The Power of Noise: Redefining Retrieval for RAG Systems

Florin Cuconasu*
cuconasu@diag.uniroma1.it
Sapienza University of Rome
Rome, Italy

Giovanni Trappolini*
trappolini@diag.uniroma1.it
Sapienza University of Rome
Rome, Italy

Federico Siciliano
siciliano@diag.uniroma1.it
Sapienza University of Rome
Rome, Italy

Simone Filice
filice.simone@gmail.com
Technology Innovation Institute
Haifa, Israel

Cesare Campagnano
campagnano@di.uniroma1.it
Sapienza University of Rome
Rome, Italy

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

Nicola Tonellotto
nicola.tonellotto@unipi.it
University of Pisa
Pisa, Italy

Fabrizio Silvestri
fsilvestri@diag.uniroma1.it
Sapienza University of Rome
Rome, Italy

ABSTRACT

Retrieval-Augmented Generation (RAG) has recently emerged as a method to extend beyond the pre-trained knowledge of Large Language Models by augmenting the original prompt with relevant passages or documents retrieved by an Information Retrieval (IR) system. RAG has become increasingly important for Generative AI solutions, especially in enterprise settings or in any domain in which knowledge is constantly refreshed and cannot be memorized in the LLM. We argue here that the retrieval component of RAG systems, be it dense or sparse, deserves increased attention from the research community, and accordingly, we conduct the first comprehensive and systematic examination of the retrieval strategy of RAG systems. We focus, in particular, on the type of passages IR systems within a RAG solution should retrieve. Our analysis considers multiple factors, such as the relevance of the passages included in the prompt context, their position, and their number. One counter-intuitive finding of this work is that the retriever's highest-scoring documents that are not directly relevant to the query (e.g., do not contain the answer) negatively impact the effectiveness of the LLM. Even more surprising, we discovered that adding random documents in the prompt improves the LLM accuracy by up to 35%. These results highlight the need to investigate the appropriate strategies when integrating retrieval with LLMs, thereby laying the groundwork for future research in this area.[1]

CCS CONCEPTS

• **Information systems** → **Novelty in information retrieval**.

[1]The code and data are available at github.com/florin-git/The-Power-of-Noise
*These authors contributed equally to this work.

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

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657834

KEYWORDS

RAG, LLM, Information Retrieval

ACM Reference Format:
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. 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, 11 pages. https://doi.org/10.1145/3626772.3657834

1 INTRODUCTION

Large Language Models (LLMs) [9] have demonstrated unprecedented proficiency in various tasks, ranging from text generation and complex question answering [6], to information retrieval (IR) tasks [22, 57]. However, LLMs have limitations in the handling of long contexts [52], a constraint that leads to an increased reliance on their pre-trained knowledge. This limitation not only confines their ability to effectively manage extended discourse, such as in books or long conversations, but also increases the probability of generating hallucinations, instances for which the model produces factually incorrect or nonsensical information [41]. To improve the accuracy of responses generated by LLMs, Retrieval-Augmented Generation (RAG) has emerged as a promising solution [28]. RAG is primarily designed to improve factual accuracy by providing the model access to auxiliary information, thereby augmenting the original prompt with information not necessarily memorized in the LLM. A key benefit of this approach is that it helps ground the prompt with relevant information that might help the LLM generate more accurate answers at inference time. At their core, RAG systems consist of two fundamental components: a *retriever* and a *generator*. The retriever is responsible for invoking an external IR system (dense and/or sparse) and feeding the selected results to a generator component.

This study focuses on the IR aspect of RAG, posing the following research question: *"What characteristics are desirable in a retriever to optimize prompt construction for RAG systems? Are current retrievers ideal?"*. We focus on the three main types of documents

(or passages[2]) that a retriever can return: *relevant*, *distracting*, and *random*. *Relevant* documents contain pertinent information that either directly answers or might inform the query. *Distracting* documents, while not directly answering the query, are semantically or contextually linked to the topic. For instance, if one asks for the color of Napoléon's horse, a passage describing the color of Joséphine de Beauharnais' (Napoléon's first wife) horse, while not containing the right information, would be highly related. *Random* documents have no relation whatsoever to the query and can be seen as a kind of informational noise within the retrieval process. One of the key goals of our study is to determine the role of each type of document and the relative value they bring to the LLM effectiveness. In particular, we verify whether there is a need to revisit some of the commonly accepted assumptions in IR systems when used in the context of LLMs. The main contributions of our work are the following:

(1) We conduct the first comprehensive study examining the impact of the type of retrieved documents in RAG on the LLM effectiveness.
(2) We propose retrieval RAG heuristics that leverage the unexpected results of this study.
(3) We release all associated code and data to the community to encourage further research.

2 RELATED WORKS

2.1 Generative Language Models

The inception of the modern LLM era can be traced back to the seminal paper titled "Attention Is All You Need" [52]. This work introduced the transformer architecture, a framework that adopts an attention mechanism instead of recurrent layers, enabling the model to capture global dependencies within the data. The following year, BERT (Bidirectional Encoder Representations from Transformers) [22] offered a significant improvement over the state-of-the-art via a novel bidirectional, unsupervised language representation. The evolution of transformer-based models continued with the development of the Generative Pre-trained Transformer (GPT) [37]. Its successor, GPT-2 [38], expanded upon this foundation with a larger scale model and demonstrated improved performance across a variety of language tasks without task-specific training. The subsequent iteration, GPT-3 [9], represented a further enhancement in model scale and capabilities, particularly in the realm of few-shot learning. Finally, recent times have seen a surge in the production of large, publicly available language models. Several actors have released their models, most notably, Llama [49, 50], Falcon [1], Mosaic MPT [47], and Phi [16, 29]. There are also versions of these models that have been fine-tuned on specific languages [5, 10, 12, 17, 43]. The proliferation and quality of these models are expanding the range of tasks and the vision they address [48, 54, 56].

2.2 Information Retrieval

Foundational information retrieval methodologies, such as the Vector Space Model and the TF-IDF scoring [42] introduced in the 1980s are the basis for quantifying textual similarity. These retrieval methods are characterized by their use of high-dimensional and sparse feature vectors and have been essential in developing a full generation of IR systems. BM25 represents the most famous current iteration [40]. A significant evolution in IR is the introduction of dense retrievers, which emerged from advancements in deep learning; they utilize low-dimensional dense vectors for textual representation, and allow to capture semantic relationships. This is in contrast to traditional IR methods (referred to as sparse in opposition to dense), which typically rely on lexical match and struggle with semantic match [32]. In the last few years, dense methods such as DPR [19] and others [15, 24] have demonstrated that they can compete with sparse methods.

