2 (top left) shows our retrieval pipeline, consisting of a datastore $\mathbf{D}$, a bi-encoder retriever θ_{bi}, and a cross-encoder reranker θ_{cross}. We first select initial candidate paragraphs using $\mathbf{D}$ and θ_{bi}, as well as external APIs, and then refine and identify the top N relevant paragraphs using θ_{cross}.

Scientific paper collection and datastore construction. While prior work often uses a small subset of scientific papers, such as arXiv papers from 2023-2024 [Zheng et al., 2024], it is important to have a diverse set of papers to improve the quality and coverage of model generation [Shao et al., 2024]. To this end, we use peS2o [Soldaini et al., 2024] as our retrieval source, which consists of open-access academic papers from S2ORC [Lo et al., 2020]. We built our datastore using peS2o v3,[2] which includes 45 million papers up to October 2024.[3] Following prior work [Shao et al., 2024], we split the main text of each paper into discrete, 250-word text blocks (as determined by white space) and concatenate the paper title to each block to formulate passages in $\mathbf{D}$. Our datastore consists of 236 million passages. To our knowledge, this is the largest open-sourced datastore for scientific literature.

Initial paragraph retrieval. We retrieve passages from three sources: (1) the peS2o datastore using our trained retriever, (2) publicly available abstract from papers returned via the Semantic Scholar API [Kinney et al., 2023] based on search keywords, and (3) publicly available texts from papers retrieved through a web search engine using the original query x.

For (1), we first generate embeddings of each passage in $\mathbf{D}$ using the passage bi-encoder θ_{bi}, which processes text chunks (e.g., queries or passages) into dense vectors [Karpukhin et al., 2020] offline. Off-the-shelf retrieval models often struggle in out-of-domain scenarios [Thakur et al., 2021]. To overcome this limitation, we develop θ_{bi} by continually pre-training Contriever [Izacard et al., 2022a] on the peS2o datastore in an unsupervised fashion to improve domain-specific retrieval performance. During inference, we encode the query using θ_{bi} and retrieve the top 100 passages through a nearest neighbor search [Karpukhin et al., 2020].

[2] `https://huggingface.co/datasets/allenai/peS2o/tree/main/data/v3`.

[3] For evaluations, we use peS2o v2, which consists of papers up to January 2023, as our main benchmarks and models were constructed before the curation of peS2o v3.

For (2), we first generate keywords from the query x using a generator LM. These keywords are then used to retrieve the top 10 papers for each, as ranked by citation count, via the Semantic Scholar Search API. This approach addresses a limitation of the Semantic Scholar API, which cannot effectively handle long, question-like search queries.

For (3), we obtain the top 10 search results using the `You.com` retrieval API,[4] restricting the search to academic platforms such as ArXiv and PubMed. If the papers are open-access, we extract and add their full texts to the candidate pool; otherwise, we include only their abstracts.

Top N paragraph reranking and finalization. After the initial stage, we have gathered over 100, or even a thousand of relevant passages per query. However, passages retrieved by the bi-encoder may include unhelpful context due to deep interactions between a query and passages, as they are encoded separately [Asai et al., 2023b]. Feeding a large number of documents that might including irrelevant content to LLMs can cause efficiency and performance issues, even with state-of-the-art models [Liu et al., 2024; Xu et al., 2024b]. To overcome these challenges, we use a cross-encoder reranker [Nogueira and Cho, 2019; Xiao et al., 2023], denoted as θ_{cross}. For each candidate paragraph, the cross-encoder reranker jointly encodes and computes the relevance score between the input query and each of the passages. We then use the relevance score to rank the passages accordingly. To train θ_{cross} for scientific domains, we fine-tune a BGE-reranker [Xiao et al., 2023] using synthetic data generated by Llama 3 70B Instruct. Specifically, we randomly generate queries based on abstracts from peS2o and retrieve the top 10 passages. For each passage, Llama 3 70B Instruct then assigns a relevance score from 1 to 5, where we consider scores of 4 or 5 as positive, and scores of 1 or 2 as negative. Passages with a score of 3 are discarded. During reranking and finalization of top N passages, we also implement additional meta-filtering, which includes: (1) limiting the number of passages per paper to three passages, and (2) incorporating normalized citation counts into relevance scores predicted by the cross-encoder.

[4] `https://api.you.com/`

Inference: Iterative Generation with Retrieval-augmented Self-Feedback

In standard retrieval-augmented generation (RAG; Lewis et al. 2020b; Ram et al. 2023), a generator LM takes in the original input x and top N retrieved passages $\mathbf{P}$ and generates the output y_0. Although effective for tasks such as question answering [Mallen, Asai, et al., 2023], this one-step generation can lead to unsupported claims [Liu et al., 2023a] or incomplete output due to missing information [Asai et al., 2024c; Jiang et al., 2023]. To address these challenges, in OPENSCHOLAR, we introduce an iterative generation approach with self-feedback, which involves three steps: (1) **initial response and feedback generation** to output the initial draft y_0 and a set of feedback on y_0; (2) **iterative refinement with additional retrieval** to improve y_0 using the feedback, and (3) **citation verification**. Full details are in the Appendix.

Initial response and feedback generation. Given the input x and retrieved passages $\mathbf{P}$, the generator LM first produces an initial response y_0 with citation markers tied to the corresponding passages in $\mathbf{P}$. After generating y_0, the LM generates a set of feedback on y_0, $\mathbf{F} = f_1, f_2, \ldots, f_T$, that is aimed at improving the initial response, wherein each feedback f_t is a natural language sentence that describes potential improvements. Although the model can generate an arbitrary number of feedback (T), we set a maximum limit of three feedback sentences for efficient inference. Unlike prior work that relies on a predefined set of feedback signals [Asai et al., 2024c], our approach allows the LM to generate flexible natural language feedback on various aspects of the response, such as organization, completeness, or additional needed information. If the feedback sequence identifies missing content (e.g., "The answer only includes empirical results on QA tasks. Add results from other task types."), the LM also generates a retrieval query for additional retrieval using the pipeline in Section 5.2.2.

Iterative refinement. We then iterate over the feedback $\mathbf{F}$ to incrementally refine the output. If f_k indicates that further retrieval is needed, the query q_k is used to retrieve additional passages, which are appended to $\mathbf{P}$ before producing y_k.[5] The LM uses the previous output y_{k-1}, the retrieved

[5] Although we could iteratively regenerate the output each time feedback is provided, doing so introduces additional latency. Empirically, we found that feedback is often diverse, addressing different aspects of generation. As a result, sequentially incorporating feedback from the initial output remains effective.

passages **P**, and newly retrieved passages if any, to generate an updated output y_k. This process is repeated until all feedback has been addressed, resulting in a final output y_T by timestep T.

Citation verification. Finally, we instruct the generator LM to verify the citations in y_t. Specifically, the generator ensures that all citation-worthy statements—scientific claims requiring justification—are adequately supported by references from the retrieved passages. If any claims lack proper citations, the LM performs a post hoc insertion to ensure that citation-worthy statements are supported by passages. In our pipeline, we do not remove sentences that lack citation-worthy statements.

Training: High-quality Synthetic Data Generation with Inference Pipeline

Building powerful LMs that can effectively synthesize scientific literature is challenging due to the lack of training data for this problem. While there are some resources to train scientific LMs [Wadden et al., 2024], most tasks do not require open-retrieval settings and are single-paper tasks. As a result, most prior work in this area [Skarlinski et al., 2024] rely on proprietary LMs, which poses challenges for reproducibility and inference costs.

We leverage our inference-time pipeline to synthetically generate high-quality training data through self-feedback, so that the resulting model can get better at generating higher-quality output without going through the self-feedback process (Figure 5.2 bottom).

Question and response generations. Our data generation process involves three steps: first, selecting the top-cited papers from **D**; second, generating information-seeking queries based on their abstracts; and third, using the OPENSCHOLAR inference-time pipeline to produce high-quality responses. We generate data using Llama 3.1 70B [Dubey et al., 2024]. Specifically, we begin by sampling 1 million paper abstracts from the peS2o dataset and gather their corresponding metadata, such as publication year or citation count. We then randomly select 10,000 papers that were published after 2017, and then prompt an LM to generate literature review questions, or information-seeking queries based on each abstract that require multiple papers to answer. Next, we employ our OPENSCHOLAR pipeline to produce the final output y_T, along with intermediate

generations such as feedback $\mathbf{F}$ and initial outputs.

Data filtering. Despite its effectiveness and scalability, synthetic data may also contain issues such as hallucinations, repetitive writing, or limited instruction-following [Li et al., 2024c]. To address this, we introduce a two-step data filtering process: pairwise-filtering and rubric-filtering, leveraging the same LM used for data generation. In pair-wise filtering, we compare the quality of model outputs y_T (output at the final step) and y_0 (initial output), and retain the output that is judged to be higher quality. We find that y_0 is preferred over y_T around 20% of the time, due to over-editing or increased redundancy after multiple iteration steps. We then evaluate the quality of the chosen response on a five-point scale across two aspects: **organization** and **factual precision and citation accuracy**. A valid model output must achieve a score of 4.5 or higher in both categories, and we discard instances whose outputs do not meet this requirement. More details are provided in the Appendix.

Data mixing and training. From this synthetic pipeline, we generate three types of training data: answer generation ($x \rightarrow y$), feedback generation ($y_0 \rightarrow \mathbf{F}$), and feedback incorporation ($y_{t-1}, f_t \rightarrow y_t$). We found that incorporating both final and intermediate outputs during training helps smaller LMs learn to generate more effective feedback. We further blend this synthetic training data with existing general-domain instruction-tuning data [Ivison et al., 2023] and scientific instruction-tuning data [Wadden et al., 2024], ensuring that 50% of the training data comes from scientific domains, while the remaining 50% is sourced from general-domain data. We also generate synthetic fact verification and boolean QA data based on sampled abstract data from peS2o. For this, we sort the papers based on citation count and select the top 100,000 papers. After data mixing, we train generator LMs on our large-scale synthetic training data. We train Llama 3.1 8B Instruct on the generated training data.

5.2.3 SCHOLARQABENCH: A Realistic Literature Review Evaluation Benchmark

We introduce SCHOLARQABENCH, a benchmark that supports diverse and realistic scientific literature synthesis tasks. SCHOLARQABENCH is designed to evaluate model capabilities in automating

Dataset	Task Format	Discipline	Size	Evaluation	Multi-paper
SciFact (Wadden et al. 2020)	Claim → Label (True or False)	Biomedicine	208	Corr, Cite	
PubMed QA (Jin et al. 2019)	Question → Answer (Yes, No)	Biomedicine	843	Corr, Cite	
QASA (Lee et al. 2023)	Question → Answer (Long-form)	Computer Science	1,375	Corr, Cite	
SCHOLAR-CS	Question → Answer† (Long-form)	Computer Science	100	Corr, Cite	✓
SCHOLAR-BIO	Question → Answer* (Long-form)	Biomedicine	1,451	Cite	✓
SCHOLAR-NEURO	Question → Answer* (Long-form)	Neuroscience	1,308	Cite	✓
SCHOLAR-MULTI	Question → Answer (Long-form)	Computer Science, Physics, Biomedicine	108	Cite, LLM, Expert	✓

Table 5.1: Overview of SCHOLARQABENCH. The top three rows show single-paper datasets adopted from prior datasets. The bottom four rows are new datasets, which we constructed by recruiting Ph.D.-level experts. Answer* indicates that the dataset comes with questions only, and answer† indicates that answer will be evaluated based on human-annotated rubrics. The evaluation columns corresponds to the multifaceted evaluations in Section 5.2.3. The "Multi-paper" column indicates whether the task requires multiple papers to answer. LLM and Expert indicate the fine-grained evaluations by evaluator LLMs (i.e., Prometheus; Kim et al. 2024) and human experts, respectively.

scientific literature review. The curation process is guided by three key factors: **Diversity of tasks**: SCHOLARQABENCH includes tasks with a range of input-output formats; **Diversity of disciplines**: Unlike previous analyses that often focus on a single discipline such as computer science, SCHOLARQABENCH spans four scientific disciplines; **Inclusion of multi-paper tasks**: Unlike prior work that focuses on understanding single, pre-selected papers, all tasks require retrieving from the entire open-access collection of full text of papers and four datasets specifically require reasoning over multiple retrieved papers.

We adopt three existing single-paper datasets, and then construct a suite of high-quality, expert annotated datasets for computer science, biomedicine, physics, and neuroscience. We also build a reliable automatic evaluation pipeline. Table 5.1 provides a list of tasks in SCHOLARQABENCH, and Figure 5.3 shows an example and an overview of the evaluation pipeline.

Curation of Single-paper Tasks

For single-paper tasks, we curate and adapt existing widely-used single-paper datasets.

SciFact. SciFact [Wadden et al., 2020] is a dataset of 1.4K expert-written scientific claims in the biomedical domain, paired with gold evidence from existing PubMed paper abstracts annotated with labels and rationales. We include validation set queries labeled as either `supports` (true) or `contradicts` (false), discarding the original gold evidence, and reformulate the task as binary open-retrieval, wherein a system needs to identify relevant papers from a large collection of papers.

PubMedQA. PubMedQA [Jin et al., 2019] has expert-annotated (yes/no/maybe) QA data on PubMed paper abstracts. Similarly to SciFact, we only keep instances with yes or no labels, and discard the original abstract passage to formulate the task as an open-retrieval setup.

QASA. QASA [Lee et al., 2023] is a single paper QA dataset that consists of question answering pairs, requiring reasoning over scientific articles in AI and ML. We evaluate the model's ability to sufficiently answer a detailed question about the target paper. While the original dataset provides three subtasks (answer selection, rationale generation and answer compositions) as well as end-to-end QA, we evaluate the models' performance based on an end-to-end QA setup.

Creation of New Multi-paper Tasks

Single-paper, closed-set tasks may provide reliable evaluations. However, they may not be reflective of realistic scenarios, in which complex, open-ended questions are asked independently from existing papers, and require multi-paper retrieval and reasoning. Few datasets [Xu et al., 2024a; Malaviya et al., 2024] explore multi-paper setups with realistic queries, and most lack a reliable evaluation pipeline or human-written references. We address this gap by curating three new long-form QA datasets, annotated by experts, for such a challenging setting. Our multi-paper tasks encompass four broad scientific disciplines.

