

Pretrained Transformers for Efficient and Robust Information Retrieval

by

Minghan Li

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2024

© Minghan Li 2024

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Member:

Benjamin C. M. Fung

Professor

School of Information Studies

McGill University

Supervisor:

Jimmy Lin

Professor

David R. Cheriton School of Computer Science

University of Waterloo

Internal Member:

Hongyang Zhang

Assistant Professor

David R. Cheriton School of Computer Science

University of Waterloo

Internal Member:

Wenhu Chen

Assistant Professor

David R. Cheriton School of Computer Science

University of Waterloo

Internal-External Member:

Amir-Hossein Karimi

Assistant Professor

Department of Electrical and Computer Engineering

University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The thesis includes contributions from eight published papers and public manuscripts. Minghan Li is the first author of the paper described in Chapter 4, 5, 6, 8, 9, 10, who is responsible for the ideation, methodology, implementation, conducting experiments, analyzing results, and paper writing. Minghan Li is the co-first author of the paper discussed in Chapter 3 and 7, who is responsible for the ideation, methodology, and paper writing. List of publications and co-authors:

- Chapter 3: Simple and Effective Unsupervised Redundancy Elimination to Compress Dense Vectors for Passage Retrieval. Xueguang Ma*, Minghan Li*, Kai Sun, Ji Xin, Jimmy Lin. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2021.
- Chapter 4: Multi-Task Dense Retrieval via Model Uncertainty Fusion for Open-Domain Question Answering. Minghan Li, Ming Li, Kun Xiong, Jimmy Lin. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2021.
- Chapter 5: CITADEL: Conditional Token Interaction via Dynamic Lexical Routing for Efficient and Effective Multi-Vector Retrieval. Minghan Li*, Sheng-Chieh Lin, Barlas Oguz, Asish Ghoshal, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, Xilun Chen*. *In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.
- Chapter 6: SLIM: Sparsified Late Interaction for Multi-Vector Retrieval with Inverted Indexes. Minghan Li, Sheng-Chieh Lin, Xueguang Ma, Jimmy Lin. *In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023.
- Chapter 7: CELI: Simple yet Effective Approach to Enhance Out-of-Domain Generalization of Cross-Encoders. Xinyu Zhang*, Minghan Li*, Jimmy Lin. *In Proceedings of the North American Chapter of the Association for Computational Linguistics*, 2024.
- Chapter 8: Can Query Expansion Improve Generalization of Strong Cross-Encoder Rankers? Minghan Li, Honglei Zhuang, Kai Hui, Zhen Qin, Jimmy Lin, Rolf Jagerman, Xuanhui Wang, Michael Bendersky *In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.
- Chapter 9: Certified Error Control of Candidate Set Pruning for Two-Stage Relevance Ranking. Minghan Li*, Xinyu Zhang*, Ji Xin, Hongyang Zhang, Jimmy Lin *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2022.
- Chapter 10: Nearest Neighbor Speculative Decoding for LLM Generation and Attribution. Minghan Li, Xilun Chen, Ari Holtzman, Beidi Chen, Jimmy Lin, Wen-tau Yih, Xi Victoria Lin. *arxiv preprint, abs/2405.19325*, 2024.

Abstract

Pretrained transformers have significantly advanced the field of information retrieval (IR) since the introduction of BERT. Through unsupervised pretraining followed by task-aware finetuning, the encoder-only BERT effectively captures the complex semantics of texts and overcomes the limitations of traditional bag-of-words systems like BM25, addressing the lexical mismatch between queries and documents. The successor models based on decoder-only pretrained transformers such as GPT and LLaMA have further improved the ranking effectiveness by scaling up the computation. Undoubtedly, these pretrained transformers have made a profound impact on search and ranking, completely changing the landscape of modern IR systems.

Consequently, pretrained transformers are now applied in various stages of a modern IR system, typically comprising a first-stage retriever, a second-stage reranker, and an optional large language model (LLM). For instance, the first-stage retriever can be a bi-encoder that separately encodes the queries and documents, and the encoded collection of documents is indexed into a vector database for nearest neighbor search. Given a query, the first-stage retriever selects the top-k similar documents from the vector database for the second-stage reranker to review. The following second-stage reranker is usually a cross-encoder which evaluates the query and documents jointly, further refining the search results. The final step is to use an LLM to summarize the retrieved information before returning it to the users, which is optional but has become more and more popular in modern search engines.

However, pretrained transformers are not without flaws. Compared to traditional IR methods, the challenges of efficiency and robustness are frequently encountered in practice. For example, first-stage retrievers and their vector indexes often suffer from redundancy, leading to increased search latency and excessive storage requirements. The second-stage rerankers, based on cross-encoder pretrained transformers, also incur high computational costs when reranking a large set of retrieved documents. In addition, the representations of pretrained transformers are usually lossy compression of the inputs and tied to certain data distributions, making them susceptible to out-of-domain queries and subtle details such as dates and locations. These issues can further impair the downstream LLM, hindering the efficient extraction of useful information from the retrieved documents and promoting hallucinations when processing biased content retrieved by adversarial queries.

Given the breadth of the challenges outlined, this thesis will delineate a specific scope of study for efficiency and robustness. Regarding efficiency, our focus will be on minimizing search latency and storage costs during the first-stage retrieval and second-stage reranking processes, along with reducing the generation latency of the downstream LLM. Regarding robustness, we will concentrate on the reliability of both retrievers and rerankers in handling out-of-domain queries, as well as ensuring the factuality and attribution accuracy in the output of the retrieval-augmented LLM. Under this context, we outline our efforts to address the efficiency and robustness challenges

in major components of a modern IR system based on pretrained transformers, ranging from first-stage retrieval—including dense, sparse, hybrid, and multi-vector retrievers—to second-stage cross-encoder reranking, as well as tackling hallucination and attribution issues in downstream LLMs from a retrieval standpoint.

For first-stage dense retrieval models, we explore index compression to reduce search latency and storage space through dimensionality reduction and product quantization. We also detail our work on improving the robustness of these models through techniques like model ensembling and integration with sparse retrieval methods. In extending these ideas to multi-vector retrieval systems, we employ dynamic lexical routing and token pruning to jointly optimize for efficiency and robustness during finetuning on ranking tasks. Additionally, we examine the integration of sparse retrieval in a multi-vector system and propose sparsified late interactions for efficient retrieval, ensuring compatibility with traditional inverted indexes and improving latency.

In the later sections, we continue to address the efficiency and robustness challenges in second-stage rerankers based on cross-encoders. We introduce a straightforward strategy for cross-encoder rerankers by adding late interaction at the last layer to handle out-of-domain, long-sequence data with minimum latency overhead. We also assess the impact of using LLMs in query expansion for high-precision cross-encoder rerankers in conventional ranking tasks, finding that traditional query expansion methods can undermine the robustness of effective rerankers. After integrating first-stage retrieval with second-stage reranking, we propose a candidate-set pruning method with high-confidence error control to accelerate the reranking process reliably, allowing users to specify their precision goals while maintaining efficiency with theoretical guarantees.

Finally, we use these optimized two-stage retrieval systems to address hallucination and attribution problems in downstream retrieval-augmented LLMs with improved efficiency for generation. We introduce a hybrid generation model that incorporates segments retrieved from the corpus directly into LLM outputs to reduce hallucinations and improve attribution in knowledge-intensive tasks. Our final system leverages our efficient and robust solutions in neural retrieval, delivering precise responses swiftly and accelerating content generation by processing multiple tokens per time step, with a post-hoc revision mechanism to balance robustness and efficiency.

Overall, this thesis aims to improve the efficiency and robustness of key components in a contemporary IR system and their integration with downstream LLMs.

Acknowledgments

I would like to first express my gratitude to my advisor, Jimmy Lin, for his invaluable guidance and support throughout my Ph.D. studies. Working with him gave me considerable freedom to pursue my interests while ensuring I remained on the right track. He taught me how to conduct independent research and various skills, including how to assess the impact of my papers, present my research effectively, and refine my writing style. I am fortunate to have had such a great mentor during my Ph.D. studies.

I would like to thank my committee members—Hongyang, Wenhui, Amir, and Benjamin—for their time in reading my lengthy thesis and providing insightful feedback.

I would like to thank all the mentors I met during my internships at Meta and Google, including Xilun, Honglei, Victoria, Scott, and other peers. Their guidance on identifying impactful problems, delivering practical solutions, and communicating effectively with team members was invaluable. The research approach I learned during my internships provided new perspectives that will undoubtedly influence my future career.

I am grateful for the support of the Waterloo Apple PhD Fellowship in Data Science and Machine Learning, which allowed me to continue my research without concerns about living conditions or accessing external resources.

I also wish to thank my lab mates and friends from my doctoral studies: Xueguang, Crystina, Jayden, and Sheng-Chieh, for being my co-authors on many papers and for all the joyful hours spent outside of work. Thanks also to Richard, Peng, and Linqing for their assistance with job referrals, and to Jiarui, Xinyu, Jin, Ehsan, Matt, Nandan, and Ronak for those informative discussions and enjoyable chats.

Reflecting on the path before my Ph.D., I am thankful for the camaraderie of my classmates during my M.Sc. studies—Mohan, Zhiyong, Zihao, Xinyu, and Siyue—for their companionship and the enjoyable times shared after classes. My gratitude extends to my M.Sc. advisors, Danijar Hafner, Jimmy Ba, and Animesh Garg, for guiding me in exploring deep reinforcement learning, and to my internship mentors at RSVP.ai, Luchen Tan, Kun Xiong, and Ming Li, for giving me the hands-on opportunity on my first NLP project.

I would also like to acknowledge my undergraduate advisors, Weishi Zheng and Martha White, as well as my mentors during my undergraduate internship at MSRA, Huanhuan Xia and Bin Shao. Thank you for teaching me the fundamentals of research and laying a solid foundation for my graduate studies.

Special thanks to Xuanhui and Kevin for helping me settle in when I first arrived in Canada. My thanks also go to Yifu and Jingyun for the countless hours we spent playing board games and video games together. Thanks for offering me free rides from time to time and letting me stay at your house when I had nowhere else to live!

Completing this journey would have been a distant reality without the unwavering support and immense sacrifices of my parents, Yongzhao Li and Xiaoqiong Liang. My parents would have had more of my accompany if their selfish son decided not to go abroad, yet they still fully supported my decision knowing that I might not be able to come home very often. I'm blessed with such parents as they have always encouraged me to pursue my passions and stood by me through countless challenges. I also owe them too much for the countless sacrifices they made which shaped me into the individual I am today. Thank you, Mom and Dad, for everything.

Lastly, and most importantly, I owe a profound debt of gratitude to my beloved wife, Nanyun Zhang. During our two years of long-distance relationship and throughout the pandemic, she has been my unswerving support and warmest companion. I cannot imagine going through those dark and gloomy days without her. She also made significant sacrifices, leaving her job in Europe and facing numerous challenges to move to Canada to marry a humble Ph.D. student without reservations. Marrying you, Nanyun, is the luckiest thing that happened in my life, and I will cherish you forever.

Dedication

To my wife, my parents, and all the good things in life.

Table of Contents

List of Figures	xv
List of Tables	xx
1 Introduction	1
1.1 Challenges of Deploying Pretrained Transformers in IR	2
1.2 Efficient and Robust Solutions for Modern IR	3
1.3 Contributions	5
2 Background	7
2.1 Bag-of-Words Retrieval	7
2.2 Approximate Nearest Neighbor Search	8
2.3 Neural Retrievers and Rerankers	9
2.3.1 Dense Retrievers	9
2.3.2 Supervised Sparse Retrievers	9
2.3.3 Multi-Vector Retrievers	11
2.3.4 Cross-Encoder Rerankers	12
2.4 Retrieval-Augmented Language Models	13
3 Efficient Dense Retrieval via Index Compression	14
3.1 Overview	14
3.2 Compressing Dense Representations	15

3.2.1	Quantifying Redundancy	15
3.2.2	Dense Vector Compression	16
3.3	Experimental Setup	17
3.4	Results	18
3.4.1	Dimensionality Reduction	19
3.4.2	Product Quantization	20
3.4.3	Hybrid Search	20
3.5	Summary	21
4	Robust Dense Retrieval via Model Uncertainty Fusion	22
4.1	Overview	22
4.2	Multi-Task Model Fusion	24
4.2.1	Ensemble Uncertainty Estimation	25
4.2.2	Model Uncertainty Fusion	26
4.2.3	Uncertainty Calibration	26
4.3	Experimental Setup	27
4.4	Results and Analysis	28
4.4.1	Benchmark Dataset Retrieval	28
4.4.2	Mixed-Dataset Retrieval	29
4.4.3	DPR-BM25 Hybrid Retrieval	30
4.4.4	Ensemble Sensitivity and Latency	32
4.4.5	Uncertainty Visualization	33
4.4.6	Storage-Latency-Flexibility Trade-off	34
4.5	Summary	34
5	Efficient Token Interaction via Dynamic Lexical Routing	36
5.1	Overview	36
5.2	A Unified Token Routing View of Multi-Vector Retrievers	39
5.2.1	Single-Vector Retrieval	39

5.2.2	Multi-Vector Retrieval	39
5.3	The CITADEL Method	40
5.3.1	Dynamic Lexical Routing	40
5.3.2	Router Optimization	41
5.3.3	Sparsely Activated Router Design	42
5.3.4	Regularization for Routing	42
5.3.5	Inverted Index Retrieval	43
5.4	Experimental Setup	44
5.4.1	Datasets	44
5.4.2	Baselines	44
5.4.3	Training	44
5.5	Results	46
5.5.1	MS MARCO Passages Retrieval	46
5.5.2	BEIR: Out-of-Domain Evaluation	47
5.6	Performance Analysis	47
5.6.1	Number of Token Interactions	47
5.6.2	Latency-Memory-Accuracy Tradeoff	47
5.6.3	Combination with Product Quantization	48
5.7	Token Routing Analysis of CITADEL	49
5.8	Ablation Studies	51
5.9	Summary	51
6	Sparsified Late Interaction with Inverted Indexes	53
6.1	Overview	53
6.2	Efficient Approximation for Sparsified Late Interaction	55
6.3	Experiments	58
6.4	Summary	60

7	Robust Cross-Encoder Rankers with Late Interaction	61
7.1	Overview	61
7.2	Late Interaction for Robust Cross-Encoder Ranking	62
7.3	Experimental Setup	64
7.4	Results and Analysis	64
7.4.1	Model Size	64
7.4.2	First-Stage Retriever	65
7.4.3	Token Dimensions	66
7.4.4	Query Length	67
7.5	Related Work	68
7.6	Summary	68
8	Multi-Step Query Expansion for Cross-Encoder Rankers	69
8.1	Overview	69
8.2	Related Work	71
8.3	Framework and Implementation	71
8.3.1	Cross-Encoder Ranking	71
8.3.2	High-Quality Keyword Generation	72
8.3.3	Minimally-Disruptive Query Modification	73
8.4	Experiments	74
8.5	Ablation Analysis	75
8.6	Summary	77
9	Certified Error Control for Two-Stage Relevance Ranking	78
9.1	Overview	78
9.2	Related Work	81
9.3	Background	81
9.3.1	Notation	81
9.3.2	Two-Stage Relevance Ranking	81

9.4	Risk-Control Prediction Set for Ranking	83
9.4.1	Settings	83
9.4.2	Truncated Risk Function for Ranking	84
9.4.3	Correction for Risk Level and Confidence	85
9.5	Experimental Setup	86
9.5.1	Datasets	86
9.5.2	Retrievers and Rerankers	87
9.5.3	Baselines	88
9.6	Results	89
9.6.1	Risk Function and Upper Bound	89
9.6.2	In-Domain Results	89
9.6.3	Out-of-Domain Results	89
9.6.4	Efficiency-Accuracy Trade offs with Certified Error Control	90
9.6.5	Confidence-Risk Correction	91
9.7	Summary	92
10	Nearest Neighbor Speculative Decoding	93
10.1	Overview	93
10.2	Background	94
10.2.1	Problem Definition	94
10.2.2	Nearest Neighbor Language Models (k NN-LM)	95
10.3	Nearest Neighbor Speculative Decoding	95
10.3.1	Two-Stage k -NN Search	95
10.3.2	Confidence-Based Output Interpolation	97
10.3.3	Dynamic Span Selection	97
10.3.4	Relaxed Speculative Decoding	97
10.4	Experiments	98
10.4.1	Benchmark Datasets	98

10.4.2	Implementation	99
10.4.3	Baselines	100
10.4.4	Main Results	100
10.4.5	Latency Analysis	102
10.4.6	Attribution and Qualitative Analysis	103
10.5	Summary	104
11	Conclusion and Future Work	105
	References	108

List of Figures

1.1	An illustration of a modern IR system based on pretrained transformers, including first-stage retrieval for high recall, second-stage reranking for high precision, and an optional process (dashed arrows) of using large language model (LLM) for generating answers.	2
2.1	Dense retriever vs supervised sparse retriever. (a) Dense retriever projects the query and the document individually onto a low-dimensional dense space. (b) Supervised sparse retriever projects the query and the document onto a high-dimensional space (usually the vocabulary) where most dimensions are set to zero.	10
2.2	ColBERT vs COIL. (a) ColBERT considers token interactions between all query token vectors and the document token vectors. (b) COIL only considers token interactions that have overlaps between the query and the document at the input.	11
3.1	Quantifying redundancy in dense vectors using explained variance ratio and normalized mutual information. The y-axis is ratio and does not have a unit.	16
3.2	Trade-offs between retrieval accuracy and index size using different combination of our compression techniques.	20
4.1	An example of ensemble uncertainty on two examples with the TriviaQA expert model. For the in-domain query, the ensemble of neural networks will tend to agree on the same document prediction, while the ensemble prediction diverges on out-of-domain query as the neural networks are encouraged to converge on different local optima.	23
4.2	An illustration of model uncertainty fusion of 3 DPR experts, each with an ensemble of 3 fully connected neural networks. Given a query, each DPR expert first retrieves top- k documents, followed by the uncertainty estimation using the corresponding ensemble. The weighted sum of predictions is then used to rerank the union of the retrieved documents.	24

4.3	Line plot of top-100 accuracy (%) and bar chart of latency (ms/query) relative to a single DPR model of our method w.r.t. the size of the ensemble on NQ, Trivia, WQ, SQuAD, and their random mixes.	32
4.4	A visualization of uncertainty estimation using mutual information between the ensemble’s predictions and parameters. Each subplot shows the prediction of 5 ensemble on the top-20 documents retrieved by the DPR expert. The uncertainty decreases as more ensemble members “agree” with each other.	33
5.1	GPU latency vs ranking quality (MRR@10) on MS MARCO passages with an A100 GPU. The <i>circle size</i> represents the relative index storage on disk. All models are trained without hard-negative mining, distillation, or further pretraining.	37
5.2	A unified token routing view over different multi-vector and single-vector retrieval models. The cylinder in (d) represents the lexical key for token routing. The grey tokens in (c) and (d) represent the tokens that do not contribute to the final similarity. For (d), CITADEL routes “ <i>light</i> ”, “ <i>theory</i> ”, and “ <i>relativity</i> ” to the “ <i>Einstein</i> ” key to avoid term mismatch using a learned routing function.	38
5.3	Normalized disk storage over token indices in an inverted list of CITADEL and COIL.	48
5.4	Latency-memory-accuracy tradeoff on MS MARCO passages using post-hoc pruning. τ is the pruning threshold.	49
5.5	Tokens in grey (pruned) have zero activated keys while bold tokens have at least one activated key. We leave out the expanded terms and routing weights due to space limit.	50
5.6	Token distribution over number of activated experts per passage. τ is the pruning threshold.	50
5.7	Token routing analysis of CITADEL. Grey tokens are deactivated, while bold tokens are routed to at least one activated key (in blue).	51
6.1	(a) The naive SLIM implementation takes the expansion of each token as sub-queries/documents. The token-level rankings are merged using scatter operations. (b) The approximate SLIM implementation first fuses the lower- and upper-bounds of scores for retrieval using Equations (6.8) and (6.9); the candidate list is then refined according to Equation (6.4).	54
6.2	Effectiveness-Efficiency trade-offs of SLIM (w/ and w/o score refinement) on MS MARCO Passages.	59

7.1	nDCG@10 improvement from late interaction on queries over different lengths. Each point represents the average of nDCG@10 improvements over the query of the corresponding length. The line is the least square polynomial fit of the points.	67
8.1	A flowchart of our method including generation, filtering, and fusion to select high-quality keywords for zero-shot ranking task with strong cross-encoder rankers. . . .	70
8.2	nDCG@10 score on DL19 using different number of keywords for RankT5.	76
9.1	Comparisons between certified and empirical error control methods on MS MARCO Passage v1. Coverage: Percentage of 100 independent runs that satisfies a pre-specified $\text{MRR@10} \geq 0.35$ (the dotted line).	79
9.2	Calibration and test procedures of candidate set pruning with certified error control. During calibration, we use the confidence region and desired risk α to find the threshold $-\hat{\lambda}$. During testing, we use the threshold to prune the retrieved documents, which guarantees control of the expected loss under α with probability $1 - \delta$	80
9.3	(Corrected) Confidence region of the risk function. (a) shows the risk function for metrics like recall; (b) shows the risk function that is truncated in ranking. (c) and (d) show two different types of corrections: Either setting the risk level to the minimum of the upper bound or increasing δ to shrink the upper bound.	85
9.4	Empirical risk function and its upper bound on one calibration-test data split. . . .	88
9.5	Trade offs between candidate set size and MRR@10 scores on MS MARCO Passage v1 based on the DPR + MonoELECTRA ranking system. The empirical coverage for each point is greater than 90%.	90
9.6	Confidence correction on MS MARCO Passage v1 using DPR + MonoELECTRA. The x -axis represents different risk thresholds (i.e., α) and the y -axis represents the percentage.	91
10.1	The NEST approach first locates the tokens in the corpus using the LM hidden states. The retrieval distribution $p_{k\text{-NN}}$ is dynamically interpolated with p_{LM} based on the retriever’s uncertainty λ_t . The token and its n -gram continuation are then selected from the mixture distribution $p_{\mathcal{M}}$, while the final span length is determined by speculative decoding to remove undesired tokens. The spans incorporated in the final generation provide direct attribution and amortize the generation latency. . .	96

10.2 Latency-accuracy trade-off and breakdown on Biography using Llama-2-Chat 70B+NEST.	
As the relaxation factor γ decreases, NEST tends to accept longer spans from the corpus. We choose $\gamma = 5e - 2$ in our main experiments, which accelerates the generation and improves the FACTSCORE.	102

List of Tables

3.1	Top- $\{20, 100\}$ retrieval accuracy of different dimensionality reduction methods evaluated at $d = \{256, 128, 64\}$ on five benchmark QA datasets. The BM25 index size is about 2.4 GB.	18
3.2	Top- $\{20, 100\}$ retrieval accuracy of three dimensionality reduction methods with different PQ settings at $d = \{256, 128, 64\}$	19
4.1	Top-20 and Top-100 retrieval accuracy (%) on benchmark QA test sets. Each score represents the percentage of top 20/100 retrieved passages that contain answers. All methods containing “DPR-MUF” stand for our Model Uncertainty Fusion (MUF) method. See Section 4.4.1 for details of different models.	27
4.2	Top-20/100 retrieval accuracy (%) on random mixes of 4 benchmark QA test sets. We average the metrics from 5 evenly-sampled subsets of 3,000 samples from NQ, Trivia, WQ, and SQuAD.	30
4.3	Top-20/100 retrieval accuracy (%) of BM25 and DPR-BM25 hybrid model on test sets of NQ, Trivia, WQ, CuratedTrec, and SQuAD.	31
5.1	In-domain evaluation on MS MARCO passages and TREC DL 2019/2020. CITADEL ⁺ is trained with cross-encoder distillation and hard-negative mining. The red region means CITADEL/CITADEL ⁺ is better than the method while the green region means that there’s no statistical significance ($p > 0.05$). “×” means not implemented and “-” means not practical to evaluate on a single CPU.	45
5.2	Out-of-domain evaluation on BEIR benchmark. nDCG@10 score is reported. Dataset Legend [42]: TC=TREC-COVID, NF=NFCorpus, NQ=NaturalQuestions, HQ=HotpotQA, FQ=FiQA, AA=ArguAna, T2=Touché-2020 (v2), Qu=Quora, DB=DBPedia, SD=SCIDOCS, Fe=FEVER, CF=Climate-FEVER, SF=SciFact.	46
5.3	Maximal number of dot products per query on MS MARCO passages. # DP: number of dot products.	48

5.4	Product Quantization. Pruning threshold is set to 0.9. nbits: ratio of total bytes to the vector dimension.	49
5.5	[CLS] ablation on MS MARCO passage. Dev: MRR@10. DL19: nDCG@10.	50
5.6	Number of routing keys for documents during training. No post-hoc pruning is applied.	52
6.1	In-domain and out-of-domain evaluation on MS MARCO Passages, TREC DL 2019/2020, and BEIR. “-” means not practical to evaluate on a single CPU. Latency is benchmarked on a single CPU and query encoding time is excluded.	56
7.1	A preview of comparison between CELI and monoBERT. Detailed results are shown in Table 7.2 and 7.3.	61
7.2	In-domain (MRR@10 on MS MARCO) and OOD (averaged nDCG@10 on BEIR) scores of CELI and two baselines (i.e., monoBERT and Mean-Pooling).	65
7.3	Averaged nDCG@10 on BEIR, reranking the top-1k candidates from each retriever. TCT : TCT-ColBERT. DPR (NQ/MS) : DPR finetuned on NQ [138] or MS MARCO, respectively.	66
7.4	MRR@10 on MS MARCO and nDCG@10 on BEIR of CELI with token representation in different dimensions (D_{tok} in Eq. (7.3)). We report scores to 4 digits here as the values are close in all conditions.	66
8.1	nDCG@10 on TREC DL 2019/2020. Directly applying existing query expansion methods on strong cross-encoder rankers can cause effectiveness deterioration.	70
8.2	nDCG@10 scores on BEIR. TC=TREC-COVID, NF=NFCorpus, NQ=NaturalQuestions, HQ=HotpotQA, FQ=FiQA, AA=ArguAna, T2=Touché-2020, Qu=Quora, DB=DBPedia, SD=SCIDOCS, Fe=FEVER, CF=Climate-FEVER, SF=SciFact, S1=Signal-1M, BA=BioASQ, R0=Robust04, TN=TREC-NEWS. *: pass the paired t-test against the other baselines ($p < 0.01$).	73
8.3	nDCG@10 score of Q2D2K, Q2K, RM3, and PRF + D2K with fusion based on RankT5.	74
8.4	Incremental ablation on the keyword fusion process. Details are introduced in Section 8.5.	76
8.5	Weighting methods for fusion on DL 19/20 based on RankT5. nDCG@10 scores are reported.	76

9.1	In-domain (MS MARCO Passage v1) and out-of-domain (Quora) first-stage retrieval test MRR@10 scores averaged over 100 random dev/test splits.	86
9.2	In-domain (MS MARCO Passage v1) and out-of-domain (Quora) second-stage reranking (with evidence fusion) test MRR@10 scores averaged over 100 random dev/test splits.	86
9.3	In-domain results on MS MARCO Passage v1 (left) and out-of-domain results on Quora (right). CEC: Certified error control. EST: Empirical score threshold. ERT: Empirical rank threshold. Confidence: $1 - \delta_c$ as in Eq (9.10). Coverage: Proportion of 100 runs that satisfy the risk constraints. Size: Average candidate set size out of 1,000. See Section 9.6.2 for details.	87
10.1	Results on text completion (upper table) and other tasks (lower table). Bold numbers indicate the best performance. PPL: Perplexity. RG: ROUGE score. Avg. Len: Average generation length. Δ BLEU/ Δ RG: The difference between the max score to correct references and the max score to incorrect references. FS: FACTSCORE with length penalty.	101
10.2	Attribution analysis. (Attribution) Ratio: Proportion of tokens that are taken from the corpus. (Attribution) Length: Average length of consecutive spans in the generation that are taken from the same document. Green : Segments taken from the corpus. Gray : Reference.	103

Chapter 1

Introduction

Recent progress in natural language processing (NLP) has been driven by the development of pre-trained transformers [319]. Capable of learning complex semantics from texts, these models are “transforming” the way users interact with digital systems. Since the introduction of BERT [57], the encoder-only transformers have achieved major milestones in many NLP tasks such as sentiment analysis and reading comprehension [369]. Meanwhile, decoder-only transformers such as GPT [253, 34, 237] and LLaMA [314, 315] revolutionize the field of text generation by scaling up the computation (i.e., pretraining on more data and using more parameters) for auto-regressive models. Unlike BERT, these generative models can produce coherent and engaging content under instructions, enabling personalized and context-aware interactions. Such capability marks a major paradigm shift from smaller neural networks, such as LSTMs [98] and CNNs [130] trending in early NLP applications, to behemoth models with billions of parameters, deepening our comprehension of the potential of these pretrained transformers.

The advancement above has also set a new foundation for information retrieval (IR), a critical domain in NLP focused on surfacing relevant information from vast collections. Traditional methods like TF-IDF [275] and BM25 [267], which rely on bag-of-words, term-matching approaches, fail to consider the semantic relationships between texts without term overlaps, rendering them less suitable for semantic-related tasks. As language models become more adept at understanding context and generating text, IR systems must evolve accordingly. Modern IR needs to leverage these advancements to effectively retrieve information and assess its relevance across various contexts, ensuring that the presented information is both accurate and contextually appropriate.

In this context, modern IR systems based on pretrained transformers [126, 197] have emerged as alternatives to traditional search models. For instance, dense passage retrieval [126] encodes the query and documents into a continuous vector space using pretrained transformers, calculating the relevance by measuring the similarity between vectors. Such representation learning methods allow the model to capture more complex relationships between queries and documents, enabling

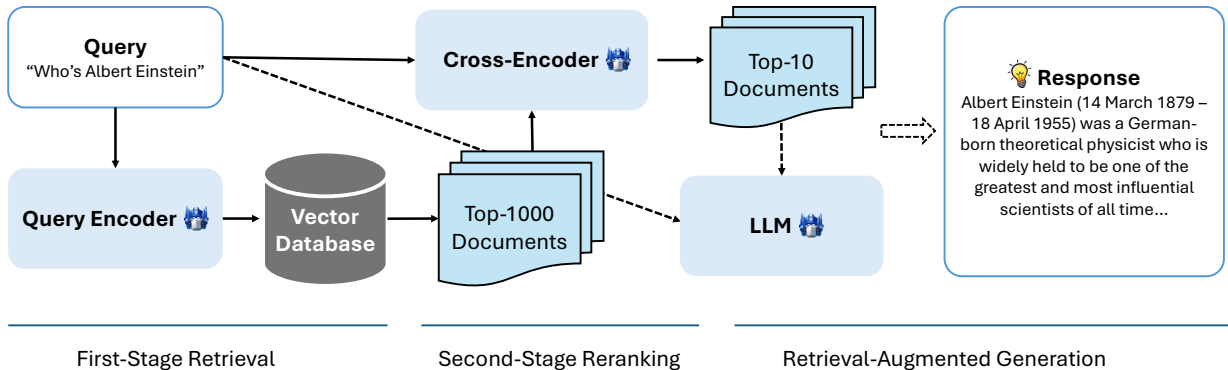


Figure 1.1: An illustration of a modern IR system based on pretrained transformers, including first-stage retrieval for high recall, second-stage reranking for high precision, and an optional process (dashed arrows) of using large language model (LLM) for generating answers.

them to achieve state-of-the-art accuracy in various IR tasks, such as open-domain question-answering [41, 147] and fact-verification tasks [226, 232].

In this thesis, we summarize our efforts to enhance the components of a modern IR system based on pretrained transformers, including first-stage retrieval and second-stage reranking, as well as downstream retrieval-augmented LLMs as shown in Figure 1.1. We begin by describing the challenges encountered in deploying modern IR systems based on pretrained transformers and then present our solutions which will be discussed in detail throughout the thesis.

1.1 Challenges of Deploying Pretrained Transformers in IR

As mentioned above, a modern IR system based on pretrained transformers typically includes a first-stage retriever aiming for high recall, a second-stage reranker targeting high precision, and an optional retrieval-augmented language model that summarizes retrieved information for the user. The first-stage retriever needs to first encode the whole corpus of documents into vectors and store them in a vector database for nearest neighbor search. However, using a pretrained transformer to encode a large collection of documents often requires significant computational resources compared to traditional IR methods. Moreover, the encoded vector databases can consume substantial storage space and demand high latency to search for relevant documents, depending on the hardware and configuration. The second-stage reranker, in contrast, does not need to store the indexed documents but instead computes the relevance score for each query-document pair on the fly. Therefore, when reranking a large candidate set retrieved from the first stage, the latency and computational costs can become computationally intractable due to

the expensive forward process of transformers. Both the high latency and storage costs can be categorized as the efficiency challenge.

The second challenge, affecting both first-stage retrievers and second-stage rerankers, is that pretrained transformer models finetuned on specific domains may not generalize well to other domain-specific nuances and jargon uncommon in other IR tasks. For example, models finetuned on general question-answering tasks might struggle to adapt to medical record retrieval. This lack of robustness can lead to sub-optimal results and even potential risks in situations that demand reliability, such as retrieving medical records and legal documents.

Lastly, the efficiency and robustness issues of the first-stage and second-stage models can further impact the downstream retrieval-augmented language models. With the rapid rise of LLMs, it is increasingly common for IR systems or search engines to employ an LLM to summarize the retrieved information before presenting it to the users. This paradigm, known as retrieval-augmented generation (RAG), can lead to difficulties for LLMs in efficiently extracting relevant content from scattered retrieved documents and may encourage hallucination when facing adversarial queries that result in retrieving biased content. For example, if a user asks, “What impact does the position of the Sun at birth have on someone’s personality?” and the search engine returns a collection of unverified research on astrology, the downstream LLM might generate incorrect conclusions based on the retrieved information without further verification.

1.2 Efficient and Robust Solutions for Modern IR

This thesis addresses the above challenges and discusses approaches to enhance the efficiency and robustness of a two-stage base IR system, as well as improve the factuality and attribution in downstream retrieval-augmented language models.

Chapter 2 lays the groundwork by establishing the foundational concepts necessary to grasp the complexities of modern neural retrieval systems. Initially, we introduce the basic types of retrieval systems in IR, namely Bag-of-words systems (BoW) and Approximate Nearest Neighbor search (ANN). Subsequently, we delve into the model architectures of diverse retrievers based on pretrained transformers, including dense retrievers, sparse retrievers, multi-vector retrievers, and cross-encoder rankers. At the end of the chapter, we briefly overview various types of retrieval-augmented LLM systems.

Chapter 3 begins an in-depth investigation into the efficiency and robustness of simple dense retrievers. We propose an unsupervised compression pipeline consisting of principal component analysis (PCA), product quantization, and hybrid search to enhance efficiency without compromising effectiveness. To bolster the robustness of single-vector dense retrieval, Chapter 4 introduces a method to train individual dense retrievers for various tasks and aggregate their predictions

during test time, utilizing uncertainty estimation as weights to determine the likelihood that a specific query pertains to each expert’s domain.

Chapter 5 expands our discussion on efficiency and robustness to more advanced retrieval systems like multi-vector retrievers. We unify various multi-vector retrieval models from a token routing perspective and propose conditional token interaction through dynamic lexical routing for efficient and effective multi-vector retrieval. Our model is designed to route each token vector to predetermined lexical “keys,” ensuring that a query token vector interacts only with document token vectors routed to the same key. This design significantly reduces computational costs while maintaining high accuracy. To better integrate this multi-vector retrieval system with existing search libraries such as Pyserini and Lucene, Chapter 6 introduces an efficient two-stage retrieval architecture that includes inverted index retrieval followed by a score refinement module to approximate late interaction, resulting in a sparse retrieval system that is faster yet as accurate as multi-vector retrievers like ColBERT. These methods exploit the redundancy in multi-vector retrieval, striking an ideal balance between efficiency and robustness.

Transitioning from first-stage retrievers to second-stage rerankers, Chapter 7 integrates the late interaction mechanism from multi-vector retrieval into the cross-encoder models. This simple approach yields a 5% improvement on BEIR benchmarks without compromising in-domain effectiveness or increasing search latency. Chapter 8 identifies two crucial steps for cross-encoders using query expansion: high-quality keyword generation and minimally disruptive query modification. We demonstrate that it is possible to further enhance the generalization of a robust neural ranker by generating keywords through a reasoning chain and aggregating the ranking results of each expanded query via self-consistency, reciprocal rank weighting, and fusion. Chapter 9 addresses the shortcomings of a commonly used candidate set pruning method in two-stage relevance ranking, which often empirically sacrifices accuracy for computational efficiency without theoretical guarantees. We propose the concept of certified error control for candidate set pruning in relevance ranking, ensuring that the test error after pruning remains within a user-specified threshold with high probability.

With the efficient and robust solutions for the base IR system in place, Chapter 10 integrates these enhanced methods with LLMs for knowledge-intensive tasks. We propose a training-free, span-level nearest neighbor language model that directly injects segments from the corpus into the LLM output. We further utilize speculative decoding with relaxation to refine the outputs, eliminating generation artifacts (e.g., grammatical errors, repetition caused by retrieval) while preserving the crucial facts sourced from the corpus. The final model is both efficient and robust, providing direct attribution to improve the factuality of the LLM’s output.

Finally, Chapter 11 outlines promising future research directions that could further enhance the efficiency and robustness of retrieval systems and LLMs, including potential advancements in model architecture, training techniques, and integration methods.

1.3 Contributions

In summary, this thesis contributes to the field of Information Retrieval (IR) by improving the efficiency and robustness of first-stage retrievers and second-stage rerankers based on pretrained transformers, as well as integrating these advancements with downstream retrieval-augmented language models to address factuality and attribution issues. We enumerate our contributions as follows:

- Unsupervised index compression for efficient dense retrieval (Chapter 3): We introduce an unsupervised compression pipeline for first-stage retrieval models, incorporating principal component analysis (PCA), product quantization, and hybrid search techniques. This methodology enhances the efficiency of dense retrieval without compromising effectiveness.
- Model uncertainty fusion for robust dense retrieval (Chapter 4): We train individual dense retrievers for various tasks and aggregate their predictions during test time. By employing uncertainty estimation as weights, we enhance the system’s overall robustness.
- Dynamic lexical routing for efficient multi-vector retrieval (Chapter 5): We establish a new conceptual framework for multi-vector retrievers, predicated on dynamic lexical routing. This strategy substantially reduces computational costs by facilitating efficient token interactions, thereby improving both the scalability and effectiveness of retrieval systems.
- Sparsified late interaction for efficient multi-vector retrieval (Chapter 6): We transform token embeddings for late interaction into a sparse representation, allowing multi-vector retrieval systems to utilize traditional inverted index search methods, thereby enhancing compatibility and efficiency.
- Injecting late-interaction for robust cross-encoder reranking (Chapter 7): We utilize robust late-interaction designs from multi-vector retrieval and apply them to cross-encoder rerankers, significantly enhancing their reliability on out-of-distribution data without a substantial increase in computational costs.
- Query expansion via LLMs for robust cross-encoder reranking (Chapter 8): We explore the use of Large Language Models (LLMs) for query expansion, aiming to bolster the robustness of cross-encoder rankers through high-quality keyword generation and minimally disruptive query modification.
- Certified error control for two-stage relevance ranking (Chapter 9): We introduce an error control method with a high probability guarantee for candidate set pruning, which optimizes reranking efficiency in a two-stage ranking system. This method allows users to specify performance goals and delivers results with a high probability of meeting those targets.

- Nearest neighbor speculative decoding for LLMs (Chapter 10): We integrate LLMs with enhanced retrieval systems for instruction-following tasks, addressing factual accuracy and source attribution issues in language generation. Through a span-level nearest-neighbor approach, we inject precise corpus segments directly into the LLM outputs, ensuring accurate and contextually relevant generations.

Overall, we believe that these contributions advance the understanding and capabilities of modern IR systems based on pretrained transformers, laying a solid foundation for future research and practical applications in IR and NLP.

Chapter 2

Background

In this chapter, we will first introduce two common search paradigms: the traditional bag-of-words retrieval versus the approximate nearest neighbor (ANN) search for vector retrieval. We will then discuss different neural retrievers and rerankers based on pretrained transformers, from dense, sparse, and multi-vector retrievers to cross-encoder rerankers. We will also briefly discuss retrieval-augmented LLMs at the end of this Chapter.

2.1 Bag-of-Words Retrieval

Bag-of-words (BoW) models estimate the relevance of a document to a query primarily based on word overlap. This model represents text documents as vectors of word counts or term frequencies, disregarding the order of words and the syntactic structures within the text. The simplicity of the BoW model makes it reasonably effective for various tasks including document classification, information retrieval, and topic modeling.

One of the most effective bag-of-words models is BM25, which refines the traditional term frequency-inverse document frequency (TF-IDF) approach by incorporating document-specific adjustments. BM25 modifies term frequency to account for diminishing returns of term repetition and includes a document length normalization parameter to handle the bias towards longer documents. Robertson and Zaragoza [265] provide a comprehensive overview of BM25, explaining its theoretical underpinnings and practical implications. The formula for BM25 is given by:

$$\text{score}(q, d) = \sum_{i=1}^n \text{IDF}(q_i) \cdot \frac{f(q_i, d) \cdot (k_1 + 1)}{f(q_i, d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{\text{avgl}})}, \quad (2.1)$$

where $f(q_i, d)$ is the term frequency of q_i in the document d , $|d|$ represents the length of the document, and avgdl is the average document length across the text collection. k_1 and b are adjustable hyper-parameters.

In application, the document bag-of-words representations are usually organized in an inverted index for efficient search. An inverted index consists of two main components: a dictionary and postings lists. The dictionary contains all the unique terms that appear in the dataset, while the postings lists record each term’s occurrences across the documents. Each entry in a postings list typically includes the document identifier and can also store additional information like term frequency, position, or context, enhancing search capabilities through more nuanced ranking and retrieval strategies. Methods such as posting compression, tiered indexing, and partitioning are applied to further optimize the efficiency of inverted indexes.

2.2 Approximate Nearest Neighbor Search

Approximate Nearest Neighbor (ANN) search techniques facilitate efficient retrieval in large-scale databases by finding close neighbors to a query in high-dimensional spaces. The aim is to achieve a balance between retrieval accuracy and computational efficiency, which is crucial in handling vast datasets where exact nearest neighbor searches are computationally prohibitive.

Quantization. Quantization techniques reduce the dimensionality and the precision of the dataset, transforming continuous features into a finite set of discrete values or bins. One popular method is Product Quantization (PQ) [123], which divides the space into a Cartesian product of low-dimensional subspace and then quantize each subspace independently. PQ demonstrates its effectiveness in reducing memory usage and improving speed without significantly compromising the accuracy.

Hierarchical Navigable Small World (HNSW). HNSW [207] constructs a layered graph where each layer is a subset of the previous one, with the top layers having fewer, more distant nodes and the bottom layers having more, closer nodes. This structure allows efficient navigation through the graph by starting searches at higher layers and descending to lower layers for more refined searching.

Inverted File System (IVF). The Inverted File System (IVF) [121] is akin to the inverted index used in traditional text retrieval but adapted for vector space. It partitions the vector space into a finite number of Voronoi cells, each represented by a centroid. Vectors are assigned to their nearest centroid, and an inverted list of vectors is maintained for each cell. At query time, only a subset of these cells are searched, reducing the computational load. The IVF approach is often combined with other techniques such as PQ for enhanced performance.

While ANN methods have significantly improved the efficiency of nearest neighbor searches, they often involve trade-offs between accuracy, speed, and memory usage. Furthermore, the choice

of parameters, such as the number of partitions in IVF or layers in HNSW, can significantly affect performance. Ongoing research in this field aims to develop more adaptive and robust methods that can automatically tune these parameters based on dataset characteristics.