2.3 Retrieve and Generate

RAG introduces a new approach in AI, combining the strengths of both retrieval-based and generative models. The concept of RAG was coined and popularized in [28], which introduced a model that combines a dense passage retriever with a sequence-to-sequence model, demonstrating substantial improvements in knowledge-intensive tasks. Similar methods/variations have also been proposed concurrently or soon after, such as [2, 4, 8, 13, 21]; see [33] for a survey on augmented language models. Researchers and practitioners have recently started to explore these RAG systems' inner workings. Notably, [44, 51] analyzed the impact of different types of documents on cascading IR/NLP systems. Other works have tried to study how attentive transformers are to their input [23, 30, 31, 39, 46]. [7] studied the effect of the retriever's similarity metric, which was found to be insufficient for reasoning. In [25, 55], authors analyzed LLM's receptiveness to external evidence against internal memory. In [60], they test the model's (in)ability to ground references.

In this paper, we want to provide the first comprehensive analysis of the implications of using a retriever module in a RAG system, studying the impact of several key factors, like the type, number, and position of documents that should augment the prompt to the LLM.

3 RAG

In this paper, we explore the application of RAG in the context of Question Answering, arguably its most popular application.

3.1 Open-Domain Question Answering

Open-domain Question Answering (OpenQA) refers to the task of developing systems capable of providing accurate and contextually relevant answers to a broad range of questions posed in natural language without limitations to specific domains or predefined datasets. In general, we want to find an answer $\mathcal{A}$ to a query q. To do so, we draw information from a corpus of documents $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$, which is usually assumed to be large in size. A prevalent approach for this task involves a two-step architecture, typically comprising a retriever and a reasoner (typically a generator). This methodology addresses the inherent complexities of OpenQA by dividing the process into distinct phases: first finding the appropriate set of documents that can potentially address the

[2]We interchangeably use here the terms "passage" or "document" to represent the indexing/retrieval unit of the IR system.

query and then synthesizing an answer, which can be consumed by the user of the QA system.

3.2 Retriever

The retriever plays a critical role in the OpenQA task. Its goal is to find a sufficiently small subset of documents $\mathcal{D}_r$ to allow the reasoner to answer the query correctly. Among the various retrieval methodologies, the use of a dense retriever has gained prominence due to its effectiveness in handling semantic matches. Dense retrieval requires transforming textual data into vector representations, which is typically achieved with a neural network, often a transformer-based encoder, like BERT [22]. The dense retriever processes both the query q and potential source documents to generate corresponding embeddings $\vec{q}$ for the query and $\vec{d_i}$ for each document $d_i \in \mathcal{D}$. The embedding process can be represented as:

$$\vec{q} = Encoder_q(q);\ \vec{d_i} = Encoder_d(d_i)$$

where $Encoder_q$ and $Encoder_d$ are neural network-based encoders, potentially sharing weights or architecture, designed to map the textual data into a vector space. Once the embeddings are generated, the retrieval process involves computing the similarity between the query embedding and each document embedding. The most common approach is to use dot product [20], defined as: $s(q, d_i) = \vec{q} \cdot \vec{d_i}$. This score quantifies the relevance of each document to the query by measuring their similarity in the embedded vector space, with higher scores indicating greater relevance. According to these scores, the top-ranked documents are selected for further processing in the generator component.

3.3 Reasoner

The second step involves a generator component in charge of synthesizing an answer, typically implemented via an LLM. Generative language models operate by predicting the probability distribution of the next token, given the previous tokens. For a given sequence of words $w_1, w_2, \ldots, w_n$, a generative language model aims to maximize the likelihood of this sequence, expressed using the chain rule of probability:

$$P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{N} P(w_i | w_1, w_2, \ldots, w_{i-1})$$

where $P(w_i|w_1, w_2, \ldots, w_{i-1})$ is the conditional probability of the word w_i given the preceding sequence of words $w_1, w_2, \ldots, w_{i-1}$. In RAG, the generative language model takes a query q and the retrieved documents $\mathcal{D}_r$ as input and generates a response by sequentially predicting the next token in the sequence. More formally,

$$P_{rag}(y|q) \approx \prod_{i}^{N} \sum_{d \in \mathcal{D}_r} p_\eta(d|q) p_\theta(y_i|q, d, y_{1:i-1}),$$

where $p_\eta(d|q)$ is the retrieval component that provides a (truncated) probability distribution for the top-scoring documents, and $p_\theta(y_i|q, d, y_{1:i-1})$ is a probability distribution parameterized by θ that generates a current token based on the previously generated tokens, the query, and the retrieved document; this role is filled by the LLM. In the case of dense retrieval, the probability distribution for the top-scoring documents may assume a functional form of the kind $p_\eta(d|q) \propto \exp(\vec{q} \cdot \vec{d})$. Given our formalization of the RAG task, we notice how the generative component p_θ depends on a given text, that is the query, and a dynamic text, that is the set of retrieved documents. We study in the next two sections the impact of changing the set of retrieved documents on the generator and, consequently, the whole end-to-end system. In particular, we aim to find the best set of documents $\mathcal{D}_r$ that a retriever should feed the generator to maximize the system's effectiveness.

4 EXPERIMENTAL METHODOLOGY

In this section, we detail the experimental framework. We start by describing the data used in the experiments and then discuss the type of documents that a retriever can return and pass to the LLM.

4.1 Natural Question Dataset

The Natural Questions (NQ) dataset [26] is a large-scale collection of real-world queries derived from Google search data. Each entry in the dataset consists of a user query and the corresponding Wikipedia page containing the answer. The NQ-open dataset [27], a subset of the NQ dataset, differs by removing the restriction of linking answers to specific Wikipedia passages, thereby mimicking a more general information retrieval scenario similar to web searches. This open-domain nature significantly impacts our experimental design, particularly in the selection and categorization of documents. Following the methodology of Lee et al. [27], our primary source for answering queries is the English Wikipedia dump as of 20 December 2018. Consistently with the Dense Passage Retrieval (DPR) approach [20], each Wikipedia article in this dump was segmented into non-overlapping passages of 100 words. A significant challenge in open-domain question answering is the potential temporal mismatch between the Wikipedia dump and the question-answer pairs in the dataset, which can lead to missing answers in the dataset, as highlighted in the AmbigQA study [34]. To mitigate this, we integrated the gold documents from the original NQ dataset into our Wikipedia document set. Given the open-domain nature of our task, there may be additional documents *relevant* to the query, i.e., containing the answer, but we will *not* consider them as *gold*. The final dataset comprises 21,035,236 documents, with 72,209 queries in the train set and 2,889 in the test set.