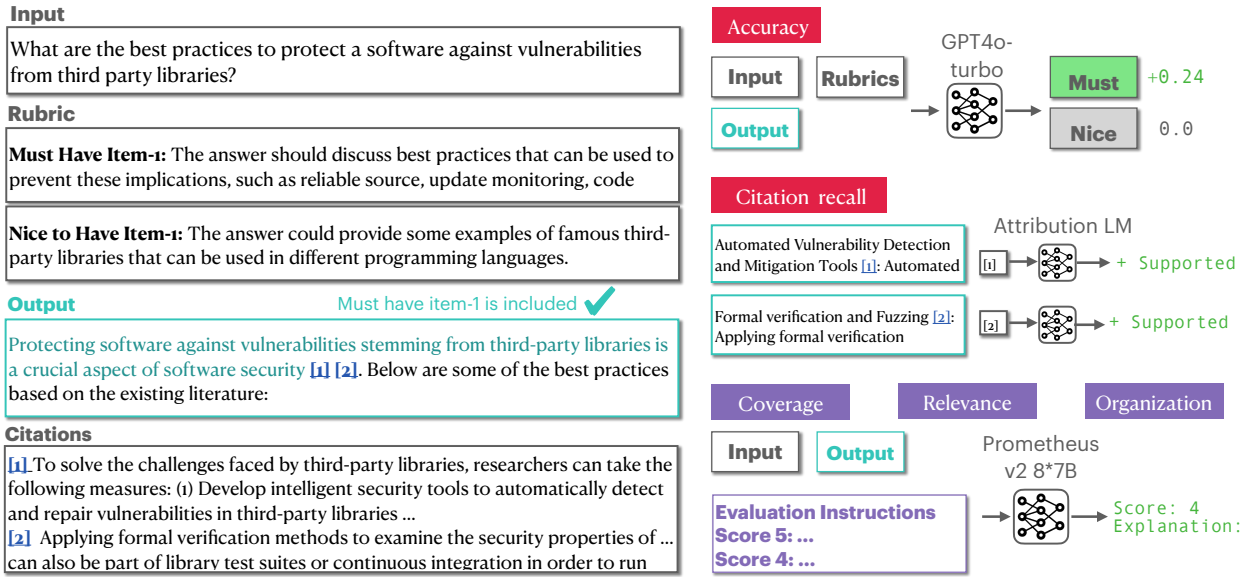


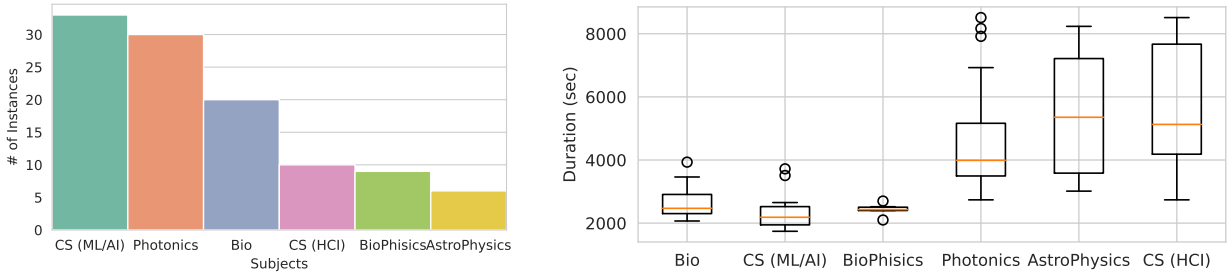
Figure 5.3: An example from SCHOLAR-CS and an overview of evaluation metrics. SCHOLAR-CS consists of 100 questions and an average of 4.4 expert-written rubrics to be satisfied. Our SCHOLARQABENCH evaluation pipeline evaluates aspects like correctness and citation accuracy.

SCHOLAR-CS. We collected 100 questions along with detailed answer rubrics for each question across various computer science disciplines by recruiting expert annotators holding Ph.D.s in the field (professors, postdoctoral researchers, and research scientists). Annotators were tasked with writing literature review questions that require multiple research papers to answer. The question topics span areas such as networks, algorithms, the Internet of Things, artificial intelligence, and human-computer interaction. Then, for each question, two other annotators searched the Web to produce a rubric listing the key ingredients for a correct answer, categorized by importance ("must-have" and "nice-to-have"), along with supporting quotes from sources for each ingredient. The annotators were instructed not to use any LLM services for this initial part of the task. After the initial web search, the annotators were shown corresponding responses from four LLM services (Claude 3.5 Sonnet, GPT4o, Perplexity Pro and an unpublished RAG prototype based on Claude 3.5) in a randomized order in case they wanted to revise their rubrics. On average, each question is annotated with 4.4 key ingredients, each supported by 4.4 quotes.

To measure agreement, we had both annotators produce rubrics for a subset of 10 randomly sampled questions. We then compute the scores for responses from the four LLM services the

	Length	Inf. time (min)	# of Citations	Prec.	Rec.	Org.	Cov.	Rel.
Human	289.5	56.8	6.0	44.4	41.5	–	–	–
OS-GPT4o	716.3	1.5	9.8	38.0	37.1	4.63	4.50	4.23
OS-8B	578.6	0.9	21.6	42.5	43.2	3.92	4.44	4.02
GPT4o	281.3	0.5	3.8	0.8	0.6	4.06	3.94	4.21
PaperQA2	166.9	0.7	3.14	53.1	42.2	3.67	3.46	4.20

Table 5.2: Human-written answer stats. Models tend to generate longer responses, citing more papers than humans. For reference, we run GPT4o without retrieval on the human evaluation queries.



(a) Subject distributions of expert-annotated answers.

(b) Average per-instance annotation time for each subject (seconds).

Figure 5.4: Distributions and annotation durations of human-written answers in SCHOLARQABENCH-Multi. (a) shows the distribution of instances per subject, and (b) shows the average annotation time per instance per subject.

annotators were exposed to using our automated approach, once for each set of annotator rubrics. Finally, we calculate Pearson's correlation coefficient among the scores for each question and compute the average. Given the subjectivity of rubric annotation, we assess agreement both with and without the general criterion included in the scores, resultin in values of 79.3 and 59.5, respectively. Figure 5.3 shows an example.

SCHOLAR-BIO, SCHOLAR-NEURO. We further collected 2,759 expert-written literature review questions in biomedicine and neuroscience, recruiting six experts who have a Ph.D. in relevant areas and are currently research scientists and engineers. The annotators were asked to choose papers from their area of expertise, and generate complex scientific questions that biomedical scientists might reasonably ask about the scientific literature based upon their parsing of those papers. We collected questions from different areas, such as bioimaging, genetics, microbiology,

and neuromodulation for each. Due to the cost of annotation, we focused solely on curating the questions.

SCHOLAR-MULTI. Lastly, we collected 108 literature review questions and expert-written answers with citations in four domains: computer science (AI/ML, HCI), Biomedicine (Bioimaging, Genetics), Physics (Astrophysics, Photonics, Bio Physics). All annotations are conducted by Ph.D. students or post-Ph.D. scientists, who have more than three years of research experience in the corresponding areas and have multiple first-author publications. We asked them to come up with questions that are related to most recent literature, and to compose answers to the questions using relevant papers that they found via search. Our annotators were instructed not to use any LLM-based systems such as ChatGPT, and told to only use general search (e.g., Google Search) or paper search systems (e.g., Semantic Scholar). Statistics of collected questions and answers are available in Table 5.2 and the distribution of subjects are in Figure 5.4a, along with the average annotation time per subject (Figure 5.4b). On average, each annotator spent 56 minutes per instance.

Metrics and Evaluation Protocols

We developed a multifaceted automatic evaluation pipeline to facilitate reproducible and efficient evaluations, complementing expert assessments. An overview of our evaluations is in Figure 5.3.

Correctness (Corr). Correctness evaluates the degree of overlap or matching of a model-generated answer and human-annotated reference. This metric is only applied to tasks with human-annotated reference answers. For short-form generation tasks given a fixed set of answer classes, namely SciFact and PubMedQA, we use accuracy as the correctness metric. For QASA, we use ROUGE-L as an evaluation metric, following Lee et al. [2023]. For SCHOLAR-CS, we develop a new long-form evaluation pipeline, which employs expert-annotated rubrics. Each rubric has two criteria: general (accounting for 40% of the score) and annotation-driven (60%). General criteria cover the evaluation of length, expertise, citations, and excerpts, while annotation-driven criteria involve assessing the presence of specific key ingredients identified by annotators. GPT4o-turbo assigns scores for each criterion, and a weighted sum is computed to obtain a final score.

Citation accuracy (Cite). Evaluating long-form responses to literature review questions requires citation accuracy: LMs should correctly attribute relevant evidence for all citation-worthy statements. In SCHOLARQABENCH, all systems generate outputs with reference numbers (e.g., `[1]`, `[2]`) linked to passages provided during inference. Following prior work [Gao et al., 2023a], we measure citation precision and recall. Specifically, we check if each citation-worthy statement has appropriate citations and if the citations support the statement (**Citation Recall**, Cite-r). For each citation, we then verify its relevance and necessity—specifically, whether the citation supports the statement and if its removal impacts the integrity of remaining citations (**Citation Precision**, Cite-p). Finally, we compute **Citation F1** (Cite-F1), and use it as a primarily metric for citation accuracy. Citation accuracy does not require gold reference answers or rubrics.

Content quality and organization (LLM, Expert). We further define key aspects to evaluate the generated answers beyond Corr or Cite alone. Specifically, we assess **Relevance** (Rel) to the question, **Coverage** (Cov) of topics (e.g., diversity of discussed papers) and depth (e.g., sufficiency of details), and **Organization and Writing Flow** (Org). These aspects are challenging to capture with standard metrics. Since LMs can effectively follow detailed evaluation rubrics [Zheng et al., 2023a; Kim et al., 2024], we use Prometheus v2 [Kim et al., 2024] to assign five-scale scores based on defined rubrics and use the same schema for human evaluations. For human evaluation (Expert), we further evaluate **Overall Usefulness** (Use). As prior studies show that LLM is less reliable when gold reference answers are not available [Kim et al., 2025], this evaluation is only applied to a task with human-annotated reference answer, namely SCHOLAR-MULTI. We analyzed the agreement between human and model assessments on fine-grained aspects and found that the model's evaluations often align with human rankings, showing higher correlation especially in organization and coverage.

5.2.4 Experiments and Results

Experimental Details

Models. First, we evaluate both open-weight and proprietary LMs, including Llama 3.1 (8B, 70B) and GPT4o (`gpt-4o-2024-05-13`). In this setup, each LM generates an answer independently,

without external retrieval, and provides a list of referenced paper titles. For evaluation, we verify whether the generated paper titles exist. If they do, we retrieve their corresponding abstracts to use as citations. For multi-paper tasks, we further evaluate other proprietary systems: Perplexity Pro,[6] and PaperQA2 [Skarlinski et al., 2024], a concurrent literature review agent system that uses GPT4o for reranking, summarization, and answer generation.[7] Then, we evaluate models using our OPENSCHOLAR-DATASTORE (+OSDS), where we retrieve the top N passages, and concatenate and feed them together with the original input. Lastly, we evaluate our proposed OPENSCHOLAR, leveraging our custom inference-time pipeline using trained 8B model models (**OS-8B**), as well as Llama 3.1 70B and GPT4o (**OS-70B**, **OS-GPT4o**).

Details of OPENSCHOLAR. We use peS2o v2 as **D**, our default datastore. We analyze the effect of different datastores. For θ_{bi} and θ_{cross} in OPENSCHOLAR, we use our trained bi-encoder and cross-encoder models, which consist of 110 million and 340 million parameters, respectively. We set the maximum number of papers from web search and Semantic Scholar to 10. For the generator LMs, we set the temperature to 0.7 and limit the maximum token count to 3,000 for response generation and 1,000 for feedback generation, and use the `vllm` package for faster inference. We train Llama 3.1 8B for two epochs on 130k training instances for two epochs. For all models, we set the number of passages input into the generator LM to five for single-paper tasks and ten for multi-paper tasks. No few-shot demonstrations are provided, except for SciFact and PubMed, for which we include one-shot demonstrations.

Results

Table 5.3 show scores for multiple aspects of the main baselines. In summary, OPENSCHOLAR achieves state-of-the performance, significantly outperforming GPT4o and their standard RAG version, as well as specialized literature review systems by a large margin.

[6] `https://www.perplexity.ai/`. We used the paid subscription version for the experiments. Note that Perplexity Search does not have an API, so we use the `selenium` toolkit and save their final prediction results from the interface. Due to this, we could not retrieve their citation information.

[7] We use PaperQA2's official codebase. As PaperQA2 does not release their retrieval corpus and requires downloading PDFs offline, we downloaded PDF files of papers suggested by our retrieval pipelines and the Semantic Scholar API. Unlike PaperQA2, we do not have access to private or license-protected papers, which may limit the effectiveness of our replication to some extent.

	Single-paper performance						Multi-paper performance						Cost
	Pub		Sci		QASA		CS		Multi		Bio	Neu	CS
Model	Corr	Cite	Corr	Cite	Corr	Cite	Corr	Cite	LLM	Cite	Cite	Cite	USD / q
Llama3-8B	61.5	0.0	66.8	0.0	14.3	0.0	41.9	0.0	3.79	0.0	0.0	0.0	0.0001
+OSDS	75.2	63.9	75.5	36.2	18.6	47.2	46.7	26.1	**4.22**	25.3	38.0	36.8	0.0001
OS-8B	**76.4**	**68.9**	**76.0**	**43.6**	**23.0**	**56.3**	**51.1**	**47.9**	4.12	**42.8**	**50.8**	**56.8**	0.003
Llama3-70B	69.5	0.0	76.9	0.0	13.7	0.0	44.9	0.0	3.82	0.0	0.0	0.0	0.0004
+OSDS	77.4	71.1	78.2	42.5	22.7	63.6	48.5	24.5	4.24	41.4	53.8	58.1	0.0004
OS-70B	**79.6**	**74.0**	**82.1**	**47.5**	**23.4**	**64.2**	**52.5**	**45.9**	4.03	**54.7**	**55.9**	**63.1**	0.01
GPT4o	65.8	0.0	77.8	0.0	**21.2**	0.0	45.0	0.1	4.01	0.7	0.2	0.1	0.006
+OSDS	**75.1**	73.7	79.3	47.9	18.3	53.6	52.4	31.1	4.03	31.5	36.3	21.9	0.01
OS-GPT4o	74.8	**77.1**	**81.3**	**56.5**	18.7	**60.4**	**57.7**	**39.5**	**4.51**	**37.5**	**51.5**	**43.5**	0.05
PaperQA2	–	–	–	–	–	–	45.6	48.0	3.82	47.2	56.7	56.0	0.3∼2.3
Perplexity	–	–	–	–	–	–	40.0	–	4.15	–	–	–	0.002**

Table 5.3: Results of SCHOLARQABENCH. CS, Multi, Bio and Neu indicate SCHOLAR-CS, SCHOLAR-MULTI, SCHOLAR-BIO and SCHOLAR-NEURO, respectively. Corr indicates correctness metrics (accuracy for PubMedQA and SciFact, ROUGE-L for QASA and Overall scores for SCHOLAR-CS) and Cite indicates citation F1. LLM indicates the average score of Org (organization), Rel (relevance), Cov (coverage) as predicted by Prometheus [Kim et al., 2024]. *PaperQA2 is based on GPT4o, and its pricing is dependent on the number of PDF files used during inference. For the 8B and 70B model costs, while evaluations were conducted on our local machines, we estimated costs based on Together.ai pricing. **We used Perplexity Pro (which requires a monthly subscription at $20 USD) and divided this cost by 9,000, which is the maximum number of queries allowed under the Pro subscription. Since the Perplexity UI does not provide snippets for each citation, we were unable to evaluate its citation accuracy.

Single-paper tasks. On single-paper tasks, OPENSCHOLAR consistently outperforms other models. OS-8B and OS-70B outperforms Llama 3.1 8B and 70B with and without retrieval augmentation in terms of final Corr and Cite in Table 5.3. OS-70B even matches or outperforms GPT4o on PubMedQA and QASA.