2.3 Neural Retrievers and Rerankers

2.3.1 Dense Retrievers

A dense passage retrieval model [113, 126] uses a bi-encoder structure where its query encoder $\eta_Q(\cdot)$ and document encoder $\eta_D(\cdot)$ are independent pretrained transformers that map the input text into a low-dimensional vector (e.g., 768 dimensions). Specifically, the similarity score s between the query q and document d can be defined as, for instance, the dot product between their encoded vectors $v_q = \eta_Q(q)$ and $v_d = \eta_D(d)$:

$$s(q, d) = v_q^T v_d. \quad (2.2)$$

To train the query encoder $\eta_Q(\cdot)$ and document encoder $\eta_D(\cdot)$, many loss functions are proposed with the same intuition that the similarity between the query q and the positive document d^+ should be the larger than any negative documents d^- in the text corpus. For example, a simplified version of the InfoNCE [318] loss for training the encoders is:

$$\mathcal{L}(q, d^+, d^- ; \eta_Q, \eta_D) = -\log \frac{\exp(s(q, d^+))}{\exp(s(q, d^+)) + \exp(s(q, d^-))}, \quad (2.3)$$

where d^- is a negative document sampled from the corpus. However, multiple negative documents can be sampled for the objective in Equation (2.3). During inference, the document encoder η_D is used to encode the whole corpus into vectors which can be further indexed for approximate nearest neighbor (ANN) search [121].

Compared to term-matching methods, dense retrievers circumvent the term mismatch problem by encoding the queries and documents into low-dimensional embeddings and using their dot product as the similarity score. However, dense retrieval is less robust on entity-heavy questions [282] and out-of-domain datasets [310], therefore calling for better solutions.

2.3.2 Supervised Sparse Retrievers

There's another class of single-vector retriever which instead of projects the inputs into low-dimensional dense vectors, these models maps the inputs to a high-dimensional lexical space, where each element represents a token in the lexicon. For example, SPLADE [67, 68] utilizes the

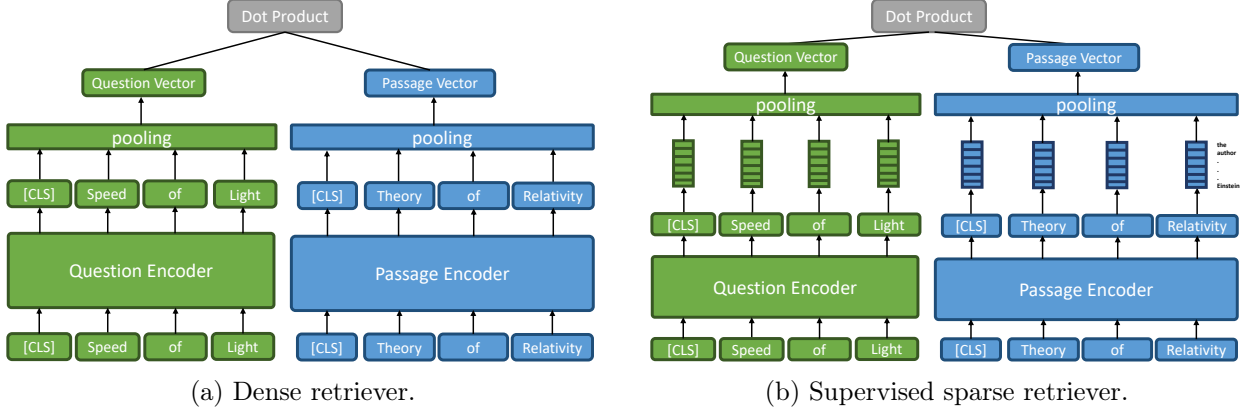


Figure 2.1: Dense retriever vs supervised sparse retriever. (a) Dense retriever projects the query and the document individually onto a low-dimensional dense space. (b) Supervised sparse retriever projects the query and the document onto a high-dimensional space (usually the vocabulary) where most dimensions are set to zero.

pretrained Mask Language Modeling (MLM) layer of BERT to project each c -dimensional token embedding to a high-dimensional, lexical space V . Each dimension corresponds to a token and has a non-negative value due to the following activation:

$$\phi_{d_j} = \log(1 + \text{ReLU}(W^T v_{d_j} + b)), \quad (2.4)$$

where $\phi_{d_j} \in \mathbb{R}^{|V|}$, v_{d_j} is the token embedding of the j th token of document d ; W and b are the weights and biases of the MLM layer. To compute the final similarity score, SPLADE pools all the token embeddings into a sequence-level representation and uses the dot product between a query and a document as the similarity:

$$s(q, d) = (\max_{i=1}^N \phi_{q_i})^T (\max_{j=1}^M \phi_{d_j}). \quad (2.5)$$

Here, max pooling is an element-wise operation and is feasible for sparse representations as the dimensions of all the token vectors are aligned (lexical space) and non-negative. The training objective is the same as Equation (2.3) but the similarity function $s(q, d)$ is switched to Equation (2.5). For inference, inverse index search is often applied by first storing the sparse representations as tokens with weights.

Sparse representations have several advantages. First, the sparse representation of documents requires less memory compared to dense models. Second, sparse vectors are more interpretable as each dimension in the embedding space corresponds to a meaningful concept or feature. However, the sparsity of SPLADE can result in the loss of fine-grained information present in dense represen-

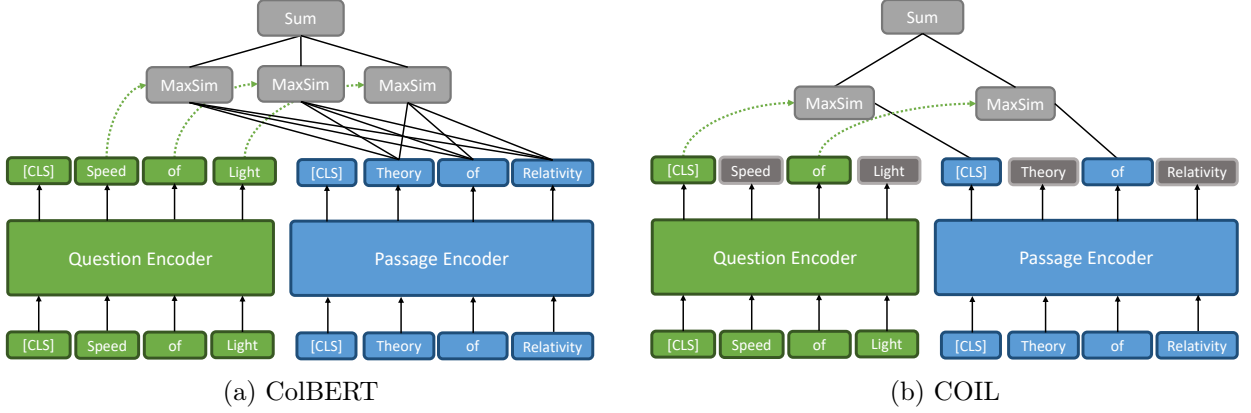


Figure 2.2: ColBERT vs COIL. (a) ColBERT considers token interactions between all query token vectors and the document token vectors. (b) COIL only considers token interactions that have overlaps between the query and the document at the input.

tations. Moreover, sparsity also directly affects both retrieval efficiency and effectiveness, making these two aspects more entangled and difficult to trade off in a post-hoc manner. Specifically, as the sparse representations are usually stored in a key-value fashion, the search latency increases as the number of non-zero entries increases and therefore sparser representations are favored in terms of latency. However, the information content stored in the sparse representations decreases as the number of non-zero entries decreases, which affects the retrieval accuracy adversely. Therefore, trade-offs between efficiency and effectiveness are often faced in sparse retrieval.

2.3.3 Multi-Vector Retrievers

ColBERT [128] proposes late interaction between the tokens in a query $q = \{q_1, q_2, \dots, q_N\}$ and tokens in a document $d = \{d_1, d_2, \dots, d_M\}$:

$$s(q, d) = \sum_{i=1}^N \max_j v_{q_i}^T v_{d_j}, \quad (2.6)$$

where v_{q_i} and v_{d_j} denotes the last-layer contextualized token embeddings of BERT. This is known as the MaxSim operation which compares each query token to all document tokens.

Another representative multi-vector approach known as COIL [72] proposes an exact match constraint on the MaxSim operation where only the embeddings with the same token id could interact with each other. Let $\mathcal{J}_i = \{j \mid d_j = q_i, 1 \leq j \leq M\}$ be the subset of document tokens

$\{d_j\}_{j=1}^M$ that have the same token ID as query token q_i , then we have:

$$s(q, d) = \sum_{i=1}^N \max_{j \in \mathcal{J}_i} v_{q_i}^T v_{d_j}, \quad (2.7)$$

It could be further combined with Equation (2.2) to improve the effectiveness if there’s no word overlap between the query and documents.

$$s(q, d) = v_q^T v_d + \sum_{i=1}^N \max_{j \in \mathcal{J}_i} v_{q_i}^T v_{d_j}. \quad (2.8)$$

Generally, the fine-grained token interaction helps the model capture nuanced relationships between the query and the document, therefore improving retrieval effectiveness in both in-domain and out-of-domain settings. However, due to its complexity, multi-vector retrieval requires more engineering efforts and low-level optimization [276, 277] for time and space efficiency. Unlike dense or sparse retrieval, this feature also makes it difficult to fit into off-the-shelf search libraries.

2.3.4 Cross-Encoder Rerankers

The reranker module is responsible for improving the ranking quality of the candidate documents returned from the first-stage retrievers. We focus on recent neural rerankers based on pretrained language models such as MonoBERT [230] and MonoELECTRA. The reranker could also be seen as a logical scoring model. However, instead of using a bi-encoder structure, a cross-encoder structure is often applied, where the question-document pairs are encoded and fed into a single model together for more fine-grained token-level interactions:

$$s_r(q, d) \doteq \zeta(\text{concat}(q, d)), \quad (2.9)$$

where ζ is the reranker that takes the question-document pair as input and outputs the similarity score s_r . One way to implement the “concat” function is using special tokens as indicators, such as [CLS] q [SEP] d [SEP].

To sum up, all the above neural retrieval models have their pros and cons, but it is not hard to see that by increasing the redundancy, retrievers can be more effective but also less efficient (see the comparison between dense retrievers and multi-vector retrievers).

2.4 Retrieval-Augmented Language Models

Retrieval-augmented language models involve external knowledge sources to improve the performance of language models on knowledge-intensive tasks. DrQA [41] combines extractive models and independent retrievers for open-domain question-answering. Follow-up works on retrieval-augmentation such as REALM [85], RAG [154], and Atlas [115] further combine the retrieval component in pretraining and finetuning for downstream knowledge-intensive tasks. [13] further divide them into three categories:

Input augmentation. REPLUG [289] and in-context RALM [259] propose to pre-pend the retrieved passages in the prompts for zero-shot factual support. Recently, Self-RAG [12] leverages special tokens to perform adaptive retrieval and different critics to iteratively refine the RALM’s output. RA-DIT [178] retrofits LLMs with retrieval capabilities via instruction finetuning.

Intermediate fusion. RETRO [31] employs a novel attention mechanism to incorporate multiple pre-processed text fragments in intermediate layers for more efficient integration of retrieved results. This approach has been successfully applied to larger decoder-only language models as demonstrated by RETRO++ [337] and InstructRetro [336]. FiD [114] applies similar an encoder-decoder structure in a zero-shot manner and achieves better performance at a document level.

Output integration. k NN-LM [127] pioneers this direction and proposes to interpolate the retrieval distribution and LM’s prediction. Follow-up works further propose adaptive interpolation methods which involve training [88, 26] and excessive tuning [62]. Another line of work proposes to jointly train the phrase encoder and LM to expand the vocabulary dynamically using the retrieved phrases, such as Copy-Generator [142] and its follow-up work [38].

Chapter 3

Efficient Dense Retrieval via Index Compression

3.1 Overview

Dense passage retrieval [126] improves end-to-end retrieval accuracy in open-domain question answering (QA) by representing queries and documents in a low-dimensional, dense vector space. However, the vastly increased space and memory demands for storing and loading the dense vectors call for effective compression methods [116]. For example, the size of the original DPR vector (flat) index on the Wikipedia corpus is about 61 GB, while its sparse counterpart—BM25 inverted index—only uses 2.4 GB. The staggering increase of around $25\times$ in space requirements only yields an average gain of 2.5% in top-100 accuracy, indicating potential redundancy in the dense representations.

In this Chapter, we quantify redundancy within the dense vectors encoded by the DPR model using explained variance ratio and mutual information. Figure 3.1 shows that the original 768 dimensions is unnecessarily large as both the mutual information and explained variance ratio start to plateau at around 256 dimensions. Based on this observation, we further propose a simple yet effective pipeline for dense retrieval compression that includes principal component analysis (PCA), product quantization (PQ), and hybrid search to reduce index size while retaining effectiveness.

We also compare other compression options, including supervised dimensionality reduction, where we finetune a linear projection layer on top of the pretrained DPR model using relevance labels. Surprisingly, we find that PCA achieves top-100 retrieval accuracy that is better than the two supervised dimensionality reduction techniques for 256 and 128 dimensions, while supervised techniques outperform (unsupervised) PCA for 64 dimensions. Our techniques support different

accuracy–space trade-offs, but one sweet spot manages to compress the dense vectors by $96\times$ with less than 4% drop in top-100 accuracy on average across five standard QA datasets. Finally, we incorporate our pipeline with the BM25 inverted index for sparse–dense hybrid search, where we can achieve $16\times$ compression without any accuracy drop compared to the original DPR results.

3.2 Compressing Dense Representations

As mentioned above, DPR’s encoders produce dense vectors of 768 dimensions by default, which we will show is unnecessarily large below. In this section, we discuss how to quantify the redundancy in the encoded vectors and how to improve DPR’s space efficiency by reducing this redundancy.

3.2.1 Quantifying Redundancy

We use two metrics to quantify the redundancy in dense vectors: *explained variance ratio* of the principal components and *mutual information* between the question vectors and passage vectors. The explained variance ratio of PCA is defined as:

$$\frac{\sum_{i=1}^m \sigma_i^2}{\sum_{i=1}^n \sigma_i^2}, \quad (3.1)$$

where σ_i^2 is the variance corresponding to the i^{th} largest eigenvalue, n is the original dimension, and m is the reduced dimension. This ratio tells us how much variance is retained by preserving the first m eigenvectors of the dense representations. Another way to evaluate redundancy is using the mutual information between the dense representation of questions Q and the retrieved passages D , which can be approximated using the classifier in Eq. (2.3):

$$\begin{aligned} \mathcal{I}(Q; D) &\approx \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{p(D|Q=q_i)} [\ln p(D | Q = q_i)] \\ &\quad - \mathbb{E}_{p(D)} \left[\ln \frac{1}{N} \sum_{i=1}^N p(D | Q = q_i) \right], \end{aligned} \quad (3.2)$$

where $\{q_i\}_{i=1}^N$ is the training/dev/test questions. This quantity is upper-bounded by $\ln N$ and the normalized mutual information is $\mathcal{I}(Q; D)/\ln N$.

Figure 3.1 shows the explained variance and mutual information of compressed vectors with different dimensions reduced by PCA. As we can see, $\sim 90\%$ variance and $\sim 99\%$ mutual information is held by the first ~ 200 dimensions. However, as the dimension further decreases, useful

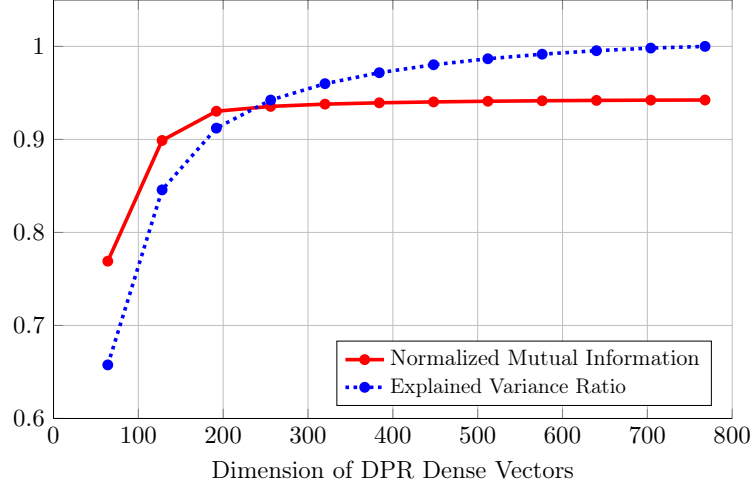


Figure 3.1: Quantifying redundancy in dense vectors using explained variance ratio and normalized mutual information. The y-axis is ratio and does not have a unit.

information is discarded at a higher rate and the dense representation starts to degrade visibly. The figure illustrates that a dimension of ~ 200 could be a sweet spot in accuracy–space trade-offs. We also find in later experiments that a dimension of 256 indeed achieves the best balance among other choices as shown in Figure 3.2, which will be discussed in Section 3.4.3.

3.2.2 Dense Vector Compression

We explore three different types of dimensionality reduction techniques given dense representations from a pretrained DPR model:

Supervised Approach We apply a linear transformation to the pretrained dense vector representations, and finetune this linear layer with relevance labels according to Eq. (2.3). Independent linear layers W_q and W_p are added to the question encoder and passage encoder, respectively. To make this compression technique a plug-and-play component, we only optimize the linear layer while freezing the rest of the networks. In addition, we add an orthogonality regularization term $\|W_p W_q^\top - I\|_2$ to the original loss function in Eq. (2.3), where I is the identity matrix. Such regularization encourages the weights in W_p and W_q to be orthogonal while retaining the most information in the original dense vectors.

Unsupervised Approach A popular technique, principle component analysis (PCA), can effectively reduce the dimensionality of high dimensional vectors while retaining most of the variance

within the original representation. We fit a linear PCA transformation using the combination of all question and passage vectors to learn a compressed representation based on a pretrained DPR model. During inference, the same transformation is applied to each question and the relevance score is the dot product between the compressed question and passage vectors.

Product Quantization On top of the supervised and unsupervised dimensionality reduction techniques described above, we further leverage product quantization (PQ), which decomposes the original d -dimensional vector into s sub-vectors. Each sub-vector is quantized using k -means and eventually stored with t bits [120]. For example, the original 768 dimension dense vector occupies 768×32 bits. By dividing it into 192 sub-vectors of 8 bits, the storage space becomes 192×8 bits, which is $1/16$ of the original size. On average, space is reduced from 32 bits to 2 bits per dimension.

3.3 Experimental Setup

Datasets and Metrics We evaluate the top- k retrieval accuracy of our compression methods on five QA datasets examined in the original DPR paper [126]: NQ, TriviaQA, WQ, CuratedTREC, and SQuAD. The top- k retrieval accuracy is defined as the fraction of questions that have at least one correct answer span in the top- k retrieved passages. Following previous work, we use $k \in \{20, 100\}$. We use the combination of NQ, TriviaQA, WQ, and CuratedTREC to train our models, following the same setting as DPR.

Model Training For DPR, instead of the original Facebook implementation, we use the implementation of [76], which takes advantage of gradient caching to save GPU memory usage and mixed precision training to speed up the learning process. We find that using a learning rate of 10^{-6} and training the model for 40 epochs achieve better effectiveness than the default DPR setting. We refer to the original DPR model, which has 768 dimensional output vectors, as DPR-768. Other hyperparameters are identical to default DPR [126].

For the compression methods, we consider the reduced dimensions $d \in \{256, 128, 64\}$ according to Figure 3.1. For the supervised approach, the linear layer is trained for one epoch with a learning rate of 10^{-3} while freezing the backbone DPR model (i.e., BERT), and we refer to these models as Linear- d . For comparison, we train models with identical architecture to Linear- d , but without freezing the BERT model, and refer to them as DPR- d .

For the unsupervised PCA approach, we fit the PCA transformation with question and passage embeddings produced by the original DPR-768 model. The question embeddings are from the original embeddings of questions in the training set. A total of 160k passage embeddings are randomly sampled from the passage embeddings of the entire corpus (i.e., original dense index).

Method	NQ		TriviaQA		WQ		CuratedTREC		SQuAD	
	top-20	top-100	top-20	top-100	top-20	top-100	top-20	top-100	top-20	top-100
DPR-768	1× compression		size: 61 GB		latency: 7570 ms					
	79.4	87.0	78.5	84.5	75.3	83.0	88.2	94.4	58.3	72.4
PCA-256	3× compression		size: 21 GB		latency: 2540 ms					
	77.2	85.5	76.5	83.4	73.5	81.9	87.3	93.8	53.7	69.2
Linear-256	76.2	85.2	75.6	83.0	71.9	81.3	86.3	93.7	51.0	67.1
DPR-256	71.8	84.1	72.1	81.2	71.8	80.6	86.6	91.8	47.2	63.3
PCA-128	6× compression		size: 11 GB		latency: 1130 ms					
	75.3	84.3	74.9	82.7	72.4	81.3	86.3	93.5	51.4	67.1
Linear-128	74.7	84.2	74.4	82.7	71.3	80.6	86.0	93.2	48.7	65.4
DPR-128	72.0	82.6	69.9	79.9	68.5	78.9	83.6	91.5	44.4	60.9
PCA-64	12× compression		size: 5.1 GB		latency: 625 ms					
	63.6	77.2	66.7	78.4	65.5	76.8	81.1	89.8	42.2	59.8
Linear-64	69.7	81.2	69.3	80.0	66.3	77.8	83.4	91.8	42.4	59.9
DPR-64	68.3	80.4	65.7	77.6	66.2	77.5	82.1	91.1	39.5	56.6
BM25	62.9	78.3	76.4	83.2	62.4	75.5	80.7	89.9	71.1	81.8

Table 3.1: Top-{20, 100} retrieval accuracy of different dimensionality reduction methods evaluated at $d = \{256, 128, 64\}$ on five benchmark QA datasets. The BM25 index size is about 2.4 GB.

We utilize the PQ features from Faiss [61]. The number of codewords for each sub-vector is fixed at 256 (i.e., using 8 bits). Following [116], we change the number of sub-vectors for dense embeddings such that the average memory of each dimension is reduced. Combined with dimensionality reduction methods, we decrease the occupied space of each dimension from 32 bits to 1 or 2 bits.

Sparse–Dense Hybrid Search [196] have shown that hybrid search significantly improves the retrieval accuracy of DPR. Therefore, we fuse the DPR’s and BM25’s retrieval results following their same rule, where the final hybrid score is calculated by $\text{score}_{\text{dense}} + \alpha \cdot \text{score}_{\text{sparse}}$. We set $\alpha = 1$, which equally weights dense and sparse scores for all hybrid search cases as it is a good default option for hybrid search in general.

3.4 Results

In this section, we characterize the trade-offs between retrieval accuracy and space requirements using different combinations of our proposed techniques. All experiments are implemented using the Pyserini IR toolkit [167].

Method	Comp.	Size	Latency	NQ		TriviaQA		WQ		CuratedTREC		SQuAD	
				top-20	top-100	top-20	top-100	top-20	top-100	top-20	top-100	top-20	top-100
DPR-768	1×	61 G	7570ms	79.4	87.0	78.5	84.5	75.3	83.0	88.2	94.4	58.3	72.4
+ PQ2	16×	3.8 G	2360ms	77.9	86.3	77.0	84.2	73.4	82.4	87.9	93.9	56.1	70.9
+ PQ1	32×	1.9 G	1080ms	73.7	83.5	74.7	82.7	71.1	81.0	86.3	92.8	52.4	68.4
PCA-256	3×	21 G	2540ms	77.2	85.5	76.5	83.4	73.5	81.9	87.3	93.8	53.7	69.2
+ PQ2	48×	1.3 G	765ms	74.8	84.1	74.5	82.6	72.2	81.0	88.2	92.7	51.7	67.5
+ PQ1	96×	642 M	382ms	67.6	79.4	68.0	79.2	66.5	78.7	82.7	90.1	45.1	61.4
PCA-128	6×	11 G	1130ms	75.3	84.3	74.9	82.7	72.4	81.3	86.3	93.5	51.4	67.1
+ PQ2	96×	642 M	416ms	72.3	82.9	72.2	81.7	71.0	80.1	85.3	91.8	48.5	64.9
+ PQ1	192×	321 M	266ms	62.6	76.4	64.3	77.2	64.8	77.1	80.0	89.2	41.4	58.7
Linear-64	12×	5.1 G	625ms	69.7	81.2	69.3	80.0	66.3	77.8	83.4	91.8	42.4	59.9
+ PQ2	192×	321 M	249ms	62.5	77.2	65.9	78.0	63.7	75.8	79.1	90.3	39.7	57.9
+ PQ1	384×	161 M	200ms	44.0	64.2	51.7	70.0	50.0	68.2	71.8	85.2	30.7	50.0

Table 3.2: Top-{20, 100} retrieval accuracy of three dimensionality reduction methods with different PQ settings at $d = \{256, 128, 64\}$.

3.4.1 Dimensionality Reduction

Table 3.1 shows the top-{20, 100} accuracy of the three dimensionality reduction techniques presented in Section 3.2.2 evaluated at $d = \{256, 128, 64\}$ on five benchmark QA datasets. DPR-768 achieves the best accuracy, which serves as the upper bound for compression. Overall, the top-100 accuracy decreases as we reduce the dense vectors to fewer dimensions. However, we see that PCA-256 only has a 0.6 ~ 1.5% drop in accuracy on {NQ, TriviaQA, WQ, CuratedTREC} while achieving 3× compression. Model quality degrades more on SQuAD, since our models are trained on the combination of the other four datasets.

We further evaluate retrieval latency for different dimensionality levels in Table 3.1. Although latency is not the focus of this work, it is still worth noticing that dimensionality reduction also reduces retrieval latency as it speeds up dot-product calculations. For example, reducing dimensionality from 768 to 256 can reduce retrieval latency by three times. The latency is measured by query encoding time + brute-force retrieval time using a machine with Intel Xeon Platinum 8160 2.10GHz CPU using Faiss FlatIP indexes. Both query encoding and retrieval are performed with a single CPU thread.

Across different dimensions, the unsupervised PCA method often works the best in terms of the trade-off between accuracy and compression rate. It is surprising that unsupervised PCA outperforms the other two supervised methods at dimensions 256 and 128. Although the supervised methods might be further improved with carefully-tuned hyperparameters, training DPR can be computationally expensive, while PCA is the more robust and economical method to achieve comparable results. Another surprising finding is that Linear- d generally outperforms DPR- d , which means that freezing the backbone DPR model and finetuning only the linear layer seem to work better than training the entire model end to end to generate compressed representations.

However, this finding may be simply due to poor hyperparameter selection.

3.4.2 Product Quantization

Product quantization (PQ) can often aggressively reduce the index size while largely preserving retrieval effectiveness. For example, PQ2 (meaning each dimension occupies 2 bits on average after PQ) reduces the size of DPR-768’s vectors by $16\times$ with only 0.7% loss in top-100 retrieval accuracy on average. Table 3.2 shows the top- $\{20, 100\}$ retrieval accuracy of our dimensionality reduction methods with different PQ settings. Combined with product quantization, the dimensionality reduction methods achieve a much higher compression rate with, in some cases, only a modest loss in retrieval accuracy. In addition, we find that PQ2 outperforms PQ1 on most datasets and dimensions, as PQ1 suffers more than twice the accuracy loss compared to PQ2. It appears that compression to 1 bit per dimension is too aggressive and represents a poor trade-off. If we restrict the retrieval accuracy drop to within 4% on average, we can compress the dense vectors by up to $96\times$, reducing the original DPR index from 61 GB to mere hundreds of MB on the Wikipedia corpus.

3.4.3 Hybrid Search

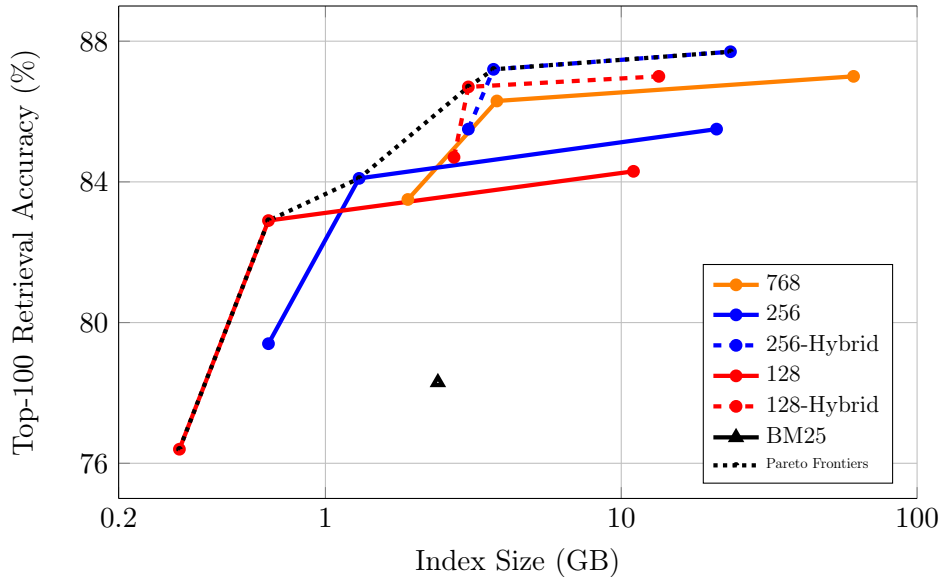


Figure 3.2: Trade-offs between retrieval accuracy and index size using different combination of our compression techniques.

Figure 3.2 shows the trade-off between retrieval accuracy and index size with different combinations of dimensionality reduction, product quantization, and hybrid search. On each curve, the points from left to right represent PQ1, PQ2, and w/o PQ, respectively. Sparse retrieval with the BM25 inverted index is shown as the black triangle. The dashed lines represent sparse–dense hybrid retrieval; these lines include the size of the BM25 inverted index. In the plot, up and to the left represents better: higher accuracy and smaller indexes.

As an example, the pipeline consisting of PCA-256, PQ2, and HS reduces the (total) index size from 61 GB to 3.7 GB (57 GB or roughly $16\times$ smaller) and even yields 0.2% gain in top-100 accuracy compared to DPR-768. The dotted black line in Figure 3.2 shows the Pareto Frontier, which can be understood as the best achievable accuracy for a particular restriction on index size. Overall, we see that the DPR-768 (orange) line does *not* lie on the frontier, which means that some combination of our techniques is strictly more accurate and smaller than the original DPR representations.

3.5 Summary

In this Chapter, we analyze the redundancy within dense representations from DPR, a popular dense retrieval model. We use explained variance ratio and normalized mutual information to quantify the redundancy in the dense representations and found that over 90% of the information is kept in the first 256 principal components. We propose a simple yet effective compression pipeline that comprises of principal component analysis (PCA), product quantization, and hybrid search. It enables flexible trade-offs between space and accuracy which drastically reduces index size while preserving end-to-end retrieval accuracy at reasonable levels. We perform extensive experiments on five question-answering datasets and demonstrate that our best pipeline achieves good accuracy–space trade-offs, for example, $48\times$ compression with less than 3% drop in top-100 retrieval accuracy on average or $96\times$ compression with less than 4% drop.

Chapter 4

Robust Dense Retrieval via Model Uncertainty Fusion

4.1 Overview

Open-domain question-answering requires finding answers to given questions from a large collection of documents [326]. Therefore, a first-stage retrieval component that selects a set of potentially answer-containing documents is often involved for the second-stage reading comprehension model [41]. Traditional term-matching methods such as TF-IDF and BM25 [267, 167] that leverage an inverted index to construct sparse textual representations have built strong baselines in the first-stage retrieval.

Recently, neural-based dense retrievers [284, 147, 85, 126] have been shown to achieve better performance in open-domain question-answering, but they often fail to generalize outside of the training data distribution. A standard solution known as *joint training* that learns a single dense retriever on the union of different datasets [204, 350] provides a solution to a certain extent. However, [126] has shown that data from different tasks might have conflicts with each other, where joint training on their union can cause performance degradation. For example, SQuAD [258] only focuses on a small set of Wikipedia documents while datasets like NQ [138] and Trivia [122] cover more entries. Therefore, careful data re-balancing and hyperparameter search are required during training.

In this Chapter, we propose another solution to multi-task learning, which trains multiple DPR experts on different datasets separately and their predictions are aggregated during test time. This is also known as *model fusion* [97] which differs from a *mixture of experts* [285] as it does not need to learn a gating function on the joint dataset. The model fusion method makes it easier to incorporate new data for continual learning without introducing conflicts, as each expert

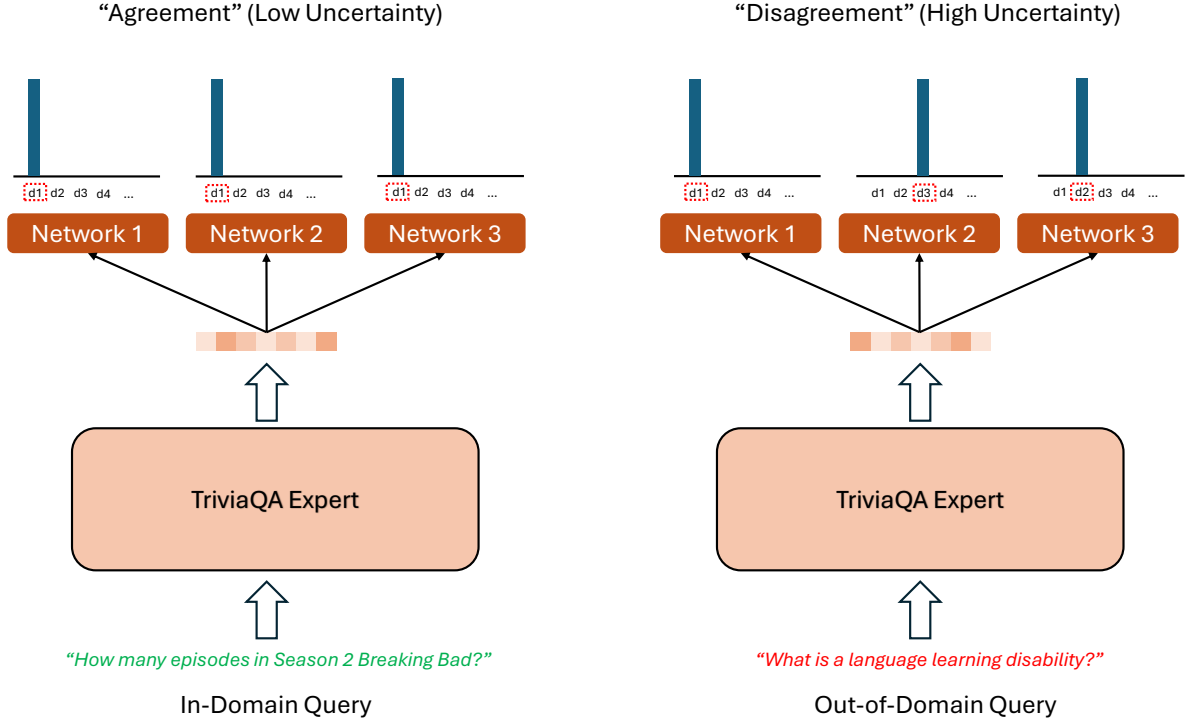


Figure 4.1: An example of ensemble uncertainty on two examples with the TriviaQA expert model. For the in-domain query, the ensemble of neural networks will tend to agree on the same document prediction, while the ensemble prediction diverges on out-of-domain query as the neural networks are encouraged to converge on different local optima.

trains on independent tasks. In addition, these experts can be trained in parallel to speed up the learning process.

However, the challenge now becomes how to aggregate different expert’s predictions without hurting their in-distribution performance. We propose model uncertainty estimation [186] as a dynamic weighting scheme, which helps the expert to identify whether a question belongs to its expertise. Intuitively, a model that overfits to a training distribution should be more uncertain about the out-of-domain data than the in-domain data. For example, the question *"How many episodes in Season 2 Breaking Bad?"* might get a high uncertainty score from an expert trained on a medical QA dataset.

In practice, we leverage ensemble uncertainty where we train an ensemble of small neural networks for each pretrained DPR expert [140]. Specifically, we represent the model uncertainty as the mutual information [246] between the ensemble’s predictions and parameters. For each question, we retrieve a set of the top- k documents using different DPR experts and then use its

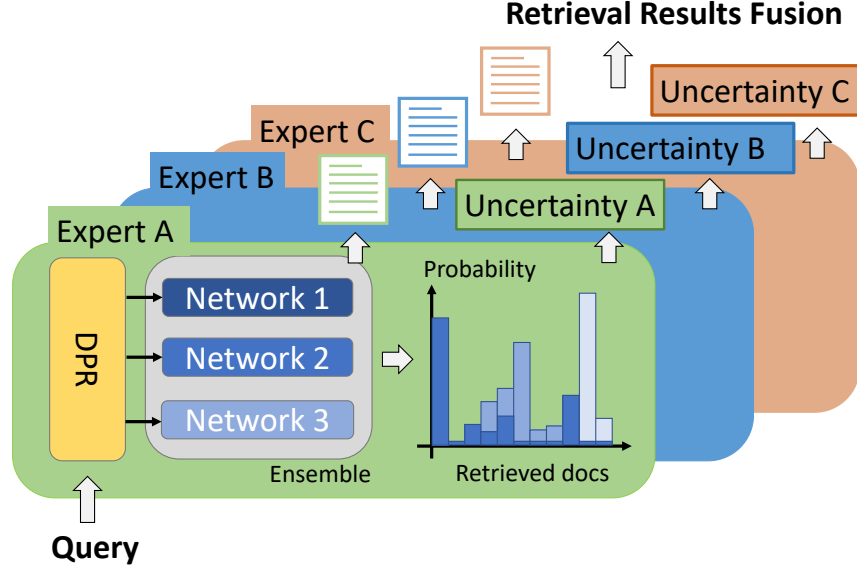


Figure 4.2: An illustration of model uncertainty fusion of 3 DPR experts, each with an ensemble of 3 fully connected neural networks. Given a query, each DPR expert first retrieves top- k documents, followed by the uncertainty estimation using the corresponding ensemble. The weighted sum of predictions is then used to rerank the union of the retrieved documents.

corresponding ensemble to compute the uncertainty score of the question. Finally, we aggregate all the expert’s predictions into a normalized weighted sum and rerank the retrieved documents. Fig. 4.2 demonstrates a simplified pipeline of our algorithm.

Extensive experiments show that our final fusion model not only outperforms individual specialists on 5 open-domain QA datasets but also outperforms the performance of the joint-training, multi-task DPR model, with up to 10% improvement in top-100 accuracy on SQuAD. Finally, our method manages to handle corpus conflicts on a mixed subset of different QA tasks, which even outperforms an oracle model using Bayesian optimization [69].

4.2 Multi-Task Model Fusion

Given m question-answering tasks and m independent experts, the goal of multi-task model fusion is to find the optimal set of weights $\{w^{(i)}\}_{i=1}^m$ to combine all experts’ predictions for each question. We use DPR as the expert model.

4.2.1 Ensemble Uncertainty Estimation

There are mainly two types of uncertainty: model uncertainty and data uncertainty [206]. Data uncertainty is often caused by mislabelling or missing features, while model uncertainty measures the confidence of the model’s predictions given the training data, which is often used to identify whether a sample is within the training domain. Therefore, we use model uncertainty for weighting the experts’ predictions, such that we know whether a question belongs to an expert’s expertise.

There are many ways to represent model uncertainty. In this work, we consider ensemble uncertainty due to its effectiveness and simplicity. The intuition is simple: the ensemble trained in a single domain will “agree” to similar predictions if in-domain samples occur, and will “disagree” otherwise. The disagreement would be more obvious if the functional space of the ensemble is complex enough, e.g., the space of neural networks, which is illustrated in Figure 4.1. To quantify such uncertainty or “disagreement”, we use *Mutual Information* (MI) [246] between the ensemble’s prediction and its parameters as the proxy. For each DPR expert, we build an ensemble of m classifiers $\{p(D \mid \Theta = \theta_i, Q = q, \mathcal{C} = \mathcal{C}_{\text{DPR}})\}_{i=1}^m$, where Θ denotes the r.v. of the ensemble parameters, θ_i denotes the parameters of the i^{th} ensemble member, and $\mathcal{C}_{\text{DPR}}$ denotes the collection of contexts retrieved by the DPR expert. For simplicity, we will use $p(D \mid \theta_i, q, \mathcal{C}_{\text{DPR}})$ as a shorthand for the distribution. The mutual information $\mathcal{I}$ between D and Θ given a question q and a collection of contexts $\mathcal{C}_{\text{DPR}}$ is:

$$\begin{aligned}
& \mathcal{I}(D; \Theta \mid q, \mathcal{C}_{\text{DPR}}) \\
&= \mathcal{H}(D \mid q, \mathcal{C}_{\text{DPR}}) - \mathcal{H}(D \mid \Theta, q, \mathcal{C}_{\text{DPR}}) \\
&= \mathbb{E}_{p(D \mid \Theta, q, \mathcal{C}_{\text{DPR}})}[\ln p(D \mid \Theta, q, \mathcal{C}_{\text{DPR}})] - \mathbb{E}_{p(D \mid q, \mathcal{C}_{\text{DPR}})}[\ln p(D \mid q, \mathcal{C}_{\text{DPR}})] \\
&\approx \frac{1}{m} \sum_{i=1}^m \mathbb{E}_{p(D \mid \theta_i, q, \mathcal{C}_{\text{DPR}})}[\ln p(D \mid \theta_i, q, \mathcal{C}_{\text{DPR}})] - \mathbb{E}_{p(D \mid q, \mathcal{C}_{\text{DPR}})} \left[\ln \left(\frac{1}{m} \sum_{i=1}^m p(D \mid \theta_i, q, \mathcal{C}_{\text{DPR}}) \right) \right], \tag{4.1}
\end{aligned}$$

where $\mathcal{H}(\cdot)$ denotes the entropy operator, $\mathbb{E}[\cdot]$ denotes the expectation operator, and the approximations are done by Monte-Carlo simulation. The approximated mutual information is also upper-bounded by the log of the number of ensemble members m :

$$\mathcal{I}(D; \Theta \mid q, \mathcal{C}_{\text{DPR}}) \leq \ln m,$$

which gives us bounded uncertainty estimation for a certain domain. We normalize the mutual information and transform it into a confidence score w for weighting the DPR expert’s prediction

of q :

$$w = 1 - \frac{\mathcal{I}(D; \Theta \mid q, \mathcal{C}_{\text{DPR}})}{\ln m}. \quad (4.2)$$

4.2.2 Model Uncertainty Fusion

Given m DPR experts that are trained on m different tasks separately, we first encode the question and document representation into dense vectors $\{v_q^{(i)}\}_{i=1}^m$ and $\{v_d^{(i)}\}_{i=1}^m$ for all m experts, where the superscript represents the expert’s id. We then build an ensemble of small neural networks, each uses the corresponding expert’s $v_q^{(i)}$ as input and outputs another vector $u_q^{(i)}$ of the same dimension. We finally optimize the same objective function w.r.t each ensemble member $u_q^{(i)}$ and $v_d^{(i)}$ for each question-answering task.

During inference, given a new question q , we first retrieve m sets of top- k documents $\{C_{\text{DPR}}^{(i)}\}_{i=1}^m = \{d_1^{(i)}, d_2^{(i)}, \dots, d_k^{(i)}\}_{i=1}^m$ using all m DPR expert’s question vectors $\{v_q^{(i)}\}_{i=1}^m$ and document vectors $\{v_{d_1}^{(i)}, v_{d_2}^{(i)}, \dots, v_{d_k}^{(i)}\}_{i=1}^m$. We then calculate the weights $\{w^{(i)}\}_{i=1}^m$ for each question according to Eq. (4.2) using the ensemble vectors $\{u_q^{(i)}\}_{i=1}^m$ and $\{C_{\text{DPR}}^{(i)}\}_{i=1}^m$ for question q . Finally, we re-rank the union of m sets of top- k documents using the uncertainty-weighted sum of each expert’s score. The final score $\mathcal{S}(q, d_j)$ of a document d_j given question q is:

$$\mathcal{S}(q, d_j) = w^{(1)} s_j^{(1)} + w^{(2)} s_j^{(2)} \dots + w^{(m)} s_j^{(m)}, \quad (4.3)$$

where $s_j^{(i)} = v_q^{(i)T} v_{d_j}^{(i)}$ according to Eq. (2.2). If we do not have a score from the i^{th} expert for a document d , we will use the minimum of $\{s_j^{(i)}\}_{j=1}^k$ as the ranking score for d . Fig. 4.2 visualizes the aforementioned retrieval fusion process of our method during inference.