4.2 Types of Documents

In our study, we categorize documents into four distinct types, each represented by a unique symbol, based on their relevance and relationship to the queries:

★ *Gold Document.* The gold document, identified by ★, refers to the original context in the NQ dataset, specifically the passage of a Wikipedia page containing the answer and contextually relevant to a given query.

🔗 *Relevant Documents.* Denoted by 🔗, relevant documents are passages that, akin to the gold document, contain the correct answer and are contextually useful for answering the query. They provide additional sources of information that are correct and pertinent to the query. Notably, the gold document is a relevant document.

🌀 *Distracting Documents.* Symbolized by 🌀, distracting documents are semantically similar to the query but do not contain the

correct answer. They serve a crucial role in evaluating the generator's proficiency in discerning between relevant and non-relevant information. In practice, these are the top-scoring retrieved documents that are not relevant.

Random Documents. Indicated by , random documents are neither related to the query nor contain the answer. They are instrumental in assessing the model's ability to handle completely unrelated information. In practice, in our tests, we will randomly sample these documents from the corpus.

In our analysis, the entire set of documents fetched by the retriever is represented by the symbol . This possibly encompasses all document types — gold, relevant, distracting, or random — and serves to discuss the retrieval output in a generalized manner without specifying individual document categories.

4.3 Document Retrieval

Our methodology utilizes a two-step approach in line with a typical RAG setting, as explained in Section 3.2. As the first component, our experiments use *Contriever* [15], a BERT-based dense retriever, as the default retriever. It is trained without supervision using a contrastive loss. To enhance the efficiency of similarity searches within our corpus, comprising about 21 million documents, we also employ the FAISS IndexFlatIP indexing system [11]. The embedding of each document and query is obtained by averaging the hidden state of the last layer of the model.

4.4 LLM Input

Upon receiving a query, the retriever selects the top-k documents from the corpus according to a given similarity measure. These documents, in conjunction with the task instruction and the query, constitute the input for the LLM to generate a response. The NQ-open dataset was structured to include only those queries whose answers consist of no more than five tokens [27]. Consequently, the LLM is tasked with extracting a query response, confined to a maximum of five tokens, from the provided documents. The input is encoded into a prompt, whose template is shown in Figure 1, beginning with the task instruction, presented in italics for clarity. This is followed by the *context*, which comprises the selected documents followed by the query string. This prompt design aligns with the methodological approach outlined in [30]. While the composition of the context will vary according to the single experiment, the instruction will always be placed at the beginning of the prompt and the query always at the end.

4.5 LLMs Tested

We consider several LLMs in our experiments. Consistently across all models, we adopt a greedy generation approach with a maximum response length of 15 tokens. Acknowledging the constraints imposed by memory and computational resources, we have implemented a model quantization strategy, reducing all models to a 4-bit representation. Besides the above prompt, the models are not provided with additional exemplars for few-shot learning, which, while of interest, is outside the scope of this paper. We conduct tests on both the *base* and the *instruct* versions of the LLMs. However, we only report on the latter, as while the behavior is consistent across both, the instruct versions demonstrate superior performance.

LLM Input - Only Gold ★

You are given a question and you MUST respond by EXTRACTING the answer (max 5 tokens) from one of the provided documents. If none of the documents contain the answer, respond with NO-RES.
Documents:
Document [3](Title: Millennium Falcon) Han Solo won the Millennium Falcon from Lando Calrissian in the card game sabacc...

Question: who owned the millennium falcon before han solo
Answer: Han Solo

Figure 1: Example LLM input with an erroneous output, highlighted in red. The input consists of an *italicized task instruction*, followed by the context (documents), and the query. The LLM's response is marked under 'Answer'. The gold color highlights both the gold document and the correct answer, "Lando Calrissian", indicating the expected source and content of the accurate response.

- *Llama2.* The 7B parameters version of the Llama2 family [50] shows state-of-the-art performance on most downstream tasks compared to models of the same size. It was trained with a 4096 tokens context window and uses multi-query attention [45].
- *Falcon.* Falcon 7B, the smallest model of the Falcon series, [1] was trained on the RefinedWeb dataset [35], a large, filtered, and deduplicated corpus. Similarly to Llama2, it uses multi-query attention, with a context length of 2048 tokens.
- *Phi-2.* This is the smallest model used in this work (2.7B parameters). Despite its modest size, it achieves performance comparable to the other models [16, 29], thanks to its pre-training on "textbook-quality" data. It has a context window of 2048 tokens.
- *MPT.* This 7B parameters model uses ALiBi attention [36, 47] for a virtually unlimited context length. In our experiments, to leverage the model's full potential, we set the limit to 2048 tokens, i.e., the same used for the model's pre-training.

4.6 Accuracy

The NQ-open dataset allows a range of potential answers for each query. Frequently, these answers are different variants of the same concept (e.g., "President D. Roosevelt" or "President Roosevelt"), while in some cases, a single query may accept multiple distinct correct answers. To evaluate the accuracy of responses generated by LLMs, we use an assessment technique in line with [18, 30]. This methodology examines whether at least one of the predefined correct answers is contained within the response produced by the LLM. We measure the correctness of the LLM's responses as either

accurate or inaccurate based on the presence of the answer in a binary fashion. Nevertheless, this evaluation strategy is not without challenges. A principal issue arises in determining response correctness, particularly in instances involving date representations or varying phrasings conveying identical meanings. For example, if the LLM generates "Roosevelt" in response to a query where the established correct answer is "President Roosevelt", the response would be deemed incorrect under our current evaluation schema. Recognizing this limitation, we acknowledge the necessity for a more advanced analysis of answer variations, which we leave to future research.

5 RESULTS

Studying the characteristics of optimal prompts for RAG systems corresponds to answering our research question **(RQ)**: "*What characteristics are desirable in a retriever to optimize prompt construction for RAG systems in order to increase the LLM effectiveness?*". More specifically, we focus on three essential elements of the configuration: type, number, and positioning of the documents, and for each, we test various prompt combinations. To facilitate the understanding of our experimental setup, we employ a streamlined schema for representing the composition of prompts via the following symbols: **[I, ★, 🔗, 🎲, 🎞, Q]**. The task instruction (**I**) and the query (**Q**) are consistently positioned at the beginning and end, respectively. The middle section varies and represents different contextual elements - in this instance, these are gold, relevant, distracting, and random, appearing in that specific sequence. Additionally, the number of contextual documents is a variable in its own right and will be reported in the results tables below.

5.1 Impact of Distracting Documents

LLM Input - Distracting 🎲 and Gold ★

Task Instruction...
Documents:
Document [1](Title: Han Solo) Before the events of the film, he and Chewbacca had lost the "Millennium Falcon" to thieves, but they reclaim the ship after it...
Document [2](Title: Millennium Falcon) The "Falcon" has been depicted many times in the franchise, and ownership has changed several times...
Document [3](Title: Millennium Falcon) Han Solo won the Millennium Falcon from Lando Calrissian in the card game sabacc...