Multi-paper tasks. OPENSCHOLAR-8B, 70B, and GPT4o (OS-8B, OS-70B and OS-GPT4o) demonstrate strong performance in multi-paper tasks. Specifically, OS-GPT4o provides a 12.7 point improvement over GPT4o alone in SCHOLAR-CS Corr and a 5.3 improvement over standard RAG. When combined with trained OS-8B, OPENSCHOLAR significantly outperforms the pipeline that uses off-the-shelf Llama 3.1 8B, showcasing the benefits of domain-specific training. Furthermore, this OPENSCHOLAR-8B outperforms proprietary systems such as GPT4o, Perplexity Pro, or PaperQA2, which uses GPT4o models for passage reranking, summarization and answer generation,

Model	Computer Science			Biomedicine		
	Total #	# of Hallucinated (↓)	Ratio (↓)	Total #	# of Hallucinated (↓)	Ratio (↓)
OS-8B	9.65	0.0	0.0	6.25	0.0	0.0
Llama 3.1 8B	5.20	4.79	92.1%	5.58	5.46	97.6%
Llama 3.1 70B	6.14	4.78	78.1%	6.98	6.74	96.6%
GPT4o	5.74	4.52	78.7%	5.24	4.97	94.8%

Table 5.4: Statistics of hallucinated papers in computer science and biomedicine domains. Our analysis revealed a significant number of non-existent cited papers in predictions made by LLMs without retrieval, which is a problem not observed in OPENSCHOLAR.

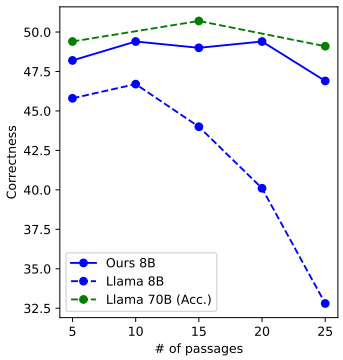
by a substantial margin. Notably, by leveraging efficient retrieval pipelines with lightweight bi-encoders, cross-encoders, and in-house models, OPENSCHOLAR-8B and OPENSCHOLAR-GPT4o achieve significantly lower costs—orders of magnitude cheaper than PaperQA2—while maintaining high performance.

Limitations of parametric LMs. On both single-paper and multi-paper tasks, we observe that non-retrieval augmented baselines struggle and retrieval is almost always conducive to achieving better performance, and models without any retrieval often struggle to generate correct citations and show limited coverage on multi-paper tasks. As shown in Table 5.4, the proportion of cited papers that actually exist is strikingly low. In particular, while models such as GPT4o and Llama 3.1 can generate plausible reference lists, we find that 78-98% of the cited papers are fabricated, with the issue being most severe in the biomedical domains. Even when citations refer to real papers, the majority of them are not substantiated by the corresponding abstracts, resulting in near-zero citation accuracy.

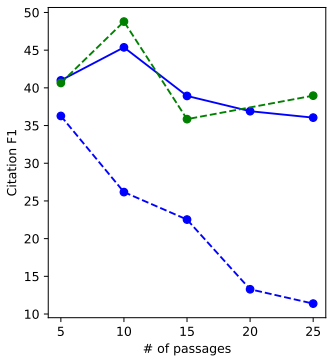
We also observe that such models also generate responses with limited coverage. On SCHOLAR-Multi, non-retrieval models (Llama 3.1 8B, 70B, and GPT4o) consistently exhibit significantly lower average scores compared to retrieval-augmented models. This discrepancy is primarily driven by substantially lower Cov scores; for instance, Llama 3.1 8B achieves a Cov score of 3.45, while Llama 3.1 8B + OSDS (a standard RAG baseline) improves the Cov score to 4.01. These results suggest that relying on models' parametric knowledge alone is particularly difficult in scientific domains, especially for smaller LMs.

	SCHOLAR-CS	
	Corr	Cite
OS-8B	51.3	47.9
- training	49.4	42.3
- reranking	49.6	28.2
- feedback	51.1	50.2
- attribution	49.3	44.0
OS-GPT4O	57.7	39.5
- reranking	52.4	22.9
- feedback	55.1	31.0
- attribution	55.6	30.6

(a) Ablation of different components of OPENSCHOLAR.



(b) Top N Ablations (Corr) on SCHOLAR-CS.



(c) Top N Ablations (Cite-F1) on SCHOLAR-CS.

Figure 5.5: Analysis on OPENSCHOLAR: (a) **Ablation studies** for key components of OPENSCHOLAR training and inference based on different underlying LMs. (b) **Top N docs:** Analysis of the effect of varying the number of context chunks for final downstream tasks. We evaluate final model performance based on citation accuracy and correctness on multi-doc QA tasks, using OPENSCHOLAR 8B and Llama 3.1 8B.

Analysis

Ablation studies. We conduct ablations to assess the effectiveness of individual components of OPENSCHOLAR (inference and training). Specifically, we remove each of the inference-time procedures: reranking, feedback and attribution. For OS-8B, we also ablate the training, where we use off-the-shelf Llama 3.1 8B Instruct without any further training.

As shown in Figure 5.5 (a), removing these components significantly impacts both the overall correctness and citation accuracy of the model outputs. Notably, removing the reranker led to substantial performance drops on both models. The pronounced decline in performance after removing feedback loops in GPT4o suggests that more powerful models greatly benefit from a self-feedback cycle, consistent with Madaan et al. [2023], while the performance drop is limited in our trained 8B. Additionally, the removal of post-hoc attribution assessments negatively affected both citation accuracy and final output correctness, highlighting the importance of ensuring that models verify their outputs. The significant performance gap between trained versus vanilla OS-8B suggests that further training on high-quality, domain-specific data is key to building efficient, task-specialized LMs. In the next analysis, we demonstrate that training has a significant impact on an LM's ability to effectively utilize more context, while maintaining citation accuracy.

Figure 5.6: Publication year distribution of the top 20 retrieved papers for SCHOLARQA-CS. By updating the datastore from peS2o v2 to the more recent peS2o v3, which includes papers until October 2024, our dense retriever can successfully retrieve more recent papers.

Number of context passages. We analyzed how varying the number of context passages (top N) impacts model performance. Specifically, we experimented with Standard RAG and OPENSCHOLAR using our trained 8B model and Llama 3.1 8B, and evaluated both generation accuracy and citation accuracy on SCHOLAR-CS. Figures 5.5 (b)(c) show the results. Although the Llama 3.1 models are trained to handle and accept a context length of up to 128K tokens, we found that its performance deteriorates after a certain context size. While increasing the top N context window from 5 to 10 does improve the model's correctness score, expanding any further actually deteriorates both correctness and citation accuracy. This suggests that even though LMs can process large numbers of passages, they may struggle to effectively use them without specialized training, particularly for smaller models.

In contrast, our trained 8B model maintains strong performance for up to $N = 20$ passages. We also found larger models such as Llama 3.1 70B to be more robust to increased context length. In terms of citation accuracy, as Figure 5.5 (c) shows, Llama 3.1 8B observes a quick decline with its citation F1 dropping to as low as 10, while our 8B LM and Llama 70B both maintain a citation F1 of around 40, albeit with some performance decline.

Effectiveness of knowledge updates via datastores. In Chapter 3, we demonstrate that Retrieval-Augmented LMs can flexibly incorporate new knowledge at inference time by simply swapping out datastores, without requiring continual training on newly available data. This capability is

especially crucial in rapidly evolving domains such as science.

To analyze this, we examine how the distribution of retrieved documents shifts when we replace datastores. Figure 5.6 compares the top retrieved papers using our retrieval systems across two datastore versions: peS2o v2 (up to January 2023) and v3 (up to October 2024). Importantly, the retrieval models were only trained on the older v2 data. Despite this, for computer science queries that require more recent sources, the systems successfully retrieve relevant papers published in late 2023 and 2024 from the updated datastore. These results highlight the strength of Retrieval-Augmented LMs in domains where timely access to up-to-date knowledge is essential.

5.2.5 Expert Evaluation

To complement our automatic evaluations and better understand the effectiveness and limitations of OPENSCHOLAR, we conducted human evaluations. This study involved over 100 literature review questions and more than 15 participants, including Ph.D. students, research scientists, and university professors with expertise in the relevant fields. In total, we curated more than 400 fine-grained expert evaluations on human and model answers.

Human Evaluation Design

Evaluations against human experts. For human evaluations, we use 108 question-answer pairs from SCHOLAR-MULTI, written by experts. We run three models on these questions to generate answers with citations: GPT4o (without external retrieval), OPENSCHOLAR with GPT4o as the generator (OS-GPT4o), and OPENSCHOLAR with our trained 8B model (OS-8B). Expert annotators are then asked to evaluate the model-generated answers against human-written answers.

Each evaluation involves presenting a question, a model-generated answer, and a human-written answer. Expert annotators then conduct fine-grained assessments of each answer and provide pairwise preference judgments between the two. For fine-grained evaluations, we use the five-scale evaluation criteria described in Section 5.2.3 (Cov, Org, Rel), with annotators scoring both model and human answers using the same rubrics. For usefulness (Use), annotators assign scores on a scale from 1-5, which we convert into three classes: Not Useful (1-2), Neutral (3), and Useful

	Fine-grained (1-5, Avg.)			Overall Usefulness	Relative to Human (%)		
	Org	Cov	Rel	Use (%)	Win	Tie	Lose
GPT4o	4.63 **(+0.4)**	4.06 **(-0.2)**	4.50 **(-0.1)**	69.7 **(-13.9)**	31.9	13.8	**54.2**
OS-8B	3.82 **(-0.3)**	4.30 **(+0.7)**	4.00 **(-0.4)**	72.1 **(+8.7)**	**50.8**	12.3	36.9
OS-GPT4o	4.47 **(+0.8)**	4.38 **(+0.9)**	4.30 **(0.0)**	80.0 **(+22.5)**	**70.0**	6.8	23.2

Table 5.5: Human evaluation results. Fine-grained aspect evaluations are conducted on a five-point scale across four aspects using our detailed instructions and rubrics. Values in parentheses represent the relative performance difference; **(+)** indicates that the model shows higher average performance, and **(-)** indicates that the human shows higher average performance.

(4-5). We then calculate the percentage of answers that fall into the Useful category. For pairwise preference, annotators either choose one of the answers or mark a "tie" if they judge both answers to be of equal quality. Optionally, experts provide explanations on why one answer is better than the other.

Expert annotators for answer writing. Our expert annotators for question and answer writing are 12 Ph.D. students and post-doctoral researchers from research institutions across the United States, all of whom have at least three years of research experience and have published multiple papers in journals or conferences in their fields. Our annotators' expert areas span computer science (Natural Language Processing, Computer Vision, Human-Computer Interaction), physics (Astrophysics, Photonics / Optics), and biomedical (Neuroscience, Bioimaging) domains, and we assign our expert annotators to questions in their expertise. On average, we paid 35-40 USD per person.

Expert annotators for evaluations. 16 expert annotators from the three fields contributed to our evaluations, with 12 of them also participating in answer generation. All expert annotators meet the same qualifications as those who composed the answers. To minimize potential biases, we ensured that annotators did not evaluate responses to their own questions by assigning evaluation tasks to different groups of experts. Each instance was reviewed by 1 to 3 expert annotators, depending on availability. The inter-annotator agreement was 0.68 using pairwise comparison with ties, and 0.70 using a relaxed approach, wherein ties were merged. On average, each expert spent five minutes per instance on evaluation, and received compensation ranging from 25–35 USD.

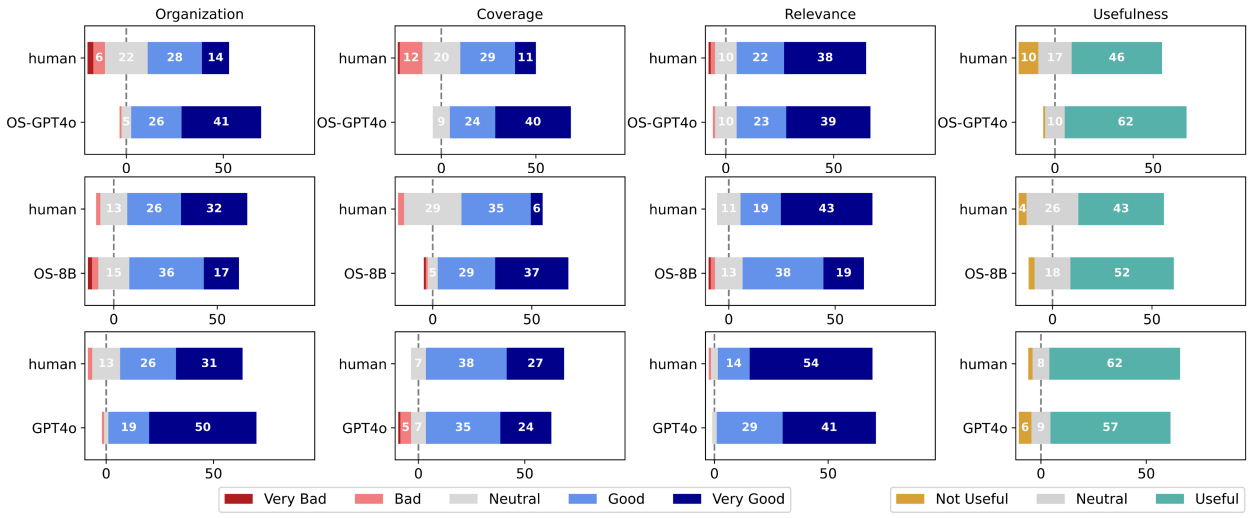


Figure 5.7: Fine-grained evaluation results. Rubric score distributions between human vs. GPT4o responses (top), human vs. OPENSCHOLAR-8B responses (middle), and human vs. OPENSCHOLAR-GPT4o responses (bottom).

Human Evaluation Results

Results of human evaluations. Table 5.5 presents the average scores for each evaluation aspect, alongside the relative win rates against human responses. Figure 5.7 illustrates the score distributions for Human, GPT4o, and OPENSCHOLAR with Llama 3 8B and GPT4o. Notably, both OS-GPT4o and our OS-8B versions outperform human answers in over 50% of cases, with their advantage primarily attributed to their ability to provide a greater breadth and depth of information (coverage; Cov). In contrast, GPT4o, which lacks retrieval capabilities, demonstrates significantly limited coverage and wins in fewer than 35% of cases, with its overall usefulness rated much lower than responses from humans and the other two models. These results highlight that even for state-of-the-art models, synthesizing and answering scientific literature review questions remains a challenging task, consistent with our findings on SCHOLARQABENCH. Overall, OPENSCHOLAR-GPT4o and OPENSCHOLAR-8B are rated as Useful in 80% and 72% of the queries, respectively.

While the performance of OPENSCHOLAR using an open 8B LM already surpasses that of human experts, the 8B model's output is judged to be less organized or fluent than the current state-of-the-art private LLM-based OPENSCHOLAR. We found that GPT4o incorporates feedback

more effectively, and tends to generate longer and more fluent outputs, leading to significantly higher organization scores compared to both the 8B OPENSCHOLAR as well as human responses.