4.2.3 Uncertainty Calibration

Despite its simplicity and effectiveness, one drawback of ensemble uncertainty is that it doesn’t have a closed-form expression and the prediction of each ensemble might have a different range [243]. Therefore, the ensemble uncertainty needs to be calibrated before fusion, such that the confidence of an expert matches its prediction accuracy. We use the Expected Calibration Error (ECE) [80] as the metric where we search for the best inverse temperature on the dev sets for each expert to minimize the ECE. As ECE mainly uses the term “confidence” which is the w in Eq. (4.2), we switch to “confidence” instead of “uncertainty” in the following. We partition the samples in the dev set into T equally-spaced bins and take the weighted average of the confidence-accuracy

Retriever	Top-20					Top-100				
	NQ	Trivia	WQ	TREC	SQuAD	NQ	Trivia	WQ	TREC	SQuAD
DPR-Single-domain	79.1	78.9	71.0	85.1	62.1	85.9	84.5	80.2	92.2	76.8
DPR-Single-worst	46.3	58.0	57.9	75.8	41.4	61.9	71.4	74.1	86.5	59.1
DPR-Multi (w/o SQuAD)	79.5	78.9	75.0	88.8	52.0	86.1	84.8	83.0	93.4	67.7
DPR-MUF (w/o SQuAD)	79.8	78.2	76.2	89.3	57.7	86.5	84.4	83.9	94.7	72.0
DPR-MUF (w/o domain)	68.9	74.1	73.5	89.6	57.7	79.4	82.1	82.5	94.4	72.0
DPR-MUF	79.5	78.6	75.9	90.2	64.6	86.4	84.7	84.0	95.0	78.3

Table 4.1: Top-20 and Top-100 retrieval accuracy (%) on benchmark QA test sets. Each score represents the percentage of top 20/100 retrieved passages that contain answers. All methods containing ‘‘DPR-MUF’’ stand for our Model Uncertainty Fusion (MUF) method. See Section 4.4.1 for details of different models.

difference:

$$\text{ECE} = \sum_{i=1}^T \frac{|B_i|}{N} |\text{conf}(B_i) - \text{acc}(B_i)|, \quad (4.4)$$

where B_i is the i^{th} bin and N is the number of samples. Functions $\text{conf}(B_i)$ and $\text{acc}(B_i)$ are the average confidence and top-1 accuracy within the i^{th} bin, respectively. Each confidence score is computed by an ensemble according to Eq. (4.2).

4.3 Experimental Setup

We follow the DPR paper [126] to train and evaluate our dense retrievers. We replicate their results on all benchmark datasets, with a maximum score difference between ours and their numbers of 1%. This work only focuses on retrieval accuracy as we only improve the retriever. We finally perform sensitivity analysis for the ensemble and uncertainty visualization of our fusion approach.

Datasets We train individual DPR models on 5 standard benchmark QA tasks: Natural Questions (NQ) [138], TriviaQA (Trivia) [122], WebQuestions (WQ) [25], CuratedTREC (TREC) [20], SQuAD-1.1 (SQuAD) [258]. We evaluate the retriever models on the test sets of the aforementioned datasets, as well as their random mixes to test the out-of-distribution performance.

For retrieval, we chunk the Wikipedia collections [85] into passages of 100 words as in [349], which yields about 21 million samples in total. We follow [126] using BM25 [267, 167] to select the positive and negative passages.

Models and Training We first train independent DPR models on the training set of NQ, TriviaQA, WQ, CuratedTREC, and SQuAD-1.1 separately following [126]. We then encode the training sets into dense vectors as the input to the ensemble. We train an ensemble of 20, 2-layer fully connected neural networks with 512 units for 100 epochs. We optimize the objective function in Eq. (2.3) with learning rate of 2e-05 using Adam [131]. We use different sub-batches and weight initialization to train each ensemble member to encourage diversity. The rest of the hyperparameter setting remains the same as described in [126].

Inference: Retrieval Fusion Given a question q during inference, a set of top- k documents is first retrieved by each DPR expert. For each expert, we use the corresponding ensemble to predict the expert’s retrieved documents and obtain a collection of dot-product scores. We then apply softmax activation on the dot-products, yielding a collection of distributions over the retrieved documents. We calculate the normalized mutual information between the ensemble’s predictions and the parameters as in Eq. (4.1) and Eq. (4.2), using it as the weight for the expert as described in Eq. (4.3) given the question.

In addition, we calibrate each ensemble’s uncertainty prediction individually using the expected calibration error (ECE) [80] according to Eq. (4.4), as each ensemble from different domains might have a different range of uncertainty. We find the lowest ECE score is achieved with the inverse temperature λ of the softmax activation setting to be 1e-3. Finally, we normalize the calibrated uncertainty and re-rank the union of retrieved documents using the uncertainty-weighted sum of experts’ scores. For documents that do not have all experts’ scores, we use the minimum of the missing expert’s prediction as the ranking score.

4.4 Results and Analysis

4.4.1 Benchmark Dataset Retrieval

Tbl. 4.1 shows retrieval performance using different types of DPR models on 5 benchmark datasets. We briefly describe each configuration below.

DPR-Single-domain: A single DPR model trained and tested on the same domain.

DPR-Single-worst: A single DPR model trained on one domain and transferred to the target test set in zero-shot and has the worst performance among all experts.

DPR-Multi (w/o SQuAD): A multi-task DPR model trained on the joint dataset of {NQ, Trivia, WQ, and Trec} without the SQuAD dataset, as implemented in [126].

DPR-MUF: Our model uncertainty fusion method using experts from {NQ, Trivia, WQ, Trec, and SQuAD}, which is our main approach.

DPR-MUF (w/o SQuAD): Our model uncertainty fusion method using experts from {NQ, Trivia, WQ, and Trec} without SQuAD to align with DPR-Multi (w/o SQuAD).

DPR-MUF (w/o domain): Our model uncertainty fusion method using all experts except DPR-Single-domain, to investigate out-of-domain generalization.

We can see from Tbl. 4.1 that our model uncertainty fusion method (DPR-MUF) achieves the best performance on almost all benchmark QA datasets except Trivia, in terms of top-20/100 accuracy. The original multi-task DPR model does not include SQuAD for joint training as “SQuAD is limited to a small set of Wikipedia documents and thus introduces unwanted bias” [126]. In comparison, our DPR-MUF that includes the SQuAD dataset significantly improves the performance on SQuAD as well as on other datasets. In addition, we find that our DPR-MUF (w/o SQuAD) not only manages to beat the joint-training DPR trained on {NQ, Trivia, WQ, and Trec}, but also outperforms it on SQuAD by a large margin (10% in top-100 accuracy). We also test the fusion of experts without the one trained on the target domain, i.e., DPR-MUF (w/o domain), whose performance turns out to be maintained at a reasonable level.

One interesting result we find in the experiments is that DPR-MUF without the CuratedTrec/WQ expert outperforms the CuratedTrec/WQ experts on their domain test sets. We suspect that the CuratedTrec and WQ datasets are too small and might be covered by other datasets. Therefore, it is not surprising that the CuratedTrec and WQ experts trained on small data regimes fail to outperform the larger expert union.

4.4.2 Mixed-Dataset Retrieval

In a real-world application, the retriever often needs to deal with questions from different sources instead of just a single task. To test the ability to retrieve out-of-distribution questions, we evenly sample 5 subsets of 3,000 test questions from 4 benchmark datasets (NQ, Trivia, WQ, SQuAD). We average top-20/100 accuracy on the 5 subsets as the final accuracy. In addition, we design two oracle models which serve as references:

DPR-Oracle-Indicator: A mixture of experts that knows the domain each question comes from, and uses the corresponding expert for retrieval.

Retriever	Top-20	Top-100
DPR-Single-NQ	65.4	76.1
DPR-Single-Trivia	66.6	77.7
DPR-Single-WQ	54.1	68.3
DPR-Single-Trec	60.5	74.0
DPR-Single-SQuAD	57.3	73.2
DPR-Multi (w/o SQuAD)	71.5	80.7
DPR-MUF (w/o SQuAD)	72.7	81.7
DPR-MUF	74.2	83.3
DPR-Oracle-Indicator	72.8	82.0
DPR-Oracle-Bayesian	73.3	82.3

Table 4.2: Top-20/100 retrieval accuracy (%) on random mixes of 4 benchmark QA test sets. We average the metrics from 5 evenly-sampled subsets of 3,000 samples from NQ, Trivia, WQ, and SQuAD.

DPR-Oracle-Bayesian: A mixture of experts that uses Bayesian optimization [69] to search for the weights. We initialize the weights with the indicator function and use scikit-optimize¹ to search for the optimal weight for 50 iterations for each question. This process is not guaranteed to find the best sets of weights as Bayesian optimization does not always find the global optimum. Although it is not the exact oracle, this is the best model we could find as exhaustive search is impractical due to its exponential time complexity.

Tbl. 4.2 shows that all single retrievers have severe performance degradation on the randomly mixed dataset, which is expected as they only specialize in their own domain. In contrast, the two multi-task models, DPR-Multi (w/o SQuAD) and DPR-MUF (w/o SQuAD), manage to maintain high scores on the random mixes, which reach the performance of the two oracle models. Moreover, DPR-MUF even outperforms both the indicator oracle and the Bayesian oracle, suggesting the benefits of using uncertainty to fuse the predictions of multiple experts from different domains.

4.4.3 DPR-BM25 Hybrid Retrieval

[126] show that DPR can be combined with BM25 to further improve retrieval performance. [196] further finetune the parameters for BM25 and obtain better accuracy using the Pyserini IR toolkit [167]. We follow the experimental setting in [196] where we re-rank the union of the top-1000 passages retrieved by DPR and BM25 separately, using the weighted sum of the two

¹<https://scikit-optimize.github.io/>

Retriever	Top-20	Top-100
NQ		
BM25	62.9	78.3
+ DPR-Single-domain	82.5	88.2
+ DPR-Multi (w/o SQuAD)	82.6	88.6
+ DPR-MUF (w/o SQuAD)	82.7	88.6
+ DPR-MUF	82.0	88.2
Trivia		
BM25	76.4	83.2
+ DPR-Single-domain	82.8	86.8
+ DPR-Multi (w/o SQuAD)	82.6	86.5
+ DPR-MUF (w/o SQuAD)	82.9	87.0
+ DPR-MUF	82.4	86.5
WQ		
BM25	62.4	75.5
+ DPR-Single-domain	74.3	82.6
+ DPR-Multi (w/o SQuAD)	77.1	84.4
+ DPR-MUF (w/o SQuAD)	77.9	84.5
+ DPR-MUF	78.1	84.9
TREC		
BM25	80.7	89.9
+ DPR-Single-domain	90.1	94.7
+ DPR-Multi (w/o SQuAD)	90.1	95.0
+ DPR-MUF (w/o SQuAD)	90.8	95.5
+ DPR-MUF	91.2	95.7
SQuAD		
BM25	71.1	81.8
+ DPR-Single-domain	75.6	84.9
+ DPR-Multi (w/o SQuAD)	75.1	84.4
+ DPR-MUF (w/o SQuAD)	76.7	86.3
+ DPR-MUF	78.7	86.7

Table 4.3: Top-20/100 retrieval accuracy (%) of BM25 and DPR-BM25 hybrid model on test sets of NQ, Trivia, WQ, CuratedTrec, and SQuAD.

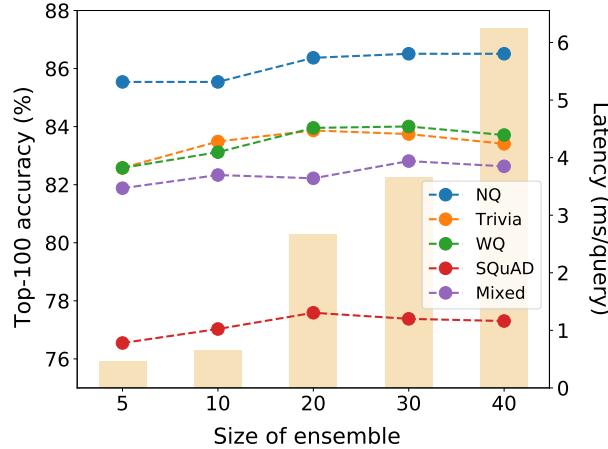


Figure 4.3: Line plot of top-100 accuracy (%) and bar chart of latency (ms/query) relative to a single DPR model of our method w.r.t. the size of the ensemble on NQ, Trivia, WQ, SQuAD, and their random mixes.

scores as the ranking value. We search for the optimal weights for BM25 and DPR on the dev set of each QA dataset.

Tbl. 4.3 shows the top- k accuracy of hybrid retrievers using the combination of different DPR models and BM25. Our model uncertainty method manages to outperform single DPR experts and multi-task, joint-training DPR on all benchmark QA datasets. Specifically, DPR-MUF (w/o SQuAD) has the best performance on NQ and Trivia, while DPR-MUF includes all experts, which has the best performance on WQ, CuratedTrec, and SQuAD. We conjecture it’s because NQ and Trivia are much larger and therefore the SQuAD expert might have more conflict with BM25.

4.4.4 Ensemble Sensitivity and Latency

In this section, we analyze how sensitive the retrieval performance of the uncertainty fusion method is w.r.t. the ensemble size. Fig. 4.3 shows the top-100 accuracy and the relative latency of different sizes of ensembles. The accuracy increases as the size of the ensemble grows until it hits 20, which then plateaus or decreases. We conjecture it is because the functional space of the ensemble is not complex enough as we only use a 2 layer neural network with 512 units as the individual component. Therefore, there are only limited ways for the model to overfit the training sets, resulting in the saturation in diversity w.r.t. the ensemble size.

However, we find that overall, these results are good enough while having reasonable latency. The latency (ms/question) of the model is measured relative to a standard DPR model, which mainly includes the ensemble forward inference time. We evaluate the inference speed on a server

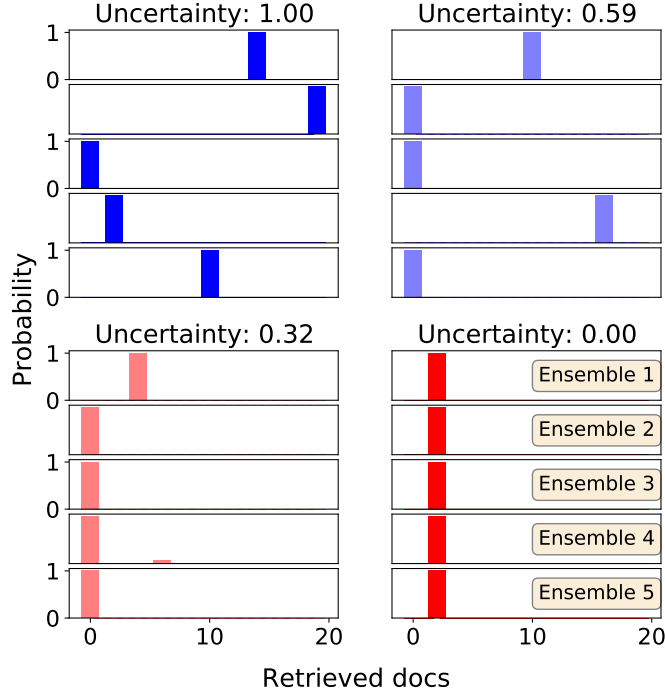


Figure 4.4: A visualization of uncertainty estimation using mutual information between the ensemble’s predictions and parameters. Each subplot shows the prediction of 5 ensemble on the top-20 documents retrieved by the DPR expert. The uncertainty decreases as more ensemble members “agree” with each other.

with an Intel Xeon CPU E5-2699 v4 @ 2.20GHz. In summary, the retrieval accuracy is stable w.r.t. the ensemble size, and one can choose the ensemble size to trade-off between accuracy and latency for different application scenarios.

4.4.5 Uncertainty Visualization

We visualize the model uncertainty in this section for better understanding. Fig. 4.4 shows 5 ensemble predictions of the top-20 documents on 4 samples from NQ with different uncertainty scores. Each strip in a subplot represents one ensemble member and all members in the same subplot share the same documents retrieved by the DPR expert on NQ. As we use small inverse temperature λ (1e-3) for the softmax distribution, the probability mass of each distribution mainly concentrates on the top-1 document, which is the tallest bar in each strip. If the top-1 predictions from different ensemble members overlap at the same document, we say these members “agree”

with each other and therefore the overall ensemble has low uncertainty. The overlap is quantified by Eq. (4.1) in practice. As we can see from Fig. 4.4, the ensemble has full uncertainty (1.0/1.0) when their top-1 predictions do not overlap at all, and has zero uncertainty when its members’ predictions completely overlap. In other cases, the more overlap or “agreement” on the top-1 prediction, the less uncertain the ensemble is.

4.4.6 Storage-Latency-Flexibility Trade-off

Despite the promising results we have shown in the previous section, the model uncertainty fusion method also has its drawbacks in open-domain question-answering. For now, all the experts are individually trained in their own domain but share a common corpus. That says if we have m experts, then the index size will grow by $O(m)$ compared to a single multi-task, joint-training model.

However, we argue that there’s no free lunch as the joint-training model suffers from other problems such as data conflict mentioned before, as well as catastrophic forgetting: If new tasks are added, the joint-training model usually requires to re-train on the union of all tasks again to maintain performance on previous tasks, while our model only needs to train on the new task’s data and the new expert can be directly added to the current set of models. Therefore, both methods have their pros and cons according to different application scenarios, and it is upon the users to consider the space-speed-flexibility trade-off. For memory and efficiency issues, possible solutions would be either learning a shared, query-agnostic index for all experts or leveraging model compression methods to compress the size of expert models.

4.5 Summary

In this Chapter, we propose a model fusion approach for multi-task dense retrieval. Instead of training a single DPR model on the union of datasets from different distributions, we leverage model uncertainty to merge different DPR expert’s predictions during test time. For each expert, we train an ensemble of small neural networks on top of the pretrained expert’s dense representations and use the mutual information between the ensemble parameters and predictions as the weight, which can be interpreted as the “disagreement” among the ensemble.

We compare our model uncertainty fusion approach with single specialists and the multi-task, joint-training DPR model on 5 benchmark QA datasets, as well as their dataset random mixes to test out-of-distribution performance. Extensive experiments show that our method manages to outperform these approaches in terms of top-20/100 accuracy on most datasets, while it can also be combined with sparse retrieval methods such as BM25 for further performance gains. Our proposed method is simple to implement and effective while enjoying the benefits of continual

learning, faster training speed as the experts can be trained in parallel, as well as the flexibility to combine experts from different domains.

Chapter 5

Efficient Token Interaction via Dynamic Lexical Routing

5.1 Overview

The goal of information retrieval is to find a set of related documents from a large data collection given a query. Traditional bag-of-words systems [267, 167] calculate the ranking scores based on the query terms appearing in each document, which have been widely adopted in many applications such as web search [226, 232] and open-domain question answering [41, 147]. Recently, dense retrieval [126] based on pretrained language models [57, 184] has been shown to be very effective. It circumvents the term mismatch problem in bag-of-words systems by encoding the queries and documents into low-dimensional embeddings and using their dot product as the similarity score (Figure 5.2a). However, dense retrieval is less robust on entity-heavy questions [282] and out-of-domain datasets [310], therefore calling for better solutions [68, 71].

In contrast, multi-vector retrieval has shown strong performance on both in-domain and out-of-domain evaluations by taking into account token-level interaction. Among them, ColBERT [128] is arguably the most celebrated method that has been the state of the art on multiple datasets so far. However, its wider application is hindered by its large index size and high retrieval latency. This problem results from the redundancy in the token interaction of ColBERT where many tokens might not contribute to the sentence semantics at all. To improve this, COIL [72] imposes an exact match constraint on ColBERT for conditional token interaction, where only token embeddings with the same token id could interact with each other. Although reducing the latency, the word mismatch problem reoccurs and the model may fail to match queries and passages that use different words to express the same meaning.

In this Chapter, we first give a unified view of existing multi-vector retrieval methods based

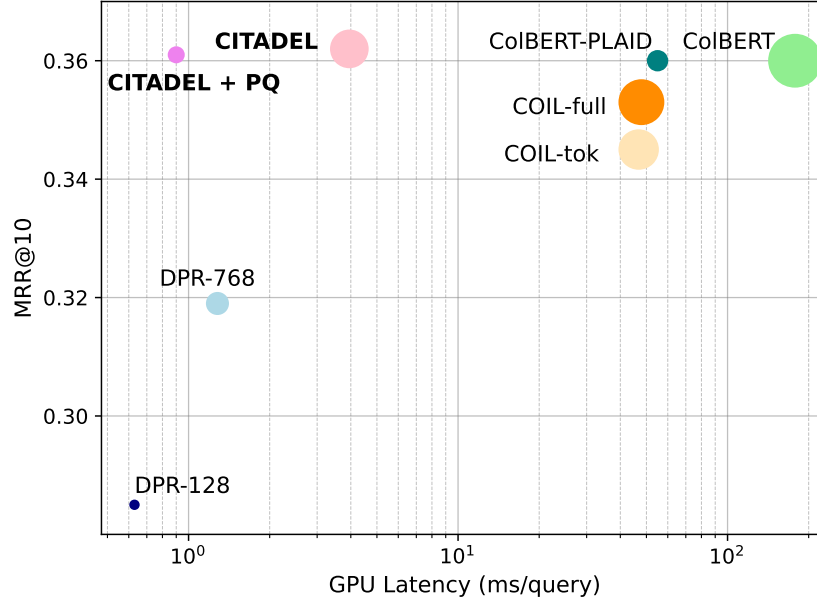


Figure 5.1: GPU latency vs ranking quality (MRR@10) on MS MARCO passages with an A100 GPU. The *circle size* represents the relative index storage on disk. All models are trained without hard-negative mining, distillation, or further pretraining.

on *token routing* (Section 5.2), providing a new lens through which we expose the limitations of current models. Under the token routing view, ColBERT could be seen as *all-to-all routing*, where each query token exhaustively interacts with all passage tokens (Figure 5.2b). COIL, on the other hand, could be seen as *static lexical routing* using an exact match constraint, as each query token only interacts with the passage tokens that have the same token id as the query token (Figure 5.2c).

In contrast, we propose a novel conditional token interaction method using *dynamic lexical routing* called CITADEL as shown in Figure 5.2d. Instead of relying on static heuristics such as exact match, we train our model to dynamically moderate token interaction so that each query token only interacts with the most relevant tokens in the passage. This is achieved by using a lexical *router*, trained end-to-end with the rest of the model, to route each contextualized token embedding to a set of activated lexical “keys” in the vocabulary. In this way, each query token embedding only interacts with the passage token embeddings that have the same activated key, which is dynamically determined during computation. As we shall see in Section 5.6.1, this learning-based routing does not lose any accuracy compared to all-to-all routing while using fewer token interactions than COIL (Section 5.3.4), leading to a highly effective and efficient retriever.

Experiments on MS MARCO passages [226] and TREC DL show that CITADEL achieves the

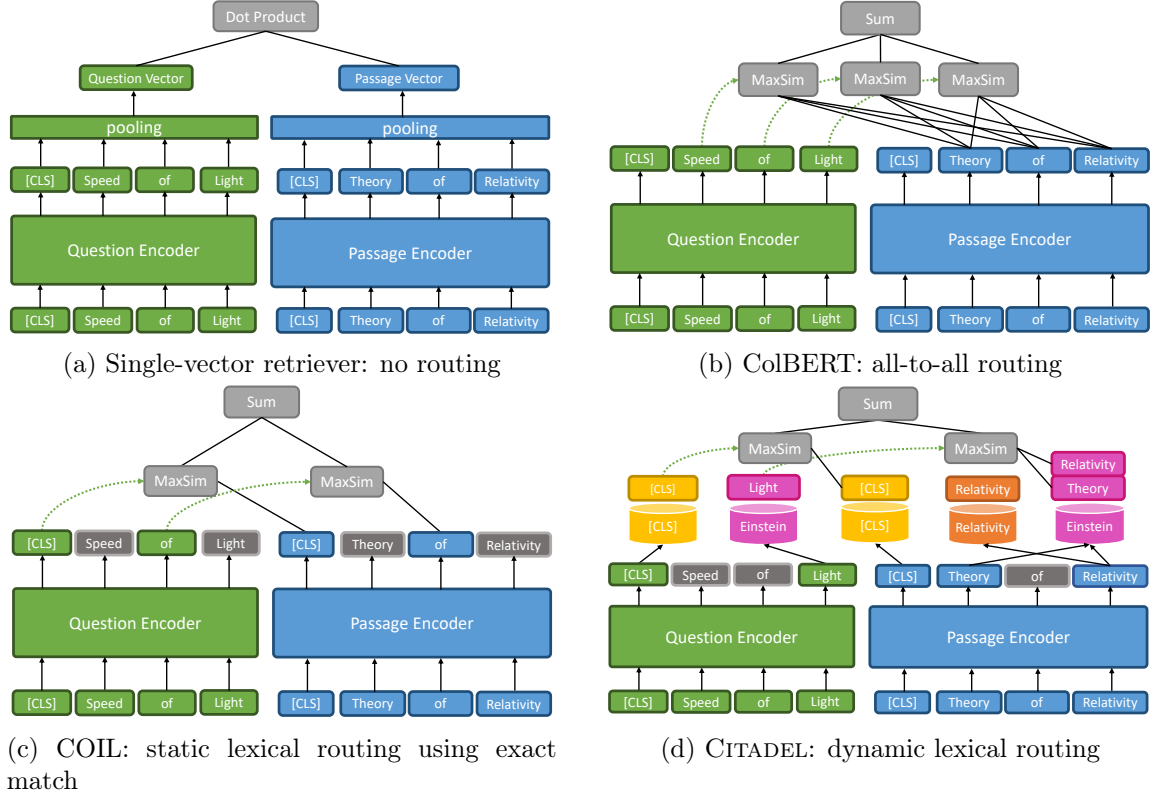


Figure 5.2: A unified token routing view over different multi-vector and single-vector retrieval models. The cylinder in (d) represents the lexical key for token routing. The grey tokens in (c) and (d) represent the tokens that do not contribute to the final similarity. For (d), CITADEL routes “*light*”, “*theory*”, and “*relativity*” to the “*Einstein*” key to avoid term mismatch using a learned routing function.

same level of accuracy as ColBERT-v2. We further test CITADEL on BEIR [310] and CITADEL still manages to keep up with ColBERT-v2 [277] which is the current state of the art. As for the latency, CITADEL can yield an average latency of 3.21 ms/query on MS MARCO passages using an A100 GPU, which is nearly 40× faster than ColBERT-v2. By further combining with product quantization, CITADEL’s index only takes 13.3 GB on MS MARCO passages and reduces the latency to 0.9 ms/query as shown in Figure 5.1.

5.2 A Unified Token Routing View of Multi-Vector Retrievers

We outline a unified view for understanding various neural retrievers using the concept of token routing that dictates token interaction.

5.2.1 Single-Vector Retrieval

Given a collection of documents and a set of queries, single-vector models [126, 113] use a bi-encoder structure where its query encoder $\eta_Q(\cdot)$ and document encoder $\eta_D(\cdot)$ are independent functions that map the input to a low-dimensional vector. Specifically, the similarity score s between the query q and document d is defined by the dot product between their encoded vectors $v_q = \eta_Q(q)$ and $v_d = \eta_D(d)$:

$$s(q, d) = v_q^T v_d. \quad (5.1)$$

As all the token embeddings are pooled before calculating the similarity score, no routing is committed as shown in Figure 5.2a.

5.2.2 Multi-Vector Retrieval

ColBERT [128] proposes late interaction between the tokens in a query $q = \{q_1, q_2, \dots, q_N\}$ and a document $d = \{d_1, d_2, \dots, d_M\}$:

$$s(q, d) = \sum_{i=1}^N \max_j v_{q_i}^T v_{d_j}, \quad (5.2)$$

where v_{q_i} and v_{d_j} denotes the last-layer contextualized token embeddings of BERT. This is known as the MaxSim operation which exhaustively compares each query token to all document tokens. We refer to this as *all-to-all routing* as shown in Figure 5.2b. The latency of ColBERT is inflated by the redundancy in the all-to-all routing, as many tokens do not contribute to the sentence semantics. This also drastically increases the storage, requiring complex engineering schemes to make it more practical [277, 276].

Another representative multi-vector approach known as COIL [72] proposes an exact match constraint on the MaxSim operation where only the embeddings with the same token id could interact with each other. Let $\mathcal{J}_i = \{j \mid d_j = q_i, 1 \leq j \leq M\}$ be the subset of document tokens

$\{d_j\}_{j=1}^M$ that have the same token ID as query token q_i , then we have:

$$s(q, d) = \sum_{i=1}^N \max_{j \in \mathcal{J}_i} v_{q_i}^T v_{d_j}, \quad (5.3)$$

It could be further combined with Equation (5.1) to improve the effectiveness if there’s no word overlap between the query and documents.

$$s(q, d) = v_q^T v_d + \sum_{i=1}^N \max_{j \in \mathcal{J}_i} v_{q_i}^T v_{d_j}. \quad (5.4)$$

We refer to this token interaction as *static lexical routing* as shown in Figure 5.2c. As mentioned in Section 5.1, the word mismatch problem could happen if $\mathcal{J}_i = \emptyset$ for all q_i , which affects the retrieval accuracy. Moreover, common tokens such as “the” will be frequently routed, which will create much larger token indexes compared to those rare words. This bottlenecks the search latency as COIL needs to frequently iterate over large token indexes.

5.3 The CITADEL Method

5.3.1 Dynamic Lexical Routing

Instead of using the wasteful all-to-all routing or the inflexible heuristics-based static routing, we would like our model to dynamically select which query and passage tokens should interact with each other based on their contextualized representation, which we refer to as *dynamic lexical routing*. Formally, the routing function (or *router*) routes each token to a set of lexical **keys** in the vocabulary and is defined as $\phi : \mathbb{R}^c \rightarrow \mathbb{R}^{|\mathcal{V}|}$ where c is the embedding dimension and $\mathcal{V}$ is the lexicon of keys. For each contextualized token embedding, the router predicts a scalar score for each key in the lexicon indicating how relevant each token is to that key. Given a query token embedding v_{q_i} and a document token vector v_{d_j} , the token level router representations are $w_{q_i} = \phi(v_{q_i})$ and $w_{d_j} = \phi(v_{d_j})$, respectively. The elements in the router representations are then sorted in descending order and truncated by selecting the top- K query keys and top- L document keys, which are $\{e_{q_i}^{(1)}, e_{q_i}^{(2)}, \dots, e_{q_i}^{(K)}\}$ and $\{e_{d_j}^{(1)}, e_{d_j}^{(2)}, \dots, e_{d_j}^{(L)}\}$ for q_i and d_j , respectively. In practice, we use $K=1$ and $L=5$ as the default option which will be discussed in Section 5.3.5 and Section 5.8. The corresponding routing weights for q_i and d_j are $\{w_{q_i}^{(1)}, w_{q_i}^{(2)}, \dots, w_{q_i}^{(K)}\}$ and $\{w_{d_j}^{(1)}, w_{d_j}^{(2)}, \dots, w_{d_j}^{(L)}\}$, respectively. The final similarity score is similar to Equation (5.3), but we substitute the static lexical routing subset $\mathcal{J}_i$ with a dynamic key set predicted by the router: $\mathcal{E}_i^{(k)} = \{j, l \mid e_{d_j}^{(l)} = e_{q_i}^{(k)}, 1 \leq j \leq M, 1 \leq l \leq L\}$ for each key $e_{q_i}^{(k)}$ of the query token q_i :

$$s(q, d) = \sum_{i=1}^N \sum_{k=1}^K \max_{j, l \in \mathcal{E}_i^{(k)}} (w_{q_i}^{(k)} \cdot v_{q_i})^T (w_{d_j}^{(l)} \cdot v_{d_j}), \quad (5.5)$$

Optionally, all [CLS] tokens can be routed to an additional *semantic* key to complement our learned lexical routing. We then follow DPR [126] to train the model contrastively. Given a query q , a positive document d^+ , and a set of negative documents D^- , the contrastive loss is:

$$\mathcal{L}_e = -\log \frac{\exp(s(q, d^+))}{\exp(s(q, d^+)) + \sum_{d^- \in D^-} \exp(s(q, d^-))}, \quad (5.6)$$

such that the distance from the query to the positive document d^+ is smaller than the query to the negative document d^- .

5.3.2 Router Optimization

To train the router representation $\phi(q)$ and $\phi(d)$, we adopt a contrastive loss such that the number of overlapped keys between a query and documents are large for positive (q, d^+) pairs and small for negative pairs (q, d^-) . We first pool the router representation for each query and document over the tokens. Given a sequence of token-level router representations $\{\phi(v_1), \phi(v_2), \dots, \phi(v_M)\}$, the sequence-level representation is defined as:

$$\Phi = \max_{j=1}^M \phi(v_j), \quad (5.7)$$

where the max operator is applied element-wise. Similar to [67], We find max pooling works the best in practice compared to other pooling methods. Subsequently, the contrastive loss for training the router is:

$$\mathcal{L}_r = -\log \frac{\exp(\Phi_q^T \Phi_{d^+})}{\exp(\Phi_q^T \Phi_{d^+}) + \sum_{d^- \in D^-} \exp(\Phi_q^T \Phi_{d^-})}. \quad (5.8)$$

In addition, we follow SPLADE [68, 67] to initialize the router with the pretrained Masked Language Modelling Layer (MLM). Without proper initialization, it is difficult to optimize the router due the large lexical space and sparse activation. With the pretrained MLM initialization, the router expands words with similar semantic meaning to sets of keys with large overlap at the beginning of training, making the contrastive loss easier to optimize.

5.3.3 Sparsely Activated Router Design

Softmax activation is commonly used for computing the routing weights in conditional computation models [64, 223]. However, softmax often yields a small probability over a large number of dimensions (in our case, about 30,000) and the product of two probability values are even smaller, which makes it not suitable for yielding the routing weights $w_{q_i}^{(k)}$ and $w_{d_j}^{(l)}$ in Equation (5.5) as the corresponding gradients are too small. Instead, we use the activation from SPLADE to compute the router representation for a token embedding v_j :

$$\phi(v_j) = \log(1 + \text{ReLU}(W^T v_j + b)), \quad (5.9)$$

where W and b are the weights and biases of the Masked Language Modeling (MLM) layer of BERT. The SPLADE activation brings extra advantages as the ReLU activation filters irrelevant keys while the log-saturation suppresses overly large “wacky” weights [202].

5.3.4 Regularization for Routing

ℓ_1 Regularization. Routing each token to more than one key increases the overall size of the index. Therefore, we propose to use ℓ_1 regularization on the router representation to encourage the router to only keep the most meaningful token interaction by pushing more routing weights to 0:

$$\mathcal{L}_s = \frac{1}{B} \sum_{i=1}^B \sum_{j=1}^T \sum_{k=1}^{|\mathcal{V}|} \phi(v_{ij})^{(k)}, \quad (5.10)$$

where $|\mathcal{V}|$ is the number of keys, B is the batch size, and T is the sequence length. As shown in Figure 5.7, CITADEL has a sparsely activated set of keys, by routing important words to multiple lexical keys while ignoring many less salient words, leading to effective yet efficient retrieval.

Load Balancing. As mentioned in Section 5.2.2, the retrieval latency of COIL is bottlenecked by frequently searching overly large token indexes. This results from the static lexical routing where common “keys” have a larger chance to be activated, which results in large token indexes during indexing. Therefore, a vital point for reducing the latency of multi-vector models is to evenly distribute each token embedding to different keys. Inspired by Switch Transformers [64], we propose to minimize the load balancing loss that approximates the expected “evenness” of the number of tokens being routed to each key:

$$\mathcal{L}_b = \sum_{k=1}^{|\mathcal{V}|} f_k \cdot p_k, \quad (5.11)$$

p_k is the batch approximation of the marginal probability of a token vector routed to the k -th key:

$$p_k = \frac{1}{B} \sum_{i=1}^B \sum_{j=1}^T \frac{\exp(W_k^T v_{ij} + b_k)}{\sum_{k'} \exp(W_{k'}^T v_{ij} + b_{k'})}, \quad (5.12)$$

where W and b are the weights and bias of the routing function in Equation (5.9) and v_{ij} is the j -th token embedding in sample i of the batch. f_k is the batch approximation of the total number of tokens being dispatched to the k -th key:

$$f_k = \frac{1}{B} \sum_{i=1}^B \sum_{j=1}^T \mathbb{I}\{\text{argmax}(p_{ij}) = k\}, \quad (5.13)$$

where $p_{ij} = \text{softmax}(W^T v_{ij} + b)$. Finally, we obtain the loss for training CITADEL:

$$\mathcal{L} = \mathcal{L}_e + \mathcal{L}_r + \alpha \cdot \mathcal{L}_b + \beta \cdot \mathcal{L}_s, \quad (5.14)$$

where $\alpha \geq 0$ and $\beta \geq 0$. The ℓ_1 and load balancing loss are applied on both queries and documents.

5.3.5 Inverted Index Retrieval

CITADEL builds an inverted index like BM25 but we use a vector instead of a scalar for each token and the doc product as the term relevance.

Indexing and Post-hoc Pruning. To reduce index storage, we prune the vectors with routing weights less than a threshold τ after training. For a key e in the lexicon $\mathcal{V}$, the token index $\mathcal{I}_e$ consists of token embeddings v_{d_j} and the routing weight $w_{d_j}^e$ for all documents d in the corpus $\mathcal{C}$ is:

$$\mathcal{I}_e = \{w_{d_j}^e \cdot v_{d_j} \mid w_{d_j}^e > \tau, 1 \leq j \leq M, \forall d \in \mathcal{C}\}. \quad (5.15)$$

We will discuss the impact of post hoc pruning in Section 5.6.2, where we find that post hoc pruning can reduce the index size by 3 $\times$ without significant accuracy loss. The final search index is defined as $\mathcal{I} = \{\mathcal{I}_e \mid e \in \mathcal{V}\}$, where the load-balancing loss in Equation (5.11) will encourage the size distribution over $\mathcal{I}_e$ to be as even as possible. In practice, we set the number of maximal routing keys of each token to 5 for the document and 1 for the query. The intuition is that the documents usually contain more information and need more key capacity, which is discussed in Section 5.8 in detail.

Token Retrieval. Given a query q , CITADEL first encodes it into a sequence of token vectors $\{v_{q_i}\}_{i=1}^N$, and then route each vector to its top-1 key e with a routing weight $w_{q_i}^e$. The final representation $w_{q_i}^e \cdot v_{q_i}$ will be sent to the corresponding token index $\mathcal{I}_E$ for vector search. The final ranking list will be merged from each query token’s document ranking according to Equation (5.5).

5.4 Experimental Setup

5.4.1 Datasets

The MS MARCO passages corpus has around 8.8 million passages with an average length of 60 words. TREC DL 2019 and 2020 contain 43 and 54 test queries whose relevance sets are densely labelled with scores from 0 to 4. For out-of-domain evaluation, we use 13 datasets from BEIR, which includes TREC-COVID [322], NFCorpus [32], Natural Questions [138], HotpotQA [376], FiQA-2018 [203], ArguAna Counterargs Corpus [331], Touché-2020 [30], Quora¹, DBPedia-Entity-v2 [86], SCIDOCS [52], FEVER [311], Climate-FEVER [58], SciFact [332].

5.4.2 Baselines

All the baseline models below are trained and evaluated under the same setting of CITADEL (e.g., datasets, hyperparameters, and hardware).

Sparse Retrievers. BM25 [267] uses the term frequency and inverted document frequency as features to compute the similarity between documents. SPLADE [68, 67] leverages the pretrained language model’s MLM layer and ReLU activation to yield sparse term importance.

Dense Retrievers. DPR [126] encodes the input text into a single vector. coCondenser [71] pretrains DPR in an unsupervised fashion before finetuning.

Multi-Vector Retrievers. ColBERT [128, 277] encodes each token into dense vectors and performs late interaction between query token vectors and document token vectors. COIL [72] applies an exact match constraint on late interaction to improve efficiency and robustness.

5.4.3 Training

For CITADEL, we use bert-base-uncased as the initial checkpoint for fine-tuning. Following COIL, we set the [CLS] vector dimension to 128, token vector dimension to 32, maximal routing keys to 5 for document and 1 for query, α and β in Equation (5.14) are set to be 1e-2 and 1e-5, respectively. We add the dot product of [CLS] vectors in Equation (2.2) to the final similarity

¹<https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs>

Models	MARCO Dev		TREC DL19		TREC DL20		Index Storage		Latency (ms/query)		
	MRR@10	R@1k	nDCG@10	R@1k	nDCG@10	R@1k	Disk (GB)	Factor ¹	Encode (GPU)	Search (GPU)	Search (CPU)
<i>Models trained with only BM25 hard negatives</i>											
BM25	0.188	0.858	0.506	0.739	0.488	0.733	0.67	×0.22	×	×	40.1
DPR-128	0.285	0.937	0.576	0.702	0.603	0.757	4.33	×1.42	7.09	0.63	430
DPR-768	0.319	0.941	0.611	0.742	0.591	0.796	26.0	×8.52	7.01	1.28	2015
SPLADE	0.340	0.965	0.683	0.813	0.671	0.823	2.60	×0.85	7.13	×	475
COIL-tok	0.350	0.964	0.660	0.809	0.679	0.825	52.5	×17.2	10.7	46.8	1295
COIL-full	0.353	0.967	<u>0.704</u>	<u>0.835</u>	<u>0.688</u>	<u>0.841</u>	78.5	×25.7	10.8	47.9	3258
ColBERT	0.360	0.968	0.694	0.830	0.676	0.837	154	×50.5	10.9	178	-
CITADEL	<u>0.362</u>	<u>0.975</u>	0.687	0.829	0.661	0.830	78.3	×25.7	10.8	3.95	520
<i>Models trained with further pretraining/hard-negative mining/distillation</i>											
coCondenser	0.382	0.984	0.674	0.820	0.684	0.839	26.0	×8.52	7.01	<u>1.28</u>	2015
SPLADE-v2	0.368	0.979	0.729	0.865	0.718	0.890	4.12	×1.35	7.13	×	2710
ColBERT-v2	0.397	0.985	0.744	0.882	0.750	0.894	29.0	×9.51	10.9	122	3275
ColBERT-PLAID ²	0.397	0.984	0.744	0.882	0.749	0.894	<u>22.1</u>	<u>×7.25</u>	10.9	55.0	<u>370</u>
CITADEL ⁺	0.399	0.981	0.703	0.830	0.702	0.859	81.3	×26.7	10.8	3.21	635

¹ Factor: Ratio of index size to plain text size.

² The PLAID implementation of ColBERT contains complex engineering schemes and low-level optimization such as centroid interaction and fast kernels.

Table 5.1: In-domain evaluation on MS MARCO passages and TREC DL 2019/2020. CITADEL⁺ is trained with cross-encoder distillation and hard-negative mining. The red region means CITADEL/CITADEL⁺ is better than the method while the green region means that there’s no statistical significance ($p > 0.05$). “×” means not implemented and “-” means not practical to evaluate on a single CPU.

score in Equation (5.5). All models are trained for 10 epochs with AdamW [188] optimizer, a learning rate of 2e-5 with 3000 warm-up steps and linear decay. Hard negatives are sampled from top-100 BM25 retrieval results. Each query is paired with 1 positive and 7 hard negatives for faster convergence. We use a batch size of 128 on MS MARCO passages with 32 A100 GPUs.

For a fair comparison with recent state-of-the-art models, we further train CITADEL using cross-encoder distillation and hard negative mining. First, we use the trained CITADEL model under the setting in the last paragraph to retrieve top-100 candidates from the corpus for the training queries. We then use the cross-encoder² to rerank the top-100 candidates and score each query-document pair. Finally, we re-initialize CITADEL with bert-base-uncased using the positives and negatives sample from the top-100 candidates scored by the cross-encoder, with a 1:1 ratio for the soft-label and hard-label loss mixing [96]. We also repeat another round of hard negative mining and distillation but it does not seem to improve the performance any further.