Question: who owned the millennium falcon before han solo
Answer: Han Solo

Figure 2: Example LLM input with an erroneous output, highlighted in red. The context of the prompt is composed of distracting documents and the gold near the query. The task instruction is as in Figure 1.

In our first set of experiments, we use a selection of 10K queries from the training set of the NQ-open dataset and assume an oracle setup in which the gold document for the query is known. To this effect, we add to the gold document a set of distracting documents, i.e., documents with high retrieval scores but not containing the answer, in order to measure their impact on the system; schematically **[I, 🎲, ★, Q]**. Figure 2 shows an example of this setup's visualization. Results of this experiment are shown in Table 1 (far, mid, and near relate to the distance between the gold document and the query; more details in the following sub-section). A critical observation emerging from this analysis is a clear pattern of progressive accuracy degradation as the number of distracting documents included in the context increases. This was observed across all LLMs, with accuracy deteriorating by more than 0.38 (−67%) in some cases. Even more importantly, adding just one distracting document causes a sharp reduction in accuracy, with peaks of 0.24 (−25%), as can be seen by comparing the row with 0 distracting documents (only gold scenario, as seen in Figure 1) with that of 1 distracting document. This experiment highlights a critical issue for RAG systems, particularly in real-world IR settings where related but non-answer-containing documents are commonplace. Our empirical analysis suggests that introducing semantically aligned yet non-relevant documents adds a layer of complexity, potentially misguiding LLMs away from the correct response. A visual explanation can be seen in Figure 3, which illustrates the attention scores within the prompt's context for a specific example in which the LLM incorrectly answers. This figure highlights the model's disproportionate focus on a distracting document (leftmost) at the expense of the gold document (rightmost), likely contributing to the erroneous response. Note that for consistency of results across LLMs, we need to account for their various input token capabilities: Llama2 can process up to 4096 tokens, but other models are limited to 2048 tokens. This led to the exclusion of evaluations with a higher number of distracting documents (namely greater than 10) as reflected by the empty values in the tables.

In addition, we wanted to verify that our results were not overly dependent on the type of dense retrieval system we used. We wanted, in particular, to check whether another dense retriever specifically trained on "hard negatives" would better distinguish between directly relevant and distracting documents, potentially leading to different results. To explore this hypothesis, we used ADORE [59], a state-of-the-art retriever trained with "dynamic hard negatives", to select the distracting documents. In scenarios with 1, 2, and 4 distracting documents in the **[I, 🎲, ★, Q]** setting with Llama2, we obtain an accuracy of 0.4068, 0.3815, and 0.3626, respectively. This is significantly lower than the baseline accuracy of 0.5642, where no distracting documents were included, and than the results obtained with Contriever in the same settings. We conclude from this that distinguishing between relevant and distracting information is a hard problem that cannot be mitigated simply by changing the dense retrieval method at this stage.

5.2 Impact of Gold Positioning

We conduct here another experiment where we systematically shift the position of the gold document within the context to study its

Table 1: Accuracy results of the LLMs when evaluated with prompts composed of the gold document ★ and a varying number of distracting 𝕊 documents. The table illustrates how the inclusion of an increasing number of distracting documents affects LLM's performance. Scenarios where the prompt exceeded the model's input limit, leading to potential data truncation, are not included (-). All values *not* marked with an asterisk * denote statistically significant changes from the gold-only document scenario [I, ★, Q] (first row), as determined by a Wilcoxon test (p-value < 0.01). Additionally, the closed-book accuracy scores for the models are as follows: Llama2 (0.1123), MPT (0.1205), Phi-2 (0.0488), Falcon (0.1083).

	Far - [I, ★, 𝕊, Q]				Mid - [I, 𝕊, ★, 𝕊, Q]				Near - [I, 𝕊, ★, Q]			
# 𝕊	Llama2	MPT	Phi-2	Falcon	Llama2	MPT	Phi-2	Falcon	Llama2	MPT	Phi-2	Falcon
0	**0.5642**	0.2148	**0.4438**	**0.4330**	**0.5642**	**0.2148**	**0.4438**	**0.4330**	**0.5642**	**0.2148**	**0.4438**	**0.4330**
1	0.4586	0.1976	0.3585	0.3469	no-mid	no-mid	no-mid	no-mid	0.4283	0.1791	0.4227	0.3602
2	0.3455	0.1913	0.3430	0.3246	0.3322	0.1802	0.3375	0.2823	0.3974	0.2002	0.3975	0.3111
4	0.2745	**0.2209***	0.3019	0.2670	0.2857	0.1775	0.2885	0.2378	0.3795	0.2059*	0.3701	0.2736
6	0.2898	0.2171*	0.2943	0.2392	0.2698	0.1424	0.2625	0.2103	0.3880	0.1892	0.3623	0.2656
8	0.2643	0.2077*	0.2513	0.1878	0.2268	0.1002	0.2360	0.1745	0.3748	0.1944	0.3423	0.2424
10	0.2537	-	-	-	0.2180	-	-	-	0.3716	-	-	-
12	0.2688	-	-	-	0.2382	-	-	-	0.3991	-	-	-
14	0.2583	-	-	-	0.2280	-	-	-	0.4118	-	-	-
16	0.2413	-	-	-	0.2024	-	-	-	0.3889	-	-	-
18	0.2348	-	-	-	0.1795	-	-	-	0.3781	-	-	-

Table 2: Accuracy results of the LLMs when evaluated with prompts composed of the gold document ★ and a varying number of random ⚅ documents. Surprisingly, increasing the number of random documents in the Near setting improves LLM's performance. Scenarios where the prompt exceeded the model's input limit, leading to potential data truncation, are not included (-). All values *not* marked with an asterisk * denote statistically significant changes from the gold-only document scenario [I, ★, Q] (first row), as determined by a Wilcoxon test (p-value < 0.01). Additionally, the closed-book accuracy scores for the models are as follows: Llama2 (0.1123), MPT (0.1205), Phi-2 (0.0488), Falcon (0.1083).