Effects of length control on model responses. While we found model outputs are often preferred over human outputs, one potential confounding factor is the significant difference in their output length–OPENSCHOLAR-GPTo and OPENSCHOLAR-8B are 2.4 times and 2.0 times longer than human-written answers, respectively, which affects human judgment [Dubois et al., 2024]. To understand the effect of output length, we conducted a controlled experiment. For a random sample of 50 questions, we generate abbreviated responses for OPENSCHOLAR-GPT4o by prompting GPT4o to create summaries of responses that fall under 300 words. This led to OPENSCHOLAR answers that average around 333 words, which is close to the average length of human answers. We then repeat the human evaluation, considering both fine-grained and overall responses. On average, the shortened GPT4o scores 4.5 for organization, 4.6 for coverage, and 4.6 for relevance. The shortened OPENSCHOLAR-GPT4o responses are preferred or tied with expert answers in 75% of the queries. The experimental results show that the model's superior performance is not merely due to the increased length of the OPENSCHOLAR answers. Moreover, human annotators' explanations often mention that both shortened OPENSCHOLAR and human answers could be improved by incorporating more details, implying that a 300-word restriction may limit the utility of answers.

Analyses on human explanations for pairwise judgments. We randomly sampled 59 instances with free-form explanations of pairwise preferences and conducted a manual analysis to identify factors that influence overall preferences. Specifically, we examined whether the explanations referenced one or more of the following four categories: organization, relevance, coverage, and citations. While the first three categories align with the fine-grained human evaluation criteria, the citation category also considers the quality of the cited papers (e.g., whether the system includes seminal papers in the field). Our analysis revealed that 12%, 23%, 29%, and 9% of the explanations cited organization, relevance, coverage, and citation accuracy, respectively, as key factors in pairwise decisions. This suggests that coverage plays a crucial role in how humans assess the quality of responses, with annotators largely favoring model-generated answers for their greater coverage

and depth of information. However, annotators also noted that the citations provided by models could be improved, pointing out that the suggested papers were occasionally outdated or less relevant compared to more representative work.

5.2.6 Summary and Limitations

In order to further research on LM-based systems that can assist scientific progress, we introduce OPENSCHOLAR and SCHOLARQABENCH, which can help navigate the complex, ever-growing task of scientific literature review. OPENSCHOLAR, a retrieval-augmented system, leverages open-checkpoint LLMs and trained retrieval models to iteratively refine scientific output, addressing challenges such as hallucinations and citation accuracy. SCHOLARQABENCH, a novel large-scale benchmark, provides a standardized way to evaluate literature review automation across multiple scientific domains. In evaluations using SCHOLARQABENCH, OPENSCHOLAR demonstrates substantial improvements, outperforming existing systems such as GPT4o and the concurrent proprietary system PaperQA2. Our expert evaluation across three scientific disciplines reveals that SCHOLARQABENCH, when paired with fully open-checkpoint models and open-access data stores, generates answers that are more helpful than those produced by expert annotators, who required an hour per annotation. This approach also significantly increases coverage. OPENSCHOLAR using our trained 8B and GPT4o achieves a 51% and 70% win rate against human-generated answers. We open-source the OPENSCHOLAR code, data, model checkpoints, datastores, and SCHOLARQABENCH, along with a public demo, to support and accelerate future research efforts. Our public demo has engaged over 30,000 users across diverse scientific disciplines. Future work can further improve OPENSCHOLAR by integrating user feedback from this platform to enhance retrieval quality, improve citation accuracy, and optimize overall usability.

We highlight several limitations of our work in this section. It is important to note that we do not claim that LM-based systems can fully automate scientific literature synthesis. To further advance research in this area, we are releasing both SCHOLARQABENCH and OPENSCHOLAR to the community.

Limitations of SCHOLARQABENCH. There are several limitations to SCHOLARQABENCH. First, due to the cost and time required to engage expert annotators—individuals with either a Ph.D. or are currently pursuing one in relevant fields—the evaluation dataset with human-written answers is relatively small (e.g., 110 for CS-LFQA and 108 for expert-written answers). This limited dataset size may introduce statistical variance and potential biases stemming from the specific expertise of the annotators. To support future research in expanding the size and scope of SCHOLARQABENCH, we open-source our data and annotation pipelines.

Second, our automatic evaluation pipelines may not always perfectly capture the quality of generated content. For example, in SCHOLAR-CS, we combine various components (e.g., length, excerpts, rubric items) using heuristically determined weight terms. Further, we discovered that often annotators asked for specific kinds of ancillary information in their rubrics—background, elaborations and challenges—even though these aspects might not be strictly required to answer the question. In our experiments, we found that LLMs are proficient at generating the background aspects, which can give them an advantage over systems that directly answer a query but do not satisfy all the constraints of the rubrics. Moreover, future systems could potentially take advantage of the stylistic biases in the rubrics and be prompted to address more rubric elements in a way that does not improve answer quality. Although we carefully analyzed the correlation between final scores and human expert evaluations, there is still room for improvement in refining which aspects should be emphasized and how these scores should be aggregated. Additionally, our evaluations of citation precision and recall are conducted at the sentence level, but we found that some sentences without direct citations are often supported by citations in adjacent sentences. As a result, our precision and recall metrics might be overly strict, potentially underestimating the true citation accuracy. We also note that our annotations were captured at particular times (July 2024 for SCHOLAR-CS and September 2024 for SCHOLAR-MULTI), and may not reflect subsequent scientific developments. Researchers who use our evaluation benchmark should ignore papers published after these dates for a fair comparison.

Lastly, SCHOLARQABENCH primarily focuses on computer science, biomedicine, and physics, with no instances from social sciences or other engineering and scientific disciplines. We recognize

that our findings may not fully generalize to other domains, particularly those with more restricted access to paper data.

Limitations of OPENSCHOLAR. While OPENSCHOLAR demonstrates strong performance on SCHOLARQABENCH and in human evaluations, as discussed in the relevant sections, our expert annotators identified several limitations. Despite these issues, we believe OPENSCHOLAR remains a valuable tool for supporting human experts.

First, as highlighted by our expert annotators, OPENSCHOLAR does not consistently retrieve the most representative or relevant papers for certain queries. Enhancing retrieval methodologies by incorporating additional information, such as citation networks or metadata such as publication recency, could significantly improve its performance. OPENSCHOLAR outputs may contain factual inaccuracies or unsupported information, particularly in versions based on our 8B model, which has limited capacity for instruction-following and scientific knowledge. Future work can explore training that further improve OPENSCHOLAR-8B. In parallel, although competitive, OPENSCHOLAR-GPT4o relies on invoking the proprietary GPT4o via the OpenAI API, which may evolve over time, rendering exact result replication a challenge. Furthermore, note that OPENSCHOLAR does not use license-protected papers at inference time. There are ongoing discussions on how to ensure fair data use in retrieval-augmented LMs, and we leave the exploration of properly incorporating copyright-protected content to future work.

We encourage future research to address these limitations and continue improving LM-based systems for scientific literature review.

Limitations of our human evaluation process. In our human evaluations, annotators performed fine-grained assessments on aspects such as coverage, relevance, organization, and usefulness, while other factors such as citation precision and recall were separately evaluated. As a result, when assessing usefulness or pairwise preferences, annotators may have focused more on the overall quality of writing instead of carefully evaluating factual correctness or citation accuracy. We leave more detailed human analysis on citation accuracy, validity, and factuality for future work.

Our evaluations were conducted by 16 Ph.D. students and postdoctoral professionals, and

we made an effort to align their expertise with the evaluated topics. However, since research often necessitates deep domain knowledge, the annotators may not have captured more nuanced differences for questions outside their immediate areas of expertise. Additionally, these evaluations were based on 108 questions that span three scientific disciplines, meaning that findings may not fully generalize to other fields or domains.

5.3 Code: CODERAG-BENCH

While LMs excel at generating code, many programs are difficult to generate using only parametric knowledge. Despite the success of Retrieval-Augmented LMs in text-centric tasks, its potential for code generation remains under-explored. We introduce CODERAG-BENCH [Wang, Asai, et al., 2025], a holistic retrieval-augmented code generation benchmark covering tasks like basic programming, open-domain, and repository-level problems and provide reproducible evaluations on both retrieval and end-to-end code generation performance. We further create a diverse, open datastore for code retrieval, aggregating sources such as competition solutions, tutorials, library documentation, StackOverflow posts, and GitHub repositories. We conducted evaluations of diverse retrieval and LMs on this new CODERAG-BENCH to measure the effectiveness and remaining limitations of Code RAG systems.

5.3.1 Benchmark Constructions

Task Curation

In this work, we focus on Python due to its prominence in code generation benchmarks, and categorize existing Python-based coding datasets into four types: basic programming, open-domain problems, repository-level problems, and code retrieval. We select widely used datasets across categories to ensure diversity (Table 5.6).

- **Basic programming:** Includes interview-style problems requiring standard Python and algorithmic reasoning. We use HumanEval [Chen et al., 2021], MBPP [Austin et al., 2021], and LiveCodeBench [Jain et al., 2025] to reduce data contamination concerns.

Type	Dataset	# Examples	# Corpus	Ground-Truth Docs	Evaluation
Basic programming	HumanEval	164	164	program solutions	execution
	MBPP	500	500	program solutions	execution
	LiveCodeBench	400	-	-	execution
Open-domain	DS-1000	1000	34,003	docs	execution
	ODEX	945	34,003	docs, stackoverflow	execution
Repository-level	RepoEval (function)	373	237	github repository	execution
	SWE-bench-Lite	300	40,868	github repository	execution
Code retrieval	CodeSearchNet-Py	22,177	22177	CSN functions	ndcg@10

Table 5.6: Overview of the datasets in CodeRAG-Bench. CSN stands for CodeSearchNet.

- **Open-domain problems:** Require external libraries beyond the standard set. We adopt DS-1000 [Lai et al., 2023] (e.g., `pandas`) and ODEX [Wang et al., 2023d], which spans 79 libraries across diverse domains.
- **Code retrieval:** We include CodeSearchNet (Python split) [Husain et al., 2019], where the goal is to retrieve the correct function for a given NL query from a pool of real GitHub functions.

Retrieval Setups and Datastore Constructions

Retrieval setups. CODERAG-BENCH supports two retrieval setups: **canonical retrieval**—retrieves documents from only the canonical datastore (e.g., retrieve documentations for open-domain problems), and **open retrieval**—retrieves documents from any datastore. We evaluate retrieval performance using only the canonical retrieval setup, while end-to-end RAG performance is assessed under both canonical and open retrieval setups. This allows us to examine the impact of incorporating diverse documents on code generation quality.

Datastore sources. We collect retrieval documents from five commonly used resources for program developers, listed in Table 5.7.

- **Programming solutions:** We construct documents by concatenating NL descriptions with canonical solutions from HumanEval and MBPP, following VoyageAI [2024].
- **Online tutorials:** We extract HTML pages from ClueWeb22 [Overwijk et al., 2022], covering tutorials from `GeeksforGeeks`, `W3Schools`, `tutorialspoint`, and `Towards Data Science`, containing code snippets and explanations.

Resource	Corpus size	Avg. length
Programming solutions	1.1k	194.6
Online tutorials	79.4k	1502.5
Library documentation	34k	953.4
StackOverflow posts	23.5M	689.2
Github files	1.7M	5135.4

Table 5.7: Five sources to form CODERAG-BENCH datastore. Our datastores cover diverse documents related to code generations and in total contains more than 25 Million documents.

- **Library documentation:** We include official Python library docs from `devdocs.io` [Zhou et al., 2023], which are especially useful for solving open-domain and repo-level problems involving external libraries.
- **StackOverflow posts:** From the RedPajama-1T `stackexchange` split [Computer, 2023], we collect SO posts containing questions, code answers, and explanations, treating each as a retrievable document.
- **GitHub repositories:** We gather high-quality repositories from the RedPajama-1T `github` split [Computer, 2023], enabling LMs to retrieve files from external repositories as context for code generation.

Evaluation Pipelines and Metrics

For retrieval, we evaluate NDCG, Precision and Recall [Thakur et al., 2021] and use NDCG@10 percentage as our primary metric, following prior work [Izacard et al., 2022a]. For code generation, we adopt the pass@k metric [Chen et al., 2021] to measure the execution correctness of programs. We evaluate the final RAG performance both in canonical and open retrieval setups.

5.3.2 Experiments and Results

Based on CODERAG-BENCH, we conduct large-scale evaluations of 10 retrievers and 10 LMs and systematically analyze when retrieval can benefit code generation models and identify remaining challenges.

Experimental Settings

Baseline retrieval models. We adopt 10 top-performing retrievers from three categories: sparse, dense, and proprietary APIs. For sparse retrievers, we use BM25 [Robertson and Zaragoza, 2009], known for its robustness in domain adaptation [Thakur et al., 2021]. Dense retrievers include BGE-base/large [Xiao et al., 2023], GIST-base/large [Solatorio, 2024], and SFR-Embedding-Mistral [Meng et al., 2024], all top-ranked on the MTEB leaderboard [Muennighoff et al., 2023]. We also include open code embedding models, Codesage-small [Zhang et al., 2024b] and Jina-v2-code [Günther et al., 2023], which are specifically trained for code retrieval. We also include two proprietary APIs: `openai-text-embedding-small-03`, selected for its cost-effectiveness, and `voyage-code-2` [VoyageAI, 2024] optimized for code retrieval. Finally, we apply reranking with BGE-reranker-base [Xiao et al., 2023] on top-100 `openai` results before generation.

Baseline generation models. We adopt both code-specific LMs and strong general text-oriented LMs. For code-specific LMs, we use StarCoder2 [Lozhkov et al., 2024], CodeGemma [Team, 2024], CodeLlama [Roziere et al., 2023], and DeepSeekCoder [Guo et al., 2024] in various sizes. For general text LMs, we include three top-performing models: Llama3 [Grattafiori et al., 2024b], Command-R [CohereAI, 2024] specially optimized for RAG, and proprietary GPT models `gpt-3.5-turbo-0125` and `gpt-4o`. We use the instruct version of all generation models if available, since they often perform better than the base versions.

Results

Tables 5.8 and 5.9 present RACG performance under canonical and open-retrieval setups, respectively. For clarity, we show a subset of model results here; please refer to Wang, Asai, et al. [2025] for the full results.

Results on the canonical retrieval setup. In the canonical setup (Table 5.8), StarCoder2-7B benefits significantly from retrieval, likely due to its limited memorization capacity [Mallen, Asai, et al., 2023]. Retrieving from small, curated sources (e.g., solutions for HumanEval, MBPP, and open-

Method	Basic Programming			Open-Domain			Repo-Level	
	HumanEval	MBPP	LCB	DS-1000	ODEX	ODEX-hard	RepoEval	SWE-bench
w/ StarCoder2-7B								
None	31.7	2.4	1.5	29.2	14.6	10.3	26.5	0.0
BM25	**43.9**	51.8	1.0	**36.7**	14.1	13.8	36.7	0.0
GIST-large	38.7	50.4	0.5	35.9	**17.3**	13.8	40.8	0.3
Voyage, code	39.0	52.6	0.3	36.0	15.3	10.3	45.8	0.3
OpenAI, small	39.0	52.6	**1.5**	35.5	15.9	**17.2**	51.2	0.0
OpenAI, rerank	34.8	**53.4**	0.5	33.4	14.1	**17.2**	**53.9**	0.3
Gold	94.5	34.8	-	30.0	17.5	17.2	42.0	0.7
w/ GPT-3.5-turbo								*GPT-4o*
None	72.6	70.8	35.3	43.7	41.7	17.2	23.9	2.3
BM25	73.2	72.4	**35.5**	36.9	**41.0**	**24.1**	30.8	6.7
GIST-large	73.2	68.2	34.8	36.7	36.2	13.8	38.3	19.3
Voyage, code	**75.0**	66.8	34.5	**37.4**	**41.0**	20.7	43.2	15.7
OpenAI, small	73.8	68.4	35.8	36.9	40.3	17.2	48.0	21.0
OpenAI, rerank	64.0	**72.6**	33.5	**37.4**	40.5	17.2	**49.6**	**21.7**
Gold	91.5	72.6	-	42.9	40.3	24.1	39.1	30.7

Table 5.8: Performance of retrieval-augmented code generation on CODERAG-BENCH. We show performance with top retrieval and generation models. We bold-type the best RACG results. We test gpt-4o on SWE-bench to show non-trivial results than gpt-3.5-turbo. Note that we exclude code canonical answer from the retrieval corpora for basic programming tasks.

domain tasks) yields consistent improvements, especially for smaller models where external context is more impactful.