²<https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2>

Methods	AA	CF	DB	Fe	FQ	HQ	NF	NQ	Qu	SF	SD	TC	T2	Avg.
<i>Models trained with only BM25 hard negatives</i>														
BM25	0.315	0.213	0.313	0.753	0.236	0.603	0.325	0.329	0.789	0.665	0.158	0.656	0.367	0.440
DPR-768	0.323	0.167	0.295	0.651	0.224	0.441	0.244	0.410	0.750	0.479	0.103	0.604	0.185	0.375
SPLADE	0.445	0.201	0.370	0.740	0.289	0.640	0.322	0.469	0.834	0.633	0.149	0.661	0.201	0.453
COIL-full	0.295	<u>0.216</u>	0.398	0.840	0.313	0.713	<u>0.331</u>	0.519	0.838	0.707	0.155	0.668	0.281	0.483
ColBERT	0.233	0.184	0.392	0.771	<u>0.317</u>	0.593	0.305	<u>0.524</u>	<u>0.854</u>	0.671	<u>0.165</u>	0.677	0.202	0.453
CITADEL	<u>0.503</u>	0.191	<u>0.406</u>	0.784	0.298	0.653	0.324	0.510	0.844	0.674	0.152	<u>0.687</u>	0.294	<u>0.486</u>
<i>Models with further pretraining/hard-negative mining/distillation</i>														
coCondenser	0.440	0.133	0.347	0.511	0.281	0.533	0.319	0.467	0.863	0.591	0.130	0.708	0.143	0.420
SPLADE-v2	0.479	0.235	0.435	<u>0.786</u>	0.336	<u>0.684</u>	0.334	0.521	0.838	0.693	0.158	0.710	0.272	0.499
ColBERT-v2	0.463	0.176	0.446	0.785	0.356	0.667	0.338	0.562	0.854	0.693	0.165	0.738	0.263	0.500
CITADEL ⁺	0.490	0.181	0.420	0.747	0.332	0.652	0.337	0.539	0.852	<u>0.695</u>	0.147	0.680	0.340	0.493
CITADEL ⁺ (w/o reg.)	0.511	0.182	0.422	0.765	0.330	0.664	0.337	0.540	0.853	0.690	0.159	0.715	<u>0.342</u>	0.501

Table 5.2: Out-of-domain evaluation on BEIR benchmark. nDCG@10 score is reported. Dataset Legend [42]: TC=TREC-COVID, NF=NFCorpus, NQ=NaturalQuestions, HQ=HotpotQA, FQ=FiQA, AA=ArguAna, T2=Touché-2020 (v2), Qu=Quora, DB=DBPedia, SD=SCIDOCs, Fe=FEVER, CF=Climate-FEVER, SF=SciFact.

5.5 Results

5.5.1 MS MARCO Passages Retrieval

We evaluate MS MARCO passages [226] and its shared tasks, TREC DL 2019/2020 passage ranking tasks [54]. Following standard practice, we train CITADEL and other baseline models on MS MARCO passages and report the results on its dev-small set and TREC DL 2019/2020 test queries. The evaluation metrics are MRR@10, nDCG@10, and Recall@1000 (i.e., R@1K).

Table 5.1 shows the in-domain evaluation results on MS MARCO passage and TREC DL 2019/2020. We divide the models into two classes: ones trained with only labels and BM25 hard negatives and the others trained with further pretraining [71], hard negative mining [367], or distillation from a cross-encoder³. CITADEL is trained only with BM25 hard negatives, while CITADEL⁺ is trained with cross-encoder distillation and one-round hard negative mining. The default pruning threshold is $\tau = 0.9$. As shown in Section 5.6.2, τ can be adjusted to strike different balances between latency, index size and accuracy. In both categories, CITADEL/CITADEL⁺ outperforms the baseline models on the MS MARCO passages dev set and greatly reduces the search latency on both GPU and CPU. For example, CITADEL⁺ achieves an average latency of 3.21 ms/query which is close to DPR-768 (1.28 ms/query) on GPU, while having a 25% higher MRR@10 score. CITADEL also maintains acceptable index sizes on disk, which can be further reduced using product quantization (Section 5.6.3). Although not able to outperform several baselines on TREC DL 2019/2020, we perform t-test ($p < 0.05$) on CITADEL and CITADEL⁺

³<https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2>

against other baselines in their sub-categories and show there is no statistical significance. The inconsistency is probably due to that we use the training data from Tevatron where each passage is paired with a title.

5.5.2 BEIR: Out-of-Domain Evaluation

We evaluate on BEIR benchmark [310] which consists of a diverse set of 18 retrieval tasks across 9 domains. We evaluate on 13 datasets following previous works [277, 67]. Table 5.2 shows the zero-shot evaluation results on BEIR. Without any pretraining or distillation, CITADEL manages to outperform all baselines in their sub-categories in terms of the average score. Compared with the distilled/pretrained models, CITADEL⁺ still manages to achieve comparable performance. Interestingly, we find that if no regularization like load balancing and L1 is applied during training, CITADEL⁺ can reach a much higher average score that even outperforms ColBERT-v2. Our conjecture is that the regularization reduces the number of token interactions and the importance of such token interaction is learned from training data. It is hence not surprising that the more aggressively we prune token interaction, the more likely that it would hurt out-of-domain accuracy that's not covered by the training data.

5.6 Performance Analysis

5.6.1 Number of Token Interactions

The actual latency is often impacted by engineering details and therefore FLOPS is often considered for comparing efficiency agnostic to the actual implementation. In our case, however, FLOPS is impacted by the vector dimension in the nearest neighbor search which is different across models. Therefore, we only compare the maximal number of dot products needed as a proxy for token interaction per query during retrieval as shown in Table 5.3. The number of dot products per query in CITADEL with pruning threshold $\tau = 0.9$ is comparable to DPR-128 and much lower than ColBERT and COIL, which is consistent with the latency numbers in Table 5.1. The reason is that CITADEL has a balanced inverted index credited to the ℓ_1 regularization and the load balancing loss as shown in Figure 5.3. By applying the load balancing loss on the router prediction, CITADEL yields a more balanced and even index distribution where its largest index fraction is $8 \times$ smaller than COIL's as shown in Figure 5.3.

5.6.2 Latency-Memory-Accuracy Tradeoff

Figure 5.4 shows the tradeoff among latency, memory, and MRR@10 on MS MARCO passages with post-hoc pruning. We try the pruning thresholds [0.5, 0.7, 0.9, 1.1, 1.3, 1.5]. We could see

Models	MRR@10	#DP $\times 10^6$
ColBERT	0.360	4213
COIL-full	0.353	45.6
CITADEL	0.362	10.5
DPR-128	0.285	8.84

Table 5.3: Maximal number of dot products per query on MS MARCO passages. # DP: number of dot products.

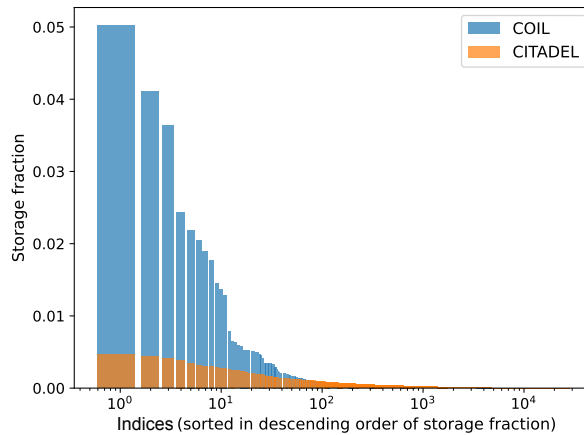


Figure 5.3: Normalized disk storage over token indices in an inverted list of CITADEL and COIL.

that the MRR@10 score barely decreases when we increase the threshold to from 0.5 to 1.1, but the latency decreases by a large margin, from about 18 ms/query to 0.61 ms/query. The sweet spots are around (0.359 MRR@10, 49.3GB, 0.61 ms/query) and (0.362 MRR@10, 78.5GB, 3.95 ms/query). This simple pruning strategy is extremely effective and readers can see in Section 5.7 that it also yields interpretable document representations.

5.6.3 Combination with Product Quantization

We could further reduce the latency and storage with product quantization [119] (PQ) as shown in Table 5.4. For nbits=2, we divide the vectors into sets of 4-dimensional sub-vectors and use 256 centroids for clustering the sub-vectors, while for nbits=1 we set the sub-vector dim to 8 and the same for the rest. With only 2 bits per dimension, the MRR@10 score on MS MARCO Dev only drops 4% but the storage is reduced by 83% and latency is reduced by 76%.

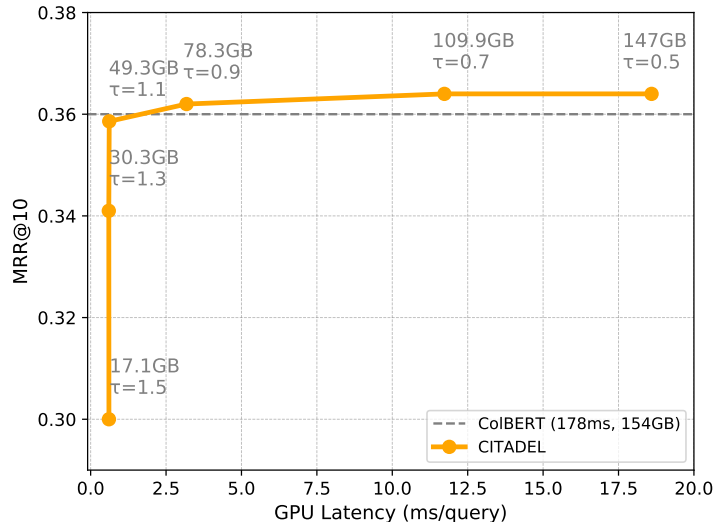


Figure 5.4: Latency-memory-accuracy tradeoff on MS MARCO passages using post-hoc pruning. τ is the pruning threshold.

Condition	MRR@10	Storage (GB)	Latency (ms)
original	0.362	78.3	3.95
nbits=2	0.361	13.3	0.90
nbits=1	0.356	11.0	0.92

Table 5.4: Product Quantization. Pruning threshold is set to 0.9. nbits: ratio of total bytes to the vector dimension.

5.7 Token Routing Analysis of CITADEL

Qualitative Analysis. We visualize CITADEL representations and the effect of posthoc pruning in Figure 5.5. By increasing the pruning threshold, more keywords are pruned and finally leaving the most central word “*arrhythmia*” activated. We provide another example in Figure 5.7. We can see that a lot of redundant words that do not contribute to the final semantics are deactivated, meaning all their routing weights are 0. For the activated tokens, we could see the routed keys are contextualized as many of them are related to *emoji* which is the theme of the document.

Quantitative Analysis. We analyze CITADEL’s token distribution over the number of activated routing keys for the whole corpus as shown in Figure 5.6. With ℓ_1 loss, around 50 tokens per passage are deactivated (i.e., all the routing weights of these 50 tokens are 0). As the pruning

Threshold τ	Sample documents from MS MARCO Passages
0.0	All medications have side effects , including drugs to treat arrhythmias . Most of the side effects aren't serious and disappear when the dose is changed or the medication is stopped . But some side effects are very serious . That's why some children are admitted to the hospital to begin the medication . Medications for Arrhythmia
0.9	All medications have side effects, including drugs to treat arrhythmias. Most of the side effects aren't serious and disappear when the dose is changed or the medication is stopped. But some side effects are very serious. That's why some children are admitted to the hospital to begin the medication. Medications for Arrhythmia
1.3	All medications have side effects, including drugs to treat arrhythmias. Most of the side effects aren't serious and disappear when the dose is changed or the medication is stopped. But some side effects are very serious. That's why some children are admitted to the hospital to begin the medication. Medications for Arrhythmia

Figure 5.5: Tokens in grey (pruned) have zero activated keys while bold tokens have at least one activated key. We leave out the expanded terms and routing weights due to space limit.

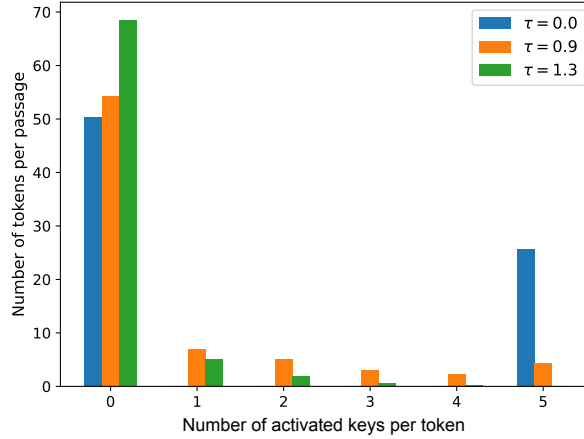


Figure 5.6: Token distribution over number of activated experts per passage. τ is the pruning threshold.

Models	Dev	DL19	Latency (ms)
COIL-full	0.353	0.704	47.9
COIL-tok	0.350	0.660	46.8
CITADEL	0.362	0.687	3.95
CITADEL-tok	0.360	0.665	1.64

Table 5.5: [CLS] ablation on MS MARCO passage. Dev: MRR@10. DL19: nDCG@10.

threshold increases, more tokens are deactivated, yielding a sparse representation for interpreting CITADEL’s behaviours.

Sample Document from MS MARCO Passages
It may be a little hard to remember (remember, forget) all of the 25 (barney, many, 25, 1625) codes (code, codes), which is why we've created (invented, stumbled, created, swirled) this dictionary (dictionary, spiked). You can find the complete facebook (facebook) emoticons (emoticon, emoji, combinations, asher, grins) list for chats (chat, chatting) and status (winked, logo, status, glided) in the table above. This list (list, kingsley, include)ing includes all the new ones which have been recently added, such as the penguin (penguin, mccann, uefa) or shark (shark, chao) emoticon (emoticon, emoji, assassins). Facebook (facebook) Emoticons (emoticon, emoji, spp, programs)

Figure 5.7: Token routing analysis of CITADEL. Grey tokens are deactivated, while bold tokens are routed to at least one activated key (in blue).

5.8 Ablation Studies

Impact of [CLS] Table 5.5 shows the influence of removing the [CLS] vector for CITADEL on MS MARCO passage. Although removing [CLS] improves the latency by a large margin, the in-domain effectiveness is also adversely affected, especially on TREC DL 2019. Nevertheless, CITADEL-tok (w/o [CLS]) still outperforms its counterpart COIL-tok in both precision and latency.

Number of Routed Experts. Table 5.6 shows the influence of changing the maximum number of keys that each document token can be routed to during training and inference on MS MARCO passage. As the number of routing keys increases, the index storage also increases rapidly but so does the MRR@10 score which plateaus after reaching 7 keys. The latency does not increase as much after 3 routing keys due to the load balancing loss.

5.9 Summary

This Chapter proposes a novel multi-vector retrieval method that achieves state-of-the-art performance on several benchmark datasets while being $40\times$ faster than ColBERT-v2 and $17\times$ faster

#Keys	MRR@10	Storage (GB)	Latency (ms)
1	0.347	53.6	1.28
3	0.360	136	14.7
5	0.364	185	18.6
7	0.370	196	20.4
9	0.367	221	19.6

Table 5.6: Number of routing keys for documents during training. No post-hoc pruning is applied.

than the most efficient multi-vector retrieval library to date, PLAID, on GPUs. By jointly optimizing for the token index size and load balancing, our new dynamic lexical routing scheme greatly reduces the redundancy in the all-to-all token interaction of ColBERT while bridging the word-mismatch problem in COIL. Experiments on both in-domain and out-of-domain datasets demonstrate the effectiveness and efficiency of our model.

Chapter 6

Sparsified Late Interaction with Inverted Indexes

6.1 Overview

Pairwise token interaction [87, 143, 390] has been widely used in information retrieval tasks [210]. Interaction-based methods enable deep coupling between queries and documents, which often outperform representation-based methods [264, 39] when strong text encoders are absent. However, with the rise of pretrained transformer models [57, 184], representation-based methods such as DPR [126] and SPLADE [68] gain more popularity as the pretrained representations capture rich semantics of the input texts. Moreover, these methods can leverage established search libraries such as FAISS [61] and Pyserini [167] for efficient retrieval. To combine the best of both worlds, models such as ColBERT [128] and COIL [72] that leverage late interaction are proposed, where the token interaction between queries and documents only happens at the last layer of contextualized embeddings. Their effectiveness and robustness are demonstrated on various retrieval and question-answering benchmarks. However, these models require token-level retrieval and aggregation, which results in large indexes and high retrieval latency. Therefore, different optimization schemes are proposed to improve efficiency in both time and space [276, 101], making multi-vector retrieval difficult to fit into off-the-shelf search libraries.

In this Chapter, we propose an efficient and compact approach called **Sparsified Late Interaction** for **Multi-vector** retrieval with inverted indexes, or SLIM. SLIM is as effective as the state-of-the-art multi-vector model ColBERT on MS MARCO Passages [226] and BEIR [310] without any distillation or hard negative mining while being more efficient. More importantly, SLIM is fully compatible with inverted indexes in existing search toolkits such as Pyserini, with only a few extra lines of code in the pre-processing and post-processing steps. In contrast, methods such as

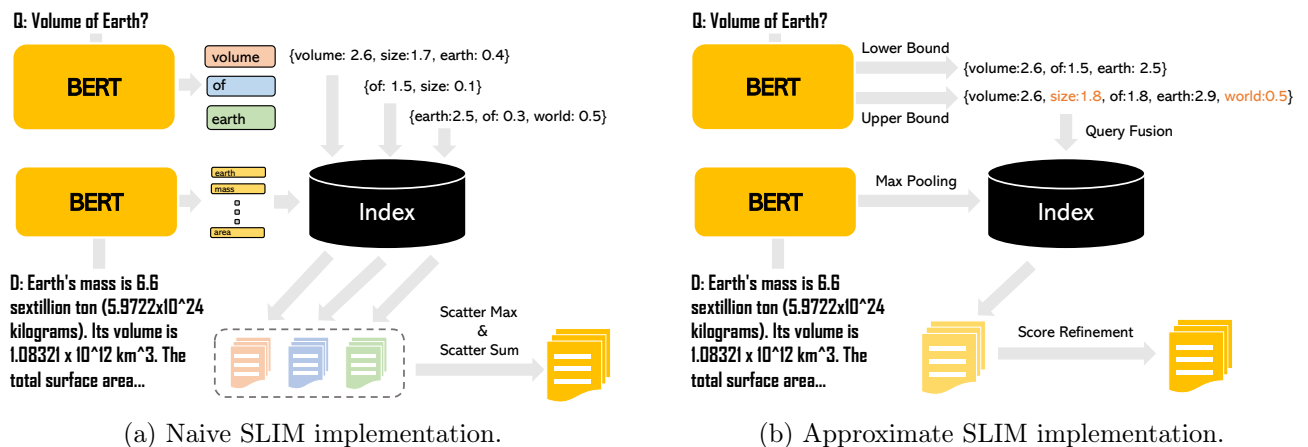


Figure 6.1: (a) The naive SLIM implementation takes the expansion of each token as sub-queries/documents. The token-level rankings are merged using scatter operations. (b) The approximate SLIM implementation first fuses the lower- and upper-bounds of scores for retrieval using Equations (6.8) and (6.9); the candidate list is then refined according to Equation (6.4).

ColBERT and COIL require custom implementations and optimizations [276, 101] in order to be practical to use. This presents a disadvantage because some researchers and practitioners may prefer neural retrievers that are compatible with existing infrastructure based on inverted indexes so that the models can be easily deployed in production.

In order to leverage standard inverted index search, SLIM projects each contextualized token embedding to a high-dimensional, sparse lexical space [68] before performing the late interaction operation. Ideally, the trained sparse representations of documents can be compressed and indexed into an inverted file system for search. Our method is the first to explore using sparse representations for multi-vector retrieval with inverted indexes to our knowledge. However, unlike previous supervised sparse retrieval methods such as uniCOIL [166] or SPLADE [68], which compute sequence-level representations for each query/document, deploying token-level sparse vectors into inverted indexes is problematic. As shown in Figure 6.1a, the high-level idea is to convert each sparse token embedding into a sub-query/document and perform token-level inverted list retrieval before aggregation (e.g., scatter sum and max). However, this is incredibly slow in practice as latency increases rapidly when the posting lists are too long.

Therefore, instead of using the naive approach, a more economical way is to approximate the sparsified late interaction with a two-stage system. We first calculate the upper and lower bounds of the sparsified late-interaction scores by swapping the order of the operators in late interaction (see Section 6.2). In this way, the token-level sparse vectors are pooled before indexing to obtain sequence-level representations for the inverted index, yielding fast retrieval speed. After retrieval,

a lightweight score refinement step is applied, where we use the sparse token vectors stored using Scipy [320] to rerank the candidate lists. As shown in Figure 6.1b, this two-stage design allows us to optimize the latency of the retrieval stage without worrying about accuracy loss as the score refinement will compensate for errors from the first stage. Experiments on MS MARCO Passages show that SLIM is able to achieve a similar ranking accuracy compared to ColBERT-v2, while using 40% less storage and achieving an 83% decrease in latency. To sum up, our contributions in this paper are three-fold:

- We are the first to use sparse representations for multi-vector retrieval with standard inverted indexes.
- We provide a two-stage implementation of SLIM by approximating late interaction followed by a score refinement step.
- SLIM is fully compatible with off-the-shelf search toolkits such as Pyserini.

6.2 Efficient Approximation for Sparsified Late Interaction

ColBERT [128] proposes late interaction between the tokens in a query $q = \{q_1, q_2, \dots, q_N\}$ and a document $d = \{d_1, d_2, \dots, d_M\}$:

$$s(q, d) = \sum_{i=1}^N \max_{j=1}^M v_{q_i}^T v_{d_j}, \quad (6.1)$$

where v_{q_i} and v_{d_j} denote the last-layer contextualized token embeddings of BERT. This operation exhaustively compares each query token to all document tokens. The latency and storage of ColBERT are bloated as many tokens do not contribute to either query or document semantics, and thus complex engineering optimizations are needed to make the model practical [277, 276].

Unlike ColBERT, which only uses contextualized token embeddings for computing similarity, SPLADE [68, 67] further utilizes the pretrained Mask Language Modeling (MLM) layer to project each c -dimensional token embedding to a high-dimensional, lexical space V . Each dimension corresponds to a token and has a non-negative value due to the following activation:

$$\phi_{d_j} = \log(1 + \text{ReLU}(W^T v_{d_j} + b)), \quad (6.2)$$

where $\phi_{d_j} \in \mathbb{R}^{|V|}$, v_{d_j} is the token embedding of the j th token of document d ; W and b are the weights and biases of the MLM layer. To compute the final similarity score, SPLADE pools all the token embeddings into a sequence-level representation and uses the dot product between a

Models	MARCO Dev		TREC DL19		TREC DL20		BEIR (13 tasks)	Space&Time Efficiency	
	MRR@10	R@1k	nDCG@10	R@1k	nDCG@10	R@1k	nDCG@10	Disk (GB)	Latency (ms/query)
<i>Models trained with only BM25 hard negatives from MS MARCO Passages</i>									
BM25	0.188	0.858	0.506	0.739	0.488	0.733	0.440	0.7	40
DPR	0.319	0.941	0.611	0.742	0.591	0.796	0.375	26.0	2015
SPLADE	0.340	0.965	0.683	0.813	0.671	0.823	0.453	2.6	475
COIL	0.353	0.967	0.704	0.835	0.688	0.841	0.483	78.5	3258
ColBERT	0.360	0.968	0.694	0.830	0.676	0.837	0.453	154.3	-
SLIM	0.358	0.962	0.701	0.824	0.640	0.854	0.451	18.2	580
<i>Models trained with further pretraining/hard-negative mining/distillation</i>									
coCondenser	0.382	0.984	0.674	0.820	0.684	0.839	0.420	26.0	2015
SPLADE-v2	0.368	0.979	0.729	0.865	0.718	0.890	0.499	4.1	2710
ColBERT-v2	0.397	0.985	0.744	0.882	0.750	0.894	0.500	29.0	3275
SLIM ⁺⁺	0.404	0.968	0.714	0.842	0.702	0.855	0.490	17.3	550

Table 6.1: In-domain and out-of-domain evaluation on MS MARCO Passages, TREC DL 2019/2020, and BEIR. “-” means not practical to evaluate on a single CPU. Latency is benchmarked on a single CPU and query encoding time is excluded.

query and a document as the similarity:

$$s(q, d) = (\max_{i=1}^N \phi_{q_i})^T (\max_{j=1}^M \phi_{d_j}). \quad (6.3)$$

Here, max pooling is an element-wise operation and is feasible for sparse representations as the dimensions of all the token vectors are aligned (lexical space) and non-negative. These two properties play a vital role in the success of SPLADE and are also important for making SLIM efficient, as we shall see later.

Similar to ColBERT’s late interaction, sparsified late interaction also takes advantages of the contextualized embeddings of BERT’s last layer. But different from ColBERT, we first apply the sparse activation in Equation (6.2) before calculating the similarity in Equation (6.1):

$$s(q, d) = \sum_{i=1}^N \max_{j=1}^M \phi_{q_i}^T \phi_{d_j}, \quad (6.4)$$

where ϕ_{d_j} and ϕ_{q_i} are the representations in Equation (6.2). However, as shown in Figure 6.1a, to keep the token-level interaction, we must convert the sparse representation of each token in a query to a sub-query and retrieve the sub-documents from the index. Moreover, the token-level rankings need to be merged to yield the final ranked list using scatter operations, which further increases latency.

Due to the impracticality of the naive implementation, a natural solution would be to approx-

imate Equation (6.4) during retrieval. We first unfold the dot product in Equation (6.4):

$$s(q, d) = \sum_{i=1}^N \max_{j=1}^M \sum_{k=1}^{|V|} \phi_{q_i}^{(k)} \phi_{d_j}^{(k)}, \quad (6.5)$$

where $\phi_{q_i}^{(k)}$ is the k th elements of ϕ_{q_i} . As each ϕ_{q_i} across different q_i all shares the same lexical space and the values are non-negative, we can easily derive its upper-bound and lower-bound:

$$\sum_{i=1}^N \max_{j=1}^M \sum_{k=1}^{|V|} e_{q_i}^{(k)} \phi_{d_j}^{(k)} \leq s(q, d) \leq \sum_{i=1}^N \sum_{k=1}^{|V|} \max_{j=1}^M \phi_{q_i}^{(k)} \phi_{d_j}^{(k)}, \quad (6.6)$$

$$e_{q_i}^{(k)} = \begin{cases} \phi_{q_i}^{(k)}, & \text{if } k = \operatorname{argmax}_k \phi_{q_i}^{(k)} \\ 0, & \text{otherwise} \end{cases} \quad (6.7)$$

Then, the lower-bound score $s_l(q, d)$ and upper-bound score $s_h(q, d)$ can be further factorized as:

$$\begin{aligned} s_l(q, d) &= \sum_{i=1}^N \max_{j=1}^M \sum_{k=1}^{|V|} e_{q_i}^{(k)} \phi_{d_j}^{(k)} = \left(\sum_{i=1}^N e_{q_i} \right)^T \left(\max_{j=1}^M \phi_{d_j} \right); \\ s_h(q, d) &= \sum_{i=1}^N \sum_{k=1}^{|V|} \max_{j=1}^M \phi_{q_i}^{(k)} \phi_{d_j}^{(k)} = \left(\sum_{i=1}^N \phi_{q_i} \right)^T \left(\max_{j=1}^M \phi_{d_j} \right) \end{aligned} \quad (6.8)$$

where both the sum and max operations will be element-wise if the targets are vectors. We see that these two equations resemble the form of SPLADE in Equation (6.3), where queries and documents are encoded into sequence-level representations independently. In this way, $\max_{j=1}^M \phi_{d_j}$ in Equation (6.8) can be pre-computed and indexed offline. To approximate $s(q, d)$ in Equation (6.5), we use the linear interpolation between the lower- and the upper-bound:

$$\begin{aligned} s_a(q, d) &= \beta \cdot s_l(q, d) + (1 - \beta) \cdot s_h(q, d) \\ &= \left(\sum_{i=1}^N (\beta \cdot e_{q_i} + (1 - \beta) \cdot \phi_{q_i}) \right)^T \left(\max_{j=1}^M \phi_{d_j} \right) \end{aligned} \quad (6.9)$$

where $s_a(q, d)$ is the approximate score of SLIM and $\beta \in [0, 1]$ is the interpolation coefficient. As shown in Equation (6.9), we can first fuse the query representation before retrieval and finetune the coefficient β on a validation set. It is worth mentioning that the approximation is applied after the SLIM model is trained using Equation (6.4). During indexing, we use Pyserini [167] to index the sequence level-representation $\max_{j=1}^M \phi_{d_j}$ and use Scipy [320] to store the sparse

token vectors. During retrieval, we use Equation (6.9) to retrieve the top- k candidates with the `LuceneImpactSearcher` in Pyserini. For score refinement, we extract the stored sparse token vectors for the top- k documents and use Equation (6.4) to refine the candidate list. This two-stage breakdown yields a good effectiveness-efficiency trade-off, as we can aggressively reduce the first-stage retrieval latency without too much accuracy loss due to the score refinement.

To further reduce the memory footprint and improve latency for the first-stage retrieval, we apply two post-hoc pruning strategies on the inverted index to remove: (1) tokens that have term importance below a certain threshold; (2) postings that exceed a certain length (i.e., the inverse document frequency is below a threshold). Since all the term importance values are non-negative, tokens with smaller weights contribute less to the final similarity. Moreover, tokens with long postings mean that they frequently occur in different documents, which yield low inverse document frequencies (IDFs) and therefore can be safely discarded.

In summary, the workflow of SLIM can be described as the following:

- **Vector Projection and Sparsification:** Each token embedding from a pretrained language model (like BERT) is projected into a high-dimensional, sparse lexical space. This is achieved using a transformation that includes a non-linear activation (e.g., ReLU) followed by a logarithmic transformation to ensure non-negativity and sparsity. The transformed sparse vectors for each token of the documents are then indexed using an inverted index structure, which allows for efficient retrieval operations.
- **Initial Retrieval Stage:** The system retrieves a preliminary set of documents based on the overlap of sparse token vectors between the query and the documents, using the inverted index. Instead of directly computing the exact interaction scores between query and document tokens, SLIM approximates these interactions using upper and lower bounds derived from the properties of the sparse vectors. This is realized by aggregating the maximum values across the dimensions of the sparse vectors, which significantly reduces computational overhead.
- **Score Refinement:** In the second stage of retrieval, a more precise score is computed for the top candidates retrieved in the initial stage. This score refinement uses the actual sparse vectors and a more detailed computation model to ensure accuracy. To further enhance performance, the model employs pruning strategies to remove less important tokens based on their contribution to the retrieval quality and their frequency across documents.

6.3 Experiments

We evaluate our model and baselines on MS MARCO Passages [226] and its shared tasks, TREC DL 2019/2020 passage ranking tasks [54]. We train the baseline models on MS MARCO Passages

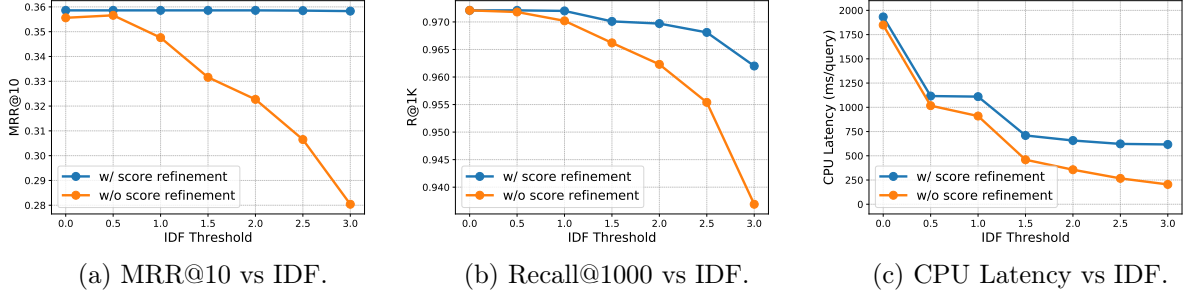


Figure 6.2: Effectiveness-Efficiency trade-offs of SLIM (w/ and w/o score refinement) on MS MARCO Passages.

and report results on its dev-small set and TREC DL 2019/2020 test queries following the same setup in CITADEL [161]. We further evaluate the models on the BEIR benchmark [310], which consists of a diverse set of 18 retrieval tasks across 9 domains. Following previous work [277, 67], we only evaluate 13 datasets due to license restrictions.¹ The evaluation metrics are MRR@10, nDCG@10, and Recall@1000 (i.e., R@1K). Latency is benchmarked on a single Intel(R) Xeon(R) Platinum 8275CL CPU @ 3.00GHz and both the batch size and the number of threads are set to 1.

We follow the training procedure of CITADEL [161] to train SLIM and apply ℓ_1 regularization on the sparse token representations for sparsity control. SLIM is trained only with BM25 hard negatives in MS MARCO Passages, while SLIM⁺⁺ is trained with cross-encoder distillation and hard negative mining. For indexing and pruning, the default token weight pruning threshold is 0.5 and the IDF threshold is 3. We use Pyserini to index and retrieve the documents and use Scipy to store the sparse token vectors (CSR matrix). For retrieval, the fusion weight β in Equation (6.9) is set to 0.01. We first retrieve the top-4000 candidates using Pyserini’s `LuceneImpactSearcher` and then use the original sparse token vectors stored by Scipy to refine the ranked list and output the top-1000 documents.

Table 6.1 shows the in-domain evaluation results on MS MARCO Passages and TREC DL 2019/2020. SLIM and SLIM⁺⁺ manage to achieve accuracy comparable to ColBERT and ColBERT-v2 on MS MARCO Passages. However, the results seem unstable on TREC DL, which is expected as TREC DL has far fewer queries. For out-of-domain evaluation, the results are a little bit worse than the current state of the art but still close to ColBERT. For latency and storage, SLIM has a much smaller disk requirement and lower latency compared to ColBERT. For example, SLIM⁺⁺ achieves effectiveness comparable to ColBERT-v2 with an 83% decrease in latency on a single CPU and using 40% less storage.

¹CQADupstack, Signal-1M(RT), BioASQ, Robust04, TREC-NEWS are excluded.

Next, we show that a two-stage implementation of SLIM provides a good trade-off between effectiveness and efficiency. Figure 6.2 plots CPU latency vs. MRR@10/Recall@1000 on MS MARCO Passages using IDF thresholding for SLIM (w/o hard-negative mining and distillation). We set the minimum IDF threshold to $\{0, 0.5, 1, 1.5, 2, 2.5, 3\}$ and first-stage retrieval candidates top- k to $\{1000, 1500, 2000, 2500, 3000, 3500, 4000\}$ before score refinement (if any). Without the score refinement, the MRR@10 and Recall@1000 scores are high when the IDF threshold is small, but effectiveness drops rapidly when we increase the IDF threshold to trade accuracy for latency. In contrast, with the score refinement step, we see that with about 0.003 loss in MRR@10 and 0.01 loss in Recall@1000, latency improves drastically from about 1800 ms/query to 500 ms/query. The reason is that the score refinement step, which takes a small amount of time compared to retrieval, compensates for the errors from the aggressive pruning at the first stage, which shows that our novel lower- and upper-bound fusion provides a good approximation of SLIM when the IDF threshold is low, and that the score refinement step is important as it allows more aggressive pruning for the first-stage retrieval.

6.4 Summary

In this Chapter, we propose an efficient yet effective implementation for multi-vector retrieval using sparsified late interaction, which yields fast retrieval speed, small index size, and full compatibility with off-the-shelf search toolkits such as Pyserini. The key ingredients of SLIM include sparse lexical projection using the MLM layer, computing the lower- and upper-bounds of the late interaction, as well as token weight and postings pruning. Experiments on both in-domain and out-of-domain information retrieval datasets show that SLIM achieves comparable accuracy to ColBERT-v2 while being more efficient in terms of both space and time.

Chapter 7

Robust Cross-Encoder Rankers with Late Interaction

7.1 Overview

The two-stage retrieve-then-rerank pipeline has been the *de facto* design for many information retrieval systems. Recently, both the retriever and the reranker systems have benefited from the advancement in pretrained language models [168]. When using the pretrained models as cross-encoders, the model views the query and document candidate together, allowing rich token interaction via the attention mechanism at all hidden layers. However, all the information then boils down to the representation of [CLS] token at the final stage, during the computation of the relevance score between the query and the document. This raises the concern of whether the single token representation is sufficient to capture all salient information.

Model	MS MARCO MRR@10	BEIR Avg. nDCG@10	Search Latency
monoBERT	0.390	0.467	1.18s
CELI	0.392	0.491	1.28s

Table 7.1: A preview of comparison between CELI and monoBERT. Detailed results are shown in Table 7.2 and 7.3.

Pretrained language models have also been adopted in the retriever stage in various ways. [126] pioneer in this direction and find that the [CLS] token embedding could well capture query or document representations, whose similarity can be used to indicate the relevance level between the query and document. This line of methods [126, 367], named as *single-vector dense*

retrievers, while being effective for the in-domain scenarios, is found to be less robust on the out-of-distribution (OOD) datasets [310], possibly due to inadequate information at token-level granularity. Methods such as further pretraining or adding token-level interaction have been applied to improve the OOD generalization, where *multi-vector retrievers* [128, 277, 161] perform the best on both the in-domain and OOD effectiveness among neural retrievers [169]. This ability is usually credited to its design that computes the similarity score based on contextualized embeddings of all tokens, which provides richer token interactions compared to the single-vector dense retrievers.

Inspired by the success of token interaction in the retriever systems, we ask the question: *Can cross-encoder also benefit from additional token interaction when computing the final similarity?* In this work, we affirm this hypothesis, showing that additional token interaction in the final-stage similarity computation indeed improves the OOD capacity for cross-encoders. We name our method **CELI** (**C**ross-**E**ncoder with **L**ate **I**nteraction), which incorporates a late interaction layer into the current cross-encoder models. As shown in Table 7.1, CELI improves averaged nDCG@10 on BEIR by 5% (from 0.467 to 0.491), while not sacrificing the in-domain score (0.390 vs. 0.392) and the search latency (1.18s vs. 1.28s). Extensive experiments show that the improvement is consistent over larger-sized models and reranking candidates from various retrievers.

7.2 Late Interaction for Robust Cross-Encoder Ranking

monoBERT. monoBERT [231] is one of the first cross-encoders [199, 229] that apply pretrained transformers in passage retrieval. Given concatenated query q and document d , the model computes relevance scores $s_{q,d}$ from the [CLS] representation on the final layer of BERT [57], formulated as follows [168, 247]:

$$s_m(q, d) = T_{[\text{CLS}]}W + b, \quad (7.1)$$

where $T_{[\text{CLS}]} \in \mathbf{R}^D$ is the [CLS] representation on the final layer, and $W \in \mathbf{R}^{D \times 1}$ and $b \in \mathbf{R}$ are the weight and bias for classification.

Some of the previous works term the models as “mono{BACKBONE}” when initialized from non-BERT pretrained Transformers, such as monoELECTRA [247]. However, since the underlying model structure remains the same, we refer to them all as monoBERT while specifying the backbones where the models are initialized.

Mean-Pooling. To study whether the improvement of CELI is attributed to the interaction between the query and the documents tokens, or simply the additional token information, we add the Mean-Pooling method as a baseline. Instead of computing the similarity score based solely

on the [CLS] representation as in Eq. (7.1), it uses the mean representation of all the tokens:

$$s_m(q, d) = \frac{1}{n} \sum_i^n (T_{tok_i} W + b), \quad (7.2)$$

where T_{tok_i} is the final-layer representation of the i -th token, and n is the total number of tokens in the input sequence. $W \in \mathbf{R}^{D \times 1}$ and $b \in \mathbf{R}$ are the weight and bias for classification, same as Eq. (7.1).

CELI. In this work, we use the simplest version of late interaction proposed by [128]. We first obtain the representation of each token in the query q and document d :

$$v_{q_i} = T_{q_i} W + b; \quad v_{d_j} = T_{d_j} W + b, \quad (7.3)$$

where q_i and d_j represent the i -th token of query q and the j -th token of document d , respectively. Similar to Eq. (7.1), $T \in \mathbf{R}^D$ refers to each token representation on the final layer. $W \in \mathbf{R}^{D \times D_{tok}}$ and $b \in \mathbf{R}^{D_{tok}}$ are the weight and bias in a projection layer, projecting the T_{tok} to a lower dimension $D_{tok} < D$.

With token representations v_{q_i} and v_{d_j} , the late interaction first computes the inner product scores between all pairs of query and document tokens, then sums up the max similarity score for each query token against all document tokens:

$$s_l(q, d) = \sum_{q_i} \max_{d_j} (v_{q_i}^T v_{d_j}). \quad (7.4)$$

Eq. (7.4) shares the same formulation as in the first-stage retrievers, and only differ in that the token representations T_{q_i} and T_{d_j} embed information from both the query and document, whereas in first-stage retrievers, they are computed independently from each other, with T_{q_i} perceiving no information from document d and vice versa.

During training, we compute the LCE loss on s_m and s_l , respectively:

$$\begin{aligned} \mathcal{L} = & lce(s_m(q, d^+), s_m(q, d_1^-), \dots, s_m(q, d_n^-)) \\ & + lce(s_l(q, d^+), s_l(q, d_1^-), \dots, s_l(q, d_n^-)), \end{aligned}$$

where d^+ is the positive document and $\{d_i^-\}_{i=1}^n$ are the negative documents to the query q .

At inference time, we sum the two scores as the final relevance score, i.e., $s_{final} = s_m + s_l$.¹

¹We have explored adding weighting terms for s_m and s_c , but only observed marginal gains. Thus we report the simplest formulation here.

7.3 Experimental Setup

All cross-encoders are trained on MS MARCO [226], a dataset composed of queries from Bing search log and a collection of passages sourced from the general Web. It contains 8.8M passages, over 500k query–document pairs for training, and 6980 queries for inference.

We implement the model based on Capreolus [378, 380], an IR toolkit for end-to-end neural ad hoc retrieval. All training configurations follow [247]: We train MS MARCO for 30k steps with a learning rate $1e - 5$ and a batch size 16. We use linear warm-up on the first 3k steps, then linearly decay the learning rate on the rest of the steps. Cross-encoders are trained on LCE loss [73, 247] with 7 negative samples.² We experimented with three backbones, all available on HuggingFace [361]: MiniLM [344],³ ELECTRA_{base},⁴ and ELECTRA_{large} [48].⁵

We use MS MARCO [226] for the in-domain evaluation and 13 datasets from BEIR [310] for OOD evaluation, which covers 10 domains including Wikipedia, Finance, Scientific, Quora, and so on.

At the inference stage, we always rerank top- $1k$ results from the first-stage retrievers. On MS MARCO, we use TCT-ColBERT [175] as the retriever following [247]. On BEIR, we use an extensive list of retrievers that covers the categories of sparse, single- and multi-vector dense retrievers. Retrievers results are produced using Pyserini [167], BEIR [310], or ColBERT [128] repository.⁶ Following the datasets standard, we report MRR@10 on MS MARCO and nDCG@10 on BEIR.

7.4 Results and Analysis

Table 7.1 provides a preview of the efficacy of CELI. In this section, we first demonstrate that such improvement is consistent over different model sizes and the first-stage retrievers, then analyze how the projected token dimension and the query length impact the improvement.

7.4.1 Model Size

Previous papers find that models with a larger number of parameters can better generate on unseen distribution [227]. Motivated by this observation, we examine whether the improvement brought by late interaction diminishes with increasing model sizes.

²We use Quadro RTX 8000 GPUs and A6000 for the experiments. On RTX 8000, the ELECTRA_{base} models took approximately 8 hours for cross-encoder training.

³[microsoft/MiniLM-L12-H384-uncased](https://github.com/microsoft/MiniLM-L12-H384-uncased)

⁴[google/electra-base-discriminator](https://github.com/google/electra-base-discriminator)

⁵[google/electra-large-discriminator](https://github.com/google/electra-large-discriminator)

⁶<https://github.com/stanford-futuredata/ColBERT>

Backbone	Model	MS MARCO MRR@10	BEIR nDCG@10
MiniLM	monoBERT	0.390	0.467
	Mean-Pooling	0.390	0.481
	CELI	0.392	0.491
ELECTRA _{base}	monoBERT	0.400	0.481
	Mean-Pooling	0.402	0.483
	CELI	0.402	0.494
ELECTRA _{large}	monoBERT	0.413	0.507
	Mean-Pooling	0.412	0.516
	CELI	0.413	0.524

Table 7.2: In-domain (MRR@10 on MS MARCO) and OOD (averaged nDCG@10 on BEIR) scores of CELI and two baselines (i.e., monoBERT and Mean-Pooling).