	Far - [I, ★, ⚅, Q]				Mid - [I, ⚅, ★, ⚅, Q]				Near - [I, ⚅, ★, Q]			
# ⚅	Llama2	MPT	Phi-2	Falcon	Llama2	MPT	Phi-2	Falcon	Llama2	MPT	Phi-2	Falcon
0	**0.5642**	0.2148	**0.4438**	**0.4330**	**0.5642**	0.2148	**0.4438**	**0.4330**	0.5642	0.2148	0.4438	**0.4330**
1	0.4733	0.2447	0.4329	0.4035	no-mid	no-mid	no-mid	no-mid	0.4862	0.2125*	0.4587	0.4091
2	0.3776	0.2639	0.4249	0.3805	0.3928	**0.2584**	0.4293	0.3612	0.5032	0.2660	**0.4614**	0.3912
4	0.3109	0.2933	0.4091	0.3468	0.3998	0.2577	0.3985	0.3462	0.5221	**0.2930**	0.4311	0.3949
6	0.3547	0.3036	0.4130	0.3250	0.4138	0.2265	0.3891	0.3196	0.5681*	0.2890	0.4388	0.3908
8	0.3106	**0.3039**	0.3812	0.2543	0.3734	0.1566	0.3596	0.2767	0.5609*	0.2911	0.4258	0.3704
10	0.3390	-	-	-	0.3675	-	-	-	0.5579*	-	-	-
12	0.3736	-	-	-	0.3641	-	-	-	0.5836	-	-	-
14	0.3527	-	-	-	0.3372	-	-	-	**0.5859**	-	-	-
16	0.3401	-	-	-	0.3159	-	-	-	0.5722	-	-	-
18	0.3466	-	-	-	0.2982	-	-	-	0.5588*	-	-	-

impact on the model's effectiveness. We define the positions of the gold document as follows:

- **Near:** placed adjacent to the query in the prompt **[I, 𝕊, ★, Q]** (as in Figure 2)
- **Mid:** inserted in the middle of the context **[I, 𝕊, ★, 𝕊, Q]**
- **Far:** positioned as far as possible from the query in the context **[I, ★, 𝕊, Q]**

Results in these settings partially corroborate evidence from [30]. The accuracy is higher when the gold document is near the query, lower when the gold document is furthest from it, and lowest when the gold document is placed in the middle of the context. For instance, Llama2, with 18 distracting documents, reaches an accuracy of 0.37, 0.23, and 0.17, respectively. These results are consistent across all models tested in the setting with distracting documents.

5.3 Impact of Noise

We devise an additional experimental setting aimed at evaluating the robustness of the RAG system against noise. To this effect, we take the gold document and add to it a certain number of documents picked at random from the corpus; see an example in Figure 4. Against our expectations, the performance does not deteriorate

Table 3: Accuracy of Llama2-7b in configurations involving random Wikipedia documents and retrieved documents [I, 🎲, 📄, Q]. Rows denote the number of random documents 🎲 added, and columns show the quantity of retrieved documents 📄. The left section reports results using Contriever, and the right section using BM25. Scenarios where the prompt exceeded the model's input limit, leading to potential data truncation, are not included (-). Each value *not* marked with an asterisk * represents a statistically significant change from the base case of retrieved documents only [I, 📄, Q] (first row), as determined by a Wilcoxon test (p-value < 0.01).

	Contriever							BM25						
# 🎲 \ # 📄	1	2	3	4	5	8	10	1	2	3	4	5	8	10
0	0.1620	0.1866	0.1876	0.1866	0.1921	0.2198	0.2108	0.2008	0.2208	0.2084	0.2028	0.2243	0.2492	0.2447
1	0.1308	0.1616	0.1717	0.1893*	0.1987*	0.2153*	0.2146*	0.1568	0.1963	0.1921	0.2115	0.2295*	0.2475*	0.2506*
2	0.1315	0.1644	0.1859*	0.2008	0.2174	0.2156*	0.2368	0.1644	0.1973	0.2080*	0.2281	0.2558	0.2495*	0.2596
3	0.1301	0.1727	0.2008	0.2316	0.2201	0.2198	0.2409	0.1568	0.2063	0.2160	0.2520	0.2579	0.2644	0.2707
5	0.1464	0.2056	0.2233	0.2240	0.2150	**0.2451**	**0.2482**	0.1772	0.2402	0.2437	0.2520	0.2554	0.2804	**0.2866**
8	0.1734	0.2066	0.2336	0.2375	0.2454	0.2416	0.2364	0.1994	0.2451	0.2579	0.2769	0.2817	**0.2859**	0.2777
10	0.1796	0.2174	0.2450	0.2502	**0.2499**	0.2420	-	0.2108	0.2589	0.2734	0.2835	**0.2935**	0.2853	-
15	0.2018	0.2354	0.2551	**0.2530**	-	-	-	0.2243	0.2686	0.2790	**0.2928**	-	-	-
16	0.2032	**0.2471**	**0.2558**	-	-	-	-	0.2323	0.2662	**0.2838**	-	-	-	-
17	0.2039	0.2426	-	-	-	-	-	**0.2326**	**0.2693**	-	-	-	-	-
18	**0.2073**	-	-	-	-	-	-	0.2309	-	-	-	-	-	-

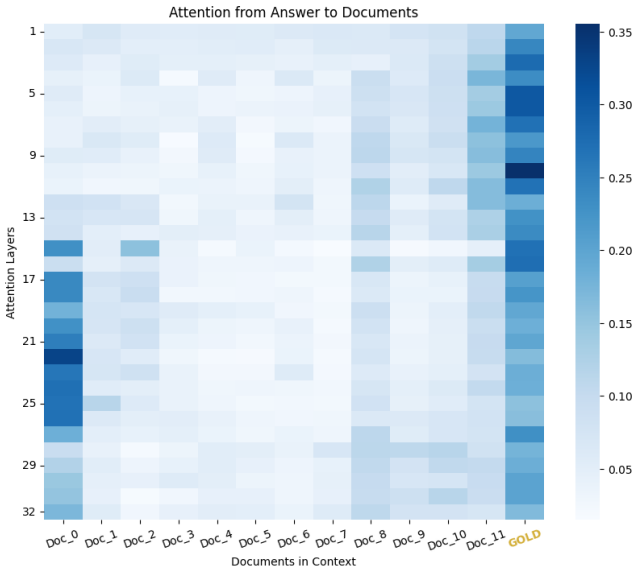

Figure 3: This heatmap depicts the attention distribution across the context documents from the example shown in Figure 2, relative to the answer generated by Llama2-7b in a prompt structured as [I, 📄, ★, Q]. Cell (i, j) denotes the mean attention that tokens in the generated answer allocate to the tokens of the i-th document within the j-th attention layer. This mean attention for each document is calculated by averaging the attention scores across all its constituent tokens.

LLM Input - Random 🎲 and Gold ★

Task instruction...
Documents:
Document [140](Title: Richard Yates (novelist)) For much of his life, Yates's work met almost universal critical acclaim, yet not one of his books sold over 12,000 copies in...
Document [242](Title: Android version history) Code name Version number Initial release date API level Security patches (No codename) 1.0 September 23...
Document [3](Title: Millennium Falcon) Han Solo won the Millennium Falcon from Lando Calrissian in the card game sabacc...

