# Old IR Methods Meet RAG

Oz Huly*
ozhuly@campus.technion.ac.il
Technion
Haifa, Israel

Idan Pogrebinsky*
idan.pog@campus.technion.ac.il
Technion
Haifa, Israel

David Carmel
david.carmel@technion.ac.il
Technion and Technology Innovation Institute
Haifa, Israel

Oren Kurland
kurland@technion.ac.il
Technion
Haifa, Israel

Yoelle Maarek
yoelle@yahoo.com
Technology Innovation Institute
Haifa, Israel

## ABSTRACT

Retrieval augmented generation (RAG) is an important approach to provide large language models (LLMs) with context pertaining to the text generation task: given a prompt, passages are retrieved from external corpora to ground the generation with more relevant and/or fresher data. Most previous studies used dense retrieval methods for applying RAG in question answering scenarios. However, recent work showed that traditional information retrieval methods (a.k.a. sparse methods) can do as well as or even better than dense retrieval ones. In particular, it was shown that Okapi BM25 outperforms dense retrieval methods, in terms of perplexity, for the fundamental text completion task in LLMs. We extend this study and show, using two popular LLMs, that a broad set of sparse retrieval methods achieve better results than all the dense retrieval methods we experimented with, for varying lengths of queries induced from the prompt. Furthermore, we found that Okapi BM25 is substantially outperformed by a term-proximity retrieval method (MRF), which is in turn outperformed by a pseudo-feedback-based bag-of-terms approach (relevance model). Additional exploration sheds some light on the effectiveness of lexical retrieval methods for RAG. Our findings call for further study of classical retrieval methods for RAG.

## CCS CONCEPTS

• **Information systems** → **Information retrieval**; **Retrieval models and ranking**;

## KEYWORDS

retrieval augmented generation

**ACM Reference Format:**
Oz Huly, Idan Pogrebinsky, David Carmel, Oren Kurland, and Yoelle Maarek. 2024. Old IR Methods Meet RAG. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3626772.3657935

*Both authors contributed equally to the paper.



*SIGIR '24, July 14–18, 2024, Washington, DC, USA*

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657935

## 1 INTRODUCTION

Large language models (LLM) are trained over massively large datasets to generate content and other digital artifacts [28]. LLM-based text generation has become the leading technology for many NLP tasks such as question answering, summarization, multi-lingual translation, story generation, and many more [2].

However, the content generated by LLMs may suffer from a variety of semantic problems [4]. The most widely known is erroneous responses, commonly called "hallucinations" [23, 27], as they are expressed as if they were actual facts. Retrieval-Augmented Generation (RAG) [1, 7, 9, 12, 15, 20] is a widely accepted methodology that helps reduce hallucinations in the generated content [23]. RAG improves the quality of the LLM response by grounding the generation process with relevant, reliable, and often fresh evidence provided by additional sources. Examples of such sources include a general knowledge-base (e.g., Wikipedia), a domain specific dataset (e.g., enterprise internal data), or even Web data. The input to the LLM (the prompt) is used to query the external sources, and retrieved results, augmented with the input, are fed to the LLM. RAG-based LLMs (RAG-LLMs) have been shown to suffer significantly less from hallucinations than regular LLMs [8, 15, 23].

RAG-LLM opens many new retrieval challenges for the IR research community, such as optimal query generation, external source selection, ranking, diversification, and many more. In this work, we focus on analyzing the effectiveness of existing retrieval methods for RAG-LLMs. In contrast to the classical IR approach of providing the most relevant results to the user query [22], we look for documents that assist the LLM with its generation task. Many retrieval approaches, in particular for knowledge intensive tasks such as question answering, are based on dense retrieval (a.k.a. semantic search), which has been proved to perform well for many NLP tasks [9]. However, Ram et al. [20] demonstrated that for text completion, Okapi BM25 [21], a sparse retrieval method, consistently and substantively outperforms dense retrieval methods. This led us to the question whether other sparse retrieval methods can do even better for RAG-LLMs.