Results show that the contribution of late interaction is consistent over model size. Table 7.2 shows in-domain and OOD scores with the models initialized from three different sizes of backbones: MiniLM, ELECTRA_{base}, and ELECTRA_{large}.⁷ The MS MARCO results rerank the top-1k candidates from TCT-ColBERT, and the BEIR results rerank the top-1k candidates from BM25.

While we observe higher average scores on BEIR as the model size increases, echoing the previous finding that larger models demonstrate better OOD generalization ability, the improvement brought by token information is consistent across the backbones. On all three models, CELI consistently improves over the two baselines. Additionally, the in-domain scores on the other two backbones are not affected as well, suggesting that the “free” gain is consistent over different model sizes.

7.4.2 First-Stage Retriever

We then extend the experiments into an extensive list of first-stage retrievers, where the retrievers are categorized as sparse, single-vector, and multi-vector dense retrievers.

Table 7.3 shows the results on BEIR reranking candidates from 9 different retrievers, covering all three categories mentioned above. Looking at the averaged nDCG@10 on BEIR, we find that late interaction consistently improves the OOD capacity when using retrievers of different natures, bringing a similar degree of improvement of 0.02–0.03.

⁷MiniLM, ELECTRA_{base}, and ELECTRA_{large} have 33M, 110M, and 340M parameters respectively.

Model	Sparse			Multi-vector Dense	
	BM25	uniCOIL	SPLADE	ColBERT v2	
monoBERT	0.467	0.426	0.469	0.467	
CELI	0.491	0.452	0.492	0.493	

Model	Single-vector Dense				
	DPR (NQ)	DPR (MS)	ANCE	TCT	TAS-B
monoBERT	0.451	0.474	0.471	0.470	0.472
CELI	0.472	0.495	0.493	0.494	0.494

Table 7.3: Averaged nDCG@10 on BEIR, reranking the top-1k candidates from each retriever. **TCT**: TCT-ColBERT. **DPR (NQ/MS)**: DPR finetuned on NQ [138] or MS MARCO, respectively.

	Projected Token Dimension (D_{tok})	MS MARCO MRR@10	BEIR nDCG@10
(1)	$D_{\text{tok}} = 1$	0.3920	0.4890
(2)	$D_{\text{tok}} = 32$	0.3920	0.4914
(3)	$D_{\text{tok}} = 128$	0.3920	0.4910
(4)	$D_{\text{tok}} = 384$	0.3900	0.4911

Table 7.4: MRR@10 on MS MARCO and nDCG@10 on BEIR of CELI with token representation in different dimensions (D_{tok} in Eq. (7.3)). We report scores to 4 digits here as the values are close in all conditions.

7.4.3 Token Dimensions

In first-stage retrieval, it is common to project the token representation into lower dimensions as restricted by indexing storage space and search efficiency. However, the representations are computed on the fly for cross-encoders, thus using token representations in higher dimensions brings no additional storage cost and only minor searching latency in the context of cross-encoders. We therefore examine whether using higher token dimensions D_{tok} brings additional improvements.

Results are shown in Table 7.4, where row (2) corresponds to the BM25 results reported in Table 7.3. Comparing rows (1–4), we find that the token dimensions have little impact on the OOD effectiveness: surprisingly, using $\text{dim} = 1$ already obtains 0.4890 on average BEIR as shown in on row (1), while increasing the dimension to $\text{dim} = 32$ and onward only provides marginal improvement.

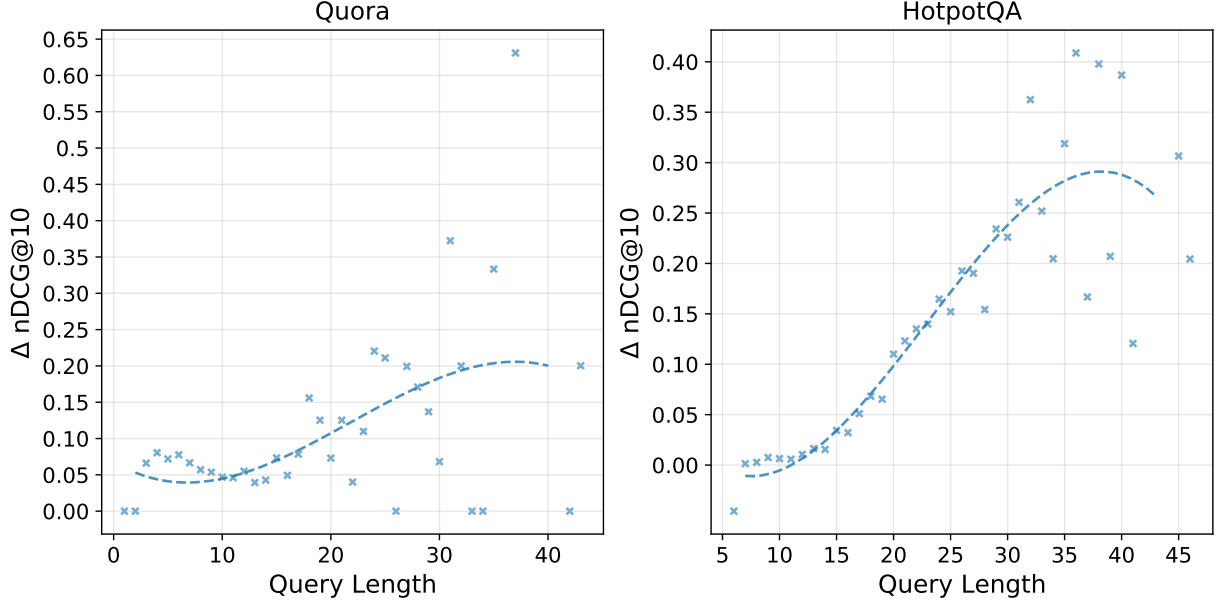


Figure 7.1: Δ nDCG@10 improvement from late interaction on queries over different lengths. Each point represents the average of Δ nDCG@10 improvements over the query of the corresponding length. The line is the least square polynomial fit of the points.

7.4.4 Query Length

Finally, we present our analysis of how the late interaction improves the OOD capacity of cross-encoders, finding that query length is a prominent indicator of the per-query improvement. Figure 7.1 plots the distribution of Δ nDCG@10 improvement by late interaction according to the query length on Quora and HotpotQA, two datasets included in BEIR.⁸ Specifically, each point represents the average of Δ nDCG@10 improvements over the query of the same length (same coordinate on the x-axis). We additionally plot an approximated polynomial line based on the points to better reveal the relationship between the query length and the improvement on Δ nDCG@10.

On both datasets, we observe a clear tendency that the late interaction brings higher improvement on longer queries. While Figure 7.1 is based on results using BM25 as the retriever, we have similar observations when reranking candidates from the other retrievers.

⁸Length determined as the number of query tokens delimited by whitespace.

7.5 Related Work

[231] is one of the first cross-encoders that apply pretrained language models on the passage retrieval task. It considers retrieval as a classification task and uses transformer encoders following the formulation of the next sentence prediction (NSP) pretraining task in BERT, where only the [CLS] vector is used to classify the query–document pair and compute the relevant score. Afterward, CEDR [199] proposes to incorporate fine-grained token interaction. However, it requires extra complex computations at all layers, which brings difficulty to implementation and adds higher computational overhead.

This line of cross-encoders has been studied and extended to other model architectures: [229] and [395] build cross-encoders on encoder-decoder architecture (e.g., T5, [255]), and [197] extend it to decoder-only architecture (e.g. LLaMA-2, [315]). Another line of cross-encoders reranks the document candidates according to the query likelihood given a passage, usually based on generative models [231, 272].

Recent works on first-stage retrieval have demonstrated the effectiveness of adding sparse information into dense retrieval [42]. The combination of the token information and dense [CLS] vector could also be done explicitly, by either adding the scores computed from [CLS] and token information or concatenating aggregated token vectors to the [CLS] vector [72, 172]. The multi-vector dense models could also be viewed under this category, where the token representation vectors jointly contribute to the relevancy computation along with the [CLS] vector [128, 161].

Our work is also connected to the interaction-based methods predating pretrained language models, where the text relevance is usually predicted based on the fine-grained similarity matrix between queries and document tokens [296, 190, 105, 240].

7.6 Summary

In this Chapter, we show that adding late interaction to existing cross-encoders brings visible improvement to its OOD capacity without hurting in-domain effectiveness, even though the cross-encoder already processes the token interaction in earlier layers. Extensive experiments on different model sizes and first-stage retrievers show that this improvement is consistent, and according to our analysis, the improvement is more prominent on longer queries. Our findings suggest that boiling all information into the [CLS] token is a sub-optimal use for cross-encoders, and further studies are required to better explore their capacities.

Chapter 8

Multi-Step Query Expansion for Cross-Encoder Rankers

8.1 Overview

Query expansion has been a core technique in information retrieval for over half a century [274, 266, 1]. The goal is to increase the retrieval accuracy by adding additional terms to the query. Conventional methods such as RM3 [144] leverage Pseudo-Relevance Feedback (PRF) to select terms from the documents retrieved for the original query as expansions [158]. Recently, large language models (LLMs) demonstrate their effectiveness in generating expansion terms for retrieval, which is known as generative query expansion [117, 341]. However, both methods or their combinations are mainly considered for improving the recall and precision of *first-stage retrievers*, yet their influence on the generalization of *second-stage, cross-encoder rerankers* remains under-explored.

This problem is interesting as additional terms usually contain more information about the query, yet they are rarely used in cross-encoder ranking. A recent study from [356] explores generative query expansion for different retrievers and rankers. They found that weaker models benefit more from expansions while stronger rankers are hurt in most cases. This counter-intuitive observation calls for further examination of current expansion methods and whether there is a way to improve the results of strong rankers using query expansion. To verify the conclusion, we also apply some popular query expansion methods to two state-of-the-art cross-encoder rankers, RankT5 [395] and MonoT5 [229]. As shown in Table 8.1, the results are consistent with [356] where we observe that the nDCG@10 scores of both rankers on TREC DL 2019/2020 and BEIR are compromised using either PRF or LLMs. More results are shown in Section 8.4 and 8.5.

In this Chapter, we identify two important steps for successfully using query expansion in cross-

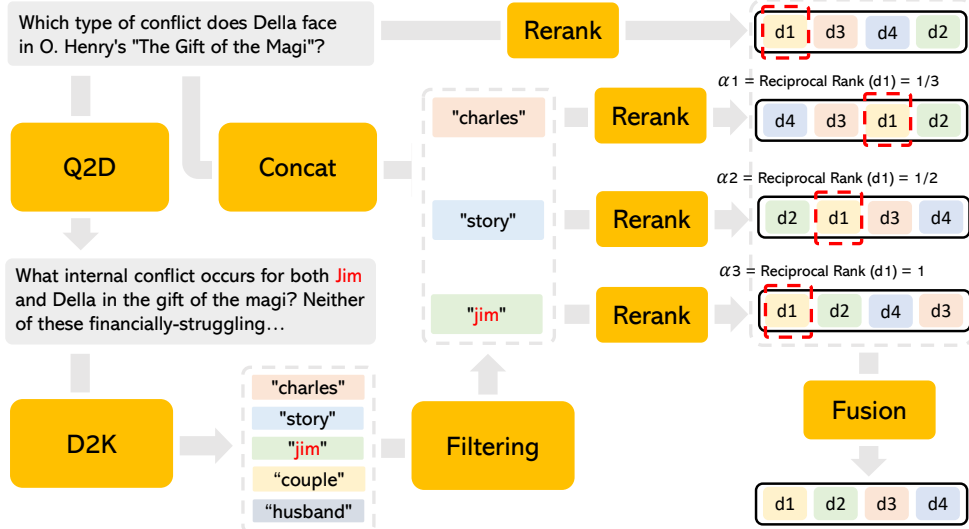


Figure 8.1: A flowchart of our method including generation, filtering, and fusion to select high-quality keywords for zero-shot ranking task with strong cross-encoder rankers.

Methods	DL19	DL20
RankT5 _{base} [395]	0.737	0.736
- w/ RM3 [144]	0.681	0.695
- w/ query2doc [341]	0.572	0.637
- w/ query2keyword [117]	0.690	0.688
- w/ Ours	0.751	0.752
MonoT5 _{base} [229]	0.695	0.720
- w/ RM3	0.673	0.693
- w/ query2doc	0.473	0.613
- w/ query2keyword	0.672	0.682
- w/ Ours	0.724	0.730

Table 8.1: nDCG@10 on TREC DL 2019/2020. Directly applying existing query expansion methods on strong cross-encoder rankers can cause effectiveness deterioration.

encoder rankers, which are not well explored by existing work focusing on retrievers: high-quality keyword generation and minimally-disruptive query modification. First, cross-encoder rankers mainly aim to improve precision-related metrics such as nDCG@10 which are sensitive to noisy keywords, while retrievers care more about recall where some low-quality expanded keywords are less influential. Therefore, strong rankers such as RankT5 have higher demand on the generation quality. Second, cross-encoder rankers are heavily based on token interactions, making them sensitive to the distributional shift in queries (e.g., number of tokens, input formats) compared to retrievers. Therefore, inserting documents in a query [341] might be less desirable.

In our experiments, we find that the most effective way is to first use an LLM to generate high-

quality, short keywords through a reasoning chain [353]. We then follow self-consistency [346] to run the above process multiple times to filter noisy keywords and select top-k candidates, ensuring the quality of the generated keywords. To mitigate distributional shift in query, we insert each keyword independently with the query and use reciprocal rank weighting [53] to combine the ranking results. Our pipeline manages to improve the nDCG@10 over the baselines for cross-encoder rankers such as RankT5 [395] on both BEIR and TREC DL 2019/2020, while other baselines that have been found effective for retrievers fail to improve such strong rankers. Our study provides a preliminary yet novel research foundation for researchers to explore query expansion for cross-encoder rankers.

8.2 Related Work

Query Expansion and Fusion Early research on query expansion concentrated on utilizing either lexical knowledge bases [27, 33, 323] or Pseudo-Relevance Feedback (PRF) [110, 270, 391]. Recent studies show that scaling up LLMs through pretraining with more extensive and higher-quality corpora [45, 34, 237, 314] can result in higher capabilities. Researchers have used large language models for generating keywords in the context of query expansion [341, 50, 117, 345].

The effectiveness of query variant fusion has been proven in previous works. [21] pioneered the fusion of multiple query variations into a single ranked list. [18] introduced Rank Biased Centroids for effective query variation fusion. [24] furthered this by applying reciprocal rank fusion [53] and CombSUM [21] with double fusion.

LLM-Based Neural Rankers MonoBERT [231] stands out as one of first cross-encoders for text reranking tasks. CEDR [199] introduces a more intricate approach by incorporating token representations at all layers of the Transformer using pre-BERT neural rerankers. More potent rankers based on LLMs have emerged to directly score the relevance between queries and documents [395, 229, 389]. Most recently, LLMs have showcased remarkable efficacy when tasked with few/zero-shot text ranking such as LRL [198], RankGPT [304], RG- k L [394], RankVicuna [249], and RankLlama [197]. Alternatively, they can perform pairwise comparisons between passages, as demonstrated by PRP [251]. Despite the zero-shot effectiveness, the multiple decoding passes render them slow and non-parallelizable.

8.3 Framework and Implementation

8.3.1 Cross-Encoder Ranking

Given a query q , the text retrieval or ranking task is to return a sorted list of documents $\{d_1, d_2, \dots, d_k\}$ from a large text corpus $\mathcal{C}$ to maximize a metric of interest. In this paper, we

assume a set of candidate documents $\{d_1, d_2, \dots, d_k\}$ generated by a first-stage retriever are given and focus on the second-stage reranker to re-order the candidate documents. The query-document pairs are encoded together for fine-grained token-level interactions:

$$s(q, d) = \phi(\text{concat}(q, d)), \quad (8.1)$$

where ϕ is the reranker and s is the similarity score. The “concat” function is implemented using special tokens as indicators, such as “Question: q Document: d ”. In the following subsection, we will introduce the framework we use for keyword generation and selection to improve the results of cross-encoders.

8.3.2 High-Quality Keyword Generation

The first step of query expansion is to generate keywords $\{w_1, w_2, \dots, w_i\}$ semantically similar to the query q . There are generally two sources of signals: The classical approach which involves corpus-based signals through Pseudo-Relevance Feedback (PRF) or more recent approaches leveraging signals from LLMs by prompting [117]. Notice that some LLM-based methods like Q2D [341] and HyDE [75] generate much longer passages or documents rather than keywords, which drift too much from the distribution of queries and deteriorate cross-encoder reranker effectiveness. In the following components of this framework, we do not consider expansion other than keywords, but will show the results of using excessively long expansion in Section 8.4. We explore four different methods for keyword generation, including both LLM-based and PRF-based methods, as well as their combinations:

- **PRF-based** methods like RM3 [144] to extract keywords from the retrieved documents $d_1, d_2, \dots$
- **LLM-based** methods to generate keywords $w_1, w_2, \dots$ like Q2K [117].
- **PRF + D2K**, which uses an LLM to extract keywords $w_1, w_2, \dots$ from the retrieved documents $d_1, d_2, \dots$
- **Q2D2K**, which uses the LLM to generate detailed documents $d'_1, d'_2, \dots$ first and then selects a set of keywords $w_1, w_2, \dots$

PRF + D2K and Q2D2K are inspired by Q2D [341] and HyDE [75]. These methods are problematic when the generated documents are directly used as queries for cross-encoder rankers due to the query distribution shift, but are useful when the generated documents are summarized into keywords for expansions. The assumption is that documents generated by a well pretrained LLM

Methods	AA	SF	NQ	Fe	Qu	HQ	FQ	CF	BA	SD	T2	TN	S1	DB	NF	R0	TC	BEIR Wiki+News	
RankT5 _{base}	0.323	0.750	0.565	0.826	0.814	0.732	0.414	0.255	0.541	0.174	0.383	0.439	0.299	0.448	0.375	0.525	0.782	0.508	0.557
- w/ RM3	0.330	0.722	0.546	0.817	0.831	0.689	0.377	0.257	0.473	0.165	0.360	0.443	0.291	0.393	0.358	0.481	0.746	0.487	0.539
- w/ query2doc	0.271	0.671	0.511	0.810	0.749	0.640	0.300	0.184	0.358	0.140	0.275	0.419	0.231	0.367	0.331	0.447	0.716	0.437	0.502
- w/ query2keyword	0.315	0.739	0.546	0.841	0.807	0.725	0.381	0.257	0.500	0.168	0.353	0.470	0.280	0.398	0.365	0.496	0.741	0.493	0.556
- w/ Ours	0.324	0.752	0.577	0.846	0.822	0.744	0.412	0.261	0.542	0.176	0.390	0.454	0.292	0.452	0.377	0.541	0.781	0.514*	0.570*
MonoT5 _{base}	0.242	0.713	0.553	0.808	0.785	0.705	0.364	0.216	0.495	0.161	0.406	0.417	0.286	0.423	0.312	0.415	0.685	0.470	0.519
- w/ RM3	0.249	0.698	0.544	0.817	0.799	0.666	0.354	0.219	0.453	0.156	0.390	0.427	0.291	0.412	0.314	0.396	0.632	0.465	0.511
- w/ query2doc	0.188	0.640	0.508	0.784	0.353	0.594	0.302	0.108	0.353	0.127	0.339	0.398	0.242	0.368	0.293	0.391	0.590	0.392	0.459
- w/ query2keyword	0.232	0.697	0.542	0.828	0.774	0.702	0.341	0.215	0.449	0.156	0.378	0.453	0.284	0.397	0.293	0.359	0.642	0.455	0.517
- w/ Ours	0.240	0.720	0.569	0.831	0.793	0.719	0.369	0.223	0.499	0.163	0.404	0.431	0.288	0.443	0.313	0.420	0.671	0.476*	0.532*

Table 8.2: nDCG@10 scores on BEIR. TC=TREC-COVID, NF=NFCorpus, NQ=NaturalQuestions, HQ=HotpotQA, FQ=FiQA, AA=ArguAna, T2=Touché-2020, Qu=Quora, DB=DBPedia, SD=SCIDOCS, Fe=FEVER, CF=Climate-FEVER, SF=SciFact, S1=Signal-1M, BA=BioASQ, R0=Robust04, TN=TREC-NEWS. *: pass the paired t-test against the other baselines ($p < 0.01$).

can already answer the question or at least contain helpful keywords. The intuition is also similar to recitation-augmented language models [305] where more knowledge can be elicited before fulfilling the task.

As mentioned in Section 8.1, the precision-related metrics, which cross-encoder rankers aim to optimize, are more sensitive to the noise in the expanded keywords. If there are noisy keywords generated from the previous stage, the effectiveness of rankers are more likely to be affected compared to retrievers. Hence, we argue that it is necessary to add a filtering stage in this framework to remove noisy keywords and increase the reliability of the expansion. We leverage self-consistency [346] in LLM literature for filtering. For LLM-based keyword generation methods which involves stochasticity, we repeat the keyword generation method multiple times and select the top-k keywords that have the highest majority votes (i.e., frequency). For deterministic methods like RM3, we simply take the keywords with the highest RM3 keyword weights.

8.3.3 Minimally-Disruptive Query Modification

One way to insert keywords is to directly concatenate them with the query. The concatenation function we use is “Question: q w_1 w_2 ... w_i Document: d ”. However, as mentioned in Section 8.1, the increasing number of keywords or excessively long expansion may overwhelm the original query, especially for cross-encoder models which rely more on query-document token-interaction. In Section 8.4, we will show that even increasing the number of keywords to 3 will result in degraded precision. To mitigate the distributional shift in query, another way is to concatenate each keyword individually with the query and fuse the final ranking results together [189]. Inserting only 1 keyword is the minimally-disruptive expansion we found for cross-encoder, which is proved to be

Methods	DL19	DL20	BEIR	Wiki+News
Q2D2K-fusion	0.751	0.752	0.514	0.570
Q2K-fusion	0.750	0.748	0.510	0.562
PRF + D2K-fusion	0.745	0.733	0.510	0.565
RM3-fusion	0.741	0.737	0.510	0.560

Table 8.3: nDCG@10 score of Q2D2K, Q2K, RM3, and PRF + D2K with fusion based on RankT5.

very robust on multiple datasets. Specifically, the new similarity scoring function will be:

$$s(q, d) = \sum_i \alpha_i \cdot \phi(\text{concat}(q, w_i, d)), \quad (8.2)$$

where the concatenation function $\text{concat}(q, w_i, d)$ is implemented as “Question: q w_i Document: d ” and α_i is the weight for the expansion w_i . We find this formulation more effective than concatenating all keywords at once as the number of keywords increases.

Previous works [393, 22] have also found that ensembling runs from different models or data augmentation can be effective for ranking. After obtaining the candidate keywords, we concatenate each keyword independently with the original query and then feed it in the cross-encoder ranker model to rerank the top-1000 candidate documents retrieved by BM25 to get a ranked list. For the fusion weights in Equation (8.2), we follow the previous work [53, 56] to weight the ranked lists using the reciprocal rank of the top-1 document in retrieved list: $\alpha_i = \frac{1}{\text{Rank}(d^+, D_i)}$ where $\text{Rank}(d^+, D_i)$ is the rank of the top-1 document d^+ retrieved for the original query in a candidate list of expansion w_i . Finally, we combine the ranking list of the original query in case all the expansions are not helpful.

8.4 Experiments

Models and Datasets For the keyword generation, we use Flan-PaLM2-S [45]. For cross-encoder ranking, we test two different rankers: MonoT5 [229] and RankT5 [395]. We reproduce the MonoT5 model using the point-wise loss in [395]. For in-domain evaluation, we evaluate on TREC DL 2019 and 2020 [54], containing 43 and 54 test queries, respectively. The relevance sets are densely labelled with scores from 0 to 4. For out-of-domain evaluation, we evaluate on 17 datasets from BEIR.

Evaluation We report the nDCG@10 metric [118] as datasets in BEIR and TREC DL are densely labeled, and the top-10 setting reflects the common use case in applications. As the LLM is finetuned on instructions set which has large overlap with the Wikipedia and News corpus in BEIR, we also report the average score on Natural Questions, FEVER, Climate-FEVER, HotpotQA, TREC-News, and Robust04 datasets (Wiki+News) besides the main score.

Inference Pipeline We use BM25’s top-1000 retrieval results as the candidates for reranking using Pyserini [167]. For RM3, we use Pyterrier [200] to extract keywords from the retrieved documents.

For keywords generation method Q2D2K, we follow the instruction template in Promptagator [56] and ask Flan-PaLM2-S to generate 2 documents based on the query and then extract 5 keywords for each document. We then repeat the process 3 times to obtain 30 keywords. For keyword generation method PRF + D2K, we replace the generated documents by BM25 retrieved documents. By default, we use Q2D2K as the keyword generation component. We then run self-consistency and select the top-3 keywords from the keyword candidates and concatenate each of them individually with the query before feeding them into the ranker.

For fusion, we use the reciprocal ranks of the top-1 document retrieved for the original query in each expansion’s reranked list as weights. We finally fuse the aggregated expansion results with the original reranking list from the cross-encoder ranker as regularization, with a coefficient of 0.3. Coefficients range from 0.1 to 0.3 work similarly but not included due to space limit.

Results Table 8.1 and 8.2 show the main results on TREC DL19/20 and BEIR benchmark, where directly applying the query expansion pipeline in retrieval to cross-encoder ranker reranking results in deteriorated effectiveness even with top-3 keyword concatenation, while our method can improve over the original cross-encoder ranker scores on both TREC DL and BEIR. The improvement on TREC DL and Wiki+News BEIR datasets are more significant compared to other datasets in BEIR, which results from the instruction finetuning step of PaLM2 as previously mentioned.

8.5 Ablation Analysis

RQ1: Whether to use LLMs or PRF for keyword generation? To compare the keyword generation quality of different methods, we fix the keyword insertion and fusion procedure while varying the keyword generation methods. Table 8.3 shows the comparison results. We can see that the keywords generation quality is reflected in the nDCG@10 scores, where our Q2D2K method manages to outperform other LLM or PRF based methods on DL 19/20 and BEIR, reflecting higher keyword quality. Besides that, fusion is also very important for maintaining cross-encoder ranker zero-shot effectiveness which will be discussed in detail in RQ2.

RQ2: How to use these keyword expansions? Figure 8.2 plots the number of keywords used for ranking fusion and its influence on the final nDCG@10 score. We can see that the improvement of our proposed method (which uses Reciprocal Rank Weighting) peaks at 3 expanded keywords

Methods	DL19	DL20	BEIR	Wiki+News
RankT5 _{base}	0.737	0.736	0.508	0.557
+mean pooling	0.748	0.750	0.514	0.572
+reciprocal rank	0.751	0.751	0.514	0.573
+original query	0.751	0.752	0.514	0.570
MonoT5 _{base}	0.695	0.720	0.470	0.519
+mean pooling	0.700	0.727	0.471	0.533
+reciprocal rank	0.713	0.733	0.475	0.532
+original query	0.724	0.730	0.476	0.532

Table 8.4: Incremental ablation on the keyword fusion process. Details are introduced in Section 8.5.

Methods	DL19	DL20
Mean pooling	0.748	0.750
Reciprocal rank	0.751	0.751
Top-k overlap	0.741	0.750
Ranker’s entropy	0.729	0.746
KL divergence	0.741	0.749
WS distance	0.743	0.751

Table 8.5: Weighting methods for fusion on DL 19/20 based on RankT5. nDCG@10 scores are reported.

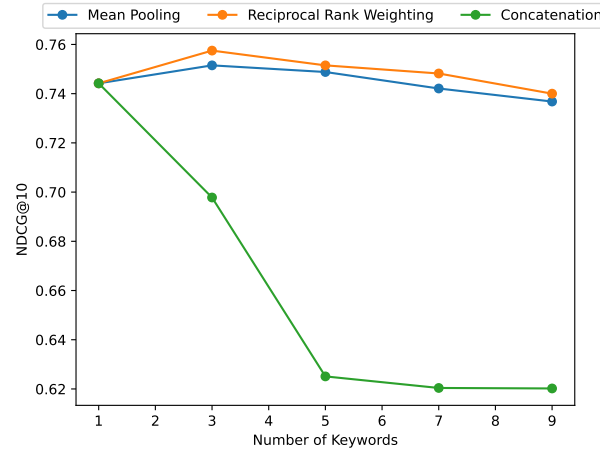


Figure 8.2: nDCG@10 score on DL19 using different number of keywords for RankT5.

and gradually diminishes with the addition of more keywords. Although the generated keywords are useful, they still bring noise which escalates as the keyword number increases.

Table 8.4 shows the ablation of the keyword fusion process, which mainly includes ranking

average, reciprocal rank weighting, and combining with original query ranking results. Adding mean fusion and reciprocal weighting consistently brings improvement to the model. As for the original query fusion, we view it as an expansion and combine its ranking with the other expansion’s ranking results using a default value of 0.3 for zero-shot evaluation. The improvement is less consistent as we did not perform hyper-parameter search on the fusion coefficient due to zero-shot evaluation, but instead use a default value of 0.3 for combining with the original query ranking results. Figure 8.2 also shows that fusion is more robust compared to keyword concatenation as the keywords increase.

Table 8.5 shows different fusion methods we tried on TREC DL 19/20. Mean pooling and reciprocal rank weighting are reported in Table 8.4. For top-k overlap, we take the overlap between the original query’s candidate list and the other expansion’s candidate lists as weights for α_i in Equation (8.2). For ranker’s entropy, we normalized the retrieved scores into a distribution for each expansion and use the reciprocal entropy of the distribution as weights. For KL divergence and Wasserstein distance, they are similar to the ranker’s entropy except that they calculate the distances between the original query’s distribution and the other expansion’s distributions. We also take the reciprocal of this distance as weights for fusion. We can see that among all the fusion techniques, the reciprocal rank weighting method has the best nDCG@10 scores on both DL 19 and 20, demonstrating the robustness of this simple fusion method.

8.6 Summary

In this Chapter, we examine the possibility of improving the generalization of cross-encoder rankers using query expansion based on the study of [356]. Our solution is to leverage an LLM to generate high-quality, concise keywords through a reasoning chain and individually evaluate the ranking scores of each expansion before aggregating them together. We observe significant improvement on BEIR and TREC DL 2019/2020 over directly using the popular query expansion methods.

Chapter 9

Certified Error Control for Two-Stage Relevance Ranking

9.1 Overview

A two-stage relevance ranking architecture has been an indispensable component for knowledge-intensive natural language processing tasks such as information retrieval (IR) [210] and open-domain question answering (OpenQA) [41]. Such a system usually consists of a high-recall first stage that retrieves a set of documents from a massive corpus and a high-precision reranker that improves the ranking of the retrieved candidate sets. The first-stage retrieval, often implemented by approximate nearest neighbor search [121] or inverted index search [167], is quite efficient while the second-stage reranking usually has high latency due to the trend of using over-parameterized pretrained language models and large candidate set sizes. Previous work in early exiting proposed to predict the ranking score using only a partial model [363, 192, 35] or prune the candidate set before reranking [342, 66] to trade accuracy off against speed. However, such methods lack accurate error control which can not provide guarantees to satisfy the exact accuracy constraints specified by users.

In this Chapter, we focus on *candidate set pruning* methods of early exiting for two-stage relevance ranking, and we show that a simple score-thresholding method can yield predictions with *certified* error control using the prediction sets theory [359, 360, 333, 19]. Instead of predicting a single target, prediction sets will yield sets that contain the desired fraction of the population with high probability. Moreover, we allow users to specify the error tolerance of their custom metrics and our method will return the pruned set of candidates that satisfies the constraints with a finite sample guarantee. Our method makes no assumption about the data distributions and models, except that the calibration data are exchangeable with the test data. Fig. 9.2 illustrates the calibration and test procedures for certified error control of candidate set pruning.

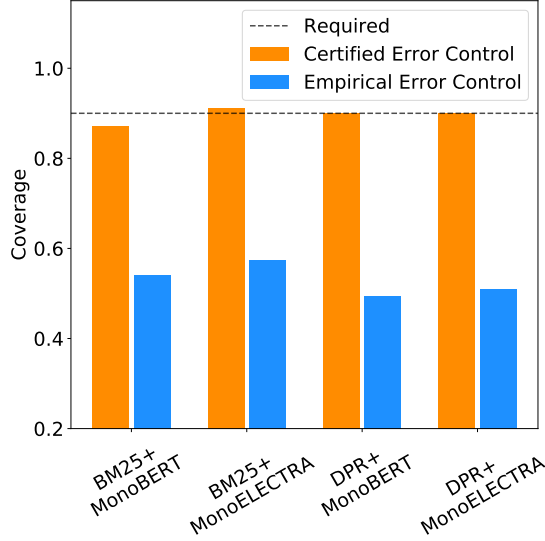


Figure 9.1: Comparisons between certified and empirical error control methods on MS MARCO Passage v1. Coverage: Percentage of 100 independent runs that satisfies a pre-specified $\text{MRR@10} \geq 0.35$ (the dotted line).

Challenges. However, directly applying prediction set methods to relevance ranking can be problematic. Unlike classification, where the true label will be eventually included in the predicted set as the set size grows, in relevance ranking, users care more about the *rank* of the positives. Therefore, a rank requirement from users might not always be satisfied as the ranking system might not be good enough to rank the positive documents correctly no matter how large the candidate set is.

Contributions. To this end, we propose to correct the coverage level pre-specified by the user if the constraints are impossible to satisfy, and it is up to the user to decide whether to abandon the prediction or accept the corrected results. Empirically, we evaluate our method on IR and OpenQA benchmarks, including MS MARCO Passage v1 [226] and Quora [310]. We also test different combinations of retrievers and rerankers for the two-stage relevance ranking system under both in-domain and out-of-domain settings. Fig. 9.1 shows the results of certified and empirical error control methods using different ranking systems on MS MARCO Passage v1, and we can see that the empirical method fails to provide the required coverage while our method succeeds to meet the requirement (see Tbl. 9.3 for more details). For example, if we pre-specify $\text{MRR@10} \geq 0.38$, we can reduce the average candidate set size from 1000 to 27, increasing the reranking speed by $37\times$ while satisfying the constraint with 90% empirical coverage. We further confirm that the risk-confidence correction of our method is able to consistently correct the risk/confidence when

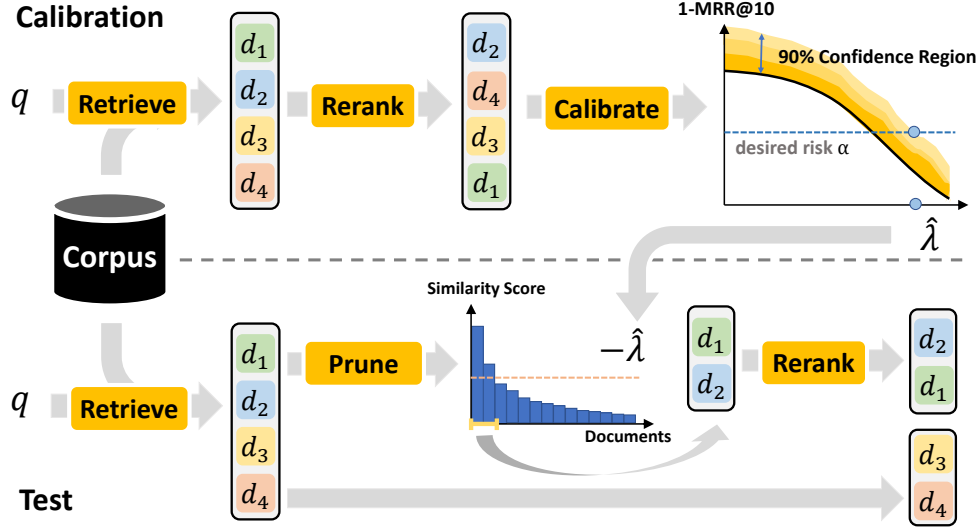


Figure 9.2: Calibration and test procedures of candidate set pruning with certified error control. During calibration, we use the confidence region and desired risk α to find the threshold $-\hat{\lambda}$. During testing, we use the threshold to prune the retrieved documents, which guarantees control of the expected loss under α with probability $1 - \delta$.

the pre-specified conditions are impossible to achieve.

To better understand this concept, imagine the user is searching for “winter boots” on an online shopping site. The site initially pulls up thousands of boots. To speed up the process and not overload the system, we use a reranking system with set pruning to quickly narrow down these thousands of boots to just 30. It does this while ensuring that the best options are still in those 30 boots with high probability, so the user does not miss out on great products. This way, the user gets fast and accurate results without sifting through thousands of options. If the system judges that finding the desired boots in the 30 search results is unrealistic, it will automatically correct the probability guarantee or the pre-specified conditions.

To sum up, our contributions are three-fold:

- We propose the first certified error control method of candidate set pruning for relevance ranking, providing a guarantee to satisfy user-specified constraints with high probability.
- We propose a risk-confidence correction method to adjust constraints that may be otherwise impossible to satisfy for ranking tasks.
- Our method achieves consistent results under both in-domain and out-of-domain settings.

With at least 90% coverage, our method returns a candidate set size less than 50 with 1 ~ 2% accuracy drop for most two-stage ranking systems.

9.2 Related Work

Early exiting for relevance ranking. Early exiting [364, 183, 365] is a popular latency-accuracy tradeoff method in document ranking. [363] and [297] proposed to output the question-document similarity score at earlier layers of a pretrained language model, while [192] and [35] proposed to use a set of decision trees in the ensemble for prediction. [36] proposed optimization strategies that allow short-circuiting score computations in additive learning systems. [342] presented a boosting algorithm for learning such cascades to optimize the tradeoff between effectiveness and efficiency. Despite their popularity, the above early exiting methods mainly use fixed rules for efficiency-accuracy trade offs without performance guarantees.

Prediction sets and cascade systems. Prediction sets are essentially tolerance regions [359, 360, 333], which are sets that contain the desired fraction of the collection with high probability. Recently, tolerance regions have been applied to yield prediction sets for deep learning models [241, 19]. In addition, conformal prediction [328, 327] has been recognized as an attractive way of producing predictive sets with finite-sample guarantees. In retrieval, structured prediction cascades [354] optimize their cascades for overall pruning efficiency, and [66] proposed a cascade system to prune the unnecessarily large conformal prediction sets for OpenQA. However, conformal prediction is only suitable for metrics like recall. Other works on inverted file systems also provide correctness guarantees on keyword and document pruning [233].

9.3 Background

9.3.1 Notation

In the rest of the paper, we will use upper-case letters (e.g., $Q, D, \dots$) to denote the random variables, script letters (e.g., $\mathcal{Q}, \mathcal{D}, \dots$) to denote the event space, and lower-case letters (e.g., $q, d, \dots$) to denote an actual value of a random variable in its event space. Specially, we use $\mathcal{X}'$ to denote the space of all possible subsets in $\mathcal{X}$ where $\mathcal{X}' = 2^{\mathcal{X}}$.

9.3.2 Two-Stage Relevance Ranking

Given a question q , the relevance ranking task is to return a sorted list of documents from a large text corpus to maximize a metric of interest. Particularly, in a two-stage ranking system,

a first-stage retriever generates a set of candidate documents $\{d_1, d_2, \dots, d_k\}$ for the second-stage reranker to re-order the candidate lists.

Retrieval

Dense Retrievers encode the question and documents separately and project them into a low-dimensional space (e.g., 768 dim, which is “lower” than the size of the corpus vocabulary). Representative methods include DPR [126], ANCE [367], ColBERT [128], and MeBERT [191].

Lexical/Sparse Retrievers use the corpus vocabulary as the basis for vector representations. Static methods include tf-idf [275] and BM25 [267]. Contextualized methods include SPLADE [68], DeepCT [55], DeepImpact [208], and COIL [72, 166].

Despite their differences, all the above methods can be viewed as a logical scoring model [165]. Let $\eta_Q : \mathcal{Q} \rightarrow \mathbb{R}^n$ be an arbitrary function that maps a question to an n dimensional vector representation, and let $\eta_D : \mathcal{D} \rightarrow \mathbb{R}^n$ be an arbitrary function that maps a document to an n dimensional vector. The similarity score s_v between a question q and a document d can be defined as:

$$s_v(q, d) \doteq \phi(\eta_q(q), \eta_d(d)), \quad (9.1)$$

where ϕ is a metric that measures the similarity between encoded vectors of $\eta(q)$ and $\eta(d)$, such as dot product or cosine similarity.

Reranking

The reranker module is responsible for improving the ranking quality of the candidate documents returned from the first-stage retrievers. We focus on recent neural rerankers based on pretrained language models such as MonoBERT [230] and MonoELECTRA. The reranker could also be seen as a logical scoring model. However, instead of using a bi-encoder structure, a cross-encoder structure is often applied, where the question-document pairs are encoded and fed into a single model together for more fine-grained token-level interactions:

$$s_r(q, d) \doteq \zeta(\text{concat}(q, d)), \quad (9.2)$$

where ζ is the reranker that takes the question-document pair as input and outputs the similarity score s_r . One way to implement the “concat” function is using special tokens as indicators, such as [CLS] q [SEP] d [SEP].

9.4 Risk-Control Prediction Set for Ranking

9.4.1 Settings

Formally, given a question $Q \in \mathcal{Q}$, a set of documents $D' \in \mathcal{D}'$ retrieved by the first-stage retriever ϕ , and a relevance-judged set of gold documents $D'_\omega \in \mathcal{D}'_\omega$, we consider a pruning function $\mathcal{T} : \mathcal{D}' \rightarrow \mathcal{P}'$, where $\mathcal{P}'$ denotes the space of subsets of D' . We then use a loss function on the pruned document sets that also depends on the reranker ζ , i.e., $L(\cdot, \cdot; \zeta) : \mathcal{D}'_\omega \times \mathcal{P}' \rightarrow \mathbb{R}$, to encode a metric of the user's interest, and seek a pruning function $\mathcal{T}$ that controls the risk (i.e., error) $R(\mathcal{T}; \zeta) = \mathbb{E}[L(D'_\omega, \mathcal{T}(D'); \zeta)]$.

Definition 1 (Certified Error Control of Candidate Set Pruning). *Let $\mathcal{T} : \mathcal{D}' \rightarrow \mathcal{P}'$ be a random function. We say that $\mathcal{T}$ is a pruning function for reranker ζ with certified error control if, with probability at least $1 - \delta$, we have $R(\mathcal{T}; \zeta) \leq \alpha$.*

The risk level $\alpha > 0$ is pre-specified by users, and the same goes for $\delta \in (0, 1)$ where 0.1/0.01 is often chosen as a rule of thumb.