Question: who owned the millennium falcon before han solo
Answer: Lando Calrissian

Figure 4: Example LLM input with a correct output, highlighted in green. The context of the prompt is composed of random documents and the gold near the query. The task instruction is as in Figure 1.

in the presence of noise, as can be seen in Table 2. Instead, we observe an improvement in performance under the best-performing setting (near **[I, 🎲, ★, Q]**), with an improvement of 0.08 (+36%) in the case of MPT. Furthermore, we observe that different models exhibit distinct behaviors. Both Llama2 and Phi-2 showed improvements in this setting when the noise is introduced furthest from the query. However, when the noise is positioned in the far **[I, ★, 🎲, Q]** and mid **[I, 🎲, ★, 🎲, Q]** settings, these models exhibit a decline in performance. Notably, this performance degradation is much less accentuated when compared to the earlier setting with distracting documents. This suggests that while Llama2 and Phi-2 can effectively handle noise far from the query, their ability to sift

through irrelevant information diminishes as the noise is placed closer to it. The MPT model presented a unique response; it showed an improvement in performance under all settings. Standing out from the rest, the Falcon model did not exhibit an improvement in performance as observed in other models with the introduction of noise. Peculiarly enough, Falcon and Llama2 do not consistently exhibit a "lost in the middle" phenomenon, having in some instances better accuracy in the mid than far setting, for instance, in the case with 8 noisy documents added.

5.4 RAG in Practice

To address our primary Research Question **(RQ)** about the characteristics of an effective RAG retriever, and following the results reported above, we now consider a more realistic scenario than an oracle setup. Namely, given a query, we retrieve a set of documents that can be either relevant or distracting. We then add random documents to this set of retrieved ones, schematically: **[I, 🎲, 📄, Q]**. For this second set of experiments, we use the test set of the NQ-open dataset. Results for this experiment, using Llama2, can be seen on the left side of Table 3. These results show that, regardless of the number of retrieved documents, adding random documents up until the context length is filled is almost always beneficial, with gains in terms of accuracy up to 0.07 (+35%) in the case of 4 retrieved documents.

5.4.1 Testing Sparse Retrievers. In an effort to validate our initial observations, we replicate our experiment using a sparse retrieval approach, specifically BM25. The corresponding results are outlined in the right section of Table 3. Consistent with earlier findings, we observe that including random documents leads to an improvement in the effectiveness of the LLM. Notably, the use of BM25 yields an average increase in accuracy of 3-4 percentage points. This improvement is attributed to the quality of documents retrieved by BM25. We quantitatively evaluate the effectiveness of the retrieval methods by computing the top-k accuracy for varying numbers of retrieved documents. Note that this heuristic, while indicative, does not capture the full spectrum of relevance. Our evaluation, based on the presence of correct answers within documents, might overlook the context-specific relevance due to potential lexical matches of the answer string in documents. Despite this limitation, this method aligns with established computational practices in literature [15, 19]. In our analysis, BM25 demonstrated higher relative top-k accuracy (0.2966, 0.4105, 0.5237, 0.6663 for $k = 1, 2, 4, 10$) compared to those of Contriever (0.2502, 0.3569, 0.4784, 0.6085 for the same k), underscoring its effectiveness in retrieving more relevant documents in our experimental setup.

5.4.2 Increasing The Randomness. While our previous experiments show the benefits of adding random documents, one might argue that these documents are not totally random as they originate from the same corpus (Wikipedia) and that they might help the LLM answer in a fashion that is consistent with the corpus. For this reason, we carry out another experiment in which random documents are drawn from a drastically different corpus in terms of tone and style, namely Reddit Webis-TLDR-17 dataset [53]. The results are outlined on the left of Table 4. The inclusion of documents from the Reddit corpus not only maintains the observed increase in accuracy but even enhances it, with an improvement of 0.023 (+9% accuracy) when compared to the previous best score. Pushing the randomness even further, we carry out another test where we consider nonsensical sentences made up of random words as random documents. Remarkably, even in this scenario, we observe a performance improvement when compared to the base case of Wikipedia random documents, as shown in the right side of Table 4.

5.4.3 Falcon. As shown in Table 2, Falcon does not reach the same performance increase when random documents are added to the gold document **[I, 🎲, ★, Q]**. Accordingly, we want to verify whether it behaves differently when adding retrieved rather than gold documents. We find that the addition of random documents on top of retrieved documents **[I, 🎲, 📄, Q]** does improve the effectiveness of Falcon; see detailed results in Table 5. These results are in contrast with the ones obtained in the oracle setting, where Falcon was robust to noise. This new finding further validates our experimental evidence, namely that, outside the oracle setting, all the tested models show an improvement when a certain amount of noise is added.

5.5 Retriever Trade-Off

The experimental evidence detailed above not only contradicts the common perception that semantically close documents are helpful for LLMs but also highlights the need for a delicate balance between relevant and random documents. When arranged as described, random documents seem to exert a positive influence on LLM accuracy. However, for the LLM to generate accurate answers, some degree of relevant information must exist in the context. On the other hand, an overabundance of retrieved documents increases the likelihood of including distracting and non-relevant information, leading to a sharp decline in performance. While establishing a formal or comprehensive theory behind these findings remains an open research challenge, we can still infer that there seems to be a trade-off between the number of relevant and totally irrelevant documents. More specifically, we observed that the best effectiveness is achieved when a minimal set of documents is initially retrieved and then supplemented with random documents until the context limit is reached. For the queries examined in this study, retrieving between 3 and 5 documents is the most effective choice. Adding more increases the risk of including too many distracting, thus counterproductive, documents. We argue here that there is a pressing need for further research towards investigating how these initial findings can be exploited. More importantly, it is evident that we have yet to refine our understanding of the retriever's role within a RAG system.

On The Unreasonable Effectiveness Of Random Documents. We cannot close this paper without attempting to explain the results shown up to this point. We refer back to our RAG formulation, particularly the conditioned function $p_\theta(y|\cdot, d)$. In hindsight, we can now state that by adding random documents to the context, we are better conditioning this function, inducing enhanced accuracy. Previous research [3, 14], particularly [58], hints that there might be cases in which a pathologically low attention entropy causes the LLM to generate degenerate outputs with a sharp decrease in performance. These episodes are named entropy collapse. Following this line of

Table 4: Accuracy of Llama2-7b in configurations involving random documents and retrieved documents by Contriever [I, 🎲, 📄, Q]. Rows denote the number of random documents 🎲 added, and columns show the quantity of retrieved documents 📄. The left section reports results with random documents from Reddit and the right section with nonsensical sentences made up of random words. Scenarios where the prompt exceeded the model's input limit, leading to potential data truncation, are not included (-). Each value *not* marked with an asterisk * represents a statistically significant change from the base case of retrieved documents only [I, 📄, Q] (first row), as determined by a Wilcoxon test (p-value < 0.01).