In this work we examine retrieval methods for RAG-LLMs for text completion, comparing not only between dense and sparse methods, but also between various sparse methods. More specifically, we experiment with the In-context RAG-LLM framework [20], which provides a testbed for measuring text generation quality through perplexity, as a function of the retrieved content that is used to enrich the input. We consider a variety of traditional (sparse)

retrieval methods, including Okapi BM25 [21], query likelihood [25], pseudo relevance feedback (RM3) [14] and Markov Random Fields (MRF) [18]. We compare them to three dense retrieval methods: DPR [11], Contriever [8], and its fine-tuned version, Contriever$^{ft}$, which was further tuned on the MSMARCO dataset [19]. The key contributions of our work are the following:

- We confirm and extend the results reported by Ram et al. [20] that sparse retrieval is more effective than dense retrieval for the text completion task, by providing a systematic analysis of various retrieval methods with various query lengths and two LLMs.
- We demonstrate that BM25, the most common representative of sparse methods for RAG-LLM, is inferior by far to other sparse methods: MRF and RM3.
- We present an additional study that sheds light on the importance of simple lexical match between the prompt and the passage retrieved to augment it.

## 2 RELATED WORK

Evaluating RAG contribution for text generation is based on measuring the quality of the LLM response, depending on the retrieved content. Es et al. [5] highlighted the need for automated evaluation for RAG-LLMs. Finardi et al. [6] examined the recall and precision of sparse retrieval (BM25) and Dense Retrieval (DPR) and their impact on question answering. A recent study by Cuconasu et al. [3] discusses the contributions of different types of documents for RAG in the context of question answering, contrasting between relevant, related, and random documents. They make the surprising claim that random documents that are totally orthogonal to the query bring unexpected value to the RAG process.

Ram et al. [20] examined RAG contribution to the LLM's text completion task, clearly showing that BM25 significantly outperforms dense retrieval methods. This result challenges the widespread practice of adopting dense retrieval as the retrieval method of choice in RAG-LLM-based tasks [1, 9, 15]. Our work, inspired by Ram et al. [20], differs from the other previous studies mentioned above by systematically examining existing retrieval methods, and attempting to understand why they perform better than the widely used dense retrieval methods.

## 3 EXPERIMENTAL SETUP

*Framework.* Our experimental framework is based on In-context RAG-LLM [20]. In this framework, the external source is the English Wikipedia dump from Dec. 20, 2018, in which articles are split into 100-word disjoint passages; each passage is considered an independent document. The retrieval system (the Retriever) is based on the Pyserini package [16]. Following Ram et al. [20], we feed the LLM with a sequence of 1024 tokens, henceforth referred to as the *original prompt*, taken from a text in the dataset so as to generate the next token. If RAG is applied, the $L$ rightmost tokens in the sequence are used to retrieve a passage list using the retrieval methods described below. The highest ranked passage is used instead of the leftmost tokens in the original sequence, together with the original rightmost tokens, to form a prompt, referred to as *RAG-prompt*. Previous studies (e.g., [20]) showed that using more than one passage for the text completion task does not bring additional value. Practically, to reduce the number of calls to the Retriever, we let the LLM generate a stride of $s$ tokens in response to an original prompt, augmented with a single retrieved passage (i.e., RAG-prompt); the stride $s$ is set to 4 following Ram et al. [20].

As in previous work [20], we use perplexity over the Wikitext-103 dataset [17] to measure generation quality for both the original text sequences and those augmented with a retrieved passage. We experiment with two LLMs: 1) GPT2-Small[1], to allow comparison with the original report [20], and 2) Llama2-7B[2] as a recent representative of a high quality LLM.