In contrast, GPT-3.5 Turbo shows limited gains from retrieval on DS-1000 and ODEX, likely because it has already memorized much of the content from these widely-used Python libraries. To test this, we evaluate performance on a harder ODEX subset featuring less popular libraries (ODEX-hard). On this subset, GPT-3.5 gains 7 points with retrieval, confirming the memorization hypothesis. Interestingly, for this harder subset, GPT-3.5 performs best when combined with BM25, while StarCoder2-7B sees larger gains from neural retrievers. On SWE-bench, most models except GPT-4o struggle due to the task's complexity. While GPT-4o improves performance with retrieval, a noticeable gap remains between oracle retrieval and RACG, highlighting the need for further advances in both retrieval and generation.

Results on the open retrieval setup. Table 5.9 shows that models like GPT-4o can effectively leverage diverse retrieval contexts in open-retrieval setup. On HumanEval, GPT-4o achieves substantial

Model	Retriever	HumanEval							ODEX						
		w/o	Prog	Tut	Docs	SO	GitHub	All	w/o	Prog	Tut	Docs	SO	GitHub	All
StarCoder	BM25	31.7	97.6	27.4	**29.3**	32.9	30.5	97.6	14.6	18.2	13.4	14.1	**11.6**	15.9	16.2
	GIST		67.1	**34.8**	26.7	32.3	**32.9**	69.1		14.6	**15.7**	**17.3**	11.4	15.5	**17.1**
	OpenAI		97.6	29.3	24.4	**36.0**	31.1	97.6		**18.7**	14.1	15.9	10.9	**16.9**	15.3
GPT-4o	OpenAI	75.6	94.5	90.2	90.9	91.5	84.8	95.1	44.6	**49.2**	44.2	47.6	40.3	39.4	39.6

Table 5.9: Open Retrieval RACG performance on CODERAG-BENCH. Comparing five retrieval sources on HumanEval and ODEX, using StarCoder2 (top) and GPT-4o (bottom).

performance gains even when retrieving from non-canonical sources such as tutorials, documentation, and StackOverflow. For ODEX, it performs best when retrieving from programming solutions, surpassing even canonical documentation sources. These results suggest that expanding datastores to include a wider range of domains and content types can significantly benefit retrieval-augmented code generation, as discussed in Shao et al. [2024]. However, as datastores grow larger, it becomes critical for models to identify the most relevant documents and generate responses conditioned on them. Notably, on ODEX, using all sources combined ("All") yields significantly worse performance than using only programming solutions or documentation.

5.4 Multilinguality: XORQA and CORA

Retrieval-Augmented LMs also offer a promising path to improving information access equity across languages. Many languages lack sufficient online content, limiting the ability of linguistic minorities to access critical information [UNESCO, 2016]. Moreover, NLP research and model development have been predominantly focused on a few high-resource languages [Yu, Asai, et al., 2022; Asai et al., 2024b]. Moreover, our prior work has also shown that in low-resource languages, many questions remain unanswerable due to the absence of documents with sufficient evidence [Asai and Choi, 2021].

To address this global challenge, we pioneered the development of cross-lingual Retrieval-Augmented LMs, where systems that retrieve and reason over multilingual context to fulfill users' information needs. Our contributions include: **(1) Dataset construction** (Section 5.4.1): We released the series of the first benchmarks for cross-lingual retrieval-augmented approaches, covering diverse languages and tasks, which have since been adopted in shared tasks; **(2) Model**

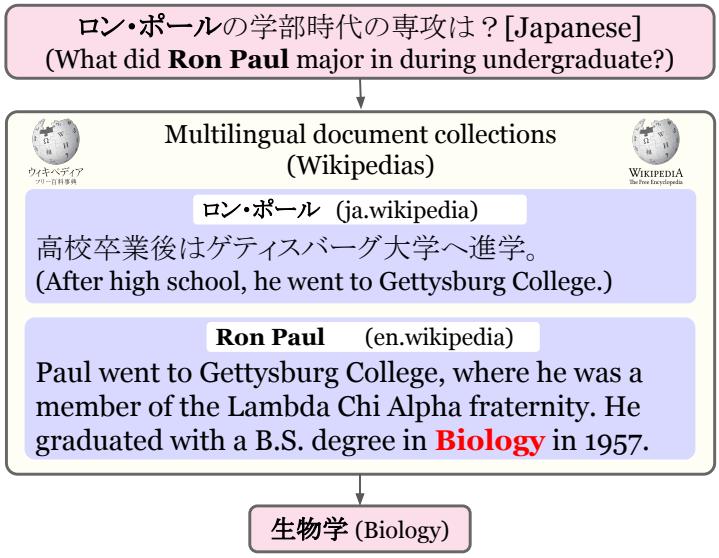


Figure 5.8: Overview of XORQA. Given a question in L_i, the model finds an answer in either English or L_i Wikipedia and returns an answer in English or L_i. L_i is one of the 7 typologically diverse languages.

development (Section 5.4.2): We introduced the first end-to-end cross-lingual Retrieval-Augmented LMs that achieve state-of-the-art performance without relying on language-specific components.

5.4.1 XORQA and Beyond: Benchmarks for Enhanced Information Access

We pioneered this area by formulating a new task, Cross-lingual Open-Retrieval Question Answering (**XORQA**), and introducing the first large-scale dataset for it, **XOR-TyDi** [Asai et al., 2021a]. This work laid the foundation for subsequent efforts, including the creation of the first open-retrieval QA datasets for African languages [Ogundepo et al., 2023], cross-lingual product QA for Amazon [Shen et al., 2023], and associated shared tasks [Asai et al., 2022b].

A New Task: Cross-lingual Open-Retrieval QA

Background. Despite rapid progress in developing models to address users' information needs, most models and datasets remain heavily focused on English [Yu, Asai, et al., 2022; Asai et al., 2024b]. Moreover, multilingual information access introduces new challenges, as the availability of information varies significantly across languages.

First, *information scarcity* limits the amount of content available in low-resource languages [Miniwatts Marketing Group, 2011]. For instance, as shown in Figure 5.8, the Japanese Wikipedia article for American politician Ron Paul omits details about his college education. This omission may reflect a lack of editorial interest from Japanese contributors in the educational backgrounds of foreign political figures. These issues make many information-seeking questions unanswerable [Asai and Choi, 2021]. Second, *information asymmetry* arises when culturally relevant information is disproportionately represented in certain languages, making it difficult for speakers of other languages to access it [Callahan and Herring, 2011]. For instance, if a question concerns a specific Japanese entity, the Japanese Wikipedia may offer a more comprehensive article than its English counterpart. Supporting retrieval in languages other than the original question is critical, as information availability across languages is uneven.

Task formulation. To address these challenges, we introduce the task of Cross-lingual Open-Retrieval Question Answering (**XORQA**), which focuses on answering multilingual questions posed by non-English speakers using a multilingual document collection. Formally, Given a question in target language L_i, a system is required to generate an answer in L_i, using a document d written in any language. Importantly, the system does not know a priori which language contains the information needed to answer the question.

Dataset Curation

Annotation pipeline. We construct a new XORQA dataset, XOR-TyDi, built upon the multilingual QA dataset TyDiQA [Clark et al., 2020], in which questions and answers are written by native speakers of 10 languages to reflect their real-world information needs. The questions in XOR-TyDi are inherited from TyDiQA and were originally unanswerable due to issues such as information scarcity and information asymmetry. XOR-TyDi addresses these limitations by enabling cross-lingual retrieval to find relevant answers from a multilingual document collection.

In particular, our annotation pipeline proceeds with four steps: 1) collection of questions from TyDiQA without a same-language answer which require cross-lingual reference to answer; 2) question translation from a target language to the pivot language of English where the missing

lang	**Question** Q_L (*Translation* Q_{pl})	**Relevant Passage** P_{pl}	**Answer** A_{pl} (*Translation* A_L)
hau	Jahohi nawa ne a kasar Malaysia? banga? (*How many states are there in Malaysia?*)	The states and federal territories of Malaysia are the principal administrative divisions of Malaysia. Malaysia is a federation of **13** states (Negeri) and 3 federal territories.	13 (*13*)
bem	Bushe Mwanawasa stadium ingisha abantu banga? (*What is the capacity of Mwanawasa Stadium?*)	The Levy Mwanawasa Stadium is a multi-purpose stadium in Ndola, Zambia. It is used mostly for football matches. The stadium has a capacity of **49,800 people**.	49,800 people (*Abantu 49800*)
wol	Man po moo niroo ag powum Softbal? (*Quel sport ressemble beaucoup au softball?*)	Ce sport est un descendant direct du **baseball** (afin de différencier les deux) mais diffère de ce dernier par différents aspects dont les cinq principaux sont les suivants.	baseball (*Bas-bal*)

Table 5.10: Examples of AfriQA questions. Table showing selected questions, relevant passages, and answers in different languages from the dataset. It also includes the human-translated versions of both questions and answers. For the primary XOR QA task, systems are expected to find the relevant passage among all Wikipedia passages, not simply the gold passage shown above.

information may exist; 3) answer retrieval in the pivot language given a set of candidate documents; 4) answer verification and translation from the pivot language back to the original language.

Statistics of XOR-TyDi. We curated a dataset of 40,000 annotated questions and answers across seven typologically diverse languages: Arabic, Bengali, Finnish, Japanese, Korean, Russian, and Telugu. Our findings show that cross-lingual retrieval significantly improves answer coverage in all languages, by as much as 40% in Bengali. As a result, we were able to find valid answers for more than 50% of the original information-seeking questions in six out of the seven languages. These results demonstrate the effectiveness of leveraging multilingual document collections to enhance answer coverage in information-seeking scenarios.

Subsequent Work

Scaling XORQA to underserved languages and real-world domains. Numerous XORQA datasets have since been developed, including our collaborative efforts aimed at supporting underserved languages and real-world domains.

In collaboration with Masakhane, we introduced **AfriQA** [Ogundepo et al., 2023], the first

open-retrieval QA dataset for African languages. AfriQA includes over 12,000 examples across 10 languages spoken throughout the African continent. As shown in Table 5.10, the majority of questions focus on entities and topics that are culturally and geographically relevant to Africa. By building this dataset from the ground up with a focus on African languages and their associated cultural contexts, we ensure greater relevance and utility for local users. Unlike the original XOR-TyDi dataset, which used English documents or documents in the target language, AfriQA uses English and French as source languages for document retrieval and annotation, reflecting the multilingual information landscape in many African regions.

It also motivated **xPQA** [Shen et al., 2023], which developed the first product-oriented cross-lingual QA dataset for shopping platforms such as Amazon, where answers often exist only in English or a small set of high-resource languages.

Shared Tasks. XORQA was later extended into the **MIA 2022 shared task** [Asai et al., 2022b], where we expanded data collection to six diverse languages, including two underrepresented ones: Tagalog and Tamil, which were added just two weeks before the submission deadline. The shared task featured two tracks: constrained and unconstrained. In the constrained track, participants could only use the official training data, while the unconstrained track allowed the use of any publicly available multilingual QA data. The top-performing system in the constrained track [Tu and Padmanabhan, 2022] leveraged mLUKE [Ri et al., 2022], a multilingual extension of LUKE [Yamada et al., 2020] introduced in Chapter 4, demonstrating the effectiveness of entity-focused contextualized representations for cross-lingual retrieval.

5.4.2 CORA: An End-to-end Cross-lingual Retrieval-Augmented LM

While cross-lingual retrieval has been studied for decades, earlier systems typically relied on translating queries or documents into a pivot language before applying retrieval or QA models [Ture and Boschee, 2016]. Such designs introduce complexity into the inference pipeline and hinder scalability to new languages. Recent advances in multilingual pre-trained LMs [Xue et al., 2021; Liu et al., 2020] have enabled strong cross-lingual performance even without explicit supervision in

the target language [Asai et al., 2024b]. Motivated by these advances, we introduced **CORA** [Asai et al., 2021b], the first end-to-end multilingual Retrieval-Augmented LM.

Method

Overview. CORA directly retrieves documents from *any* language for questions asked in *any* target language, and then generates answers in the target language conditioned on those passages. The CORA inference consists of two steps of (i) retrieving passages using Multilingual Dense Passage Retriever (mDPR) and (ii) generating an answer based on the retrieved passages using Multilingual Answer Generator (mGEN). CORA fine-tunes the core components using semi-automatically constructed supervision by aligning multilingual documents and answers via Wikipedia and Wikidata language links.

mDPR. mDPR extends Dense Passage Retriever (DPR; Karpukhin et al., 2020) to a multilingual setting. mDPR uses an iterative training approach to fine-tune a pre-trained multilingual LM (e.g., mBERT; Devlin et al., 2019) to encode passages and questions separately. Once training is done, the representations for all passages from $\mathcal{D}^{multi}$ are computed offline and stored locally. At inference, mDPR independently obtains an embedding of the question. It retrieves k passages with the k highest relevance scores to the question, regardless of languages, as described in prior Chapters.

mGEN. We use a multilingual autoregressive LM (e.g., mT5; Xue et al., 2021) to generate answers in the target language token by token, conditioned on the retrieved multilingual passages. This setup allows the model to produce answers in the target language L using evidence from documents written in various languages, without relying on machine translation, as done in prior work. In addition, the multilingual generator can generalize to unseen target languages that are not present in the XORQA training data, enabling broader applicability in low-resource settings.

Training. We automatically mine training data using external language links and train mDPR and mGEN iteratively. In particular, each iteration proceeds over two stages of updating model parameters with available training data and mining new training data *cross-lingually* by Wikipedia

language links and predictions made by the models. This approach does not require any additional human annotations or machine translation, and can be applied to many new languages with low resources.

Experiments and Results

Our experiments show that CORA advances the state of the art on two multilingual open QA datasets, XOR-TyDi [Asai et al., 2021a] and MKQA [Longpre et al., 2021], across 26 typologically diverse languages; CORA achieves gains of 23.4 and 4.7 F1 points in XOR-TyDi and MKQA respectively, where MKQA data is not used for training. Moreover, CORA achieves F1 scores of roughly 30 over 8 languages on MKQA that have no training data or even reference Wikipedia documents, outperforming the state-of-the-art approach by 5.4 F1 points. Our controlled experiments and human analyses illustrate the impact of many-to-many cross-lingual retrieval in improving multilingual open QA performance. We further observe that through cross-lingual retrieval, CORA can find answers to 20% of the multilingual questions that are valid but are originally annotated as *unanswerable* by humans due to the lack of evidence in the English knowledge sources.