	Random from Reddit							Random Words						
# 🎲 \ # 📄	1	2	3	4	5	8	10	1	2	3	4	5	8	10
0	0.1620	0.1866	0.1876	0.1866	0.1921	0.2198	0.2108	0.1620	0.1866	0.1876	0.1866	0.1921	0.2198	0.2108
1	0.1693*	0.1931	0.1845*	0.1907	0.2008	0.2084	0.2084	0.1744	0.1924*	0.1969	0.2077	0.2091	0.2139*	0.2073*
2	0.1886	0.2018	0.2101	0.2143	0.2160	0.2222*	0.2219	0.1765	0.1855*	0.2094	0.2122	0.2181	0.2045	0.2084*
3	0.1897	0.2108	0.2212	0.2340	0.2371	0.2326	0.2319	0.1755	0.1990	0.2166	0.2201	0.2288	0.2032	0.2156*
5	0.1897	0.2215	0.2388	0.2468	0.2409	**0.2769**	**0.2451**	0.1862	0.2139	0.2319	0.2367	0.2232	0.2184*	0.2278
8	0.2011	0.2326	0.2354	0.2489	0.2440	0.2568	0.2364	0.1973	0.2274	0.2319	0.2316	0.2305	0.2357	**0.2412**
10	0.2053	0.2326	0.2451	0.2534	**0.2551**	0.2658	-	0.2053	0.2271	0.2340	0.2385	**0.2406**	**0.2499**	-
15	0.2240	0.2489	**0.2689**	**0.2786**	-	-	-	0.2215	0.2416	**0.2589**	**0.2634**	-	-	-
16	0.2240	0.2561	0.2676	-	-	-	-	**0.2219**	0.2437	0.2568	-	-	-	-
17	**0.2243**	**0.2565**	-	-	-	-	-	0.2201	**0.2450**	-	-	-	-	-
18	0.2240	-	-	-	-	-	-	0.2177	-	-	-	-	-	-

Table 5: Accuracy of Falcon-7b on Reddit data in the random + retrieved setting [I, 🎲, 📄, Q]. Rows denote the number of random documents 🎲 added, and columns show the quantity of retrieved documents 📄. Scenarios where the prompt exceeded the model's input limit, leading to potential data truncation, are not included (-). Each value *not* marked with an asterisk * represents a statistically significant change from the base case (first row), as determined by a Wilcoxon test (p-value < 0.05).

# 🎲 \ # 📄	1	2	3	4	5	9
0	0.1568	0.1717	0.1855	0.1938	0.1942	**0.1998**
1	0.1551*	0.1793*	0.1897*	0.1924*	0.1976*	-
2	0.1529*	0.1762*	0.1938*	0.2011*	0.1976*	-
3	0.1599*	0.1727*	0.1911*	0.2021*	**0.2118**	-
4	0.1606*	0.1758*	0.1959	0.2073	0.2108	-
5	0.1627*	0.1762*	0.2000	**0.2108**	-	-
6	0.1651*	0.1848	**0.2004**	-	-	-
7	0.1675	**0.1848**	-	-	-	-
8	**0.1682**	-	-	-	-	-

research, we measure the entropy of the attention scores in the case where only the gold document is supplied **[I, ★, Q]** against the case in which random documents are added **[I, 🎲, ★, Q]**. We find that when we introduce random documents, the entropy of the systems has a 3X increase. Although these experiments show a pattern, we cannot yet answer this question in a definitive manner. While out of the scope of this work, which focuses on the retriever component of RAG systems, we believe it is highly important to investigate the reasons for which the LLM shows this behavior. Future studies should aim to elucidate why this noisy state is more advantageous and identify the characteristics that contribute to its effectiveness.

6 CONCLUSIONS

In this paper, we conducted the first comprehensive study focusing on the impact of retrieved documents on the RAG framework, aiming to understand the traits required in a retriever to optimize prompt construction for a RAG system. This study led to several important findings, including two unexpected ones. First, the position of relevant information should be placed near the query; otherwise, the model seriously struggles to attend to it. Second, in contrast to common perception, top-scoring retrieved documents that do not contain the answer, when added to a prompt, negatively impact the LLM effectiveness. Finally, and even more surprisingly, random, noisy documents are actually helpful in increasing the accuracy of these systems when correctly positioned within a prompt. While we have proposed heuristics to exploit these findings, further research is needed both to uncover the inner mechanisms behind this behavior and to develop a new generation of information retrieval techniques that are specifically designed to interact with the generative component.

ACKNOWLEDGMENTS

This work is supported by the Spoke "FutureHPC & BigData" of the ICSC – Centro Nazionale di Ricerca in High-Performance Computing, Big Data and Quantum Computing, the Spoke "Human-centered AI" of the M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research", SERICS (PE00000014), IR0000013 - SoBigData.it, funded by European Union – NextGenerationEU, the FoReLab project (Departments of Excellence), and the NEREO PRIN project funded by the Italian Ministry of Education and Research Grant no. 2022AEFHAZ. This work was carried out while Florin Cuconasu was enrolled in the Italian National Doctorate on Artificial Intelligence run by the Sapienza University of Rome.

REFERENCES

[1] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The Falcon Series of Open Language Models. arXiv:2311.16867 [cs.CL]

[2] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. arXiv:2310.11511 [cs.CL]

[3] Giuseppe Attanasio, Debora Nozza, Dirk Hovy, and Elena Baralis. 2022. Entropy-based attention regularization frees unintended bias mitigation from lists.

[4] Andrea Bacciu, Florin Cuconasu, Federico Siciliano, Fabrizio Silvestri, Nicola Tonellotto, and Giovanni Trappolini. 2023. RRAML: Reinforced Retrieval Augmented Machine Learning. In *Proceedings of the Discussion Papers - 22nd International Conference of the Italian Association for Artificial Intelligence (AIxIA 2023 DP) co-located with 22nd International Conference of the Italian Association for Artificial Intelligence (AIxIA 2023), Rome, Italy, November 6-9, 2023 (CEUR Workshop Proceedings, Vol. 3537)*, Roberto Basili, Domenico Lembo, Carla Limongelli, and Andrea Orlandini (Eds.). CEUR-WS.org, 29–37. https://ceur-ws.org/Vol-3537/paper4.pdf

[5] Andrea Bacciu, Giovanni Trappolini, Andrea Santilli, Emanuele Rodolà, and Fabrizio Silvestri. 2023. Fauno: The Italian Large Language Model that will leave you senza parole!. In *Proceedings of the 13th Italian Information Retrieval Workshop (IIR 2023), Pisa, Italy, June 8-9, 2023 (CEUR Workshop Proceedings, Vol. 3448)*, Franco Maria Nardini, Nicola Tonellotto, Guglielmo Faggioli, and Antonio Ferrara (Eds.). CEUR-WS.org, 9–17. https://ceur-ws.org/Vol-3448/paper-24.pdf

[6] Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open LLM Leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.