*Evaluation Measures.* Our goal is to measure the impact of retrieved passages on the generation quality of RAG-LLM. Hence, we compare the perplexity of the texts generated using the original prompt and the RAG-prompt. Measuring the resulting perplexity when using the highest retrieved passage provides a somewhat partial picture about the effectiveness of a retrieval method for RAG, specifically, with respect to lower ranks. Hence, we use a measure, novel to this study, DCG@k ($k = 5$), to estimate the expected contribution to text generation, in terms of resulting perplexity, of the $k$ highest ranked passages in a retrieved list[3]. We use $S(g)$ to denote the ratio of the perplexity value obtained when using the original prompt, to the one obtained when using the RAG-prompt with a retrieved passage $g$. Recall that lower perplexity indicates better text generation quality. We assign each passage $g$ among the $k$ highest ranked a label based on $S(g)$: 0 if $0 \leq S(g) \leq 1$, 1 if $1 < S(g) \leq 2$, and 2 if $S(g) > 2$. We then compute DCG at cutoff $k$.[4]

*Retrieval Methods.* We use the following sparse retrieval methods, and set their free-parameter values using a validation set as described below. Parameters that are not specified below were set to the default values of Pyserini.

- **BM25** - A probabilistic retrieval method [21], used in many RAG studies as a representative sparse method. We used the Pyserini interface to Lucene implementation, with $k_1$ set to values in {0.05, 0.1, 0.2,..., 0.6}.
- **Query Likelihood** (QL) - A unigram language-model-based method [24, 25] applied with two smoothing methods: QL-JM – Jelinek-Mercer Smoothing (with smoothing parameter set to values in {0.1,..., 0.6}), and QLD – Dirichlet Prior Smoothing (with smoothing parameter set to values in $\{1, 50, 100, 200\}$).
- **RM3** - Relevance Model #3, is a classic pseudo-feedback-based method that leverages unigram language models [10, 13]. We use the Pyserini RM3 version, which is based on BM25 scores rather than query likelihood scores. The original query weight was set to values in $\{0.25, 0.5, 0.75\}$; the number of top-retrieved passages used to construct the relevance model was in $\{3, 5, 7, 10\}$; and, the number of terms to which the model was clipped was in $\{5, 10, 33, 66, 100, 200\}$; the BM25 parameter values were those selected when BM25 was used alone.
- **MRF** - The Markov Random Field model with the sequential dependence method (SDM) [18]. This approach quantifies the occurrence (ordered and unordered) of consecutive pairs of query

[1] https://huggingface.co/gpt2
[2] https://huggingface.co/meta-llama/Llama-2-7b
[3] Note that each of the $k$ passages is used *separately* as described above.
[4] The thresholds for the labels were set based on histogram of $S(g)$ values over the validation set for the Okapi BM25 retrieval method: there was approximately a normal distribution for $S(g) \leq 2$ and then a long tail.

terms in windows of eight terms in the passages to be retrieved. We implemented SDM using LuceneIndri[5] since Pyserini has no implementation for this method. The weights of unigrams, bigrams and unordered bigrams were set to values in {0.8, 0.1, 0.1}, {0.8, 0.05, 0.15} and {0.8, 0.15, 0.05}, respectively. Jelinek-Mercer smoothing was used with the smoothing parameter set to values in {0.1, 0.2, 0.3, 0.4, 0.5, 0.6}.

We used the following pre-trained dense retrieval methods.

- **DPR** [11] - This is a popular pre-trained dense retrieval model for passage retrieval trained by contrastive learning.
- **Contriever** [8]. A DPR model, unsupervisedly trained on Wikipedia passages by cropping a subtext from the passage, to be marked as a query, and considering the remaining text as the relevant document, forming a positive training pair. We experimented with the pre-trained model and with Contriever$^{ft}$, a variant further fine-tuned on the MSMARCO dataset.

*Additional details.* We report the quality of the generated texts, in terms of perplexity and DCG@5, for varying lengths of queries: $L \in \{8, 16, 24, 32, 40, 48, 64, 96\}$. Recall that we use each retrieval method to feed the highest ranked passage to the LLM. The free parameters of each method (specified above), for each query length, were set using a grid search over a random subset of 5% of the Wikitext-103 validation set. This resulted in 3108 retrieval calls for GPT2 and 3667 calls for Llama2. The differences are due to the different tokenizers applied by the LLMs. The Wikitext-103 test set contains 71052 strides for GPT2 and 83267 strides for Llama2. We determine statistically significant performance differences (perplexity and DCG@k) using the two tailed paired t-test with $p = 0.05$. Bonferroni correction was applied for multiple hypothesis testing.