Pruning Function and Risk Function

We use a calibration set to certify the error control of the pruning function and apply the certified pruner during testing. Let $\{Q_i, D'_i, D'_{\omega_i}\}_{i=1}^m$ be an i.i.d. sampled set of random variables representing a calibration set of queries, candidate document sets, and gold document sets. For the pruning function, we define a parameter $\lambda \in \Lambda$ as its index, which is essentially a score threshold with the following property:

$$\lambda_1 < \lambda_2 \Rightarrow \mathcal{T}_{\lambda_1}(d') \subset \mathcal{T}_{\lambda_2}(d'). \quad (9.3)$$

Let $L(D'_\omega, P'; \zeta) : \mathcal{D}'_\omega \times \mathcal{P}' \rightarrow \mathbb{R}_{\geq 0}$ be a loss function on the pruned subsets. In ranking, we could take, for instance, $L(D'_\omega, P'; \zeta) = 1 - \text{MRR@10}(D'_\omega, P'_\zeta)$, where MRR@10 is a popular measurement of ranking quality and P'_ζ is the reranked version P' by ζ . In general, the loss function has the following nesting property:

$$P'_1 \subset P'_2 \Rightarrow L(D'_\omega, P'_1; \zeta) \geq L(D'_\omega, P'_2; \zeta). \quad (9.4)$$

That is, larger sets lead to smaller losses (i.e., monotonicity) [19]. We then define the risk of a pruning function $\mathcal{T}_\lambda$ to be

$$R(\mathcal{T}_\lambda; \zeta) = \mathbb{E}[L(D'_\omega, \mathcal{T}_\lambda(D'); \zeta)].$$

Confidence Region

In practice, to find the parameter λ , we need to search across the collection of functions $\{\mathcal{T}_\lambda\}_{\lambda \in \Lambda}$ and estimate their risks on the calibration set. However, the true risk is often unknown and the

empirical risk function is often used as an approximation:

$$\widehat{R}(\mathcal{T}_\lambda; \zeta) = \frac{1}{m} \sum_{i=1}^m L(D'_{\omega i}, \mathcal{T}_\lambda(D'_i); \zeta).$$

To compute the confidence region, we leverage the concentration inequalities and assume that we have access to a pointwise confidence region for the risk function for each λ :

$$\Pr(R(\mathcal{T}_\lambda; \zeta) \leq \widehat{R}_\delta^+(\mathcal{T}_\lambda; \zeta)) \geq 1 - \delta, \quad (9.5)$$

where $\widehat{R}_\delta^+(\mathcal{T}_\lambda; \zeta)$ is the upper bound of the empirical risk $\widehat{R}(\mathcal{T}_\lambda; \zeta)$. [19] presented a generic strategy to obtain such bounds by inverting a concentration inequality as well as concrete bounds for various settings. For this paper, we use the Waudby-Smith-Ramdas (WSR) bound [351] which is adaptive to variance.

We choose the smallest λ such that the entire confidence region to the right of λ falls below the target risk level α following [19]:

$$\hat{\lambda} \doteq \inf \left\{ \lambda \in \Lambda : \widehat{R}_\delta^+(\mathcal{T}_{\lambda'}; \zeta) < \alpha, \forall \lambda' \geq \lambda \right\}. \quad (9.6)$$

In this way, $\mathcal{T}_{\hat{\lambda}}$ is a pruning function with certified error control. In the following sections, we will discuss the problems of truncated risk functions in relevance ranking and how to modify the certification for impossible constraints.

9.4.2 Truncated Risk Function for Ranking

In Section 9.4.1, we mentioned that the risk function is often related to the metrics that we care about. For example, in ranking, metrics such as MRR@K are often used to assess the ranking quality, where K is the maximal set size that we choose. For example, the MRR@10 score given a set of questions $\{q_i\}_{i=1}^m$, the positive document sets $\{d'_i\}_{i=1}^m$, and the retrieved document candidate sets $\{d'_{\omega i}\}_{i=1}^m$ is

$$\text{MRR@10} = \frac{1}{m} \sum_{i=1}^m \frac{1}{f(d'_i, d'_{\omega i})}, \quad (9.7)$$

where

$$f(d'_i, d'_{\omega i}) = \begin{cases} +\infty, & \text{if } r_i > 10; \\ r_i, & \text{otherwise,} \end{cases} \quad (9.8)$$

and

$$r_i = \min_j (r(d'_i, d'_{\omega ij})).$$

$d'_{\omega ij}$ means the j^{th} positive document for query i and $r(d'_i, d'_{\omega ij})$ means the rank of $d'_{\omega ij}$ in the candidate set d'_i . If we use $1 - \text{MRR@10}$ for the empirical risk function $\widehat{R}(\mathcal{T}_\lambda; \zeta)$, we can see that the risk function and its upper-bound plateaus after a certain λ value (Fig. 9.3b) due to the threshold function in Eq (9.8). Therefore, naive certification will fail if the risk level is specified

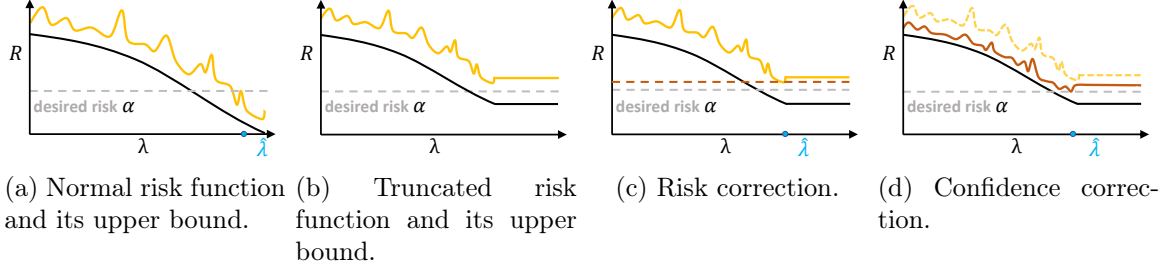


Figure 9.3: (Corrected) Confidence region of the risk function. (a) shows the risk function for metrics like recall; (b) shows the risk function that is truncated in ranking. (c) and (d) show two different types of corrections: Either setting the risk level to the minimum of the upper bound or increasing δ to shrink the upper bound.

too low.

9.4.3 Correction for Risk Level and Confidence

To this end, we propose a safe certification method, which will automatically correct the risk level α or the confidence $1 - \delta$ if the risk level specified by the user is too low. Given a specific (α, δ) pair, the basic idea is that if the minimum of the upper confidence bound $\widehat{R}_\delta^+(\mathcal{T}_\lambda; \zeta)$ over all possible λ values is bigger than the specified risk level α , we will either replace the risk level with the best-possible minimal risk that we could achieve:

$$\alpha_c \doteq \inf_{\lambda \in \Lambda} \widehat{R}_\delta^+(\mathcal{T}_\lambda; \zeta), \quad (9.9)$$

where α_c is the calibrated risk level as shown in Fig. 9.3c, or increase δ to shrink the upper-confidence bound until $\delta = 1$:

$$\delta_c \doteq \inf \left\{ \delta \in (0, 1] : \inf_{\lambda \in \Lambda} \widehat{R}_\delta^+(\mathcal{T}_\lambda; \zeta) \leq \alpha \right\}, \quad (9.10)$$

where δ_c is the calibrated significance level as shown in Fig. 9.3d. Therefore, the previous pruning function found in Eq (9.6) does not necessarily hold for some specific (α, δ) values in ranking, and we propose a new theorem for the corrected version of certification:

Theorem 1 (Correction of Certified Error Control). *In the setting of Section 9.4.1, assume that there exists $\alpha > 0$ and $\delta > 0$ such that for every $\hat{\lambda} \in \Lambda$, $\widehat{R}_\delta^+(\mathcal{T}_{\hat{\lambda}}; \zeta) > \alpha$. In this case, $\mathcal{T}_{\hat{\lambda}}$ is no longer a pruning function with certified error control. Instead, with the corrected risk level α_c and confidence δ_c in Eq (9.9) and (9.10), there exists $\hat{\lambda} \in \Lambda$ such that,*

$$\Pr(R(\mathcal{T}_{\hat{\lambda}}; \zeta) \leq \alpha_c) \geq 1 - \delta,$$

or

$$\Pr(R(\mathcal{T}_{\hat{\lambda}}; \zeta) \leq \alpha) \geq 1 - \delta_c.$$

Retriever	MARCO Quora	
BM25	0.185	0.781
DPR	0.311	0.434
UniCOIL	0.348	0.659

Table 9.1: In-domain (MS MARCO Passage v1) and out-of-domain (Quora) first-stage retrieval test MRR@10 scores averaged over 100 random dev/test splits.

Retriever	MARCO Quora	
BM25+MonoBERT	0.369	0.840
DPR+MonoBERT	0.378	0.728
UniCOIL+MonoBERT	0.383	0.832
BM25+MonoELECTRA	0.399	0.823
DPR+MonoELECTRA	0.415	0.653
UniCOIL+MonoELECTRA	0.415	0.817

Table 9.2: In-domain (MS MARCO Passage v1) and out-of-domain (Quora) second-stage reranking (with evidence fusion) test MRR@10 scores averaged over 100 random dev/test splits.

In both cases, $\mathcal{T}_{\hat{\lambda}}$ is a pruning function with certified error control.

Proofs are provided in [162].

9.5 Experimental Setup

9.5.1 Datasets

In this paper, we evaluate our method on the following datasets: MS MARCO Passage v1 [226] contains 8.8M English passages with an average length of around 55 tokens, which is a standard retrieval benchmark for comparing in-domain results. Quora Duplicate Questions¹ [310] contains 522K passages with an average length of around 11 tokens and mostly consist of duplicate entity questions that were found to be challenging for out-of-domain generalization of neural retrievers like DPR [282]. The above datasets label documents with shallow judgments [381] for each query and therefore MRR@10 is often used as the evaluation metric. Other datasets such as TREC DL 2019 [324] use metrics like nDCG, but such densely labeled data are very scarce and therefore they are not suitable for finite-sample calibration, which we leave for future work.

¹<https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs>

Methods	MRR@10	Confidence	Coverage	Size
<i>BM25 + MonoBERT (required MRR@10=0.350)</i>				
CEC	0.359	0.870	0.870	451
EST	0.350	-	0.540	139
ERT	0.350	-	0.510	149
<i>UniCOIL + MonoBERT (required MRR@10=0.350)</i>				
CEC	0.360	0.900	0.900	15
EST	0.350	-	0.480	9
ERT	0.352	-	0.460	7
<i>DPR + MonoBERT (required MRR@10=0.350)</i>				
CEC	0.360	0.900	0.900	22
EST	0.350	-	0.500	11
ERT	0.353	-	0.730	11
<i>BM25 + MonoELECTRA (required MRR@10=0.380)</i>				
CEC	0.389	0.894	0.910	602
EST	0.381	-	0.580	280
ERT	0.381	-	0.550	246
<i>UniCOIL + MonoELECTRA (required MRR@10=0.380)</i>				
CEC	0.390	0.900	0.910	18
EST	0.380	-	0.520	13
ERT	0.383	-	0.750	9
<i>DPR + MonoELECTRA (required MRR@10=0.380)</i>				
CEC	0.389	0.900	0.900	27
EST	0.381	-	0.580	16
ERT	0.382	-	0.580	17

Methods	MRR@10	Confidence	Coverage	Size
<i>BM25 + MonoBERT (required MRR@10=0.780)</i>				
CEC	0.789	0.900	0.910	2
EST	0.780	-	0.510	2
ERT	0.780	-	0.600	3
<i>UniCOIL + MonoBERT (required MRR@10=0.780)</i>				
CEC	0.790	0.900	0.900	16
EST	0.780	-	0.560	13
ERT	0.782	-	0.640	14
<i>DPR + MonoBERT (required MRR@10=0.620)</i>				
CEC	0.632	0.900	0.940	40
EST	0.620	-	0.500	30
ERT	0.621	-	0.490	29
<i>BM25 + MonoELECTRA (required MRR@10=0.780)</i>				
CEC	0.790	0.900	0.900	3
EST	0.780	-	0.460	3
ERT	0.780	-	0.560	2
<i>UniCOIL + MonoELECTRA (required MRR@10=0.780)</i>				
CEC	0.790	0.900	0.910	3
EST	0.785	-	0.540	3
ERT	0.782	-	0.600	3
<i>DPR + MonoELECTRA (required MRR@10=0.620)</i>				
CEC	0.634	0.900	0.950	282
EST	0.620	-	0.530	155
ERT	0.620	-	0.540	149

Table 9.3: In-domain results on MS MARCO Passage v1 (left) and out-of-domain results on Quora (right). CEC: Certified error control. EST: Empirical score threshold. ERT: Empirical rank threshold. Confidence: $1 - \delta_c$ as in Eq (9.10). Coverage: Proportion of 100 runs that satisfy the risk constraints. Size: Average candidate set size out of 1,000. See Section 9.6.2 for details.

9.5.2 Retrievers and Rerankers

For retrievers, we use DPR (dense retriever), BM25 (static lexical retriever), and UniCOIL (contextualized lexical retriever). For rerankers, we use two cross-encoder models, MonoBERT [230] and MonoELECTRA [247]. Although some of these models are no longer state of the art, they remain competitive and have been widely adopted by researchers in the community as points of reference. For the pipeline, we use the retriever (e.g., DPR) to retrieve the top-1000 candidates from the corpus, and then use the reranker (e.g., MonoELECTRA) to rerank the retrieved candidate sets. We believe the above choices cover the basic types of modern two-stage ranking systems; our approach is model agnostic and can be easily applied to other models as well. For implementation, we use off-the-shelf pretrained models from Pyserini [167] and Caprelous [379].

Finally, for rerankers based on neural networks, we need to consider both in-domain and out-

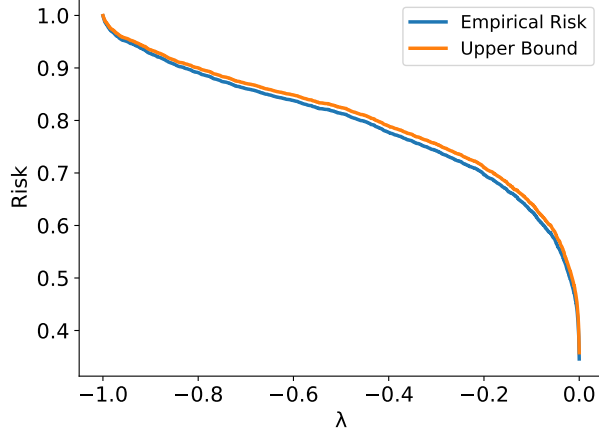


Figure 9.4: Empirical risk function and its upper bound on one calibration-test data split.

of-domain situations, as it is possible that the reranker overfits to certain domains and has worse out-of-domain effectiveness than the retriever. To solve this, we linearly interpolate the score of the retriever ϕ and the reranker ζ for each (q, d) pair during reranking:

$$s_f(q, d) = \beta \cdot \phi(\eta_q(q), \eta_d(d)) + (1 - \beta) \cdot \zeta(\text{concat}(q, d)),$$

which is known as evidence fusion [195]. The weight $\beta \in [0, 1]$ is searched on the calibration set for the best MRR@10 score, such that the fusion model will consistently yield better ranking results than both ζ and ϕ . Tbl. 9.1 and 9.2 show the retrieval and reranking effectiveness with evidence fusion under both in-domain and out-of-domain settings.

9.5.3 Baselines

We modify two *empirical error control* methods from [36] as the baselines:

Empirical Score Threshold (EST). A score threshold on the calibration set such that the pruned MRR@10 just meets the required score.

Empirical Rank Threshold (ERT). Similar to EST, but we tune a threshold on the rank of the documents to prune the candidate set instead.

9.6 Results

9.6.1 Risk Function and Upper Bound

Fig. 9.4 shows the empirical risk function and its WSR upper bound of the pruning function $\mathcal{T}_\lambda$ on Quora using the DPR + MonoELECTRA ranking system. We can see that the empirical risk function is a monotone function and the minimum of the risk is greater than 0, which is consistent with the assumptions we made in Section. 9.4. In addition, we can see that the bound is very tight, providing a good estimation of the true risk.

9.6.2 In-Domain Results

Empirically, if we set the risk threshold $\alpha = 0.62$ and confidence level $1 - \delta = 0.9$, then there should be at least 90% of independent runs (i.e., coverage) for which the MRR@10 score is greater than 0.38 (i.e., $1 - \alpha$). To verify this, we mix the test set and dev set and then randomly sample a calibration set of size 5,000 and a test set of size 6,980, repeating for 100 trials.

Tbl. 9.3 (left) shows the MRR@10 score, corrected confidence, empirical coverage, and average candidate set size on MS MARCO Passage v1. We choose different performance thresholds for different ranking systems such that the final reranking score is around 1 ~ 2% less than the highest obtainable score, which is a very typical setting in real-world applications. We use $\delta = 0.1$ for all experiments, but it could be corrected if the risk threshold is unable to be satisfied. For example, for BM25 + MonoBERT, the confidence is corrected from 0.90 to 0.87, which is more consistent with the empirical coverage.

We can see that our method achieves the required risk constraint with the required coverage (i.e., confidence) for multiple ranking systems, while the average candidate set size is also drastically reduced from 1,000 to less than 50. In comparison, although they are able to obtain smaller candidate set sizes, both empirical error control methods do not achieve the expected coverage. Despite the fact that we can choose other thresholds to achieve better coverage, it is unclear how much accuracy should be sacrificed in order to achieve the required coverage.

9.6.3 Out-of-Domain Results

We also test our method on Quora under an out-of-domain setting, where we use the retrievers and rerankers trained on MS MARCO Passage v1 as the prediction models. We can see from Tbl. 9.1 and Tbl. 9.2 that the out-of-domain retrieval and reranking results are drastically different from the MS MARCO dataset, where BM25 outperforms the other neural retrievers. This is because the Quora dataset mostly consists of duplicate, entity-based questions, which are naturally biased toward static lexical retrievers.

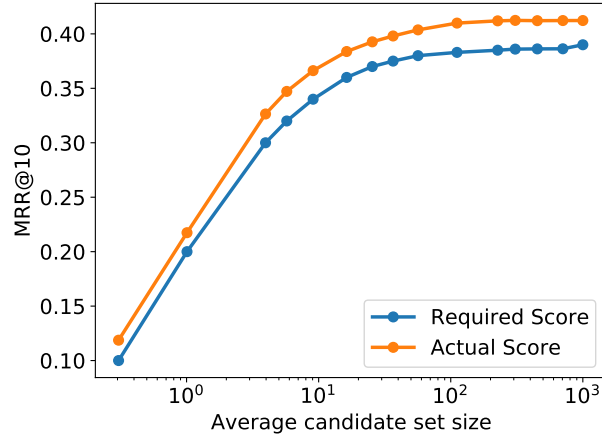


Figure 9.5: Trade offs between candidate set size and MRR@10 scores on MS MARCO Passage v1 based on the DPR + MonoELECTRA ranking system. The empirical coverage for each point is greater than 90%.

Similar to the in-domain experiments, we calibrate the pruning function on the calibration set with 5,000 data points and test it on a test set with 10,000 data points over 100 trial runs. Tbl. 9.3 (right) shows the MRR@10 scores of different ranking systems. However, unlike the in-domain setting, the ranking effectiveness of the first stage retrievers varies a lot under the out-of-domain setting and it is hard to align the pruning results for intuitive comparison. Therefore, we set different MRR@10 thresholds such that the effectiveness drop is around 1 ~ 10% to align the results of different ranking systems. The results are similar to the in-domain experiments, where the certified error control method manages to provide a performance guarantee while the empirical method fails to do so. Our method also yields reasonable set sizes that are close to the empirical baseline’s. This is consistent with our claims that our method does not make assumptions about data distributions and prediction models, as long as the data from the calibration set and test set are exchangeable.

9.6.4 Efficiency-Accuracy Trade offs with Certified Error Control

In this section, we investigate the guarantee of the overall trade offs between efficiency and accuracy. Fig. 9.5 illustrates the efficiency-accuracy trade offs results on MS MARCO Passage v1 using DPR + MonoELECTRA. Similarly, we set the confidence level $1 - \delta$ to be 0.9. Our method (blue line) achieves the best MRR@10 score at around 20% of the original top-1000 candidate set size, which is a very good trade off between accuracy and efficiency. In addition, the MRR@10 score of our method (blue line) is higher than the specified score threshold (orange line) with at least 90% coverage, which further verifies the guarantee claims we made about our method. We

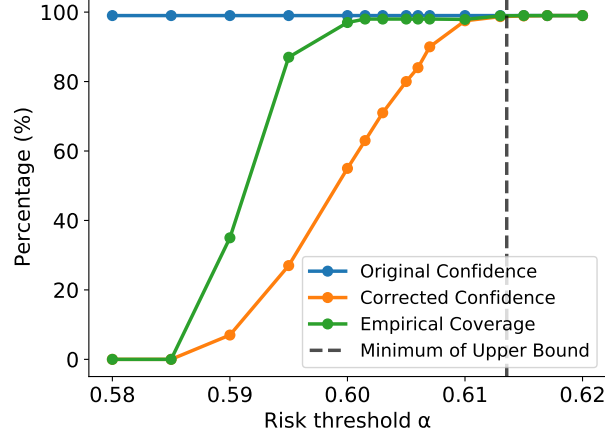


Figure 9.6: Confidence correction on MS MARCO Passage v1 using DPR + MonoELECTRA. The x -axis represents different risk thresholds (i.e., α) and the y -axis represents the percentage.

can see that our method achieves a good trade off while satisfying different values of the risk level α .

9.6.5 Confidence-Risk Correction

In Section 9.4.3, we mentioned that it might be impossible to achieve the risk threshold if it is specified too low in ranking tasks. Our solution approaches this problem by correcting either the risk threshold α or the significance level δ as shown in Fig. 9.3c and 9.3d. In practice, the risk threshold correction is rather straightforward: if the risk is lower than the minimal upper bound over all λ , we just reset the risk threshold to the minimal upper bound as shown in Eq (9.9). For the confidence correction, we need to fix the risk threshold and shrink the upper bound by increasing the significance δ until the confidence $(1 - \delta)$ is 0 as shown in Eq (9.10). Fig. 9.6 shows the confidence correction of our method. We use the DPR+MonoELECTRA ranking system whose best reranking MRR@10 score is 0.41 on the dev small set, meaning that the minimal risk is around 0.59, and the minimal UCB of the risk function is around 0.61 (the vertical line). From right to left, we can see that the confidence does not change too much until the risk passes 0.6137. As the risk threshold decreases, the confidence (the orange line) also gradually decreases, which is consistent with the empirical coverage (the green line).

9.7 Summary

We present a theoretically principled method for candidate set pruning of two-stage ranking systems, allowing users to control a customized loss under the desired threshold with high probability. We further propose to correct the risk threshold or confidence level if the desired risk cannot be achieved given the ranking system. Experiments performed under in-domain (MS MARCO Passage v1) and out-of-domain (Quora) settings show that our method provides a consistent performance guarantee to candidate set pruning across multiple ranking systems.

Chapter 10

Nearest Neighbor Speculative Decoding

10.1 Overview

Large language models (LLMs) have demonstrated strong potential as multi-task solvers, excelling in a wide range of applications [34, 45, 314, 6]. Despite their advanced capabilities, LLMs frequently encounter the problem of hallucination, particularly when dealing with long-tail knowledge that is less represented in their training data [125, 10]. To address this limitation, retrieval augmentation incorporates information retrieval and nearest neighbor search from a non-parametric data store to enhance evidence-based and situated reasoning with LLMs. The resulting semi-parametric LMs exhibit reduced tendency to generate unsupported content [127, 31, 289, 288, 10].

However, the effectiveness of retrieval-augmented language models (RALMs) in ensuring *accurate and reliable content generation* varies. The widely used in-context retrieval-augmentation (RA) regime [259, 289, 288] softly biases the LM output distribution by prepending retrieved content to the input, which does not reliably guarantee faithful attribution of information. Approaches such as k NN-LM [127] modify the LM output with a non-parametric token distribution derived from nearest-neighbor matches in a corpus, which provide more direct attribution but have also been shown to degrade the quality of text generation [343]. Additionally, retrieval augmentation can significantly increase the *generation latency* due to the time required for the retrieval processes to complete and the subsequent expansion of the LM’s context.

In this Chapter, we propose **Nearest Neighbor Speculative Decoding** (NEST). This new semi-parametric language modeling approach is capable of incorporating real-world text spans of arbitrary length into the generations of an off-the-shelf LM, leading to improved quality and latency. NEST extends the standard k NN-LM approach, which interpolates the output distribution of an LM using the distribution of possible next tokens retrieved from a corpus [127]. It conducts an additional passage retrieval step at the beginning to limit the need to store and search over all

tokens in the corpus, offering a balanced trade-off between search accuracy and efficiency. At each inference step, NEST performs content generation with three sub-steps:

1) Confidence-based interpolation. We use a novel *Relative Retrieval Confidence* (RRC) score to measure the uncertainty of the token retriever and use it as the interpolation coefficient of the output probability mixture. This enables flexible adaptation of the LM’s output to different downstream tasks through dynamic interpolation with the token retrieval results.

2) Dynamic span selection. Inspired by the Copy Generator (COG) [142], NEST selects not only the best token predicted by the mixture probability, but also extends to the span containing from that token in the corpus when the token retrieval confidence exceeds a predefined threshold.

3) Relaxed speculative decoding. If a span of more than one tokens is selected, it undergoes evaluation based on the mixture probability. Through a rejection procedure similar to that in speculative decoding [151], only a prefix deemed highly likely by the mixture probability is accepted.

Evaluated on various free-form generation tasks—including question answering, text completion, and factuality-aware generation—using Llama-2-Chat models [315] of different sizes, NEST demonstrates superior performance compared to both the base LM and the standard k NN-LM under a zero-shot setting. For example, combined with NEST, the Llama-2-Chat 70B model demonstrates 42.3% improvement of ROUGE-1 on WikiText-103 and 21.6% improvement of FACTSCORE on Biography. Furthermore, NEST performs competitively with respect to in-context retrieval-augmentation on MMLU, Pile-of-Law, and TruthfulQA. We further demonstrate that the two approaches can be combined to enhance generation quality and attribution. Additionally, by generating multiple tokens at each time step, NEST significantly improves the efficiency of long-form generation. For Llama-2-Chat 70B, it achieves a $1.8\times$ speedup in inference time without compromising attribution or fluency.

10.2 Background

10.2.1 Problem Definition

Given an input x , a mixture model $\mathcal{M}$ predicts the output y consisting of segments $\{y_1, y_2, \dots, y_T\}$. In our setting, $\mathcal{M}$ may produce multiple tokens at a time step t , and therefore y_t indicates the t -th segment consisting of at most n tokens where $1 \leq |y_t| \leq n$. Let $\{w_t^{(1)}, w_t^{(2)}, \dots, w_t^{(n)}\}$ be the tokens in segment y_t , we use $p_{\mathcal{M}}(w|x, y_{<t})$ to denote the distribution of the next token, and use $p_{\mathcal{M}}(w = w_t^{(1)}|x, y_{<t})$ to denote the probability of $w_t^{(1)}$ of the next segment y_t .

10.2.2 Nearest Neighbor Language Models (k NN-LM)

The mixture model $\mathcal{M}$ involves a pretrained LM and key-value datastore $(\mathcal{K}, \mathcal{V})$ that enables approximate nearest neighbors search without further training or finetuning.

Key-value datastore. To create the datastore $(\mathcal{K}, \mathcal{V})$ using the LM for a corpus $\mathcal{D}$, let $f(\cdot)$ be the mapping from input sequence c to the hidden states h of the LM at some fixed layer. Let w be the next word of c in the corpus $\mathcal{D}$. For a sample (c_i, w_i) in $\mathcal{D}$ after segmentation, we define the i -th key-value pair (k_i, v_i) in $(\mathcal{K}, \mathcal{V})$ as (h_i, w_i) , where $h_i = f(c_i)$. The whole datastore is thus defined as the set of all possible key-value pairs in $\mathcal{D}$:

$$(\mathcal{K}, \mathcal{V}) = \{(h_i, w_i) | (c_i, w_i) \in \mathcal{D}\}. \quad (10.1)$$

The size of the datastore $(\mathcal{K}, \mathcal{V})$ is proportional to the total number of tokens in corpus $\mathcal{D}$. This brings difficulty in scaling the size of the corpus and the model, which may require massive storage space and computational resources.

Probability interpolation. During inference, the language model outputs the token distribution $p_{\text{LM}}(w|x, y_{<t})$, together with the hidden state q_t . The model uses q_t as a query to search the datastore $(\mathcal{K}, \mathcal{V})$ and retrieve the r -nearest neighbors π according to the similarity $s(q, k)$ between a query q and a key k . The final non-parametric distribution $p_{k\text{-NN}}(w|x, y_{<t})$ is computed using a softmax function over the similarity of all retrieved neighbors:

$$p_{k\text{-NN}}(w|x, y_{<t}) \propto \sum_{(k_i, v_i) \in \pi} \mathbb{I}_{w=v_i} \exp(\mu \cdot s(q_t, k_i)), \quad (10.2)$$

where μ is the inverse temperature. We use $1/\sqrt{\dim(q_t)}$ for μ in practice where $\dim(q_t)$ is the hidden state dimension. This is similar to computing attention in the Transformer model [319]. For similarity $s(q, k)$, we follow [127] and use the negative squared ℓ_2 distance. Items not in π are assigned with 0 probability based on the indicator function $\mathbb{I}_{w=v_i}$.

Finally, the next token is sampled from the mixture distribution $p_{\mathcal{M}}$ of the non-parametric distribution $p_{k\text{-NN}}$ and the parametric distribution p_{LM} using a fixed hyper-parameter $\lambda \in [0, 1]$:

$$p_{\mathcal{M}}(w|x, y_{<t}) = \lambda \cdot p_{\text{LM}}(w|x, y_{<t}) + (1 - \lambda) \cdot p_{k\text{-NN}}(w|x, y_{<t}). \quad (10.3)$$

10.3 Nearest Neighbor Speculative Decoding

10.3.1 Two-Stage k -NN Search

As mentioned in Section 10.2.2, maintaining a token-level key-value store can be expensive in terms of both latency and storage. To provide a better trade-off between latency and accuracy, we adopt the two-stage design, which is widely applied in information retrieval and search engines.

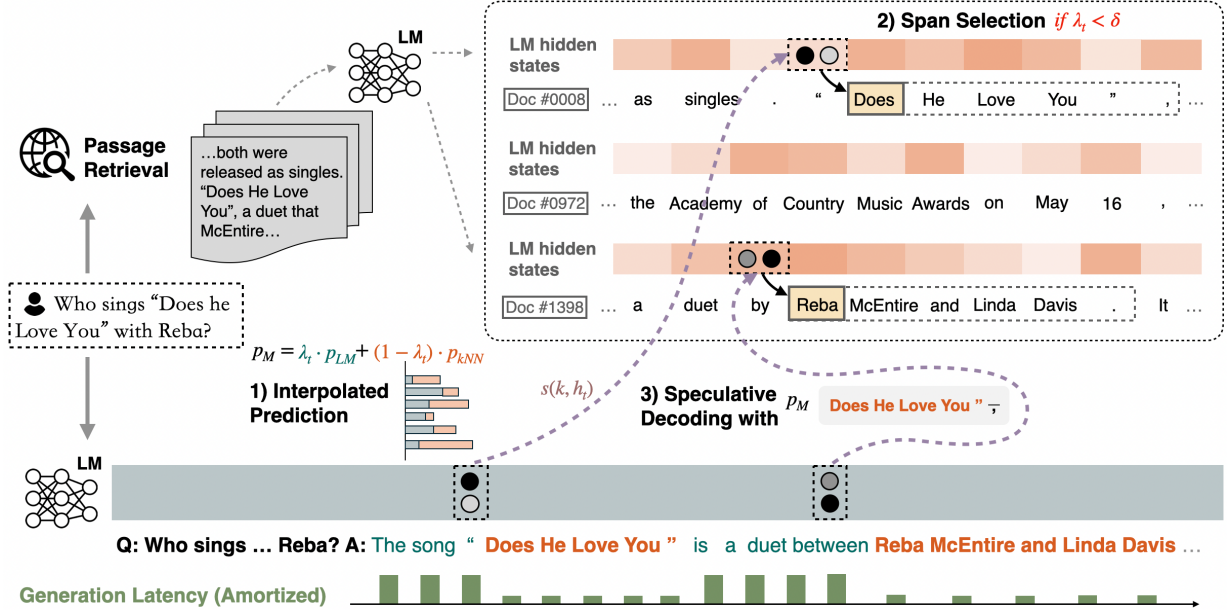


Figure 10.1: The NEST approach first locates the tokens in the corpus using the LM hidden states. The retrieval distribution $p_{k\text{-}NN}$ is dynamically interpolated with p_{LM} based on the retriever’s uncertainty λ_t . The token and its n -gram continuation are then selected from the mixture distribution p_M , while the final span length is determined by speculative decoding to remove undesired tokens. The spans incorporated in the final generation provide direct attribution and amortize the generation latency.

First-stage passage retrieval Given the corpus $\mathcal{D}$, we segment the documents into separate passages of less than m tokens each. We then encode the corpus and use a hybrid retrieval system to select the relevant passages, as dense retrievers are good at handling semantics in queries [126] and sparse retrievers are good at lexical matching [282].

Second-stage k -NN token search After obtaining the top- b retrieved passages $\{d_1, d_2, \dots, d_b\}$ at time step t , we use the encoder $f(\cdot)$ of LM to encode the prefixes of all tokens as keys as shown in Figure 10.1. The key-value datastore $(\mathcal{K}, \mathcal{V})$ therefore is created *on the fly*. Similarly, we use the negative squared ℓ_2 distance as the similarity function and q_t as the queries to search for the top- r nearest neighbors π in $(\mathcal{K}', \mathcal{V}')$.

The two-stage design provides a trade-off between search latency and accuracy and the passage-level index only takes a fraction of the token-level index in Section 10.2.2. In addition, the first-stage passage search also acts as a filter to remove deceptively similar tokens in non-relevant contexts.

10.3.2 Confidence-Based Output Interpolation

Similar to Equation (10.3), we linearly interpolate the language model’s distribution p_{LM} and non-parametric distribution $p_{k\text{-NN}}$ using a coefficient λ_t for a time step t in generation. The difference is that we use the token retrieval score to compute λ_t :

$$\lambda_t = \sigma \left(\left(\frac{\min_i |s(q_t, k_i)|}{\max_i |s(q_t, k_i)|} - \alpha \right) / \tau \right), \quad (10.4)$$

where σ is the sigmoid function and the min-max ratio expresses the uncertainty of the k -NN component. We use the sigmoid activation to re-center and re-scale this uncertainty, where α is the offset and τ is the scale for the sigmoid function. We refer to this method as Relative Retrieval Confidence (RRC).

If the downstream task does not involve generation, such as perplexity evaluation and multi-choice tasks, our method will end at Equation (10.4). The mechanisms introduced in the following sections are only applied to generation, including token/span selection and post-hoc revision.

10.3.3 Dynamic Span Selection

Directly sampling tokens from the mixture distribution $p_{\mathcal{M}}$ might escalate the exposure bias since the seemingly coherent tokens might be retrieved from completely different sources. To maintain coherence, we extend the context of the current sampled token by using its n -gram continuation in the corpus. Given the current time step t , we first select the next token w_t from the mixture distribution $p_{\mathcal{M}}$. However, the sampled token w_t may correspond to multiple retrieved w_i (i.e., the value v_i), in the neighbors π which have different n -gram continuations. We use a simple max-pooling strategy to select the starting token $w_t^{(1)}$ of the n -gram from π :

$$w_t^{(1)} = \underset{\{w_i | w_i = w_t, w_i \in \pi\}}{\operatorname{argmax}} p_{k\text{-NN}}(w = w_i | x, y_{<t}) \quad (10.5)$$

The corresponding n -gram for time step t is $\{w_t^{(1)}, w_t^{(2)}, \dots, w_t^{(n)}\}$ where n is fixed hyper-parameter. The final output is determined by the interpolation coefficient λ_t in Equation (10.4):

$$y_t = \begin{cases} w_t, & \text{if } \lambda_t > \delta; \\ \{w_t^{(1)}, w_t^{(2)}, \dots, w_t^{(n)}\}, & \text{otherwise.} \end{cases} \quad (10.6)$$

where δ is a threshold and y_t is the segment output at time step t .

10.3.4 Relaxed Speculative Decoding

Despite the dynamic selection, the hyper-parameter n is hard to control over different tasks. To produce spans with adaptive length, we take inspiration from [151], where we use $\mathcal{M}$ to revise the proposed n -gram. However, the proposal distribution $q(w|x, y_{<t})$ is unknown besides the first

token $w_t^{(1)}$. Therefore, we use a relaxed version of speculative decoding that upper bounds the acceptance probability. The probability of accepting the token $w_t^{(i)}$ in a span is:

$$P(\text{accept token } w_t^{(i)}) = \min \left(1, \frac{p_{\mathcal{M}}(w = w_t^{(i)} \mid x, y_{<t}, w_t^{(1)}, w_t^{(2)}, \dots, w_t^{(i-1)})}{\gamma \cdot \max_w p_{\mathcal{M}}(w \mid x, y_{<t}, w_t^{(1)}, w_t^{(2)}, \dots, w_t^{(i-1)})} \right), \quad (10.7)$$

where $\gamma \in (0, 1]$ is the relaxation factor, which is referred to as “leniency” by [151]. The smaller γ is, the less often $\mathcal{M}$ rejects the draft. If token $w_t^{(i)}$ is rejected, we will remove all the tokens from $w_t^{(i)}$ to $w_t^{(n)}$, and then re-select a token $w_t^{(i)}$ from the distribution $p_{\mathcal{M}}$ without going through the span selection. The computation for processing multiple tokens can be parallelized and NEST can thus maintain the latency or even accelerate the generation. Moreover, suppose all tokens in the draft are not rejected. In that case, we will directly fetch the n -gram’s continuation in the corpus and use it for the next draft proposal until rejection, removing the reliance on the hyper-parameter n .

Once the n -gram is accepted, the corresponding parts are masked in the corpus and will never be used again in this generation. This is to prevent the k -NN component from repetitively retrieving the same segments in a small key-value store $(\mathcal{K}', \mathcal{V}')$.

10.4 Experiments

We evaluate NEST and other baselines on various tasks including text completion, question-answering, fact-verification, and multi-choice tasks, providing a comprehensive picture of factuality, fluency, and attribution of NEST in different domains. In all experiments, we focus on evaluating instruction-following models. We use Llama-2-chat under a zero-shot setting, where we remove the few-shot demonstrations from the instructions to simulate the realistic usage of these models.

10.4.1 Benchmark Datasets

Text completion. **WikiText-103** [214] is a standard benchmark for language modeling, extracted from the set of verified articles on Wikipedia. **Pile of Law** [91] is a growing dataset of legal and administrative data. We use the datasets¹ from Huggingface and further split the test data into validation and test sets. For language modeling, we report the perplexity score. For free-form generation, we report ROUGE-1, 2, L [163] and MAUVE [244].

¹<https://huggingface.co/datasets/pile-of-law/pile-of-law/tree/main>

Question answering. We select four knowledge-intensive question-answering datasets, including Natural Questions (NQ) [138], TriviaQA (TQA) [122], HotpotQA (HQA) [376], and MedMCQA (MQA) [239]. Since the in-context demonstrations are removed for free-form generation, we use answer-level recall (i.e., Hit@1) [126] which checks if the output contains any correct answers instead of exact match.

Fact verification. We evaluate a biography-generation task [221] and TruthfulQA [177] which is a benchmark for testing false beliefs or misconceptions. We use FACTSCORE [221] for biography. For TruthfulQA, we follow [177] which uses the difference between the max similarity to a true reference answer and the max similarity to a false reference answer for BLEU and ROUGE-1.

Closed-set tasks. MMLU (Massive Multitask Language Understanding) [92] benchmark covers 57 subjects across STEM, the humanities, the social sciences, and more. We report the macro accuracy for each domain.

10.4.2 Implementation

Knowledge Sources. Wikipedia (CC BY-SA 3.0): For tasks except text completion on Pile of Law, we use the Wikipedia 2021 dump released by [115] as the knowledge source and follow the same pre-processing procedures in RA-DIT [178], yielding ~ 33 M passages with each less than 200 tokens. Pile of Law (CC BY-NC-SA 4.0): We use the training split from Huggingface and select only the English data. We then follow the same procedure applied in Wikipedia, yielding a corpus containing ~ 15 M passages after filtering.

Inference setting. k NN-LM and NEST share the same first-stage retriever. We use DRAGON+ [170] and BM25 [267] to encode the segments into dense and sparse vectors, respectively. Given the input, we query both the dense and sparse indexes at the same time and retrieve their corresponding top- $(b \cdot l)$ passages. We linearly interpolate the similarity scores between the two search results (also known as fusion) and sort them before selecting the top- b passages. The number of passage candidates b is set to be 40 and the scaling factor l is set to be 100. For RA, we use the top-3 passages in the prompt due to the context window limit. We further combine NEST and RA since they are independent methods. Greedy decoding is used during generation.

Evaluation setting. For text completion tasks and perplexity evaluation, we use 128 tokens as the prefix and the consecutive 256 tokens as the target. For the other tasks, we use 128 tokens as the max generation length for question answering and 512 for fact verification. For retrieval-based models, only the prefix will be used for retrieval. Hyper-parameters of all baselines and NEST are

tuned on the dev set of WikiText-103, NQ, and Biography. Each baseline uses the same hyper-parameters for all tasks evaluated. We first tune the related hyper-parameters for perplexity and then tune the rest for generation metrics to reduce the search space.

10.4.3 Baselines

Base LMs. We evaluate publicly available, instruction-tuned language models, Llama-2-chat series², with model sizes ranging from 7B, 13B to 70B.

Two-Stage k NN-LM. We apply the two-stage strategy described in Section 10.3.1 to k NN-LM as well, where we retrieve the top- b passages and encode a key-value datastore $(\mathcal{K}', \mathcal{V}')$ on the fly.

In-Context Retrieval Augmentation (RA). A common retrieval-augmentation method is adding the retrieved evidence into the prompt. We perform retrieval given the only input instead of retrieving new passages every k step due to the expense of refreshing the kv-cache.

10.4.4 Main Results

Table 10.1 shows the main results of NEST and other baselines. For **language modeling**, RA-NEST is able to achieve the lowest complexity on both WikiText-103 and Pile of Law. For **text completion**, RA has the best MAUVE scores and ROUGE scores in Wikitext-103 while RA-NEST works better for 7B and 13B models on Pile of Law. We observe that for legal documents, quoting the exact clauses from the source might be more favorable compared to Wikipedia.

For **question-answering**, RA-NEST tends to work better for smaller models (7B and 13B) in general. The gap between base LMs and other methods diminishes for 70B LMs, which is consistent with previous work where retrieval is found most useful for smaller models [31].

For **fact-verification**, NEST is able to outperform the base LMs but underperform RA in terms of the FACTSCORE. RA-NEST is able to outperform RA for the 70B model. The degradation for RA-70B is caused by generating shorter claims which is punished by the FACTSCORE. On TruthfulQA, the semi-parametric LMs consistently outperform base LMs and RAs where in-context retrieval seems to have a negative effect on the scores. This is because TruthfulQA is an adversarial dataset containing difficult questions where in-context RA is more susceptible to the “evidence” in the prompt (e.g., astrology and myths). In contrast, NEST only interpolates the results at the output level and therefore performs better in this case. The combination RA-NEST is also affected by the in-context retrieval.