Chapter 6

Conclusion and Future Work

6.1 Summary of The Thesis

In recent years, the NLP and broader AI communities have witnessed the remarkable capabilities of parametric LMs. By scaling both the amount of training data (e.g., up to 10 trillion tokens) and model size (e.g., over 1 trillion parameters), these models have achieved impressive performance across a wide range of tasks and gained widespread real-world adoption. However, fundamental limitations persist, including hallucinations, poor adaptability, and the high cost of training. Those limitations raise practical concerns as these models are deployed at scale.

This dissertation investigates the root causes of these challenges, which largely stem from the monolithic nature of current parametric LMs, and advocates for a paradigm shift toward Augmented LMs. In this new framework, LMs are designed, trained, and deployed alongside complementary modules to enhance their capabilities and address core limitations. Specifically, this dissertation focuses on the understanding, modeling, and real-world applications of Retrieval-Augmented LMs as a central approach to overcoming these issues. We have (1) conducted one of the earliest large-scale scientific studies of Retrieval-Augmented LMs to rigorously assess their effectiveness, (2) proposed new foundational approaches that go beyond simple combinations of existing models to support broader applications, and (3) applied these systems to address real-world challenges. Below, we provide a summary of the main takeaways from each chapter.

Chapter 2: An Overview of Retrieval-Augmented LMs. This chapter introduced the foundations of Retrieval-Augmented LMs, covering key definitions, historical developments, architectural and training taxonomies, and standard evaluation protocols. Drawing from our prior studies and tutorial [Asai et al., 2023a, 2024d], we highlight how retrieval augmentation has evolved from task-specific QA systems to a central paradigm in modern LM applications.

- **Foundations and preliminaries (Section 2.1):** We define Retrieval-Augmented LMs as systems that integrate a parametric LM with an external retriever and datastore, enabling inference-time access to external knowledge. We contrast this setup with monolithic LMs and review preliminaries.

- **Historical evolution (Section 2.2):** We trace the development of Retrieval-Augmented LMs from early open-domain QA systems to modern, versatile architectures. We also highlight the rise of IC RAG methods, which are often referred to simply as RAG nowadays, that enhance powerful off-the-shelf LMs (e.g., GPT-3) with retrieval at inference time, requiring no additional training.

- **Architectural taxonomy (Section 2.3.1):** We categorize Retrieval-Augmented LMs' architectures along three key axes: (1) how retrieved text is incorporated (input augmentation, intermediate fusion, output interpolation), (2) retrieval frequency (one-time, periodic, adaptive), and (3) the granularity of retrieved units (text chunks, tokens, phrases). Each design choice reflects tradeoffs in efficiency, expressiveness, and scalability.

- **Training strategies (Section 2.3.2):** We compare three major training paradigms: independent training (e.g., off-the-shelf retrievers and LMs), sequential training (e.g., retriever frozen or trained first), and joint training. We discuss challenges such as index update costs and techniques like asynchronous updates and in-batch approximations.

- **Evaluation protocols (Section 2.3.3):** We review commonly used datasets (e.g., NQ, HotpotQA, ASQA, ELI5) and evaluation metrics (e.g., EM/F1, ROUGE, Answer Match). In addition, we highlight emerging evaluation dimensions, as well as the growing use of model-based and human evaluation frameworks in recent work.

Chapter 3: Establishing the Necessity of Retrieval-Augmented LMs. We began by discussing the limitations of merely scaling parametric, monolithic LMs to address persistent challenges, and demonstrated that Retrieval-Augmented offer a more effective and efficient path forward through large-scale scientific studies.

- **Reducing hallucinations (Section 3.2)**: We showed that even very large LMs struggle to memorize long-tail knowledge that is underrepresented during pretraining. Scaling model size from 1 billion to 175 billion parameters does not resolve this issue: LMs continue to fail on questions targeting such long-tail facts [Mallen, Asai, et al., 2023]. In contrast, we demonstrated that Retrieval-Augmented LM can significantly reduce hallucinations on long-tail queries by dynamically retrieving and integrating relevant external knowledge at inference time. However, retrieved context is not always helpful as irrelevant or misleading information can cause models to fail counterfactually, even when they possess the correct parametric knowledge. To address this, we introduced the concept of adaptive retrieval augmentation—retrieving only when needed—to combine the strengths of parametric and nonparametric knowledge. Experimental results show that this approach improves both accuracy and efficiency.

- **Improving training efficiency (Section 3.3)**: We further demonstrated that Retrieval-Augmented LM can substantially reduce the growing training costs associated with parametric LMs. Specifically, we showed that (1) by offloading the burden of memorization, Retrieval-Augmented LM can achieve more compute efficient scaling, doubling performance under the same training-time GPU FLOPs [Shao et al., 2024], and (2) by swapping in updated datastores, Retrieval-Augmented LM can incorporate new knowledge efficiently without requiring continual retraining on freshly collected data [Kasai et al., 2022].

Our extensive studies on the effectiveness of Retrieval-Augmented LMs have spurred a wave of follow-up work, including deeper analysis of parametric LM limitations [Mishra et al., 2024], advances in model design [Asai et al., 2024c], and new evaluation methodologies [Hong et al., 2024].

Chapter 4: Building New Foundations for Retrieval-Augmented LMs. While naive RAG systems combining off-the-shelf LMs and retrieval modules without additional training have seen rapid adoption, they suffer from numerous limitations that lead to real-world failures.

- **Challenges in RAG systems (Section 4.2):** We first systematically discussed the challenges across all stages of such naive RAG pipelines. To build more reliable and broadly useful systems, we argue that both the LMs and retrievers must be redesigned and trained specifically for retrieval-augmented use cases.

- **Designing LMs for retrieval augmentation (Section 4.3):** We introduced new training strategies and model architectures that enable LMs to more effectively leverage retrieval [Asai et al., 2024c, 2022a; Mishra et al., 2024]. SELF-RAG [Asai et al., 2024c] enables LMs to dynamically decide when to retrieve (planning), generate responses, and self-evaluate their outputs using special reflection tokens. Through joint training and inference-time integration of these capabilities, SELF-RAG produces more adaptive RAG systems with improved performance and significantly higher citation accuracy.

- **Developing versatile and scalable Retrievers (Section 4.4):** We described our efforts to advance retrieval methods by increasing their versatility, improving robustness to complex queries, and enhancing efficiency [Asai et al., 2023b, 2020; Yamada et al., 2020; Lin et al., 2023; Shao et al., 2024; Yamada et al., 2021]. We introduced TART [Asai et al., 2023b], the first instruction-following retrieval system capable of dynamically adapting to different retrieval behaviors based on user instructions. This work involved a new task formulation, training strategies, and datasets, which contributed to the development of today's most effective retrieval systems. We also explored improvements in retrieval architectures and pre-training strategies. We developed PATHRETRIEVER [Asai et al., 2020], a neural graph-based retrieval system that adaptively retrieves documents in sequence to better handle complex search queries. In addition, we introduced **LUKE** [Yamada et al., 2020], a pre-trained masked LM tailored for entity-centric queries. To address the practical challenges of hosting large-scale datastores efficiently, we proposed **BPR** [Yamada et al., 2021], which generates binarized

embeddings to significantly reduce the storage and memory demands of neural retrieval systems.

Our contributions have seen broad adoption in the research community, forming the foundation for many leading RAG models [Yan et al., 2024; Nguyen et al., 2024] and retrieval models [Lee et al., 2025; Wei et al., 2024a; Muennighoff et al., 2025]. Furthermore, several of these methods have been applied to new applications in safety-critical domains such as biomedicine [Esteva et al., 2021; Jeong et al., 2024a] and science [Wang et al., 2025a]. Our models have been officially supported by major open-source libraries and used in production by both large technology companies and emerging startups, making substantial industry impacts.

Chapter 5: Making Real-World Impacts with Retrieval-Augmented LMs. Building on the foundations from earlier chapters, we show how Retrieval-augmented LMs can drive real-world applications. We focus on three high-impact domains: scientific research, code generation, and multilingual information access.

- **Synthesizing scientific literature (Section 5.2):** We introduced OPENSCHOLAR [Asai et al., 2024a], a fully open Retrieval-Augmented LM designed to assist scientists in conducting literature reviews. To support rigorous and reproducible evaluation, we also introduced SCHOLARQABENCH, a benchmark covering four scientific disciplines. Expert annotators curated information-seeking questions and long-form answers with detailed citations. OPENSCHOLAR introduces the largest, up-to-date corpus of scientific papers across disciplines, a self-reflective generation framework for improved inference, and new open models trained on the synthetic data based on the datastore and inference-time pipeline. Our results show that OPENSCHOLAR not only outperforms proprietary models but is also preferred over expert-written answers in blind evaluations. We released the first public demo of its kind, which has since been used by over 30,000 researchers and practitioners.

- **Enhancing code generation with heterogeneous knowledge (Section 5.3):** We presented CODERAG-BENCH [Wang, Asai, et al., 2025], which demonstrates that Retrieval-Augmented

LMs can significantly improve code generation, particularly in scenarios requiring access to other files in a repository or reasoning over long-tail knowledge (e.g., less commonly used libraries).

- **Improving multilingual information access across world languages (Section 5.4):** We described our work on addressing information access inequality across languages [Asai et al., 2021a, 2022b, 2021b; Asai and Choi, 2021; Asai et al., 2024b; Shen et al., 2023; Ogundepo et al., 2023]. Many world languages suffer from sparse web content, limiting the effectiveness of conventional LMs. We pioneered cross-lingual retrieval-augmented generation, enabling systems to retrieve and reason over documents in multiple languages. We introduced XORQA [Asai et al., 2021a], a new task and the first large-scale dataset in this area, which has since led to multiple shared tasks and follow-up benchmarks. We also developed CORA [Asai et al., 2021b], the first end-to-end multilingual Retrieval-Augmented LM, which outperforms prior systems that rely on language-specific components like translation, achieving strong performance using a single retriever and generator across all languages.

6.2 Future Directions

We envision future LMs as Augmented LMs, designed to interact with external modules for greater reliability, adaptability, and efficiency. While Retrieval-Augmented LMs and more broadly, Augmented LMs such as LM agents have garnered significant attention in the research community, many open challenges remain across different layers of the system stack. Below, we outline several key directions for future work.

Reasoning over heterogeneous knowledge at scale. While this dissertation has primarily focused on Retrieval-Augmented LM with text-only datastores, the future of AI increasingly involves reasoning over heterogeneous knowledge sources including images, audio, video, structured databases, and domain-specific information such as genomic data or scientific diagrams. These diverse modalities are central to many real-world applications but remain underexplored in the context of retrieval-augmented systems.

We conducted one of the earliest work on multimodal retrieval and reasoning, spanning tables, images, and knowledge graphs [Talmor and Berant, 2018; Asai et al., 2020]. However, such systems often relied on separate pipelines for each modality, with post-hoc integration, leading to inefficiencies and limiting generalization. Today, we are beginning to see the emergence of multimodal LMs that can natively handle diverse inputs and outputs, including our latest multilingual, multimodal LM, Pangea [Yue et al., 2025]. Yet, retrieval-augmented systems that operate across modalities, especially beyond the common pairing of text and images, remain largely uncharted. In future work, we aim to build Retrieval-Augmented LM that seamlessly integrate diverse forms of knowledge and scale to underexplored domains and modalities.

Towards module orchestrations. As tasks grow more complex, a promising future direction is orchestrating multiple modules such as multiple LMs, retrieval engines, symbolic tools, simulators, into cohesive systems that go beyond the capabilities of any single model. While retrieval remains a central component, additional modules can support reasoning, adaptation, and computation. For instance, symbolic solvers and reward models can guide optimization, simulators can assist in scientific or engineering domains, and specialized LMs or external APIs can enhance performance on domain-specific tasks while reducing cost. However, major challenges exist across all stages of the LM development pipeline: data, learning and inference.

First, there is a lack of large-scale corpora that naturally capture interactions across multiple modules. As demonstrated in SELF-RAG [Asai et al., 2024c], synthetic data can be highly effective for enabling new behaviors. However, scaling such approaches to cover a broader range of modules and strategies remains challenging due to the combinatorial explosion of possible interaction trajectories, only a small fraction of which lead to correct outcomes. Future work should focus on advancing synthetic and semi-synthetic data generation pipelines to support learning from multi-step, multi-agent interactions. Additionally, SELF-RAG generates training data offline and trains LMs on the static training data using a standard supervised learning approach. However, more dynamic training approaches, such as online reinforcement learning with external modules, could provide further benefits. These methods may enhance the model's ability to explore and discover novel trajectories. Such trajectories are often unlikely to emerge from either synthetic or

human-curated datasets.

Achieving pareto-optimal inference of such multi-module systems is also particularly important. Recent studies show that test-time scaling (e.g., generating multiple reasoning paths or iterative outputs) can improve accuracy. However, these improvements often come with increased latency and computational cost. In Shao et al. [2024], we showed that Retrieval-Augmented LMs can significantly reduce training-time compute, achieving more efficient scaling; that is, better performance given the same computational budget during training. We believe similar trends may hold for inference-time scaling as well. Our future work develops orchestration strategies that balance accuracy and efficiency, enabling real-time decision-making without compromising performance. This includes methods for selective execution, adaptive routing, and caching across modules to support fast, cost-effective inference.

New frontiers of Augmented LMs. Building on these technical foundations, our future work will explore new frontiers in applying Augmented LMs to high-impact, high-stakes domains, such as science, healthcare, law, and education, where information is abundant yet fragmented, and where accuracy, attribution, and transparency are critical. Our system, OPENSCHOLAR, has demonstrated how Retrieval-Augmented LMs can support scientific workflows by automating literature reviews while preserving transparency and incorporating expert feedback. However, real-world deployments also expose new risks, including safety concerns, legal issues such as copyright [Chen et al., 2024a], and potential overreliance on model-generated content. Looking ahead, we aim to develop both systems and rigorous evaluation frameworks in close collaboration with domain experts. By maintaining an expert-in-the-loop approach, we seek to enable responsible, trustworthy, and impactful applications of LMs in domains where precision and reliability are essential.

Bibliography

Nancy E Adams. 2015. Bloom's taxonomy of cognitive learning objectives. *Journal of the Medical Library Association*.

Shubham Agarwal, Issam H Laradji, Laurent Charlin, and Christopher Pal. 2024. Litllm: A toolkit for scientific literature review. *arXiv preprint arXiv:2402.01788*.

Microsoft Research AI4Science and Microsoft Azure Quantum. 2023. The impact of large language models on scientific discovery: a preliminary study using gpt-4.

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-domain question answering goes conversational via question rewriting. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Akari Asai and Eunsol Choi. 2021. Challenges in information-seeking QA: Unanswerable questions and paragraph retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

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

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

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'arcy, and 1 others. 2024a. Openscholar: Synthesizing scientific literature with retrieval-augmented lms. *Under Review*.

Akari Asai, Jungo Kasai, Jonathan Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021a. 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*.

Akari Asai, Sneha Kudugunta, Xinyan Yu, Terra Blevins, Hila Gonen, Machel Reid, Yulia Tsvetkov, Sebastian Ruder, and Hannaneh Hajishirzi. 2024b. BUFFET: Benchmarking large language models for few-shot cross-lingual transfer. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Akari Asai, Shayne Longpre, Jungo Kasai, Chia-Hsuan Lee, Rui Zhang, Junjie Hu, Ikuya Yamada, Jonathan H. Clark, and Eunsol Choi. 2022b. MIA 2022 shared task: Evaluating cross-lingual open-retrieval question answering for 16 diverse languages. In *Proceedings of the Workshop on Multilingual Information Access (MIA)*.

Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. 2023a. Retrieval-based language models and applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Tutorial)*.

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2023b. Task-aware retrieval with instructions. In *Findings of the Association for Computational Linguistics*.

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

Akari Asai, Xinyan Yu, Jungo Kasai, and Hanna Hajishirzi. 2021b. One question answering model

for many languages with cross-lingual dense passage retrieval. In *Proceedings of the Thirty-Fifth Annual Conference on Neural Information Processing Systems*.

Akari Asai, Zexuan Zhong, Danqi Chen, Pang Wei Koh, Luke Zettlemoyer, Hannaneh Hajishirzi, and Wen tau Yih. 2024d. Reliable, adaptable, and attributable language models with retrieval. *Preprint*, arXiv:2403.03187.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBPedia: A nucleus for a web of open data. In *Proceedings of International Semantic Web Conference*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2025. ResearchAgent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, and 1 others. 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 *Proceedings of the 2000 Neural Information Processing Systems Conference*.

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*.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*.

Bernd Bohnet, Vinh Q Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, and 1 others. 2022. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*.

Neal R. Brodnik, Samuel Carton, Caelin Muir, Satanu Ghosh, Doug Downey, McLean P. Echlin, Tresa M. Pollock, and Samantha Daly. 2023. Perspective: Large Language Models in Applied Mechanics. *Journal of Applied Mechanics*.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a" siamese" time delay neural network. *Proceedings of the 7th International Conference on Neural Information Processing Systems*.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*.

Ewa S Callahan and Susan C Herring. 2011. Cultural bias in Wikipedia content on famous persons. *JASIST*.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Tianyu Cao, Neel Bhandari, Akhila Yerukola, Akari Asai, and Maarten Sap. 2025. Out of style: Rag's fragility to linguistic variation. *arXiv preprint arXiv:2504.08231*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. RQ-RAG: Learning to refine queries for retrieval augmented generation. In *Proceedings of the First Conference on Language Modeling*.

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*.

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. How is chatgpt's behavior changing over time?

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Tong Chen, Akari Asai, Niloofar Mireshghallah, Sewon Min, James Grimmelmann, Yejin Choi, Hannaneh Hajishirzi, Luke Zettlemoyer, and Pang Wei Koh. 2024a. CopyBench: Measuring literal and non-literal reproduction of copyright-protected text in language model generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and Dong Yu. 2024b. Dense X retrieval: What retrieval granularity should we use? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

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*.

Chenglei Chi, Qiaozi Cheng, Zheng Wen, Rongzhe Lin, Chunyang Wen, Zhaowei Wang, Cuiling Gao, Jian Zhang, Xu Jiang, Jian Yin, and 1 others. 2024. Uni-SMART: Universal science multimodal analysis and research transformer. *arXiv preprint arXiv:2403.10301*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018a. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018b. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2022. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tai, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*.

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*.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *TACL*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Federico Cocchi, Nicholas Moratelli, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. 2025. Augmenting multimodal llms with self-reflective tokens for knowledge-based visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2025*.

CohereAI. 2024. Command r.

Together Computer. 2023. Redpajama: An open source recipe to reproduce llama training dataset.

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 *Proceedings of the Eleventh International Conference on Learning Representations*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. In *Proceedings of the Seventh International Conference on Learning Representations*.

Michiel de Jong, Yury Zemlyanskiy, Joshua Ainslie, Nicholas FitzGerald, Sumit Sanghai, Fei Sha, and William Cohen. 2023. FiDO: Fusion-in-decoder optimized for stronger performance and faster inference. In *Findings of the Association for Computational Linguistics: ACL 2023*.

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 *Proceedings of the Tenth International Conference on Learning Representations*.

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*.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Andrew Drozdov, Shufan Wang, Razieh Rahimi, Andrew McCallum, Hamed Zamani, and Mohit Iyyer. 2022. You can't pick your neighbors, or can you? when and how to rely on retrieval in the kNN-LM. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. In *Proceedings of the First Conference on Language Modeling*.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacafarm: a simulation framework for methods that learn from human feedback. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*.

Andre Esteva, Anuprit Kale, Romain Paulus, Kazuma Hashimoto, Wenpeng Yin, Dragomir Radev, and Richard Socher. 2021. Covid-19 information retrieval with deep-learning based semantic search, question answering, and abstractive summarization. *NPJ digital medicine*.

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

Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. 2024. Mol-instructions: A large-scale biomolecular instruction dataset for large language models. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Qiang Feng, Yuxi Li, Jintao Zou, Zhiwei Li, Zhiqiang Ding, Chao Zhang, Qinyan Zhang, Xueqi Hu, Weihao Peng, Xiangyu Meng, and 1 others. 2023. K2: A foundation language model for geoscience knowledge understanding and generation. *arXiv preprint arXiv:2306.05064*.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023a. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Shangyi Geng, Wenting Zhao, and Alexander M Rush. 2025. Great memory, shallow reasoning: Limits of kNN-LMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Sachin Goyal, Christina Baek, J Zico Kolter, and Aditi Raghunathan. 2025. Context-parametric inversion: Why instruction finetuning can worsen context reliance. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024a. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024b. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *Proceedings of the First Conference on Language Modeling*.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Xiaodan Gu, Zhen Wang, Zhengliang Shi, Hongyan Li, Xiaoye Chen, and Dehong Cheng. 2024. Me-llama: Foundation model for medical language understanding and generation. *arXiv preprint arXiv:2402.12749*.

Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, Tanguy Abel, Mohammad Kalim Akram, Susana Guzman, Georgios Mastrapas, Saba Sturua, Bo Wang, and 1 others. 2023. Jina embeddings 2: 8192-token general-purpose text embeddings for long documents. *arXiv preprint arXiv:2310.19923*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, and 1 others. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

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

Shirley Anugrah Hayati, Raphael Olivier, Pravalika Avvaru, Pengcheng Yin, Anthony Tomasic, and Graham Neubig. 2018. Retrieval-based neural code generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of the Ninth International Conference on Learning Representations*.

Xanh Ho, Jiahao Huang, Florian Boudin, and Akiko Aizawa. 2025. Llm-as-a-judge: Reassessing the performance of llms in extractive qa. *arXiv preprint arXiv:2504.11972*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Giwon Hong, Aryo Pradipta Gema, Rohit Saxena, Xiaotang Du, Ping Nie, Yu Zhao, Laura Perez-Beltrachini, Max Ryabinin, Xuanli He, Clémentine Fourrier, and 1 others. 2024. The hallucinations leaderboard–an open effort to measure hallucinations in large language models. *arXiv preprint arXiv:2404.05904*.

Chengyu Huang, Zeqiu Wu, Yushi Hu, and Wenya Wang. 2024. Training language models to generate text with citations via fine-grained rewards. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2023. Mlagentbench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*.

W. John Hutchins. 2004. The Georgetown-IBM experiment demonstrated in January 1954. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas: Technical Papers*.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, and 1 others. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*.

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

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *Proceedings of the Ninth 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*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022b. ATLAS: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

Minbyul Jeong, Jiwoong Sohn, Mujeen Sung, and Jaewoo Kang. 2024a. Improving medical reasoning through retrieval and self-reflection with retrieval-augmented large language models. *Bioinformatics*.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024b. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Pengcheng Jiang, Xueqiang Xu, Jiacheng Lin, Jinfeng Xiao, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025. s3: You don't need that much data to train a search agent via rl. *arXiv preprint arXiv:2505.14146*.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research 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*.

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*.

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*, Toronto, Canada. Association for Computational Linguistics.

Go Kamoda, Akari Asai, Ana Brassard, and Keisuke Sakaguchi. 2025. Quantifying the influence of evaluation aspects on long-form response assessment. In *Proceedings of the 31st International Conference on Computational Linguistics*.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning*.

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

Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith, Yejin Choi, and Kentaro Inui. 2022. Realtime QA: What's the answer right now? In *Proceedings of the 37th International Conference on Neural Information Processing Systems (Datasets and Benchmarks)*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019.

Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

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

Daniel Khashabi, Amos Ng, Tushar Khot, Ashish Sabharwal, Hannaneh Hajishirzi, and Chris Callison-Burch. 2021. GooAQ: Open question answering with diverse answer types. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

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*.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024. Prometheus: Inducing fine-grained evaluation capability in language models. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Seungone Kim, Juyoung Suk, Ji Yong Cho, Shayne Longpre, Chaeeun Kim, Dongkeun Yoon, Guijin Son, Yejin Cho, Sheikh Shafayat, Jinheon Baek, and 1 others. 2025. The BiGGen Bench: A principled benchmark for fine-grained evaluation of language models with language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Rodney Michael Kinney, Chloe Anastasiades, Russell Authur, Iz Beltagy, Jonathan Bragg, Alexandra Buraczynski, Isabel Cachola, Stefan Candra, Yoganand Chandrasekhar, Arman Cohan, Miles Crawford, Doug Downey, Jason Dunkelberger, Oren Etzioni, Rob Evans, Sergey Feldman, Joseph Gorney, David W. Graham, F.Q. Hu, and 29 others. 2023. The semantic scholar open data platform. *arXiv preprint arXiv:2301.10140*.

Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. 2023. Pretraining language models with human preferences. In *Proceedings of the 40th International Conference on Machine Learning*.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

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*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-tau Yih, Daniel Fried, Sida Wang, and Tao Yu. 2023. DS-1000: a natural and reliable benchmark for data science code generation. In *Proceedings of the 40th International Conference on Machine Learning*.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

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

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

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*.

Yoonjoo Lee, Kyungjae Lee, Sunghyun Park, Dasol Hwang, Jaehyeon Kim, Hong-in Lee, and Moontae Lee. 2023. QASA: advanced question answering on scientific articles. In *Proceedings of the Fortieth International Conference on Machine Learning*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. 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, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

Junfeng Li, Junjie Gao, Siru Zhang, Yiwen Wang, Xinhang Yan, Hongyan Liu, Shiping Yang, Jie Qiao, and Qian Zhan. 2024a. BioMistral: A collection of open-source pretrained large language models for biomedicine. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024b. The dawn after the dark: An empirical study on factuality hallucination in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024c. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Scott Yih. 2024. RA-DIT: Retrieval-augmented dual instruction tuning. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Frederick Liu, Siamak Shakeri, Hongkun Yu, and Jing Li. 2021. EncT5: Fine-tuning t5 encoder for non-autoregressive tasks. *arXiv preprint arXiv:2110.08426*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*.

Nelson F Liu, Tianyi Zhang, and Percy Liang. 2023a. Evaluating verifiability in generative search engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Shanshan Liu, Xin Zhang, Sheng Zhang, Hui Wang, and Weiming Zhang. 2019a. Neural machine reading comprehension: Methods and trends. *Applied Sciences*.

Xueqing Liu, Chi Wang, Yue Leng, and ChengXiang Zhai. 2018. LinkSO: A dataset for learning to retrieve similar question answer pairs on software development forums. In *Proceedings of the 4th ACM SIGSOFT International Workshop on NLP for Software Engineering*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*.

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

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Shayne Longpre, Yi Lu, and Joachim Daiber. 2021. MKQA: A linguistically diverse benchmark for multilingual open domain question answering. *Transactions of the Association for Computational Linguistics*.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, and 1 others. 2024. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The AI Scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022. QUARK: Controllable text generation with reinforced unlearning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*.

Hongyin Luo, Yung-Sung Chuang, Yuan Gong, Tianhua Zhang, Yoon Kim, Xixin Wu, Danny Fox, Helen Meng, and James Glass. 2023. Sail: Search-augmented instruction learning. *arXiv preprint arXiv:2305.15225*.

Renqian Luo, Liai Sun, Yingce Xie, Zhiting Jiang, Yangbin Gu, Kun Shi, Dejia Xiong, Sheng He, Zhen Xu, and Tao Qin. 2022. Biogpt: Generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-Refine: Iterative refinement with self feedback. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*.

Jean Maillard, Vladimir Karpukhin, Fabio Petroni, Wen-tau Yih, Barlas Oguz, Veselin Stoyanov, and Gargi Ghosh. 2021. Multi-task retrieval for knowledge-intensive tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2024. ExpertQA: Expert-curated questions and attributed answers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

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*.

Maria Marina, Nikolay Ivanov, Sergey Pletenev, Mikhail Salnikov, Daria Galimzianova, Nikita Krayko, Vasily Konovalov, Alexander Panchenko, and Viktor Moskvoretskii. 2025. Llm-independent adaptive rag: Let the question speak for itself. *arXiv preprint arXiv:2505.04253*.

Rui Meng, Ye Liu, Shafiq Rayhan Joty, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. 2024. SFR-Embedding-Mistral:enhance text retrieval with transfer learning. *Salesforce AI Research Blog*.

C Metz and K Weise. 2025. Ai is getting more powerful, but its hallucinations are getting worse. *The New York Times*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, and 1 others. 2021. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. In *NeurIPS 2020 Competition and Demonstration Track*.

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*.

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

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023a. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Sewon Min, Weijia Shi, Mike Lewis, Xilun Chen, Wen-tau Yih, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2023b. Nonparametric masked language modeling. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Miniwatts Marketing Group. 2011. Internet world stats: Usage and population statistics.

Mihran Miroyan, Tsung-Han Wu, Logan Kenneth King, Tianle Li, Anastasios N. Angelopoulos, Wei-Lin Chiang, Narges Norouzi, and Joseph E. Gonzalez. 2025. Introducing the search arena: Evaluating search-enabled ai.

Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. 2024. Fine-grained hallucination detection and editing for language models. In *Proceedings of the First Conference on Language Modeling*.

Niklas Muennighoff. 2022. SGPT: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2025. Generative representational instruction tuning. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, and 1 others. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Tuan Dung Nguyen, Yuan-Sen Ting, Ioana Ciuca, Charles O'Neill, Ze-Chang Sun, Maja Jabłońska, Sandor Kruk, Ernest Perkowski, Jack Miller, Jason Jason Jingsh Li, Josh Peek, Kartheik Iyer, Tomasz Rozanski, Pranav Khetarpal, Sharaf Zaman, David Brodrick, Sergio J. Rodriguez Mendez, Thang Bui, Alyssa Goodman, and 5 others. 2023. AstroLLaMA: Towards specialized foundation models in astronomy. In *Proceedings of the Second Workshop on Information Extraction from Scientific Publications*.

Xuan-Phi Nguyen, Shrey Pandit, Senthil Purushwalkam, Austin Xu, Hailin Chen, Yifei Ming, Zixuan Ke, Silvio Savarese, Caiming Xong, and Shafiq Joty. 2024. Sfr-rag: Towards contextually faithful llms. *arXiv preprint arXiv:2409.09916*.

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

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*.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint*.