## 4 EXPERIMENTAL RESULTS

*4.0.1 Sparse vs. Dense.* We first study the effectiveness of the different retrieval methods for RAG-LLM, in the context of the text completion task. As noted in Section 3, we report perplexity and DCG@5 based on the generated text[6]; lower and higher values of perplexity and DCG@5, respectively, indicate better performance.

Table 1 presents our main result; the query length is fixed to 32 as in previous work [20]; we show below that the relative performance patterns are the same as those for other query lengths. We can see that in all cases (except for DPR with Llama2), the resultant performance of using the retrieved passage to augment the original propmt (i.e., using RAG) is better (often substantially) than using only the original prompt ("No Retrieval"). Furthermore, in line with Ram et al.'s [20] findings for BM25, *all* sparse methods outperform all dense methods for both LLMs.

We see in Table 1 that the performance of BM25 and the two query likelihood methods (QL-JM and QLD) is statistically indistinguishable. These three bag-of-terms methods are always outperformed by MRF; some of the differences are statistically significant. These findings attest to the merits of utilizing term proximity information (MRF) for RAG for the text completion task.

RM3 is always the best performing method in Table 1 for both LLMs and both evaluation measures. It statistically significantly outperforms each of the other methods for at least one LLM and one evaluation measure. Recall that RM3 is a pseudo-feedback-based query expansion approach (with unigram language models) originally designed to cope with the vocabulary mismatch between short queries and relevant documents [10, 13]. Here, we use RM3 with 32-token queries to retrieve a passage for prompt augmentation.

Regarding the dense methods, we see in Table 1 that DPR miserably fails, with performance worse than that of not using RAG for Llama2. Contriever performs slightly better; however, its fine-tuned version, Contriever$^{ft}$, is inferior probably due to the incompatibility of its training data (questions paired with relevant passages) for the text completion task.

*4.0.2 Query length.* Figures 1, 2 and 3 present the effect of query length ($L \in \{8, 16, 32, 40, 48, 64, 96\}$) on the resultant generation quality of using the retrieval methods. Due to computational constraints, we report the performance after free-parameter tuning over the validation set, which is smaller than the test set. (Refer to Section 3 for details.) Hence, the numbers in the figures are not comparable to those in Table 1 which were reported for the test set.

**Table 1: Resultant generation quality of using the original prompt ("No Retrieval") vs. augmenting it with a retrieved passage. 'b', 'm' and 'r' mark statistically significant differences with BM25, MRF and RM3, respectively. The best result in a column is boldfaced.**

| | GPT2-Small | | Llama2-7B |
|---|---|---|---|
| Method | Perplexity | DCG@5 | Perplexity |
| No Retrieval | $37.487^{b}_{mr}$ | – | $12.475^{b}_{mr}$ |
| BM25 | $29.348_{r}$ | $1.767_{r}$ | $10.850_{m}$ |
| QL-JM | $29.328_{r}$ | $1.762_{mr}$ | $10.856_{m}$ |
| QLD | $29.342_{r}$ | $1.765_{r}$ | $10.864_{m}$ |
| MRF | 29.172 | $1.770_{r}$ | $10.786^{b}$ |
| RM3 | $\mathbf{29.024^{b}}$ | $\mathbf{1.838^{b}_{m}}$ | **10.718** |
| DPR | $37.157^{b}_{mr}$ | $1.477^{b}_{mr}$ | $12.612_{mr}$ |
| Contriever | $31.925^{b}_{mr}$ | $1.681^{b}_{mr}$ | $11.399_{r}$ |
| Contriever$^{ft}$ | $32.394^{b}_{mr}$ | $1.674^{b}_{mr}$ | $11.567_{mr}$ |