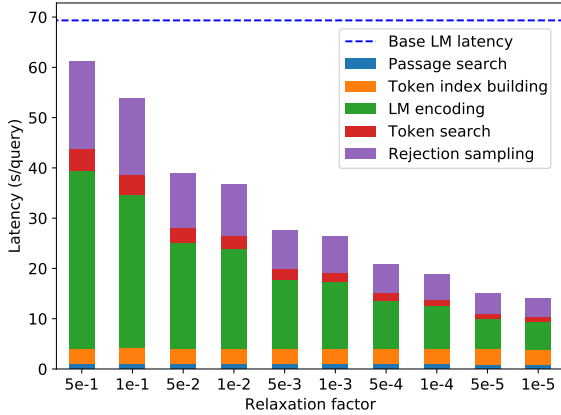
²<https://huggingface.co/meta-llama/Llama-2-70b-chat-hf>

Models	Wiktext-103						Pile of Law					
	PPL(↓)	MAUVE	RG-1	RG-2	RG-L	Avg. Len	PPL(↓)	MAUVE	RG-1	RG-2	RG-L	Avg. Len
Llama-2-Chat _{7B}	14.6	58.8	15.8	3.7	14.4	175.4	10.1	80.7	19.1	5.5	17.1	211.4
+RA	7.2	74.6	35.7	23.1	34.4	204.5	7.1	84.7	23.1	8.9	21.1	222.0
+ <i>k</i> NN-LM	9.8	82.5	23.7	7.7	21.7	238.2	8.8	81.1	19.4	5.7	17.4	214.3
+NEST	8.4	73.2	28.4	14.2	27.1	218.4	8.1	88.0	23.7	8.7	21.5	226.5
+RA-NEST	6.4	72.6	35.2	22.7	34.0	202.0	6.7	90.0	24.4	9.0	22.2	232.1
Llama-2-Chat _{13B}	12.0	75.9	19.9	4.9	18.0	218.4	8.2	72.8	17.5	5.3	15.7	181.7
+RA	6.5	91.5	38.9	24.2	37.2	249.3	5.9	86.6	23.6	9.1	21.5	228.7
+ <i>k</i> NN-LM	8.6	76.3	23.7	8.2	21.9	238.5	7.4	71.5	17.7	5.3	15.9	183.7
+NEST	7.2	67.1	29.3	15.6	28.1	207.1	6.8	86.0	22.9	8.7	20.9	212.3
+RA-NEST	5.8	86.8	38.6	24.0	37.0	245.5	5.7	90.1	24.7	9.2	22.4	229.4
Llama-2-Chat _{70B}	9.9	88.6	22.9	6.2	20.8	239.6	6.9	93.4	23.0	7.1	20.7	250.1
+RA	5.3	91.6	40.5	26.1	38.8	235.9	4.9	95.5	26.3	10.1	24.0	253.2
+ <i>k</i> NN-LM	7.1	83.6	26.1	9.6	24.1	253.9	6.3	94.4	23.1	7.2	20.8	251.3
+NEST	6.3	82.6	32.6	17.2	31.1	236.3	5.9	95.4	25.6	9.4	23.2	251.3
+RA-NEST	4.8	90.0	40.2	25.9	38.6	233.1	4.7	97.6	26.2	9.5	23.7	253.6

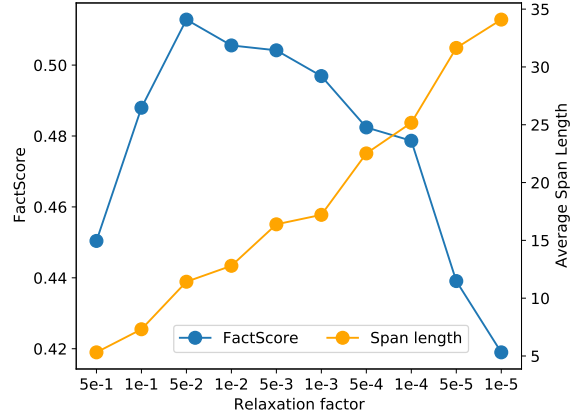
Models	TQA	NQ	HQA	MQA	Avg.	TruthfulQA		Biography		MMLU				
	Answer-Level Recall					ΔBLEU	ΔRG-1	FS	# Facts	Human.	STEM	Social	Other	Avg.
Llama-2-Chat _{7B}	61.1	38.9	30.6	9.3	35.0	-0.02	0.42	27.2	71.2	37.8	32.6	38.9	39.6	37.2
+RA	69.5	48.4	44.1	12.8	43.7	-0.34	0.18	56.5	67.1	41.8	35.3	42.2	43.3	40.7
+ <i>k</i> NN-LM	63.4	42.4	33.5	9.5	37.2	0.13	0.66	30.6	59.8	38.0	33.1	39.2	40.1	37.6
+NEST	61.5	43.2	33.5	10.2	37.1	0.03	0.45	38.9	58.2	42.0	35.4	42.0	43.4	40.7
+RA-NEST	69.0	48.8	45.3	13.3	44.1	-0.32	0.21	55.1	57.7	37.9	32.7	39.3	39.8	37.4
Llama-2-Chat _{13B}	63.5	42.3	32.6	10.2	37.2	0.13	0.81	28.8	49.9	41.5	35.0	40.2	43.8	40.1
+RA	70.9	51.6	44.6	14.0	45.3	-0.16	0.25	59.1	51.2	43.4	37.4	43.5	46.4	42.7
+ <i>k</i> NN-LM	64.7	43.5	34.2	11.2	38.4	0.20	0.95	31.1	46.1	41.4	34.7	40.6	44.2	40.2
+NEST	64.2	44.2	34.3	10.9	38.4	0.29	0.98	35.7	47.2	41.3	34.9	40.2	43.7	40.0
+RA-NEST	70.9	51.7	45.3	14.7	45.7	-0.14	0.25	58.4	52.4	43.5	37.7	43.5	46.7	42.8
Llama-2-Chat _{70B}	74.0	50.1	39.5	12.8	44.1	0.14	0.70	34.2	58.8	43.5	37.9	44.4	47.0	43.2
+RA	75.5	55.4	52.5	16.0	49.9	-0.13	0.40	52.9	42.1	45.9	39.7	46.2	48.6	45.1
+ <i>k</i> NN-LM	74.6	51.2	40.2	13.5	44.9	0.08	0.58	36.1	54.4	44.0	37.4	44.1	47.1	43.2
+NEST	74.2	51.6	41.4	13.8	45.2	0.17	0.70	41.6	56.2	43.8	38.0	44.4	47.8	43.5
+RA-NEST	75.4	55.2	52.4	16.3	49.8	-0.19	0.31	59.2	53.8	45.8	39.7	46.2	48.9	45.1

Table 10.1: Results on text completion (upper table) and other tasks (lower table). Bold numbers indicate the best performance. PPL: Perplexity. RG: ROUGE score. Avg. Len: Average generation length. ΔBLEU/ΔRG: The difference between the max score to correct references and the max score to incorrect references. FS: FACTScore with length penalty.

For **closed-set tasks**, NEST is comparable to RA and RA-NEST manages to achieve the best macro scores on average. Overall, NEST is able to outperform base LMs and *k*NN-LM’s on most tasks while being on par with RA. The combination of RA and NEST further improves over the two methods on some tasks. Despite the limited improvement, we will show that NEST is able to provide better attribution and latency in the following sections.



(a) Latency Breakdown.



(b) Span length and FACTScore on Biography.

Figure 10.2: Latency-accuracy trade-off and breakdown on Biography using Llama-2-Chat 70B+NEST. As the relaxation factor γ decreases, NEST tends to accept longer spans from the corpus. We choose $\gamma = 5e - 2$ in our main experiments, which accelerates the generation and improves the FACTScore.

10.4.5 Latency Analysis

Latency breakdown. The combination of dynamic span selection and relaxed speculative decoding can improve the latency of the LLM generation by quick draft proposal and processing multiple tokens at a time step. Figure 10.2a shows the latency breakdown of a NEST-70B model ($\alpha = 0.3, \tau = 0.1, \delta = 0.5$) for different relaxation factors on the Biography validation data. The latency experiment is done on $8 \times A100$ GPUs (for model parallelization) and 32 CPU threads (for search). The batch size is set to 1. We use internal, research-purpose implementation of the base Llama-2-chat model which did not optimize for latency. As we can see, the LM encoding time takes about half of the latency, while the sum of the others takes the rest. Noticeably, the cost of passage search and token index building stay relatively constant per query, while the others are related to the number of tokens processed per time step. Still, even with extra retrieval overheads, the slowest NEST model is faster than the base LM, showing the efficacy of span selection and speculative decoding.

Latency-accuracy trade-off. To understand why NEST can accelerate generation, we first show the latency-accuracy trade-off by tuning the relaxation factor in Figure 10.2b. The smaller γ is, the less often NEST rejects a segment retrieved from the corpus, which enables more tokens to be processed in parallel. The average proposed span length in Figure 10.2a can increase from 5 tokens to 35 tokens at each time step as the relaxation factor gets smaller. Combined with

Task	Models	Ratio (%)	Length	Example (Generated by NEST-70B)
NQ	NEST _{7B}	93.4	18.4	Question: who sings does he love me with reba?
	NEST _{13B}	52.4	4.4	Answer: The song "Does He Love You" [1] is a duet
	NEST _{70B}	58.8	4.3	between Reba McEntire and Linda Davis. It was
	RA-NEST _{7B}	77.5	3.1	released in August 1993 as the first single from Reba's
	RA-NEST _{13B}	33.2	3.0	compilation album Greatest Hits Volume 2. It is one of
	RA-NEST _{70B}	44.9	3.9	country music's several songs about a love triangle. [2]
Bio	NEST _{7B}	95.5	27.9	Question: Tell me a bio of Taral Hicks.
	NEST _{13B}	53.9	10.6	Answer: Taral Hicks (born September 21, 1974) is an American actress
	NEST _{70B}	58.6	7.0	and singer. Hicks is best known for her acting in such films as 1993's
	RA-NEST _{7B}	50.3	5.1	American crime drama film A Bronx Tale and her singing in such work
	RA-NEST _{13B}	48.5	5.9	as her 1997 [3] debut studio album This Time which peaked at No. 4 on
	RA-NEST _{70B}	80.7	11.0	Billboard Bubbling Under Hot 100 Singles. Raised in Teaneck, New Jersey, Hicks graduated from Teaneck High School in 1994 [3]. She is the younger sister of actress and singer D'atra Hicks.

Table 10.2: Attribution analysis. (Attribution) Ratio: Proportion of tokens that are taken from the corpus. (Attribution) Length: Average length of consecutive spans in the generation that are taken from the same document. **Green**: Segments taken from the corpus. **Gray**: Reference.

Figure 10.2a, we can reach the conclusion that fetching longer spans from the corpus results in lower generation latency per query. For the accuracy, the FACTSCORE on Biography validation data shows that there is a sweet spot around $\gamma = 5e - 2$ where both low latency and high accuracy can be achieved at the same time.

10.4.6 Attribution and Qualitative Analysis

One of the most important features of NEST is providing attribution directly at a span level, where the reference for the corresponding statement is accurate since it is directly taken from the corpus. Table 10.2 shows the attribution ratio, average attributed span length, and two examples for analysis. For NQ and Biography tasks, depending on the model and hyper-parameters in Equation (10.4) and (10.7), the ratio of tokens that can be traced back to the corpus ranges from 33.2% to 95.5%. In addition, it is more desirable to have consecutive segments that come from the same source so that consistent attribution can be provided, and the average length of spans taken from the corpus ranges from 3.0 to 27.9 tokens. This feature provides span-level attribution for most claims in the LLM generation. To our knowledge, neither of the baselines can achieve the same granularity and preciseness for the attribution as NEST.

10.5 Summary

This Chapter presents NEST, an inference-time revision method for LMs that improve their factuality and attribution through nearest neighbor speculative decoding. Leveraging two-stage k -NN search, relative retrieval confidence, dynamic span selection, and relaxed speculative decoding, NEST improves both validation perplexity and free-form generation quality on nine different tasks. Its performance can be further improved when combined with in-context retrieval augmentation. With these results, we demonstrate that NEST is capable of generating text grounded to real-world sources in low latency while maintaining fluency.

Chapter 11

Conclusion and Future Work

This thesis has presented multiple efficient and robust solutions to the field of IR based on pre-trained transformers. Throughout, we have explored different model architectures, compression techniques, and training strategies that have significantly pushed the boundaries of modern IR systems. Our work has demonstrated substantial advancements in first-stage retrievers and second-stage rerankers, where each model was evaluated under various scenarios to understand their strengths and limitations, providing deep insights into their practical applications. The integration of IR and LLMs further addresses the issues of factual accuracy and source attribution in generation. Specifically, our contributions can be summarized as follows:

- We implemented an unsupervised compression pipeline utilizing principal component analysis (PCA), product quantization, and hybrid search techniques. This pipeline has significantly reduced the computational costs and improved the scalability of retrieval systems.
- We introduced individual training of dense retrievers for different tasks, enhancing robustness by aggregating predictions at test time using uncertainty estimation. This approach has been applied to cross-encoder rankers to improve reliability on out-of-distribution data.
- We developed CITADEL and SLIM methods for multi-vector retrieval, optimizing token interactions to reduce computational demands while maintaining high retrieval accuracy. CITADEL utilizes dynamic lexical routing, and SLIM integrates a sparsified late interaction with inverted indexes, enabling compatibility with traditional bag-of-words systems.
- We improved the robustness of cross-encoder rankers by incorporating late interaction layers, which significantly enhances performance on out-of-distribution data by optimizing token-level interactions within documents. We also studied how to use query expansion to improve the robustness of cross-encoder rankers by high-quality keyword generation and minimally disruptive query modification.

- We proposed a method for certified error control in two-stage relevance ranking systems, enabling precise error control during candidate set pruning. This method uses a truncated risk function to provide theoretical guarantees for error bounds, improving the reliability of the candidate set pruning method.
- We developed the NEST approach, extending k NN-LM by adding a passage retrieval step at the beginning, and dynamically adapting the output using confidence-based interpolation, dynamic span selection, and speculative decoding. This method improves the generation quality and reduces the latency of the LLM.

The integration of pretrained transformers with retrieval systems presents a powerful avenue for improving the access and reliability of information across various domains. As this field continues to evolve, the potential to redefine performance benchmarks and practical applications of AI in IR is immense. These advancements will address the growing demands for efficient and robust information retrieval systems across various digital platforms.

At the end of the thesis, we discuss several future directions for further research on the efficiency and robustness of modern IR systems.

- **Instruction-Following Retrieval:** One of the most important capabilities of LLMs is being able to follow the instructions of users given the prompt. Specifying the user’s intent in the prompt will be a useful feature for the retriever as well to increase its robustness and efficiency. For example, for the instruction “Find a sentence from a scientific paper to check if the statement is correct or not”, instead of searching over the entire internet, the retriever knows it only needs to look up scientific papers and therefore will only search the relevant academic databases. This can drastically reduce search latency while avoiding error accumulation of methods that use an intent classifier. Similar works [11, 301] and benchmarks [235, 355] have shown that retriever models based on pretrained transformers can follow simple instructions after finetuning, and yet challenges remain for more complex intents and partially unfulfillable queries. It will be also interesting to see how to leverage existing LLMs that understand instructions well and apply them for instruction-following retrieval in zero-shot or few-shots.
- **Interpretable Reranking beyond Relevance:** The second-stage rerankers we explore in Chapter 7, 8, and 9 are mostly based on encoder-only pretrained transformers such as BERT. The common reranking paradigm is to feed the query and document jointly to the rerankers and compute a relevance score for the query-document pair. However, it is sometimes hard to interpret the relevance score as to which part of the document is useful for answering the query. Recent works [65, 383] that leverage encoder-decoder or decoder-only pretrained transformers can generate explanations along with the relevance scores to point out why the documents are assigned with such scores. However, it is found that outputting relevance

scores before the explanations performs better than the other way around, leaving room for improvement in chain-of-thought style [353] reranking (i.e., generating explanations before computing the score).

- **Planning for Retrieval in Generation:** In Chapter 10, we showed that inserting retrieved information into the LLM output can improve attribution and factuality. However, retrieving at every step still consumes a considerable amount of latency. Therefore, it is tempting to predict whether to retrieve at every step before retrieval and generation to save computation. Recent work [12, 339] on planning for retrieval has shown that it is possible to skip some retrieval steps during retrieval-augmented generation with a slight accuracy drop. The challenges remain when the knowledge base changes or gets updated, as the planner can not know whether the relevant information is in the knowledge base now, impeding its practical deployment in real applications. It will also be interesting to see fast planners that make decisions based on a coarse search of the knowledge base to determine whether there is useful information in the corpus.

References

- [1] Nasreen Abdul-Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. Umass at trec 2004: Novelty and hard. *Computer Science Department Faculty Publication Series*, page 189, 2004.
- [2] Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. Explanations for CommonsenseQA: New dataset and models. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3050–3065. Association for Computational Linguistics, 2021.
- [3] Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, et al. Raft: A real-world few-shot text classification benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [4] Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *Association for Computing Machinery*, 20(4):357–389, oct 2002.
- [5] Dario Amodei, Christopher Olah, J. Steinhardt, P. Christiano, J. Schulman, and Dandelion Mané. Concrete problems in ai safety. *ArXiv*, abs/1606.06565, 2016.
- [6] Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul R. Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick

Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, YaGuang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Gura, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Merey, Martin Baeuml, Zhifeng Chen, Laurent El Shafey, Yujing Zhang, Olcan Sercinoglu, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, Alexandre Frechette, Charlotte Smith, Laura Culp, Lev Proleev, Yi Luan, Xi Chen, James Lottes, Nathan Schucher, Federico Lebron, Alban Rustemi, Natalie Clay, Phil Crone, Tomas Kocisky, Jeffrey Zhao, Bartek Perz, Dian Yu, Heidi Howard, Adam Bloniarz, Jack W. Rae, Han Lu, Laurent Sifre, Marcello Maggioni, Fred Alcober, Dan Garrette, Megan Barnes, Shantanu Thakoor, Jacob Austin, Gabriel Barth-Maron, William Wong, Rishabh Joshi, Rahma Chaabouni, Deeni Fatiha, Arun Ahuja, Gaurav Singh Tomar, Evan Senter, Martin Chadwick, Ilya Kornakov, Nithya Attaluri, Iñaki Iturrate, Ruibo Liu, Yunxuan Li, Sarah Cogan, Jeremy Chen, Chao Jia, Chenjie Gu, Qiao Zhang, Jordan Grimstad, Ale Jakse Hartman, Xavier Garcia, Thanumalayan Sankaranarayana Pillai, Jacob Devlin, Michael Laskin, Diego de Las Casas, Dasha Valter, Connie Tao, Lorenzo Blanco, Adrià Puigdomènech Badia, David Reitter, Mianna Chen, Jenny Brennan, Clara Rivera, Sergey Brin, Shariq Iqbal, Gabriela Surita, Jane Labanowski, Abhi Rao, Stephanie Winkler, Emilio Parisotto, Yiming Gu, Kate Olszewska, Ravi Addanki, Antoine Miech, Annie Louis, Denis Teplyashin, Geoff Brown, Elliot Catt, Jan Balaguer, Jackie Xiang, Pidong Wang, Zoe Ashwood, Anton Briukhov, Albert Webson, Sanjay Ganapathy, Smit Sanghavi, Ajay Kannan, Ming-Wei Chang, Axel Stjerngren, Josip Djolonga, Yuting Sun, Ankur Bapna, Matthew Aitchison, Pedram Pejman, Henryk Michalewski, Tianhe Yu, Cindy Wang, Juliette Love, Junwhan Ahn, Dawn Bloxwich, Kehang Han, Peter Humphreys, Thibault Sellam, James Bradbury, Varun Godbole, Sina Samangooei, Bogdan Damoc, Alex Kaskasoli, Sébastien M. R. Arnold, Vijay Vasudevan, Shubham Agrawal, Jason Riesa, Dmitry Lepikhin, Richard Tanburn, Srivatsan Srinivasan, Hyeontaek Lim, Sarah Hodgkinson, Pranav Shyam, Johan Ferret, Steven Hand, Ankush Garg, Tom Le Paine, Jian Li, Yujia Li, Minh Giang, Alexander Neitz, Zaheer Abbas, Sarah York, Machel Reid, Elizabeth Cole, Aakanksha Chowdhery, Dipanjan Das, Dominika Rogozińska, Vitaliy Nikolaev, Pablo Sprechmann, Zachary Nado, Lukas Zilka, Flavien Prost, Luheng He, Marianne Monteiro, Gaurav Mishra, Chris Welty, Josh Newlan, Dawei Jia, Miltiadis Allamanis, Clara Huiyi Hu, Raoul de Liedekerke, Justin Gilmer, Carl Saroufim, Shruti Rijhwani, Shaobo Hou, Disha Shrivastava, Anirudh Baddepudi, Alex Goldin, Adnan Ozturel, Albin Cassirer, Yunhan Xu, Daniel Sohn, Devendra Sachan, Reinald Kim Amplayo, Craig Swanson, Dessie Petrova, Shashi Narayan, Arthur Guez, Siddhartha Brahma, Jessica Landon, Miteyan Patel, Ruizhe Zhao, Kevin Vilella, Luyu Wang, Wenhao Jia, Matthew Rahtz, Mai Giménez, Legg Yeung, James Keeling, Petko Georgiev, Diana Mincu, Boxi

Wu, Salem Haykal, Rachel Saputro, Kiran Vodrahalli, James Qin, Zeynep Cankara, Abhanshu Sharma, Nick Fernando, Will Hawkins, Behnam Neyshabur, Solomon Kim, Adrian Hutter, Priyanka Agrawal, Alex Castro-Ros, George van den Driessche, Tao Wang, Fan Yang, Shuo yiin Chang, Paul Komarek, Ross McIlroy, Mario Lučić, Guodong Zhang, Wael Farhan, Michael Sharman, Paul Natsev, Paul Michel, Yamini Bansal, Siyuan Qiao, Kris Cao, Siamak Shakeri, Christina Butterfield, Justin Chung, Paul Kishan Rubenstein, Shivani Agrawal, Arthur Mensch, Kedar Soparkar, Karel Lenc, Timothy Chung, Aedan Pope, Loren Maggiore, Jackie Kay, Priya Jhakra, Shibo Wang, Joshua Maynez, Mary Phuong, Taylor Tobin, Andrea Tacchetti, Maja Trebacz, Kevin Robinson, Yash Katariya, Sebastian Riedel, Paige Bailey, Kefan Xiao, Nimesh Ghelani, Lora Aroyo, Ambrose Slone, Neil Houlsby, Xuehan Xiong, Zhen Yang, Elena Gribovskaya, Jonas Adler, Mateo Wirth, Lisa Lee, Music Li, Thais Kagohara, Jay Pavagadhi, Sophie Bridgers, Anna Bortsova, Sanjay Ghemawat, Zafarali Ahmed, Tianqi Liu, Richard Powell, Vijay Bolina, Mariko Iinuma, Polina Zablotskaia, James Besley, Da-Woon Chung, Timothy Dozat, Ramona Comanescu, Xiance Si, Jeremy Greer, Guolong Su, Martin Polacek, Raphaël Lopez Kaufman, Simon Tokumine, Hexiang Hu, Elena Buchatskaya, Yingjie Miao, Mohamed Elhawaty, Aditya Siddhant, Nenad Tomasev, Jinwei Xing, Christina Greer, Helen Miller, Shereen Ashraf, Aurko Roy, Zizhao Zhang, Ada Ma, Angelos Filos, Milos Besta, Rory Blevins, Ted Klimenko, Chih-Kuan Yeh, Soravit Changpinyo, Jiaqi Mu, Oscar Chang, Mantas Pajarskas, Carrie Muir, Vered Cohen, Charline Le Lan, Krishna Haridasan, Amit Marathe, Steven Hansen, Sholto Douglas, Rajkumar Samuel, Mingqiu Wang, Sophia Austin, Chang Lan, Jiepu Jiang, Justin Chiu, Jaime Alonso Lorenzo, Lars Lowe Sjösund, Sébastien Cevey, Zach Gleicher, Thi Avrahami, Anudhyan Boral, Hansa Srinivasan, Vittorio Selo, Rhys May, Konstantinos Aisopos, Léonard Hussenot, Livio Baldini Soares, Kate Baumli, Michael B. Chang, Adrià Recasens, Ben Caine, Alexander Pritzel, Filip Pavetic, Fabio Pardo, Anita Gergely, Justin Frye, Vinay Ramasesh, Dan Horgan, Kartikeya Badola, Nora Kassner, Subhrajit Roy, Ethan Dyer, Víctor Campos Campos, Alex Tomala, Yunhao Tang, Dalia El Badawy, Elspeth White, Basil Mustafa, Oran Lang, Abhishek Jindal, Sharad Vikram, Zhitao Gong, Sergi Caelles, Ross Hemsley, Gregory Thornton, Fangxiaoyu Feng, Wojciech Stokowiec, Ce Zheng, Phoebe Thacker, Çağlar Ünlü, Zhishuai Zhang, Mohammad Saleh, James Svensson, Max Bileschi, Piyush Patil, Ankesh Anand, Roman Ring, Katerina Tsihlias, Arpi Vezer, Marco Selvi, Toby Shevlane, Mikel Rodriguez, Tom Kwiatkowski, Samira Daruki, Keran Rong, Allan Dafoe, Nicholas FitzGerald, Keren Gu-Lemberg, Mina Khan, Lisa Anne Hendricks, Marie Pellat, Vladimir Feinberg, James Cobon-Kerr, Tara Sainath, Maribeth Rauh, Sayed Hadi Hashemi, Richard Ives, Yana Hasson, Eric Noland, Yuan Cao, Nathan Byrd, Le Hou, Qingze Wang, Thibault Sottiaux, Michela Paganini, Jean-Baptiste Lespiau, Alexandre Moufarek, Samer Hassan, Kaushik Shivakumar, Joost van Amersfoort, Amol Mandhane, Pratik Joshi, Anirudh Goyal, Matthew Tung, Andrew Brock, Hannah Sheahan, Vedant Misra, Cheng Li, Nemanja Rakićević, Mostafa Dehghani, Fangyu Liu, Sid Mittal, Junhyuk Oh, Seb Noury, Eren Sezener, Fantine Huot, Matthew Lamm, Nicola De

Cao, Charlie Chen, Sidharth Mudgal, Romina Stella, Kevin Brooks, Gautam Vasudevan, Chenxi Liu, Mainak Chain, Nivedita Melinkeri, Aaron Cohen, Venus Wang, Kristie Seymore, Sergey Zubkov, Rahul Goel, Summer Yue, Sai Krishnakumaran, Brian Albert, Nate Hurley, Motoki Sano, Anhad Mohananey, Jonah Joughin, Egor Filonov, Tomasz Kepa, Yomna Eldawy, Jiawern Lim, Rahul Rishi, Shirin Badiezhadegan, Taylor Bos, Jerry Chang, Sanil Jain, Sri Gayatri Sundara Padmanabhan, Subha Puttagunta, Kalpesh Krishna, Leslie Baker, Norbert Kalb, Vamsi Bedapudi, Adam Kurzrok, Shuntong Lei, Anthony Yu, Oren Litvin, Xiang Zhou, Zhichun Wu, Sam Sobell, Andrea Siciliano, Alan Papir, Robby Neale, Jonas Bragagnolo, Tej Toor, Tina Chen, Valentin Anklin, Feiran Wang, Richie Feng, Milad Gholami, Kevin Ling, Lijuan Liu, Jules Walter, Hamid Moghaddam, Arun Kishore, Jakub Adamek, Tyler Mercado, Jonathan Mallinson, Siddhinita Wandekar, Stephen Cagle, Eran Ofek, Guillermo Garrido, Clemens Lombriser, Maksim Mukha, Botu Sun, Hafeezul Rahman Mohammad, Josip Matak, Yadi Qian, Vikas Peswani, Pawel Janus, Quan Yuan, Leif Schelin, Oana David, Ankur Garg, Yifan He, Oleksii Duzhyi, Anton Älgmyr, Timothée Lotz, Qi Li, Vikas Yadav, Luyao Xu, Alex Chinien, Rakesh Shivanna, Aleksandr Chuklin, Josie Li, Carrie Spadine, Travis Wolfe, Kareem Mohamed, Subhabrata Das, Zihang Dai, Kyle He, Daniel von Dincklage, Shyam Upadhyay, Akanksha Maurya, Luyan Chi, Sebastian Krause, Khalid Salama, Pam G Rabinovitch, Pavan Kumar Reddy M, Aarush Selvan, Mikhail Dektiarev, Golnaz Ghiasi, Erdem Guven, Himanshu Gupta, Boyi Liu, Deepak Sharma, Idan Heimlich Shtacher, Shachi Paul, Oscar Akerlund, François-Xavier Aubet, Terry Huang, Chen Zhu, Eric Zhu, Elico Teixeira, Matthew Fritze, Francesco Bertolini, Liana-Eleonora Marinescu, Martin Bölle, Dominik Paulus, Khyatti Gupta, Tejasi Latkar, Max Chang, Jason Sanders, Roopa Wilson, Xuewei Wu, Yi-Xuan Tan, Lam Nguyen Thiet, Tulsee Doshi, Sid Lall, Swaroop Mishra, Wanming Chen, Thang Luong, Seth Benjamin, Jasmine Lee, Ewa Andrejczuk, Dominik Rabiej, Vipul Ranjan, Krzysztof Styrz, Pengcheng Yin, Jon Simon, Malcolm Rose Harriott, Mudit Bansal, Alexei Robsky, Geoff Bacon, David Greene, Daniil Mirylenka, Chen Zhou, Obaid Sarvana, Abhimanyu Goyal, Samuel Andermatt, Patrick Siegler, Ben Horn, Assaf Israel, Francesco Pongetti, Chih-Wei "Louis" Chen, Marco Selvatici, Pedro Silva, Kathie Wang, Jackson Tolins, Kelvin Guu, Roey Yogeve, Xiaochen Cai, Alessandro Agostini, Maulik Shah, Hung Nguyen, Noah Ó Donnaile, Sébastien Pereira, Linda Friso, Adam Stambler, Adam Kurzrok, Chenkai Kuang, Yan Romanikhin, Mark Geller, ZJ Yan, Kane Jang, Cheng-Chun Lee, Wojciech Fica, Eric Malmi, Qijun Tan, Dan Banica, Daniel Balle, Ryan Pham, Yanping Huang, Diana Avram, Hongzhi Shi, Jasjot Singh, Chris Hidey, Niharika Ahuja, Pranab Saxena, Dan Dooley, Srividya Pranavi Potharaju, Eileen O'Neill, Anand Gokulchandran, Ryan Foley, Kai Zhao, Mike Dusenberry, Yuan Liu, Pulkit Mehta, Ragha Kotikalapudi, Chalence Safranek-Shrader, Andrew Goodman, Joshua Kessinger, Eran Globen, Prateek Kolhar, Chris Gorgolewski, Ali Ibrahim, Yang Song, Ali Eichenbaum, Thomas Brovelli, Sahitya Potluri, Preethi Lahoti, Cip Baetu, Ali Ghorbani, Charles Chen, Andy Crawford, Shalini Pal, Mukund Sridhar, Petru Gurita, Asier Mujika, Igor Petrovski, Pierre-Louis Cedoz, Chenmei Li, Shiyuan Chen, Niccolò Dal

Santo, Siddharth Goyal, Jitesh Punjabi, Karthik Kappaganthu, Chester Kwak, Pallavi LV, Sarmishta Velury, Himadri Choudhury, Jamie Hall, Premal Shah, Ricardo Figueira, Matt Thomas, Minjie Lu, Ting Zhou, Chintu Kumar, Thomas Jurdi, Sharat Chikkerur, Yenai Ma, Adams Yu, Soo Kwak, Victor Ähdel, Sujeewan Rajayogam, Travis Choma, Fei Liu, Aditya Barua, Colin Ji, Ji Ho Park, Vincent Hellendoorn, Alex Bailey, Taylan Bilal, Huanjie Zhou, Mehrdad Khatir, Charles Sutton, Wojciech Rządowski, Fiona Macintosh, Konstantin Shagin, Paul Medina, Chen Liang, Jinjing Zhou, Pararth Shah, Yingying Bi, Attila Dankovics, Shipra Banga, Sabine Lehmann, Marissa Bredesen, Zifan Lin, John Eric Hoffmann, Jonathan Lai, Raynald Chung, Kai Yang, Nihal Balani, Arthur Bražinskas, Andrei Sozanschi, Matthew Hayes, Héctor Fernández Alcalde, Peter Makarov, Will Chen, Antonio Stella, Liselotte Snijders, Michael Mandl, Ante Kärrman, Paweł Nowak, Xinyi Wu, Alex Dyck, Krishnan Vaidyanathan, Raghavender R, Jessica Mallet, Mitch Rudominer, Eric Johnston, Sushil Mittal, Akhil Udathu, Janara Christensen, Vishal Verma, Zach Irving, Andreas Santucci, Gamaleldin Elsayed, Elnaz Davoodi, Marin Georgiev, Ian Tenney, Nan Hua, Geoffrey Cideron, Edouard Leurent, Mahmoud Alnahlawi, Ionut Georgescu, Nan Wei, Ivy Zheng, Dylan Scandinaro, Heinrich Jiang, Jasper Snoek, Mukund Sundararajan, Xuezhi Wang, Zack Ontiveros, Itay Karo, Jeremy Cole, Vinu Rajashekhar, Lara Tumei, Eyal Ben-David, Rishub Jain, Jonathan Uesato, Romina Datta, Oskar Bunyan, Shimu Wu, John Zhang, Piotr Stanczyk, Ye Zhang, David Steiner, Subhajit Naskar, Michael Azzam, Matthew Johnson, Adam Paszke, Chung-Cheng Chiu, Jaume Sanchez Elias, Afroz Mohiuddin, Faizan Muhammad, Jin Miao, Andrew Lee, Nino Vieillard, Jane Park, Jiageng Zhang, Jeff Stanway, Drew Garmon, Abhijit Karmarkar, Zhe Dong, Jong Lee, Aviral Kumar, Luowei Zhou, Jonathan Evens, William Isaac, Geoffrey Irving, Edward Loper, Michael Fink, Isha Arkatkar, Nanxin Chen, Izhak Shafran, Ivan Petrychenko, Zhe Chen, Johnson Jia, Anselm Levskaya, Zhenkai Zhu, Peter Grabowski, Yu Mao, Alberto Magni, Kaisheng Yao, Javier Snaider, Norman Casagrande, Evan Palmer, Paul Suganthan, Alfonso Castaño, Irene Giannoumis, Wooyeol Kim, Mikołaj Rybiński, Ashwin Sreevatsa, Jennifer Prendki, David Soergel, Adrian Goedeckemeyer, Willi Gierke, Mohsen Jafari, Meenu Gaba, Jeremy Wiesner, Diana Gage Wright, Yawen Wei, Harsha Vashisht, Yana Kulizhskaya, Jay Hoover, Maigo Le, Lu Li, Chimezie Iwuanyanwu, Lu Liu, Kevin Ramirez, Andrey Khorlin, Albert Cui, Tian LIN, Marcus Wu, Ricardo Aguilar, Keith Pallo, Abhishek Chakladar, Ginger Perng, Elena Allica Abellan, Mingyang Zhang, Ishita Dasgupta, Nate Kushman, Ivo Penchev, Alena Repina, Xihui Wu, Tom van der Weide, Priya Ponnappalli, Caroline Kaplan, Jiri Simsa, Shuangfeng Li, Olivier Dousse, Fan Yang, Jeff Piper, Nathan Ie, Rama Pasumarthi, Nathan Lintz, Anitha Vijayakumar, Daniel Andor, Pedro Valenzuela, Minnie Lui, Cosmin Paduraru, Daiyi Peng, Katherine Lee, Shuyuan Zhang, Somer Greene, Duc Dung Nguyen, Paula Kurylowicz, Cassidy Hardin, Lucas Dixon, Lili Janzer, Kiam Choo, Ziqiang Feng, Biao Zhang, Achintya Singhal, Dayou Du, Dan McKinnon, Natasha Antropova, Tolga Bolukbasi, Orgad Keller, David Reid, Daniel Finchelstein, Maria Abi Raad, Remi Crocker, Peter Hawkins, Robert Dadashi, Colin Gaffney, Ken Franko, Anna Bulanova, Rémi Leblond,

Shirley Chung, Harry Askham, Luis C. Cobo, Kelvin Xu, Felix Fischer, Jun Xu, Christina Sorokin, Chris Alberti, Chu-Cheng Lin, Colin Evans, Alek Dimitriev, Hannah Forbes, Dylan Banarse, Zora Tung, Mark Omernick, Colton Bishop, Rachel Sterneck, Rohan Jain, Jiawei Xia, Ehsan Amid, Francesco Piccinno, Xingyu Wang, Praseem Banzal, Daniel J. Mankowitz, Alex Polozov, Victoria Krakovna, Sasha Brown, MohammadHossein Bateni, Dennis Duan, Vlad Firoiu, Meghana Thotakuri, Tom Natan, Matthieu Geist, Ser tan Girgin, Hui Li, Jiayu Ye, Ofir Roval, Reiko Tojo, Michael Kwong, James Lee-Thorp, Christopher Yew, Danila Sinopalnikov, Sabela Ramos, John Mellor, Abhishek Sharma, Kathy Wu, David Miller, Nicolas Sonnerat, Denis Vnukov, Rory Greig, Jennifer Beattie, Emily Caveness, Libin Bai, Julian Eisenschlos, Alex Korchemniy, Tomy Tsai, Mimi Jasarevic, Weize Kong, Phuong Dao, Zeyu Zheng, Frederick Liu, Fan Yang, Rui Zhu, Tian Huey Teh, Jason Sanmiya, Evgeny Gladchenko, Nejc Trdin, Daniel Toyama, Evan Rosen, Sasan Tavakkol, Linting Xue, Chen Elkind, Oliver Woodman, John Carpenter, George Papamakarios, Rupert Kemp, Sushant Kafle, Tanya Grunina, Rishika Sinha, Alice Talbert, Diane Wu, Denese Owusu-Afriyie, Cosmo Du, Chloe Thornton, Jordi Pont-Tuset, Pradyumna Narayana, Jing Li, Saaber Fatehi, John Wieting, Omar Ajmeri, Benigno Urias, Yeongil Ko, Laura Knight, Amélie Héliou, Ning Niu, Shane Gu, Chenxi Pang, Yeqing Li, Nir Levine, Ariel Stolovich, Rebeca Santamaria-Fernandez, Sonam Goenka, Wenny Yustalim, Robin Strudel, Ali Elqursh, Charlie Deck, Hyo Lee, Zonglin Li, Kyle Levin, Raphael Hoffmann, Dan Holtmann-Rice, Olivier Bachem, Sho Arora, Christy Koh, Soheil Hassas Yeganeh, Siim Pöder, Mukarram Tariq, Yanhua Sun, Lucian Ionita, Mojtaba Seyedhosseini, Pouya Tafti, Zhiyu Liu, Anmol Gulati, Jasmine Liu, Xinyu Ye, Bart Chrzasczcz, Lily Wang, Nikhil Sethi, Tianrun Li, Ben Brown, Shreya Singh, Wei Fan, Aaron Parisi, Joe Stanton, Vinod Koverkathu, Christopher A. Choquette-Choo, Yunjie Li, TJ Lu, Abe Ittycheriah, Prakash Shroff, Mani Varadarajan, Sanaz Bahargam, Rob Willoughby, David Gaddy, Guillaume Desjardins, Marco Cornero, Brona Robenek, Bhavishya Mittal, Ben Albrecht, Ashish Shenoy, Fedor Moiseev, Henrik Jacobsson, Alireza Ghaffarkhah, Morgane Rivi  re, Alanna Walton, Cl  ment Crepy, Alicia Parrish, Zongwei Zhou, Clement Farabet, Carey Radebaugh, Praveen Srinivasan, Claudia van der Salm, Andreas F  djeland, Salvatore Scellato, Eri Latorre-Chimoto, Hanna Klimczak-Pluci  nska, David Bridson, Dario de Cesare, Tom Hudson, Piermaria Mendolicchio, Lexi Walker, Alex Morris, Matthew Mauger, Alexey Guseynov, Alison Reid, Seth Odoom, Lucia Loher, Victor Cotruta, Madhavi Yenugula, Dominik Grewe, Anastasia Petrushkina, Tom Duerig, Antonio Sanchez, Steve Yadlowsky, Amy Shen, Amir Globerson, Lynette Webb, Sahil Dua, Dong Li, Surya Bhupatiraju, Dan Hurt, Haroon Qureshi, Ananth Agarwal, Tomer Shani, Matan Eyal, Anuj Khare, Shreyas Rammohan Belle, Lei Wang, Chetan Tekur, Mihir Sanjay Kale, Jinliang Wei, Ruoxin Sang, Brennan Saeta, Tyler Liechty, Yi Sun, Yao Zhao, Stephan Lee, Pandu Nayak, Doug Fritz, Manish Reddy Vuyyuru, John Aslanides, Nidhi Vyas, Martin Wicke, Xiao Ma, Evgenii Eltyshev, Nina Martin, Hardie Cate, James Manyika, Keyvan Amiri, Yelin Kim, Xi Xiong, Kai Kang, Florian Luisier, Nilesch Tripuraneni, David Madras, Mandy Guo, Austin Waters, Oliver Wang, Joshua Ainslie, Jason Baldridge, Han Zhang,

Garima Pruthi, Jakob Bauer, Feng Yang, Riham Mansour, Jason Gelman, Yang Xu, George Polovets, Ji Liu, Honglong Cai, Warren Chen, XiangHai Sheng, Emily Xue, Sherjil Ozair, Christof Angermueller, Xiaowei Li, Anoop Sinha, Weiren Wang, Julia Wiesinger, Emmanouil Koukoumidis, Yuan Tian, Anand Iyer, Madhu Gurumurthy, Mark Goldenson, Parashar Shah, MK Blake, Hongkun Yu, Anthony Urbanowicz, Jennimaria Palomaki, Chrisantha Fernando, Ken Durden, Harsh Mehta, Nikola Momchev, Elahe Rahimtoroghi, Maria Georgaki, Amit Raul, Sebastian Ruder, Morgan Redshaw, Jinhyuk Lee, Denny Zhou, Komal Jalan, Dinghua Li, Blake Hechtman, Parker Schuh, Milad Nasr, Kieran Milan, Vladimir Mikulik, Juliana Franco, Tim Green, Nam Nguyen, Joe Kelley, Aroma Mahendru, Andrea Hu, Joshua Howland, Ben Vargas, Jeffrey Hui, Kshitij Bansal, Vikram Rao, Rakesh Ghiya, Emma Wang, Ke Ye, Jean Michel Sarr, Melanie Moranski Preston, Madeleine Elish, Steve Li, Aakash Kaku, Jigar Gupta, Ice Pasupat, Da-Cheng Juan, Milan Someswar, Tejvi M., Xinyun Chen, Aida Amini, Alex Fabrikant, Eric Chu, Xuanyi Dong, Amruta Muthal, Senaka Buthpitiya, Sarthak Jauhari, Nan Hua, Urvashi Khandelwal, Ayal Hitron, Jie Ren, Larissa Rinaldi, Shahar Drath, Avigail Dabush, Nan-Jiang Jiang, Harshal Godhia, Uli Sachs, Anthony Chen, Yicheng Fan, Hagai Taitelbaum, Hila Noga, Zhuyun Dai, James Wang, Chen Liang, Jenny Hamer, Chun-Sung Ferng, Chenel Elkind, Aviel Atias, Paulina Lee, Vít Listík, Mathias Carlen, Jan van de Kerkhof, Marcin Pikuś, Krunoslav Zaher, Paul Müller, Sasha Zykova, Richard Stefanec, Vitaly Gatsko, Christoph Hirnschall, Ashwin Sethi, Xingyu Federico Xu, Chetan Ahuja, Beth Tsai, Anca Stefanoiu, Bo Feng, Keshav Dhandhaniah, Manish Katyal, Akshay Gupta, Atharva Parulekar, Divya Pitta, Jing Zhao, Vivaan Bhatia, Yashodha Bhavnani, Omar Alhadlaq, Xiaolin Li, Peter Danenberg, Dennis Tu, Alex Pine, Vera Filippova, Abhipso Ghosh, Ben Limonchik, Bhargava Urala, Chaitanya Krishna Lanka, Derik Clive, Yi Sun, Edward Li, Hao Wu, Kevin Hongtongsak, Ianna Li, Kalind Thakkar, Kuanysh Omarov, Kushal Majmundar, Michael Alverson, Michael Kucharski, Mohak Patel, Mudit Jain, Maksim Zabelin, Paolo Pelagatti, Rohan Kohli, Saurabh Kumar, Joseph Kim, Swetha Sankar, Vineet Shah, Lakshmi Ramachandruni, Xiangkai Zeng, Ben Bariach, Laura Weidinger, Amar Subramanya, Sissie Hsiao, Demis Hassabis, Koray Kavukcuoglu, Adam Sadosky, Quoc Le, Trevor Strohman, Yonghui Wu, Slav Petrov, Jeffrey Dean, and Oriol Vinyals. Gemini: A family of highly capable multimodal models. *arXiv*, abs/2312.11805, 2024.

- [7] Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepey, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin,

Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. PaLM 2 technical report. *arXiv*, abs/2305.10403, 2023.

- [8] Nima Asadi and Jimmy Lin. Effectiveness/efficiency tradeoffs for candidate generation in multi-stage retrieval architectures. *Proceedings of the 36th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2013.
- [9] Akari Asai, Matt Gardner, and Hannaneh Hajishirzi. Evidentiality-guided generation for knowledge-intensive NLP tasks. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2226–2243, Seattle, United States, July 2022. Association for Computational Linguistics.
- [10] Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. Retrieval-based language models and applications. In Yun-Nung (Vivian) Chen, Margot Margot, and Siva Reddy, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pages 41–46, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [11] Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. Task-aware retrieval with instructions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3650–3675, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [12] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *ArXiv*, abs/2310.11511, 2023.