Odunayo Ogundepo, Tajuddeen R. Gwadabe, Clara E. Rivera, Jonathan H. Clark, Sebastian Ruder, David Ifeoluwa Adelani, Bonaventure F. P. Dossou, Abdou Aziz Diop, Claytone Sikasote, Gilles Hacheme, Happy Buzaaba, Ignatius Ezeani, Rooweither Mabuya, Salomey Osei, Chris Emezue, Albert Njoroge Kahira, Shamsuddeen Hassan Muhammad, Akintunde Oladipo, Abraham Toluwase Owodunni, and 33 others. 2023. AfriQA: Cross-lingual open-retrieval question answering for African languages. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. 2024. INSTRUCTIR: A benchmark for instruction following of information retrieval models. *arXiv preprint arXiv:2402.14334*.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike,

and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*.

Arnold Overwijk, Chenyan Xiong, Xiao Liu, Cameron VandenBerg, and Jamie Callan. 2022. ClueWeb22: 10 billion web documents with visual and semantic information. *arXiv preprint arXiv:2211.15848*.

Md Rizwan Parvez, Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2021. Retrieval augmented code generation and summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

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

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*.

Long N Phan, James T Anibal, Hieu Tran, Shaurya Chanana, Erol Bahadroglu, Alec Peltekian, and Grégoire Altan-Bonnet. 2021. Scifive: a text-to-text model for biomedical literature. *arXiv preprint arXiv:2106.03598*.

Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Dmytro Okhonko, Samuel Broscheit, Gautier Izacard, Patrick Lewis, Barlas Oğuz, Edouard Grave, Wen-tau Yih, and Sebastian Riedel. 2021. The web is your oyster–knowledge-intensive nlp against a very large web corpus.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. MAUVE: Measuring the gap between neural text and human text

using divergence frontiers. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

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, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 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*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *Proceedings of the First Conference on Language Modeling*.

Ryokan Ri, Ikuya Yamada, and Yoshimasa Tsuruoka. 2022. mLUKE: The power of entity representations in multilingual pretrained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

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*.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*.

R. Rosenfeld. 2000. Two decades of statistical language modeling: where do we go from here? *Proceedings of the IEEE*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, and 1 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

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*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, and 21 others. 2022. Multitask prompted training enables zero-shot task generalization. In *Proceedings of the Tenth International Conference on Learning Representations*.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceedings of the Fifth International Conference on Learning Representations*.

Shishir G Shaikh, Jaideep Ramachandran, Varnith Nanda, Benjamin Lunt, Miltiadis Allamanis, Daman Sharma, Sebastien Bubeck, and Prateek Jain. 2023. Darwin: Data analytics and reasoning with large language models for science. *arXiv preprint arXiv:2308.13565*.

Rulin Shao, Jacqueline He, Akari Asai, Weijia Shi, Tim Dettmers, Sewon Min, Luke Zettlemoyer, and Pang Wei Koh. 2024. Scaling retrieval-based language models with a trillion-token datastore. In *Proceedings of the Thirty-Eighth Annual Conference on Neural Information Processing Systems*.

Xiaoyu Shen, Akari Asai, Bill Byrne, and Adria De Gispert. 2023. xPQA: Cross-lingual product question answering in 12 languages. In *Proceedings of the Thirty-Eighth Annual Conference on Neural Information Processing Systems (Industry Track)*.

Zejiang Shen, Kyle Lo, Lauren Yu, Nathan Dahlberg, Margo Schlanger, and Doug Downey. 2022. Multi-lexsum: Real-world summaries of civil rights lawsuits at multiple granularities. In *Proceedings of the 36th Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*.

Weijia Shi, Julian Michael, Suchin Gururangan, and Luke Zettlemoyer. 2022. Nearest neighbor zero-shot inference. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, and 1 others. 2022. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2025. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, and 1 others. 2023a. Large language models encode clinical knowledge. *Nature*.

Zeming Singhal, Charles Sutton, Adam Mottram, Owain Lavelle, Iz Beltagy, Leonardo Neves, Kyle Lo, Stephanie Hyland, Michael Wainwright, Alexander Wettig, and 1 others. 2023b. MEDITRON-70B: Scaling medical pretraining for large language models. *arXiv preprint arXiv:2311.16079*.

Michael D. Skarlinski, Sam Cox, Jon M. Laurent, James D. Braza, Michaela Hinks, Michael J. Hammerling, Manvitha Ponnapati, Samuel G. Rodriques, and Andrew D. White. 2024. Language agents achieve superhuman synthesis of scientific knowledge. *arXiv preprint arXiv:2409.13740*.

Aivin V. Solatorio. 2024. Gistembed: Guided in-sample selection of training negatives for text embedding fine-tuning.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, and 1 others. 2024. Dolma: An open corpus of three trillion tokens for language model pretraining research. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Zhiqing Sun, Xuezhi Wang, Yi Tay, Yiming Yang, and Denny Zhou. 2023. Recitation-augmented language models. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

CodeGemma Team. 2024. Codegemma: Open code models based on gemma.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Thirty-Fifth Annual Conference on Neural Information Processing Systems (Datasets and Benchmarks Track)*.

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*.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, and 1 others. 2024. SciCode: A research coding benchmark curated by scientists. In *Proceedings of the Thirty-Eighth Annual Conference on Neural Information Processing Systems (Datasets and Benchmarks Track)*.

Yonglong Tian, Andrew Luo, Xingyuan Sun, Kevin Ellis, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. 2019. Learning to infer and execute 3d shape programs. In *Proceedings of the Seventh International Conference on Learning Representations*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue:

Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*.

Zhucheng Tu and Sarguna Janani Padmanabhan. 2022. MIA 2022 shared task submission: Leveraging entity representations, dense-sparse hybrids, and fusion-in-decoder for cross-lingual question answering. In *Proceedings of the Workshop on Multilingual Information Access (MIA)*.

Ferhan Ture and Elizabeth Boschee. 2016. Learning to translate for multilingual question answering. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

UNESCO. 2016. A decade of promoting multilingualism in cyberspace through the international normative instrument: Unesco's recommendation concerning the promotion and use of multilingualism and universal access to cyberspace (2003). *Multilingualism in Cyberspace*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Thirty-First Annual Conference on Neural Information Processing Systems*.

Ellen M Voorhees and 1 others. 1999. The trec-8 question answering track report. In *Proceedings of the Eighth Text REtrieval Conference*.

VoyageAI. 2024. voyage-code-2: Elevate your code retrieval.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

David Wadden, Kejian Shi, Jacob Morrison, Aakanksha Naik, Shruti Singh, Nitzan Barzilay, Kyle Lo, Tom Hope, Luca Soldaini, Shannon Zejiang Shen, and 1 others. 2024. SciRIFF: A resource to enhance language model instruction-following over scientific literature. In *Proceedings of the*

Thirty-Eighth Annual Conference on Neural Information Processing Systems (Datasets and Benchmarks Track).

Boxin Wang, Wei Ping, Lawrence McAfee, Peng Xu, Bo Li, Mohammad Shoeybi, and Bryan Catanzaro. 2024a. Instructretro: Instruction tuning post retrieval-augmented pretraining. In *Proceedings of the Forty-First International Conference on Machine Learning*.

Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, Anima Anandkumar, and Bryan Catanzaro. 2023a. Shall we pretrain autoregressive language models with retrieval? a comprehensive study. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Shufan Wang, Yixiao Song, Andrew Drozdov, Aparna Garimella, Varun Manjunatha, and Mohit Iyyer. 2023b. kNN-LM does not improve open-ended text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Yidong Wang, Qi Guo, Wenjin Yao, Hongbo Zhang, Xin Zhang, Zhen Wu, Meishan Zhang, Xinyu Dai, Min zhang, Qingsong Wen, Wei Ye, Shikun Zhang, and Yue Zhang. 2024b. AutoSurvey: Large language models can automatically write surveys. In *Proceedings of the Thirty-Eighth Annual Conference on Neural Information Processing Systems*.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, and 1 others. 2023c. How far can

camels go? exploring the state of instruction tuning on open resources. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (Datasets and Benchmarks)*.

Yubo Wang, Xueguang Ma, Ping Nie, Huaye Zeng, Zhiheng Lyu, Yuxuan Zhang, Benjamin Schneider, Yi Lu, Xiang Yue, and Wenhu Chen. 2025a. Scholarcopilot: Training large language models for academic writing with accurate citations. *arXiv preprint arXiv:2504.00824*.

Zhiruo Wang, Shuyan Zhou, Daniel Fried, and Graham Neubig. 2023d. Execution-based evaluation for open-domain code generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics.

Zilong Wang, Zifeng Wang, Long Le, Steven Zheng, Swaroop Mishra, Vincent Perot, Yuwei Zhang, Anush Mattapalli, Ankur Taly, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. 2025b. Speculative RAG: Enhancing retrieval augmented generation through drafting. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

Zora Zhiruo Wang*, Akari Asai*, Xinyan Velocity Yu, Frank F. Xu, Yiqing Xie, Graham Neubig, and Daniel Fried. 2025. CodeRAG-bench: Can retrieval augment code generation? In *Findings of the Association for Computational Linguistics: NAACL 2025*.

Cong Wei, Yang Chen, Haonan Chen, Hexiang Hu, Ge Zhang, Jie Fu, Alan Ritter, and Wenhu Chen. 2024a. Uniir: Training andnbsp;benchmarking universal multimodal information retrievers. In *Proceedings of the 18th European Conference on Computer Vision*.

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

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024b. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V

Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. In *Proceedings of the Thirty-Sixth Conference on Neural Information Processing Systems*.

Joseph Weizenbaum. 1966. ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*.

Sean Welleck, Jiacheng Liu, Ximing Lu, Hannaneh Hajishirzi, and Yejin Choi. 2022. Naturalprover: Grounded mathematical proof generation with language models. In *Proceedings of the Thirty-Sixth Annual Conference on Neural Information Processing Systems*.

Orion Weller, Benjamin Chang, Sean MacAvaney, Kyle Lo, Arman Cohan, Benjamin Van Durme, Dawn Lawrie, and Luca Soldaini. 2025. FollowIR: Evaluating and teaching information retrieval models to follow instructions. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Rhiannon Williams. 2024. Why google's ai overviews gets things wrong.

Chaoyi Wu, Weixiong Lin, Xiaoman Zhang, Ya Zhang, Weidi Xie, and Yanfeng Wang. 2024. Pmc-llama: toward building open-source language models for medicine. *Journal of the American Medical Informatics Association*.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-grained human feedback gives better rewards for language model training. In *Proceedings of the Thirty-Seventh Annual Conference on Neural Information Processing Systems*.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2023. C-pack: Packaged resources to advance general chinese embedding. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Fangyuan Xu, Kyle Lo, Luca Soldaini, Bailey Kuehl, Eunsol Choi, and David Wadden. 2024a. KIWI: A dataset of knowledge-intensive writing instructions for answering research questions. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024b. RECOMP: Improving retrieval-augmented lms with compression and selective augmentation. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023. A critical evaluation of evaluations for long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. 2021. Efficient passage retrieval with hashing 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*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation.

Kaiyu Yang, Aidan M Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. 2023a. LeanDojo: Theorem proving with retrieval-augmented language models. In *Proceedings of the Thirty-Seventh Annual Conference on Neural Information Processing Systems (Datasets and Benchmarks Track)*.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and E. Cambria. 2023b. Large language models for automated open-domain scientific hypotheses discovery. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with Freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2024. FLASK: Fine-grained language model evaluation based on alignment skill sets. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive Semiparametric Language Models. *Transactions of the Association for Computational Linguistics*.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023a. Generate rather than retrieve: Large language models are strong context generators. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Wenhao Yu, Chenguang Zhu, Zhihan Zhang, Shuohang Wang, Zhuosheng Zhang, Yuwei Fang, and Meng Jiang. 2022. Retrieval augmentation for commonsense reasoning: A unified approach. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Xinyan Yu*, Akari Asai*, Trina Chatterjee, Junjie Hu, and Eunsol Choi. 2022. Beyond counting datasets: A survey of multilingual dataset construction and necessary resources. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Xinyan Yu, Sewon Min, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023b. CREPE: Open-domain question answering with false presuppositions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*.

Hongyi Yuan, Zheng Yuan, Ruyi Gan, Jiaxing Zhang, Yutao Xie, and Sheng Yu. 2022. BioBART: Pretraining and evaluation of a biomedical generative language model. In *The 21st Workshop on Biomedical Language Processing (BioNLP)*.

Xiang Yue, Yueqi Song, Akari Asai, Seungone Kim, Jean de Dieu Nyandwi, Simran Khanuja, Anjali Kantharuban, Lintang Sutawika, Sathyanarayanan Ramamoorthy, and Graham Neubig. 2025. Pangea: A fully open multilingual multimodal llm for 39 languages. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

Matei Zaharia, Omar Khattab, Lingjiao Chen, Jared Quincy Davis, Heather Miller, Chris Potts, James Zou, Michael Carbin, Jonathan Frankle, Naveen Rao, and Ali Ghodsi. 2024. The shift from models to compound ai systems.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Dan Zhang, Ziniu Hu, Sining Zhoubian, Zhengxiao Du, Kaiyu Yang, Zihan Wang, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024a. SciInstruct: a self-reflective instruction annotated dataset for training scientific language models. In *Proceedings of the Thirty-Eighth Annual Conference on Neural Information Processing Systems (Datasets and Benchmarks Track)*.

Dejiao Zhang, Wasi Ahmad, Ming Tan, Hantian Ding, Ramesh Nallapati, Dan Roth, Xiaofei Ma, and Bing Xiang. 2024b. Code representation learning at scale. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Michael Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Tianhua Zhang, Hongyin Luo, Yung-Sung Chuang, Wei Fang, Luc Gaitskell, Thomas Hartvigsen, Xixin Wu, Danny Fox, Helen Meng, and James Glass. 2023a. Interpretable unified language checking. *arXiv preprint arXiv:2304.03728*.

Xinyu Zhang, Nandan Thakur, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Mehdi Rezagholizadeh, and Jimmy Lin. 2023b. Miracl: A multilingual retrieval dataset covering 18 diverse languages. *Transactions of the Association for Computational Linguistics*.

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*.

Yuqi Zhang, Zihao Zhao, Lanqing Hu, Shuai Wang, Penghui Jiao, Min Leng, Yuzhi Liu, Guotong Li, Chengming Xu, Chenhui Lin, and 1 others. 2024c. BioMedGPT: Open multimodal generative pre-trained transformer for biomedicine. *Nature Medicine*.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2023. Dense text retrieval based on pretrained language models: A survey.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. 2024. WildChat: 1m chatGPT interaction logs in the wild. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023a. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Proceedings of the Thirty-Seventh Annual Conference on Neural Information Processing Systems (Datasets and Benchmarks Track)*.

Yuxiang Zheng, Shichao Sun, Lin Qiu, Dongyu Ru, Cheng Jiayang, Xuefeng Li, Jifan Lin, Binjie Wang, Yun Luo, Renjie Pan, and 1 others. 2024. Openresearcher: Unleashing ai for accelerated scientific research. *arXiv preprint arXiv:2408.06941*.

Zhiling Zheng, Zichao Rong, Nakul Rampal, Christian Borgs, Jennifer T Chayes, and Omar M Yaghi. 2023b. A gpt-4 reticular chemist for guiding mof discovery. *Angewandte Chemie International Edition*.

Zexuan Zhong, Tao Lei, and Danqi Chen. 2022. Training language models with memory augmentation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Shuyan Zhou, Uri Alon, Frank F. Xu, Zhengbao Jiang, and Graham Neubig. 2023. Docprompting: Generating code by retrieving the docs. In *The Eleventh International Conference on Learning Representations*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.