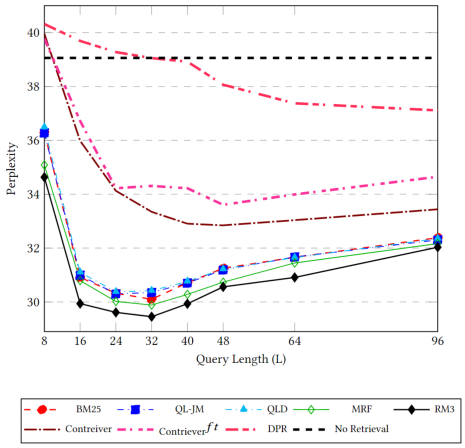


**Figure 1: GPT2-small: Effect of query length on perplexity.**

[5]https://github.com/lemurproject/Lucindri/

[6]We do not report DCG@5 for Llama2-7B due to computational constraints.

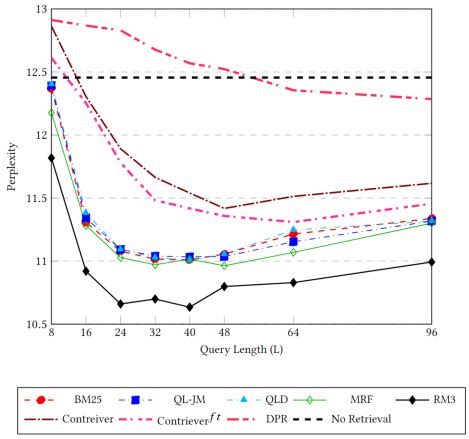


**Figure 2: Llama2-7B: Effect of query length on perplexity.**

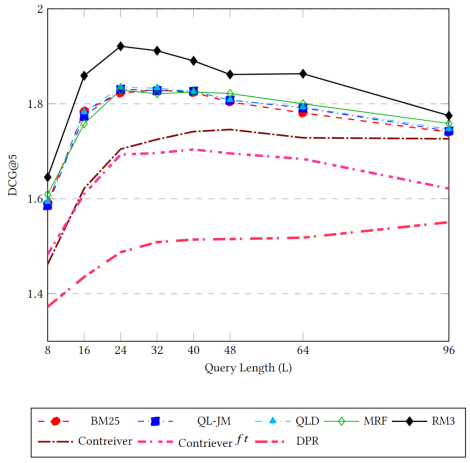


**Figure 3: GPT2-Small: Effect of query length on DCG@5.**

Figures 1 and 2 exhibit the same patterns despite the difference in scale between the two LLMs. The relative performance patterns of the retrieval methods in Figures 1, 2 and 3 are completely aligned across query lengths ($L$) with those reported in Table 1 for $L = 32$. Specifically, we observe the clear dominance of the sparse methods over dense methods. RM3 performs the best (with emphasized superiority for $L \in \{24, 32, 40\}$) and MRF is the second best. DPR performs poorly, with gains over "No Retrieval" only for queries longer than 40 and 48 tokens, for GPT2 and Llama2, respectively,

We also see in Figures 1, 2 and 3 that most sparse methods reach their best performance with queries of $L = 32$ tokens for GPT2, and $L \in \{32, 40, 48\}$ for Llama2. Queries shorter than $L = 16$ tokens severely hurt the performance of all methods. Long queries also hurt performance, except for DPR which keeps improving with increased query length.

*4.0.3 Lexical match.* The clear dominance of sparse over dense retrieval methods implies to the importance of a lexical match between the prompt and the augmenting passage in terms of resultant

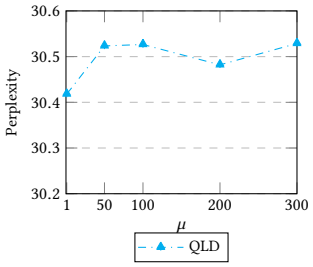


**Figure 4: GPT2-Small: Effect of the Dirichlet smoothing parameter ($\mu$) on the perplexity when using QLD with queries of $L = 32$ tokens.**

generation quality. To further study this point, we re-ranked the top-1000 passages retrieved by BM25 by the number of occurrences of unique query terms in the passage; ties were broken by BM25 scores. The resultant perplexity (over the test data) for GPT2-Small and Lllama2-7B was 29.58 and 10.796, respectively; the DCG@5 for GPT2-Small was 1.734. This performance is statistically indistinguishable from that of BM25, QL-JM and QLD, which was reported in Table 1 for Lllama2 (but not for GPT), albeit slightly lower. This leads us to conclude that there seems to be relatively limited merit in using advanced term-weighting schemes for RAG in the text completion task.