[7] Parishad BehnamGhader, Santiago Miret, and Siva Reddy. 2023. Can Retriever-Augmented Language Models Reason? The Blame Game Between the Retriever and the Language Model. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 15492–15509. https://doi.org/10.18653/v1/2023.findings-emnlp.1036

[8] 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, Baltimora, 2206–2240.

[9] 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.

[10] Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and Effective Text Encoding for Chinese LLaMA and Alpaca. *arXiv preprint arXiv:2304.08177* (2023). https://arxiv.org/abs/2304.08177

[11] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The Faiss library. (2024). arXiv:2401.08281 [cs.LG]

[12] Garrachonr. 2023. LlamaDos. https://github.com/Garrachonr/LlamaDos.

[13] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*. PMLR, Vienna, 3929–3938.

[14] David T Hoffmann, Simon Schrodi, Nadine Behrmann, Volker Fischer, and Thomas Brox. 2023. Eureka-Moments in Transformers: Multi-Step Tasks Reveal Softmax Induced Optimization Problems.

[15] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning.

[16] Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. 2023. Phi-2: The surprising power of small language models.

[17] jphme. 2023. Llama-2-13b-chat-german. https://huggingface.co/jphme/Llama-2-13b-chat-german.

[18] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*. JMLR.org, Honolulu, Hawaii, USA, Article 641, 12 pages.

[19] 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.

[20] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[21] Zixuan Ke, Weize Kong, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Bridging the Preference Gap between Retrievers and LLMs. *arXiv preprint arXiv:2401.06954* (2024).

[22] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, Vol. 1. Association for Computational Linguistic, Minneapolis, 2.

[23] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context.

[24] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. ACM, Xi'an, 39–48.

[25] Bevan Koopman and Guido Zuccon. 2023. Dr ChatGPT tell me what I want to hear: How different prompts impact health answer correctness. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 15012–15022. https://doi.org/10.18653/v1/2023.emnlp-main.928

[26] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. https://doi.org/10.1162/tacl_a_00276

[27] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, 6086–6096. https://doi.org/10.18653/V1/P19-1612

[28] 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.

[29] Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report.

[30] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts.

[31] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 8086–8098. https://doi.org/10.18653/v1/2022.acl-long.556

[32] C Manning, P Raghavan, and H Schutze. 2008. *Term weighting, and the vector space model.* Cambridge University Press Cambridge, Cambridge. 109–133 pages.

[33] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey.

[34] Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering Ambiguous Open-domain Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 5783–5797. https://doi.org/10.18653/v1/2020.emnlp-main.466

[35] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only. arXiv:2306.01116 [cs.CL]

[36] Ofir Press, Noah Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. https://openreview.net/forum?id=R8sQPpGCv0

[37] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

[38] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[39] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models.

[40] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[41] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for Building an Open-Domain Chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty (Eds.). Association for Computational Linguistics, Online, 300–325. https://doi.org/10.18653/v1/2021.eacl-main.24

[42] Gerard Salton and Michael J. McGill. 1983. Introduction to modern information retrieval. *McGraw-Hill* (1983).

[43] Andrea Santilli and Emanuele Rodolà. 2023. Camoscio: an Italian Instruction-tuned LLaMA. arXiv:2307.16456 [cs.CL]

[44] Artsiom Sauchuk, James Thorne, Alon Halevy, Nicola Tonellotto, and Fabrizio Silvestri. 2022. On the Role of Relevance in Natural Language Processing Tasks. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Madrid, 1785–1789.

[45] Noam Shazeer. 2019. Fast Transformer Decoding: One Write-Head is All You Need. arXiv:1911.02150 [cs.NE]

[46] Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. 2021. Do long-range language models actually use long-range context?

[47] MosaicML NLP Team et al. 2023. Introducing mpt-7b: A new standard for open-source, ly usable llms.

[48] Gabriele Tolomei, Cesare Campagnano, Fabrizio Silvestri, and Giovanni Trappolini. 2023. Prompt-to-OS (P2OS): Revolutionizing Operating Systems and Human-Computer Interaction with Integrated AI Generative Models. In *5th IEEE International Conference on Cognitive Machine Intelligence, CogMI 2023, Atlanta, GA, USA, November 1-4, 2023*. IEEE, 128–134. https://doi.org/10.1109/COGMI58952.2023.00027

[49] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models.

[50] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models.

[51] Giovanni Trappolini, Andrea Santilli, Emanuele Rodolà, Alon Y. Halevy, and Fabrizio Silvestri. 2023. Multimodal Neural Databases. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 2619–2628. https://doi.org/10.1145/3539618.3591930

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[53] Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to Learn Automatic Summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, Lu Wang, Jackie Chi Kit Cheung, Giuseppe Carenini, and Fei Liu (Eds.). Association for Computational Linguistics, Copenhagen, Denmark, 59–63. https://doi.org/10.18653/v1/W17-4508

[54] Shuai Wang, Liang Ding, Li Shen, Yong Luo, Bo Du, and Dacheng Tao. 2024. OOP: Object-Oriented Programming Evaluation Benchmark for Large Language Models. *arXiv preprint arXiv:2401.06628* (2024).

[55] Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2023. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *The Twelfth International Conference on Learning Representations*.

[56] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. 2024. OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments. arXiv:2404.07972 [cs.AI]

[57] Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained Transformers for Text Ranking: BERT and Beyond. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Tutorials*, Greg Kondrak, Kalina Bontcheva, and Dan Gillick (Eds.). Association for Computational Linguistics, Online, 1–4. https://doi.org/10.18653/v1/2021.naacl-tutorials.1

[58] Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M Susskind. 2023. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*. PMLR, PMLR, Hawaii, 40770–40803.

[59] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing Dense Retrieval Model Training with Hard Negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. New York, NY, USA, 1503–1512.

[60] Guido Zuccon, Bevan Koopman, and Razia Shaik. 2023. ChatGPT Hallucinates when Attributing Answers. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region* (Beijing, China) *(SIGIR-AP '23)*. Association for Computing Machinery, New York, NY, USA, 46–51. https://doi.org/10.1145/3624918.3625329