- [13] Akari Asai, Zexuan Zhong, Danqi Chen, Pang Wei Koh, Luke Zettlemoyer, Hannaneh Hajishirzi, and Wen tau Yih. Reliable, adaptable, and attributable language models with retrieval. *arXiv*, abs/2403.03187, 2024.
- [14] Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-david, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Fries, Maged Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xi-angru Tang, Dragomir Radev, Mike Tian-jian Jiang, and Alexander Rush. PromptSource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [15] Ashraf Bah Rabiou and Ben Carterette. Pdf: A probabilistic data fusion framework for retrieval and ranking. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval, ICTIR '16*, page 31–39, New York, NY, USA, 2016.
- [16] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
- [17] Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. Sparterm: Learning term-based sparse representation for fast text retrieval. *ArXiv*, abs/2010.00768, 2020.
- [18] Peter Bailey, Alistair Moffat, Falk Scholer, and Paul Thomas. Retrieval consistency in the presence of query variations. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17*, page 395–404, New York, NY, USA, 2017.
- [19] Stephen Bates, A. Angelopoulos, Lihua Lei, Jitendra Malik, and Michael I. Jordan. Distribution-free, risk-controlling prediction sets. *J. ACM*, 68:43:1–43:34, 2021.
- [20] Petr Baudiš and Jan Šedivý. Modeling of the question answering task in the YodaQA system. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 222–228. Springer, 2015.
- [21] N.J. Belkin, P. Kantor, E.A. Fox, and J.A. Shaw. Combining the evidence of multiple query representations for information retrieval. *Information Processing and Management*, 31(3):431–448, 1995. The Second Text Retrieval Conference (TREC-2).

- [22] Michael Bendersky, Honglei Zhuang, Ji Ma, Shuguang Han, Keith Hall, and Ryan McDonald. Rrf102: Meeting the trec-covid challenge with a 100+ runs ensemble. *arXiv preprint arXiv:2010.00200*, 2020.
- [23] Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In *Large Scale Kernel Machines*. MIT Press, 2007.
- [24] Rodger Benham and J. Shane Culpepper. Risk-reward trade-offs in rank fusion. In *Proceedings of the 22nd Australasian Document Computing Symposium, ADCS '17*, New York, NY, USA, 2017.
- [25] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.
- [26] Rishabh Bhardwaj, George Polovets, and Monica Sunkara. Adaptation approaches for nearest neighbor language models. *arXiv*, abs/2211.07828, 2023.
- [27] J. Bhogal, A. Macfarlane, and P. Smith. A review of ontology based query expansion. *Inf. Process. Manage.*, 43(4):866–886, jul 2007.
- [28] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.
- [29] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ B. Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021.
- [30] Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. Overview of touché 2020: Argument retrieval: Extended abstract. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International*

Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings, page 384–395, Berlin, Heidelberg, 2020. Springer-Verlag.

- [31] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. *ArXiv*, abs/2112.04426, 2022.
- [32] Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. A full-text learning to rank dataset for medical information retrieval. In Nicola Ferro, Fabio Crestani, Marie-Francine Moens, Josiane Mothe, Fabrizio Silvestri, Giorgio Maria Di Nunzio, Claudia Hauff, and Gianmaria Silvello, editors, *Advances in Information Retrieval*, pages 716–722, Cham, 2016. Springer International Publishing.
- [33] Abdoulahi Boubacar and Zhendong Niu. Concept based query expansion. In *2013 Ninth International Conference on Semantics, Knowledge and Grids*, pages 198–201, 2013.
- [34] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [35] Francesco Busolin, Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. Learning early exit strategies for additive ranking ensembles. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [36] Berkant Barla Cambazoglu, Hugo Zaragoza, Olivier Chapelle, Jiang Chen, Ciya Liao, Zhao-hui Zheng, and Jon Degenhardt. Early exit optimizations for additive machine learned ranking systems. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM 2010)*, 2010.
- [37] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-SNLI: Natural language inference with natural language explanations. *Advances in Neural Information Processing Systems*, 31, 2018.

- [38] Bowen Cao, Deng Cai, Leyang Cui, Xuxin Cheng, Wei Bi, Yuexian Zou, and Shuming Shi. Retrieval is accurate generation. *arXiv*, abs/2402.17532, 2024.
- [39] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *ArXiv*, abs/1803.11175, 2018.
- [40] Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv*, abs/2302.01318, 2023.
- [41] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics.
- [42] Xilun Chen, Kushal Lakhotia, Barlas Oguz, Anchit Gupta, Patrick Lewis, Stan Peshterliev, Yashar Mehdad, Sonal Gupta, and Wen-tau Yih. Salient phrase aware dense retrieval: Can a dense retriever imitate a sparse one? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 250–262, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [43] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. March 2023.
- [44] Ke-Li Chiu and Rohan Alexander. Detecting hate speech with gpt-3. *arXiv preprint arXiv:2103.12407*, 2021.
- [45] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanu-malayan Sankaranarayanan Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta,

- Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.
- [46] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models. *CoRR*, abs/2309.03883, 2023.
- [47] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, 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, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416, 2022.
- [48] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020.
- [49] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try arc, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.
- [50] Vincent Claveau. Neural text generation for query expansion in information retrieval. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 202–209, 2021.
- [51] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [52] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. SPECTER: Document-level representation learning using citation-informed transformers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online, July 2020. Association for Computational Linguistics.
- [53] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’09, page 758–759, New York, NY, USA, 2009.

- [54] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Fernando Campos, and Ellen M. Voorhees. Overview of the trec 2020 deep learning track. *arXiv preprint arXiv:2003.07820*, 2020.
- [55] Zhuyun Dai and Jamie Callan. Context-aware term weighting for first stage passage retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1533–1536, 2020.
- [56] Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith Hall, and Ming-Wei Chang. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*, 2023.
- [57] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [58] Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. Climate-fever: A dataset for verification of real-world climate claims. *arXiv preprint arXiv:2012.00614*, 2021.
- [59] Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander H. Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander I. Rudnicky, Jason D. Williams, Joelle Pineau, Mikhail S. Burtsev, and Jason Weston. The second conversational intelligence challenge (convai2). *CoRR*, abs/1902.00098, 2019.
- [60] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of Wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [61] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv*, abs/2401.08281, 2024.
- [62] Andrew Drozdov, Shufan Wang, Razieh Rahimi, Andrew McCallum, Hamed Zamani, and Mohit Iyyer. You can’t pick your neighbors, or can you? when and how to rely on retrieval in the kNN-LM. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2997–3007, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

- [63] Avia Efrat and Omer Levy. The turking test: Can language models understand instructions? *CoRR*, abs/2010.11982, 2020.
- [64] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- [65] Fernando Ferraretto, Thiago Laitz, Roberto Lotufo, and Rodrigo Nogueira. Exaranker: Explanation-augmented neural ranker. *arXiv*, abs/2301.10521, 2023.
- [66] Adam Fisch, Tal Schuster, T. Jaakkola, and Regina Barzilay. Efficient conformal prediction via cascaded inference with expanded admission. In *Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)*, 2021.
- [67] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. Splade v2: Sparse lexical and expansion model for information retrieval. *ArXiv*, abs/2109.10086, 2021.
- [68] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [69] Peter I. Frazier. A tutorial on Bayesian optimization. *ArXiv*, abs/1807.02811, 2018.
- [70] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.
- [71] Luyu Gao and Jamie Callan. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [72] Luyu Gao, Zhuyun Dai, and Jamie Callan. COIL: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3030–3042, Online, June 2021. Association for Computational Linguistics.
- [73] Luyu Gao, Zhuyun Dai, and Jamie Callan. Rethink training of BERT rerankers in multi-stage retrieval pipeline. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part II*, page 280–286, Berlin, Heidelberg, 2021. Springer-Verlag.

- [74] Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y. Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. Rarr: Researching and revising what language models say, using language models. *arXiv*, abs/2210.08726, 2023.
- [75] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [76] Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling deep contrastive learning batch size under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 316–321, Online, August 2021. Association for Computational Linguistics.
- [77] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.
- [78] Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [79] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.
- [80] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.
- [81] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM ’16*, New York, NY, USA, 2016.
- [82] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3887–3896. PMLR, 2020.

- [83] Fredrik K Gustafsson, Martin Danelljan, and Thomas B. Schon. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 318–319, 2020.
- [84] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.
- [85] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML’20. JMLR.org, 2020.
- [86] Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. Dbpedia-entity v2: A test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’17, page 1265–1268, New York, NY, USA, 2017.
- [87] Hua He and Jimmy Lin. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 937–948, San Diego, California, June 2016. Association for Computational Linguistics.
- [88] Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. Efficient nearest neighbor language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5703–5714, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [89] Zhenyu He, Zexuan Zhong, Tianle Cai, Jason D. Lee, and Di He. Rest: Retrieval-based speculative decoding. *arXiv*, abs/2311.08252, 2024.
- [90] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. *scikit-optimize/scikit-optimize*. Zenodo, Sep 2020.
- [91] Peter Henderson, Mark Simon Krass, Lucia Zheng, Neel Guha, Christopher D Manning, Dan Jurafsky, and Daniel E. Ho. Pile of law: Learning responsible data filtering from the law and a 256GB open-source legal dataset. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.
- [92] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv*, abs/2009.03300, 2021.

- [93] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [94] Karl Moritz Hermann, Tomáš Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701, 2015.
- [95] Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.
- [96] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015.
- [97] Minh Hoang, Nghia Hoang, Bryan Kian Hsiang Low, and Carleton Kingsford. Collective model fusion for multiple black-box experts. In *International Conference on Machine Learning*, pages 2742–2750. PMLR, 2019.
- [98] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [99] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv*, abs/2203.15556, 2022.
- [100] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv:2010.02666*, 2020.
- [101] Sebastian Hofstätter, Omar Khattab, Sophia Althammer, Mete Sertkan, and Allan Hanbury. Introducing neural bag of whole-words with colberter: Contextualized late interactions using enhanced reduction. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management, CIKM ’22*, page 737–747, New York, NY, USA, 2022.
- [102] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In

Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '21, page 113–122, New York, NY, USA, 2021.

- [103] Sebastian Hofstätter, Jiecao Chen, Karthik Raman, and Hamed Zamani. Multi-task retrieval-augmented text generation with relevance sampling. *arXiv*, abs/2207.03030, 2022.
- [104] Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2023, Toronto, Canada, July 9-14, 2023, pages 14409–14428. Association for Computational Linguistics, 2023.
- [105] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. *ArXiv*, abs/1503.03244, 2014.
- [106] Xiaoyu Hu and Jing Lei. A distribution-free test of covariate shift using conformal prediction. *arXiv: Methodology*, 2020.
- [107] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.
- [108] Shudong Huang Ian Soboroff and Donna Harman. Trec 2019 news track overview. pages 4,18, 2019.
- [109] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *ArXiv*, abs/2212.04089, 2022.
- [110] Ayyoob Imani, Amir Vakili, Ali Montazer, and Azadeh Shakery. Deep neural networks for query expansion using word embeddings. In *Advances in Information Retrieval: 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14–18, 2019, Proceedings, Part II 41*, pages 203–210. Springer, 2019.
- [111] Robert Iv, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. FRUIT: Faithfully reflecting updated information in text. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686, Seattle, United States, July 2022. Association for Computational Linguistics.

- [112] Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, Xian Li, Brian O’Horo, Gabriel Pereyra, Jeff Wang, Christopher Dewan, Asli Celikyilmaz, Luke Zettlemoyer, and Ves Stoyanov. OPT-IML: scaling language model instruction meta learning through the lens of generalization. *CoRR*, abs/2212.12017, 2022.
- [113] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*, 2022.
- [114] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online, April 2021. Association for Computational Linguistics.
- [115] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24(1), mar 2024.
- [116] Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Sebastian Riedel, and Edouard Grave. A memory efficient baseline for open domain question answering. *ArXiv*, abs/2012.15156, 2020.
- [117] Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. Query expansion by prompting large language models. *arXiv preprint arXiv:2305.03653*, 2023.
- [118] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.
- [119] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.
- [120] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:117–128, 2011.
- [121] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021.
- [122] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics.

- [123] Herve Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.
- [124] Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. Evaluating open-domain question answering in the era of large language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [125] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning, ICML’23*. JMLR.org, 2023.
- [126] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, November 2020. Association for Computational Linguistics.
- [127] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through Memorization: Nearest Neighbor Language Models. In *International Conference on Learning Representations (ICLR)*, 2020.
- [128] Omar Khattab and Matei A. Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.
- [129] Hyunwoo Kim, Youngjae Yu, Liwei Jiang, Ximing Lu, Daniel Khashabi, Gunhee Kim, Yejin Choi, and Maarten Sap. Prosocialdialog: A prosocial backbone for conversational agents. In *EMNLP*, 2022.
- [130] Yoon Kim. Convolutional neural networks for sentence classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.
- [131] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015*, 2015.
- [132] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

- [133] Mojtaba Komeili, Kurt Shuster, and Jason Weston. Internet-augmented dialogue generation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, *ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8460–8478, 2022.
- [134] Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations—democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*, 2023.
- [135] Brian Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2012.
- [136] Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. In *ICML*, 2012.
- [137] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1378–1387, New York, New York, USA, 20–22 Jun 2016. PMLR.
- [138] 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. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.
- [139] 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. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 08 2019.
- [140] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 6402–6413, 2017.

- [141] Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. QED: A framework and dataset for explanations in question answering. *Transactions of the Association for Computational Linguistics*, 9:790–806, 2021.
- [142] Tian Lan, Deng Cai, Yan Wang, Heyan Huang, and Xian-Ling Mao. Copy is all you need. In *The Eleventh International Conference on Learning Representations*, 2023.
- [143] Wuwei Lan and Wei Xu. Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. In Emily M. Bender, Leon Derczynski, and Pierre Isabelle, editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3890–3902, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.
- [144] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Association for Computing Machinery, SIGIR '01*, page 120–127, New York, NY, USA, 2001.
- [145] Teven Le Scao, Thomas Wang, Daniel Hesslow, Stas Bekman, M Saiful Bari, Stella Biderman, Hady Elsahar, Niklas Muennighoff, Jason Phang, Ofir Press, Colin Raffel, Victor Sanh, Sheng Shen, Lintang Sutawika, Jaesung Tae, Zheng Xin Yong, Julien Launay, and Iz Beltagy. What language model to train if you have one million GPU hours? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 765–782, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [146] Jinhyuk Lee, Zhuyun Dai, Sai Meher Karthik Duddu, Tao Lei, Iftekhhar Naim, Ming-Wei Chang, and Vincent Y. Zhao. Rethinking the role of token retrieval in multi-vector retrieval. *ArXiv*, abs/2304.01982, 2023.
- [147] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy, July 2019. Association for Computational Linguistics.
- [148] Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale N Fung, Mohammad Shoeybi, and Bryan Catanzaro. Factuality enhanced language models for open-ended text generation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 34586–34599. Curran Associates, Inc., 2022.
- [149] Seokju Lee, Sunghoon Im, Stephen Lin, and In So Kweon. Learning monocular depth in dynamic scenes via instance-aware projection consistency. In *AAAI*, 2021.
- [150] Lihua Lei and Emmanuel J. Candès. Conformal inference of counterfactuals and individual treatment effects. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2021.

- [151] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR, 23–29 Jul 2023.
- [152] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.
- [153] Patrick Lewis, Barlas Ögüz, Wenhan Xiong, Fabio Petroni, Wen-tau Yih, and Sebastian Riedel. Boosted dense retriever. *arXiv preprint arXiv:2112.07771*, 2021.
- [154] 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. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS ’20, Red Hook, NY, USA, 2020. Curran Associates Inc.
- [155] 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. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401, 2021.
- [156] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.
- [157] Aitor Lewkowycz, Anders Johan Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Venkatesh Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.
- [158] Hang Li, Ahmed Mourad, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. Pseudo relevance feedback with deep language models and dense retrievers: Successes and pitfalls. *Association for Computing Machinery*, 41(3), apr 2023.

- [159] Minghan Li, Ming Li, Kun Xiong, and Jimmy Lin. Multi-task dense retrieval via model uncertainty fusion for open-domain question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 274–287, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [160] Minghan Li, Sheng-Chieh Lin, Xueguang Ma, and Jimmy Lin. Slim: Sparsified late interaction for multi-vector retrieval with inverted indexes. *ArXiv*, abs/2302.06587, 2023.
- [161] Minghan Li, Sheng-Chieh Lin, Barlas Oguz, Asish Ghoshal, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. CITADEL: Conditional token interaction via dynamic lexical routing for efficient and effective multi-vector retrieval. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11891–11907, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [162] Minghan Li, Xinyu Zhang, Ji Xin, Hongyang Zhang, and Jimmy Lin. Certified error control of candidate set pruning for two-stage relevance ranking. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 333–345, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [163] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.
- [164] Jimmy Lin. The neural hype and comparisons against weak baselines. In *ACM SIGIR Forum*, volume 52, pages 40–51. ACM New York, NY, USA, 2019.
- [165] Jimmy Lin. A proposed conceptual framework for a representational approach to information retrieval. *ArXiv*, abs/2110.01529, 2021.
- [166] Jimmy Lin and Xueguang Ma. A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. *arXiv preprint arXiv:2106.14807*, 2021.
- [167] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai, editors, *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2356–2362. ACM, 2021.
- [168] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. *Pretrained transformers for text ranking: BERT and beyond*. Springer Nature, 2022.

- [169] Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. How to train your DRAGON: Diverse augmentation towards generalizable dense retrieval. *arXiv preprint arXiv:2302.07452*, 2023.
- [170] Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6385–6400, Singapore, December 2023. Association for Computational Linguistics.
- [171] Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen tau Yih, and Xilun Chen. Flame: Factuality-aware alignment for large language models. *arXiv*, abs/2405.01525, 2024.
- [172] Sheng-Chieh Lin, Minghan Li, and Jimmy Lin. Aggretriever: A simple approach to aggregate textual representations for robust dense passage retrieval. *Transactions of the Association for Computational Linguistics*, 11:436–452, 2023.
- [173] Sheng-Chieh Lin and Jimmy Lin. A dense representation framework for lexical and semantic matching. *Association for Computing Machinery*, 41(4), apr 2023.
- [174] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. Distilling dense representations for ranking using tightly-coupled teachers. *arXiv:2010.11386*, 2020.
- [175] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 163–173, Online, August 2021. Association for Computational Linguistics.
- [176] Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. Multi-stage conversational passage retrieval: An approach to fusing term importance estimation and neural query rewriting. *ACM Transactions on Information Systems (TOIS)*, 39(4):1–29, 2021.
- [177] Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [178] Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Richard James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. RA-DIT: Retrieval-augmented dual instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024.

- [179] Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O’Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona T. Diab, Veselin Stoyanov, and Xian Li. Few-shot learning with multilingual language models. *CoRR*, abs/2112.10668, 2021.
- [180] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *ACL*, 2017.
- [181] Jerry Liu. LlamaIndex. 11 2022.
- [182] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021.
- [183] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online, July 2020.
- [184] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.
- [185] Zheng Liu and Yingxia Shao. Retromae: Pre-training retrieval-oriented transformers via masked auto-encoder. *arXiv:2205.12035*, 2022.
- [186] Antonio Loquercio, Mattia Segu, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 5(2):3153–3160, 2020.
- [187] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018.
- [188] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net, 2019.
- [189] Xiaolu Lu, Oren Kurland, J. Shane Culpepper, Nick Craswell, and Ofri Rom. Relevance modeling with multiple query variations. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR ’19*, page 27–34, New York, NY, USA, 2019.
- [190] Zhengdong Lu and Hang Li. A deep architecture for matching short texts. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

- [191] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345, 2021.
- [192] Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, R. Perego, and Salvatore Trani. Query-level early exit for additive learning-to-rank ensembles. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.
- [193] Hongyin Luo, Yung-Sung Chuang, Yuan Gong, Tianhua Zhang, Yoon Kim, Xixin Wu, Danny Fox, Helen Meng, and James R. Glass. SAIL: search-augmented instruction learning. *CoRR*, abs/2305.15225, 2023.
- [194] Yuanhua Lv and ChengXiang Zhai. A comparative study of methods for estimating query language models with pseudo feedback. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1895–1898, 2009.
- [195] Xueguang Ma, Kai Sun, Ronak Pradeep, Minghan Li, and Jimmy Lin. Another look at DPR: Reproduction of training and replication of retrieval. In *Proceedings of the 44th European Conference on Information Retrieval (ECIR 2022), Part I*, pages 613–626, Stavanger, Norway, 2022.
- [196] Xueguang Ma, Kai Sun, Ronak Pradeep, and Jimmy Lin. A replication study of dense passage retriever. *ArXiv*, abs/2104.05740, 2021.
- [197] Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. Fine-tuning llama for multi-stage text retrieval. *arXiv preprint arXiv:2310.08319*, 2023.
- [198] Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*, 2023.
- [199] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR’19, page 1101–1104, New York, NY, USA, 2019.
- [200] Craig Macdonald, Nicola Tonellotto, Sean MacAvaney, and Iadh Ounis. Pyterrier: Declarative experimentation in python from bm25 to dense retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM ’21, page 4526–4533, New York, NY, USA, 2021.
- [201] David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

- [202] Joel Mackenzie, Andrew Trotman, and Jimmy Lin. Wacky weights in learned sparse representations and the revenge of score-at-a-time query evaluation. *ArXiv*, abs/2110.11540, 2021.
- [203] Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. Www’18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW ’18, page 1941–1942, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.
- [204] Jean Maillard, Vladimir Karpukhin, Fabio Petroni, Wen-tau Yih, Barlas Oguz, Veselin Stoyanov, and Gargi Ghosh. 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 (Volume 1: Long Papers)*, pages 1098–1111, Online, August 2021. Association for Computational Linguistics.
- [205] Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. Expertqa: Expert-curated questions and attributed answers. *arXiv*, abs/2309.07852, 2024.
- [206] Andrey Malinin and Mark J. F. Gales. Predictive uncertainty estimation via prior networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pages 7047–7058, 2018.
- [207] Yu. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *arXiv*, abs/1603.09320, 2018.
- [208] Antonio Mallia, Omar Khattab, Nicola Tonellotto, and Torsten Suel. Learning passage impacts for inverted indexes. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [209] Antonio Mallia, Michal Siedlaczek, Joel Mackenzie, and Torsten Suel. PISA: performant indexes and search for academia. In *Proceedings of the Open-Source IR Replicability Challenge co-located with 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, OSIRRC@SIGIR 2019, Paris, France, July 25, 2019.*, pages 50–56, 2019.
- [210] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

- [211] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*, 2020.
- [212] Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730, 2018.
- [213] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.
- [214] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.
- [215] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Rae Ying Yee Wong, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating generative llm serving with speculative inference and token tree verification. *ArXiv*, abs/2305.09781, 2023.
- [216] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *International Conference on Learning Representations*, 2018.
- [217] Katharine Miller. How Do We Fix and Update Large Language Models? *Stanford University Human-Centered Artificial Intelligence*, 2023.
- [218] Sewon Min, Jordan L. Boyd-Graber, C. Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, T. Kwiatkowski, Patrick Lewis, Yuxiang Wu, Heinrich Kuttler, L. Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Edouard Grave, Ikuya Yamada, Sonse Shi-maoka, Masatoshi Suzuki, Shumpei Miyawaki, S. Sato, Ryo Takahashi, J. Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, P. Smrz, Hao Cheng, Y. Shen, X. Liu, Pengcheng He, W. Chen, Jianfeng Gao, Barlas Oğuz, Xilun Chen, V. Karpukhin, Stanislav Peshter-liev, Dmytro Okhonko, M. Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen tau Yih. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. *ArXiv*, 2101.00133, 2021.
- [219] Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China, November 2019. Association for Computational Linguistics.

- [220] Sewon Min, Suchin Gururangan, Eric Wallace, Hannaneh Hajishirzi, Noah A. Smith, and Luke Zettlemoyer. Silo language models: Isolating legal risk in a nonparametric datastore. *arXiv*, abs/2308.04430, 2023.
- [221] Sewon Min, Kalpesh Krishna, Xinxu Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore, December 2023. Association for Computational Linguistics.
- [222] Ioannis Mollas, Zoe Chrysopoulou, Stamatis Karlos, and Grigorios Tsoumakas. Ethos: an online hate speech detection dataset. *arXiv preprint arXiv:2006.08328*, 2020.
- [223] Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. Multimodal contrastive learning with limoe: the language-image mixture of experts. *ArXiv*, abs/2206.02770, 2022.
- [224] Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel Bowman. Crows-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, 2020.
- [225] Radford M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.
- [226] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*, 2016.
- [227] 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. Large dual encoders are generalizable retrievers. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [228] Rodrigo Nogueira and Kyunghyun Cho. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583, 2017.
- [229] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. In Trevor Cohn, Yulan He, and Yang Liu, editors,

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 708–718, Online, November 2020. Association for Computational Linguistics.

- [230] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. Multi-stage document ranking with BERT. *ArXiv*, abs/1910.14424, 2019.
- [231] Cicero Nogueira dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. Beyond [CLS] through ranking by generation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1722–1727, Online, November 2020. Association for Computational Linguistics.
- [232] Natasha Noy, Matthew Burgess, and Dan Brickley. Google dataset search: Building a search engine for datasets in an open web ecosystem. In *28th Web Conference (WebConf 2019)*, 2019.
- [233] Alexandros Ntoulas and Junghoo Cho. Pruning policies for two-tiered inverted index with correctness guarantee. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’07, page 191–198, New York, NY, USA, 2007.
- [234] Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.
- [235] Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. Instructir: A benchmark for instruction following of information retrieval models. *arXiv*, abs/2402.14334, 2024.
- [236] Yasumasa Onoe, Michael JQ Zhang, Eunsol Choi, and Greg Durrett. CREAK: A dataset for commonsense reasoning over entity knowledge. *arXiv preprint arXiv:2109.01653*, 2021.
- [237] OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [238] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.
- [239] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In Gerardo

- Flores, George H Chen, Tom Pollard, Joyce C Ho, and Tristan Naumann, editors, *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR, 07–08 Apr 2022.
- [240] Liang Pang, Yanyan Lan, J. Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. Text matching as image recognition. *ArXiv*, abs/1602.06359, 2016.
 - [241] Sangdon Park, Osbert Bastani, Nikolai Matni, and Insup Lee. PAC confidence sets for deep neural networks via calibrated prediction. *ArXiv*, abs/2001.00106, 2020.
 - [242] Sangdon Park, Osbert Bastani, James Weimer, and Insup Lee. Calibrated prediction with covariate shift via unsupervised domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, pages 3219–3229. PMLR, 2020.
 - [243] Tim Pearce, Felix Leibfried, and Alexandra Brintrup. Uncertainty in neural networks: Approximately Bayesian ensembling. In *International Conference on Artificial Intelligence and Statistics*, pages 234–244. PMLR, 2020.
 - [244] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.
 - [245] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Large Margin Classif.*, 10, 06 2000.
 - [246] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.
 - [247] Ronak Pradeep, Yuqi Liu, Xinyu Zhang, Yilin Li, Andrew Yates, and Jimmy Lin. Squeezing water from a stone: A bag of tricks for further improving cross-encoder effectiveness for reranking. In *Advances in Information Retrieval*, pages 655–670, Cham, 2022. Springer International Publishing.
 - [248] Ronak Pradeep, Rodrigo Nogueira, and Jimmy J. Lin. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *ArXiv*, abs/2101.05667, 2021.
 - [249] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. Rankvicuna: Zero-shot listwise document reranking with open-source large language models. *arXiv preprint arXiv:2309.15088*, 2023.

- [250] Yujie Qian, Jinhyuk Lee, Sai Meher Karthik Duddu, Zhuyun Dai, Siddhartha Brahma, Iftexhar Naim, Tao Lei, and Vincent Y Zhao. Multi-vector retrieval as sparse alignment. *arXiv preprint arXiv:2211.01267*, 2022.
- [251] Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024.
- [252] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proc. NAACL*, pages 5835–5847, June 2021.
- [253] Alec Radford, Jong Wook Kim, and Jeff Wu. Gpt-2 output dataset. 2021. Last Updated: 02-17-21.
- [254] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [255] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.
- [256] Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, 2019.
- [257] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don’t know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics.
- [258] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.
- [259] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.

- [260] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.
- [261] Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. Measuring Attribution in Natural Language Generation Models. *Computational Linguistics*, 49(4):777–840, 12 2023.
- [262] Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabisa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *The Eleventh International Conference on Learning Representations*, 2023.
- [263] Siva Reddy, Danqi Chen, and Christopher D Manning. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019.
- [264] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [265] Stephen Robertson. On term selection for query expansion. *Journal of Documentation*, 46:359–364, 12 1990.
- [266] Stephen E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146, 1976.
- [267] Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.
- [268] Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. Getting closer to AI complete question answering: A set of prerequisite real tasks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8722–8731. AAAI Press, 2020.
- [269] Guilherme Moraes Rosa, Luiz Bonifacio, Vitor Jeronymo, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, and Rodrigo Nogueira. No parameter left behind: How distillation and model size affect zero-shot retrieval. *ArXiv*, arXiv preprint arXiv:2206.02873, 2022.

- [270] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. Using word embeddings for automatic query expansion. *arXiv preprint arXiv:1606.07608*, 2016.
- [271] Christopher Rytting and David Wingate. Leveraging the inductive bias of large language models for abstract textual reasoning. *Advances in Neural Information Processing Systems*, 34:17111–17122, 2021.
- [272] Devendra Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3781–3797, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [273] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.
- [274] Gerard Salton. *The SMART retrieval system—experiments in automatic document processing*. Prentice-Hall, Inc., 1971.
- [275] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523, 1988.
- [276] Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. Plaid: An efficient engine for late interaction retrieval. *ArXiv*, abs/2205.09707, 2022.
- [277] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 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*, Seattle, United States, July 2022. Association for Computational Linguistics.
- [278] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [279] Timo Schick, Jane Dwivedi-Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. Peer: A collaborative language model. *arXiv*, abs/2208.11663, 2022.
- [280] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to Information Retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

- [281] Thomas Scialom, Tuhin Chakrabarty, and Smaranda Muresan. Continual-t0: Progressively instructing 50+ tasks to language models without forgetting. *CoRR*, abs/2205.12393, 2022.
- [282] Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [283] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.
- [284] Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441, Florence, Italy, July 2019. Association for Computational Linguistics.
- [285] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarczyk, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.
- [286] Tao Shen, Xiubo Geng, Chongyang Tao, Can Xu, Kai Zhang, and Daxin Jiang. Unifier: A unified retriever for large-scale retrieval. *arXiv*, abs/2205.11194, 2022.
- [287] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR, 23–29 Jul 2023.
- [288] Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, June 2024.
- [289] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, June 2024.
- [290] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

- [291] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.
- [292] Kurt Shuster, Jack Urbanek, Emily Dinan, Arthur Szlam, and Jason Weston. Dialogue in the wild: Learning from a deployed role-playing game with humans and bots. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 611–624, Online, August 2021. Association for Computational Linguistics.
- [293] Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, et al. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022.
- [294] Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. Can you put it all together: Evaluating conversational agents’ ability to blend skills. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2021–2030, 2020.
- [295] Jasper Snoek, Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D. Sculley, Joshua V. Dillon, Jie Ren, and Zachary Nado. Can you trust your model’s uncertainty? Evaluating predictive uncertainty under dataset shift. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 13969–13980, 2019.
- [296] Richard Socher, Eric Hsin-Chun Huang, Jeffrey Pennington, A. Ng, and Christopher D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Neural Information Processing Systems*, 2011.
- [297] Luca Soldaini and Alessandro Moschitti. The cascade transformer: an application for efficient answer sentence selection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5697–5708, Online, July 2020.
- [298] Benjamin Spector and Chris Re. Accelerating llm inference with staged speculative decoding. *arXiv*, abs/2308.04623, 2023.
- [299] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

- [300] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.
- [301] Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. One embedder, any task: Instruction-finetuned text embeddings. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada, July 2023. Association for Computational Linguistics.
- [302] Axel Suarez, Dyaa Albakour, David Corney, Miguel Martinez, and José Esquivel. A data collection for evaluating the retrieval of related tweets to news articles. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40*, pages 780–786. Springer, 2018.
- [303] Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. Head-to-tail: How knowledgeable are large language models (llm)? a.k.a. will llms replace knowledge graphs? *arXiv*, abs/2308.10168, 2023.
- [304] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? investigating large language models as re-ranking agents. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore, December 2023. Association for Computational Linguistics.
- [305] Zhiqing Sun, Xuezhi Wang, Yi Tay, Yiming Yang, and Denny Zhou. Recitation-augmented language models. In *The Eleventh International Conference on Learning Representations*, 2023.
- [306] Oyvind Tafjord, Peter Clark, Matt Gardner, Wen-tau Yih, and Ashish Sabharwal. QUAREL: A dataset and models for answering questions about qualitative relationships. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7063–7071. AAAI Press, 2019.
- [307] Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. *arXiv preprint arXiv:2012.13048*, 2020.
- [308] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy

- Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics, 2019.
- [309] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. *GitHub repository*, 2023.
 - [310] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.
 - [311] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.
 - [312] Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D Manning, and Chelsea Finn. Fine-tuning language models for factuality. In *The Twelfth International Conference on Learning Representations*, 2024.
 - [313] Kushal Tirumala, Aram H. Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In *NeurIPS*, 2022.
 - [314] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
 - [315] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith,

- Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kam-badur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *arXiv*, abs/2307.09288, 2023.
- [316] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):1–28, 2015.
- [317] Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 673–683, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [318] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2019.
- [319] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [320] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.
- [321] Deepak Vohra and Deepak Vohra. Using Elasticsearch. *Pro Couchbase Development: A NoSQL Platform for the Enterprise*, pages 175–196, 2015.
- [322] Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. Trec-covid: Constructing a pandemic information retrieval test collection. *arXiv preprint arXiv:2005.04474*, 2020.

- [323] Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, page 61–69, Berlin, Heidelberg, 1994. Springer-Verlag.
- [324] Ellen M. Voorhees and Angela Ellis, editors. *Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC 2019, Gaithersburg, Maryland, USA, November 13-15, 2019*, volume 1250 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2019.
- [325] Ellen M. Voorhees et al. Overview of the trec 2003 robust retrieval track. In *Trec*, pages 69–77, 2003.
- [326] Ellen M. Voorhees and Dawn M. Tice. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece, May 2000. European Language Resources Association (ELRA).
- [327] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005.
- [328] Vladimir Vovk, Alexander Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999)*, 1999.
- [329] Vladimir Vovk, Ivan Petej, Paolo Toccaceli, and Alexander Gammerman. Conformal calibrators. In *COPA*, 2020.
- [330] Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. Exploring and predicting transferability across nlp tasks. In *EMNLP*, 2020.
- [331] Henning Wachsmuth, Shahbaz Syed, and Benno Stein. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251, Melbourne, Australia, July 2018. Association for Computational Linguistics.
- [332] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online, November 2020. Association for Computational Linguistics.
- [333] Abraham Wald. An extension of Wilks' method for setting tolerance limits. *Annals of Mathematical Statistics*, 14:45–55, 1943.

- [334] David Wan and Mohit Bansal. FactPEGASUS: Factuality-aware pre-training and fine-tuning for abstractive summarization. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1010–1028, Seattle, United States, July 2022. Association for Computational Linguistics.
- [335] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Tal Linzen, Grzegorz Chrupała, and Afra Alishahi, editors, *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics.
- [336] Boxin Wang, Wei Ping, Lawrence McAfee, Peng Xu, Bo Li, Mohammad Shoeybi, and Bryan Catanzaro. Instructretro: Instruction tuning post retrieval-augmented pretraining. *arXiv*, abs/2310.07713, 2023.
- [337] Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, Anima Anandkumar, and Bryan Catanzaro. Shall we pretrain autoregressive language models with retrieval? a comprehensive study. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7763–7786, Singapore, December 2023. Association for Computational Linguistics.
- [338] Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. Does it make sense? and why? a pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, 2019.
- [339] Junjie Wang, Mingyang Chen, Binbin Hu, Dan Yang, Ziqi Liu, Yue Shen, Peng Wei, Zhiqiang Zhang, Jinjie Gu, Jun Zhou, Jeff Z. Pan, Wen Zhang, and Huajun Chen. Learning to plan for retrieval-augmented large language models from knowledge graphs. *arXiv*, abs/2406.14282, 2024.
- [340] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. SIMLM: Pre-training with representation bottleneck for dense passage retrieval. *arXiv preprint arXiv:2207.02578*, 2022.
- [341] Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore, December 2023. Association for Computational Linguistics.

- [342] Lidan Wang, Jimmy Lin, and Donald Metzler. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th International ACM SIGIR Conference on Research and development in Information Retrieval*, pages 105–114, 2011.
- [343] Shufan Wang, Yixiao Song, Andrew Drozdov, Aparna Garimella, Varun Manjunatha, and Mohit Iyyer. *k*NN-LM does not improve open-ended text generation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15023–15037, Singapore, December 2023. Association for Computational Linguistics.
- [344] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS’20, Red Hook, NY, USA, 2020. Curran Associates Inc.
- [345] Xiao Wang, Sean MacAvaney, Craig Macdonald, and Iadh Ounis. Generative query reformulation for effective adhoc search. *arXiv preprint arXiv:2308.00415*, 2023.
- [346] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.
- [347] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13484–13508. Association for Computational Linguistics, 2023.
- [348] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krима Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. Super-naturalinstructions: Generalization via declarative instructions on 1600+ NLP tasks. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 5085–5109. Association for Computational Linguistics, 2022.

- [349] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5878–5882, Hong Kong, China, 2019. Association for Computational Linguistics.
- [350] Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. Retrieval, re-ranking and multi-task learning for knowledge-base question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 347–357, 2021.
- [351] Ian Waudby-Smith and Aaditya Ramdas. Variance-adaptive confidence sequences by betting. *ArXiv*, abs/2010.09686, 2020.
- [352] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.
- [353] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.
- [354] David J. Weiss and Ben Taskar. Structured prediction cascades. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, 2010.
- [355] Orion Weller, Benjamin Chang, Sean MacAvaney, Kyle Lo, Arman Cohan, Benjamin Van Durme, Dawn Lawrie, and Luca Soldaini. Followir: Evaluating and teaching information retrieval models to follow instructions. *arXiv*, abs/2403.15246, 2024.
- [356] Orion Weller, Kyle Lo, David Wadden, Dawn Lawrie, Benjamin Van Durme, Arman Cohan, and Luca Soldaini. When do generative query and document expansions fail? a comprehensive study across methods, retrievers, and datasets. In Yvette Graham and Matthew Purver, editors, *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1987–2003, St. Julian’s, Malta, March 2024. Association for Computational Linguistics.
- [357] Yeming Wen, Dustin Tran, and Jimmy Ba. BatchEnsemble: An alternative approach to efficient ensemble and lifelong learning. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

- [358] Peter West, Chandra Bhagavatula, Jack Hessel, Jena Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. Symbolic knowledge distillation: from general language models to commonsense models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4602–4625, Seattle, United States, July 2022. Association for Computational Linguistics.
- [359] Samuel Stanley Wilks. Determination of sample sizes for setting tolerance limits. *Annals of Mathematical Statistics*, 12:91–96, 1941.
- [360] Samuel Stanley Wilks. Statistical prediction with special reference to the problem of tolerance limits. *Annals of Mathematical Statistics*, 13:400–409, 1942.
- [361] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020.
- [362] Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. UnifiedSKG: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 602–631, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [363] Ji Xin, Rodrigo Nogueira, Yaoliang Yu, and Jimmy Lin. Early exiting BERT for efficient document ranking. In *Proceedings of SustainLP: Workshop on Simple and Efficient Natural Language Processing*, pages 83–88, Online, November 2020.
- [364] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online, July 2020. Association for Computational Linguistics.
- [365] Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. BERxiT: Early exiting for BERT with better fine-tuning and extension to regression. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 91–104, Online, April 2021.

- [366] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 55–64, New York, NY, USA, 2017.
- [367] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)*, 2021.
- [368] Wenhan Xiong, Hong Wang, and William Yang Wang. Progressively pretrained dense corpus index for open-domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2803–2815, Online, April 2021. Association for Computational Linguistics.
- [369] Hu Xu, Bing Liu, Lei Shu, and Philip Yu. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [370] Jing Xu, Arthur Szlam, and Jason Weston. Beyond goldfish memory: Long-term open-domain conversation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2022, Dublin, Ireland, May 22–27, 2022, pages 5180–5197, 2022.
- [371] Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. Efficient passage retrieval with hashing for open-domain question answering. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 979–986, Online, August 2021. Association for Computational Linguistics.
- [372] Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. 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 (Volume 2: Short Papers)*, pages 979–986, 2021.
- [373] Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Enabling the use of Lucene for information retrieval research. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries,

and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256. ACM, 2017.

- [374] Wei Yang, Kuang Lu, Peilin Yang, and Jimmy Lin. Critically examining the "neural hype" weak baselines and the additivity of effectiveness gains from neural ranking models. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 1129–1132, 2019.
- [375] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [376] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October–November 2018. Association for Computational Linguistics.
- [377] Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Hua-jun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore, December 2023. Association for Computational Linguistics.
- [378] Andrew Yates, Siddhant Arora, Xinyu Zhang, Wei Yang, Kevin Martin Jose, and Jimmy Lin. Capreolus: A toolkit for end-to-end neural ad hoc retrieval. In *Proceedings of the 13th International Conference on Web Search and Data Mining, WSDM '20*, page 861–864, New York, NY, USA, 2020.
- [379] Andrew Yates, Siddhant Arora, Xinyu Zhang, Wei Yang, Kevin Martin Jose, and Jimmy Lin. Capreolus: A toolkit for end-to-end neural ad hoc retrieval. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining (WSDM 2020)*, pages 861–864, Houston, Texas, 2020.
- [380] Andrew Yates, Kevin Martin Jose, Xinyu Zhang, and Jimmy Lin. Flexible ir pipelines with capreolus. In *Proceedings of the 29th ACM International Conference on Information Knowledge Management, CIKM '20*, page 3181–3188, New York, NY, USA, 2020.

- [381] Emine Yilmaz and Stephen Robertson. Deep versus shallow judgments in learning to rank. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 662–663, 2009.
- [382] Wenpeng Yin, Jia Li, and Caiming Xiong. Contintin: Continual learning from task instructions. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3062–3072, 2022.
- [383] Puxuan Yu, Daniel Cohen, Hemank Lamba, Joel Tetreault, and Alex Jaimes. Explain then rank: Scale calibration of neural rankers using natural language explanations from large language models. *arXiv*, abs/2402.12276, 2024.
- [384] Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhui Chen. MAMmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024.
- [385] Salah Zaiem and Fatiha Sadat. Sequence to sequence learning for query expansion. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 10075–10076, 2019.
- [386] Jingtao Zhan, Jiabin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1503–1512, 2021.
- [387] Jingtao Zhan, Jiabin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. RepBERT: Contextualized text embeddings for first-stage retrieval. *arXiv:2006.15498*, 2020.
- [388] Hongming Zhang, Xinran Zhao, and Yangqiu Song. Winowhy: A deep diagnosis of essential commonsense knowledge for answering winograd schema challenge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5736–5745, 2020.
- [389] Xinyu Zhang, Minghan Li, and Jimmy Lin. Improving out-of-distribution generalization of neural rerankers with contextualized late interaction. *arXiv preprint arXiv:2302.06589*, 2023.
- [390] Yinan Zhang, Raphael Tang, and Jimmy Lin. Explicit pairwise word interaction modeling improves pretrained transformers for english semantic similarity tasks. *ArXiv*, abs/1911.02847, 2019.

- [391] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. Contextualized query expansion via unsupervised chunk selection for text retrieval. *Information Processing & Management*, 58(5):102672, 2021.
- [392] Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. Lima: Less is more for alignment. *arXiv*, abs/2305.11206, 2023.
- [393] Honglei Zhuang, Zhen Qin, Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. Ensemble distillation for bert-based ranking models. In *Proceedings of the 2021 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '21)*, 2021.
- [394] Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan, Xuanhui Wang, and Michael Bendersky. Beyond yes and no: Improving zero-shot pointwise llm rankers via scoring fine-grained relevance labels. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2024.
- [395] Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Association for Computing Machinery, SIGIR '23*, page 2308–2313, New York, NY, USA, 2023.