Additional support for this conclusion is provided in Figure 4, where we present the performance (over the validation set) of QLD as a function of the Dirichlet smoothing parameter $\mu$; GPT2-small is the LLM. We see that the best performance (lowest perplexity) is attained for a very small value of $\mu$ (1) and then the performance somewhat degrades and levels off.[7] Small $\mu$ means minimal smoothing with most weight put on the passage's terms; this also amounts to a minimal inverse document frequency effect [26]. These findings support our understanding that lexical match is a highly important factor in determining passage effectiveness for augmenting a prompt for text completion.

## 5 CONCLUSIONS AND FUTURE WORK

We demonstrated the superiority of sparse over dense retrieval methods for retrieval augmented generation for the LLM text completion task. Most sparse methods we studied had not been used in past work for RAG. Most notably, we showed that pseudo-feedback-based query expansion outperforms a term proximity model, which itself outperforms the commonly used Okapi BM25. Further exploration demonstrated the importance of increased simple lexical match between a prompt and the retrieved text used to augment it.

We plan to extend the retrieval methods' comparison to include additional approaches, and to explore evaluation measures in addition to perplexity and the corresponding DCG.

**Acknowledgments**. We thank the reviewers for their comments. This work was supported in part by the Israel Science Foundation (grant no. 403/22).

[7] Further decreasing $\mu$ to 0.001 results in even lower perplexity than that for 1.

# REFERENCES


[1] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*. PMLR, 2206–2240.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[3] Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The Power of Noise: Redefining Retrieval for RAG Systems. arXiv:2401.14887 [cs.IR]

[4] Virginia Dignum. 2019. *Responsible artificial intelligence: how to develop and use AI in a responsible way*. Vol. 2156. Springer.

[5] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. RAGAS: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217* (2023).

[6] Paulo Finardi, Leonardo Avila, Rodrigo Castaldoni, Pedro Gengo, Celio Larcher, Marcos Piau, Pablo Costa, and Vinicius Caridá. 2024. The Chronicles of RAG: The Retriever, the Chunk and the Generator. *arXiv preprint arXiv:2401.07883* (2024).

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

[8] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118* (2021).

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

[10] Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *Proceedings of SIGIR*.

[11] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen Tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*. Association for Computational Linguistics (ACL), 6769–6781.

[12] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172* (2019).

[13] Victor Lavrenko and W. Bruce Croft. 2001. Relevance-Based Language Models. In *Proceedings of SIGIR*. 120–127.

[14] Victor Lavrenko and W Bruce Croft. 2017. Relevance-based language models. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 260–267.

[15] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[16] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*. 2356–2362.

[17] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models. In *International Conference on Learning Representations*.

[18] Donald Metzler and W Bruce Croft. 2005. A markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. 472–479.

[19] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *choice* 2640 (2016), 660.

[20] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-Context Retrieval-Augmented Language Models. *Transactions of the Association for Computational Linguistics* (2023). https://arxiv.org/abs/2302.00083

[21] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (apr 2009), 333–389. https://doi.org/10.1561/1500000019

[22] Gerard Salton. 1989. Automatic text processing: The transformation, analysis, and retrieval of. *Reading: Addison-Wesley* 169 (1989).

[23] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. 3784–3803.

[24] Fei Song and W. Bruce Croft. 1999. A General Language Model for Information Retrieval. In *Proceedings of CIKM*. 316–321.

[25] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)* 22, 2 (2004), 179–214.

[26] ChengXiang Zhai and John D. Lafferty. 2001. A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. In *Proceedings of SIGIR*. 334–342.

[27] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. arXiv:2309.01219 [cs.CL]

[28] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).