Fine Tuning vs. Retrieval Augmented Generation for Less Popular Knowledge

Heydar Soudani
Radboud University
Nijmegen, The Netherlands
heydar.soudani@ru.nl

Evangelos Kanoulas
University of Amsterdam
Amsterdam, The Netherlands
e.kanoulas@uva.nl

Faegheh Hasibi
Radboud University
Nijmegen, The Netherlands
faegheh.hasibi@ru.nl

Abstract

Language Models (LMs) memorize a vast amount of factual knowledge, exhibiting strong performance across diverse tasks and domains. However, it has been observed that the performance of these models diminishes when dealing with less-popular or low-frequency concepts, for example, in domain-specific applications. The two prominent approaches to enhance the performance of LMs on less frequent topics are Retrieval Augmented Generation (RAG) and fine-tuning (FT) over synthetic data. This paper explores and evaluates the impact of RAG and FT on customizing LMs in handling low-frequency entities in question answering tasks. We conduct extensive experiments on twelve LMs of varying size and type and different FT methods, data augmentation, and retrieval models. Our findings indicate that while FT boosts performance across entities of varying popularity, RAG surpasses FT by a large margin, particularly for least popular factual knowledge. Additionally, the success of both RAG and FT approaches is amplified by improving retrieval and data augmentation techniques. Fine-tuning, while beneficial for small LMs, requires extensive resources. To address this issue, we propose the new *Stimulus RAG* approach that surpasses the effectiveness of fine-tuning based approaches, thereby eliminating the need for the costly data augmentation and fine-tuning steps for enriching LMs with less popular factual knowledge.

CCS Concepts

• **Computing methodologies** → **Natural language generation**; • **Information systems** → **Question answering**; **Novelty in information retrieval**; **Language models**.

Keywords

Retrieval Augmented Generation, Fine Tuning, Data Augmentation

ACM Reference Format:
Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. 2024. Fine Tuning vs. Retrieval Augmented Generation for Less Popular Knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region (SIGIR-AP '24), December 9–12, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3673791.3698415

This work is licensed under a Creative Commons Attribution International 4.0 License.

SIGIR-AP '24, December 9–12, 2024, Tokyo, Japan
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0724-7/24/12
https://doi.org/10.1145/3673791.3698415

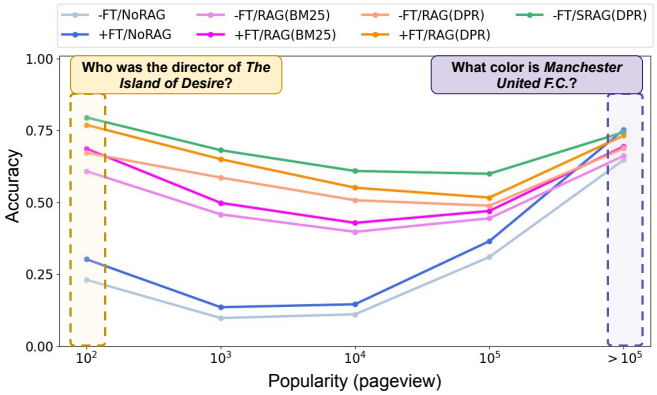

Figure 1: Comparison of RAG and fine-tuning on StableLM2 performance in question answering over factual knowledge. RAG-based approaches significantly enhance the performance of the vanilla StableLM2, outperforming fine-tuning by a large margin. Our proposed SRAG approach outperforms all models, including the fine-tuning based approaches.

1 Introduction

Language Models (LMs) exhibit outstanding capabilities in executing tasks that demand extensive memorization of factual data [13]. However, their memorization capabilities are constrained when dealing with less frequent entities [19, 27, 39, 51], and even the largest models may encounter the well-known "hallucination" problem [47] and temporal degradation [29]. Consequently, when LMs are intended for deployment in less resourced domains, customization becomes imperative to ensure optimal performance. A common example is within the industrial setup, where chatbots or Question Answering (QA) systems need to accurately answer users' questions about a proprietary knowledge graph or intra-company terminology with limited textual description [48, 60].

Retrieval-Augmented Generation (RAG) and Fine-Tuning (FT) stand out as two prominent approaches for adapting LMs to specific domains [17, 41, 43, 48]. RAG retrieves relevant information from a document corpus and enhances LM's response generation through the implementation of in-context learning (ICL) [15, 60]. Conversely, FT approach updates model weights to become adept at recalling specific information and enhance its memorization capabilities during inference [6]. In the context of less popular knowledge, where limited data is available, data augmentation methods are utilized to generate synthetic training data, serving as an initial step towards FT [49, 50]. Despite existing research on enhancing LM's memorization with RAG [39], no work to our knowledge

has compared RAG with with knowledge obtained through FT, particularly for less popular knowledge.

In this paper, we aim to understand which approach is more appropriate for customizing LMs for less resourced domains. Specifically, we seek to answer this research question: ***(RQ1)*** *How does RAG compare to fine-tuning for question answering over less popular factual knowledge, and which factors affect their performance?* To address this question, we conduct a comprehensive comparison of RAG and fine-tuning methods for less popular knowledge, assuming that textual descriptions, albeit limited, are available for a specific domain and application. We, therefore, collect Wikipedia documents related to QA datasets over long-tail entities and apply two methods of knowledge injection: parametric knowledge injection using FT and non-parametric knowledge injection using RAG. For the FT approach, the LM is fine-tuned with synthetically generated QAs from these documents using data augmentation approaches [3, 55]. For the RAG approach, we use retrievers to rank the most relevant documents for a query. We investigate how the effectiveness of these methods is affected by the following aspects: (i) fine-tuning method; i.e., full FT vs. parameter efficient fine-tuning (PEFT), (ii) data augmentation method, (iii) LM type and size; i.e., decoder only vs. encoder-decoder models and varying size, ranging from 80M to 11B parameters, and (iv) retrieval model performance. Through exhaustive experimentation on twelve LMs and different setup of fine-tuning, data augmentation, and retrieval models, we arrive at the following conclusions:

- **Fine-tuning method:** Comparing full FT with PEFT (i.e., QLoRA [18]) for LMs with less than 2 billion parameters, full FT is more effective than PEFT in the downstream task. PEFT, however, preserves the reasoning ability of LMs (needed for RAG) and outperforms full FT when the fine-tuned models are used in combination with RAG (Table 2).
- **Data augmentation method:** Comparing prompt-based and a state-of-the-art fine-tuned [55] QA generation models, the prompt-based method demonstrates better performance for the downstream task. This suggests that the high-quality synthetic data generated by large LMs can better assist LMs with memorizing new knowledge, compared to the greater volume of data generated by the fine-tuned model (Table 2).
- **LM type and size:** Comparing decoder-only with encoder-decoder LMs (Flan-T5 models of various sizes), decoder-only models outperform encoder-decoder models of similar size. Interestingly, larger LMs generally do not benefit from fine-tuning, while smaller ones do. Therefore, a small fine-tuned LM with RAG can perform on par or better than a large LM; e.g., StableLM2 (1.6B) vs. Llama3 (8B) (Table 3).
- **Retrieval model:** Comparing retrievers with varying performance in the RAG system, we observe that as the popularity of factual knowledge increases, the performance of the retriever decreases (Figure 7). Moreover, the performance of the RAG system increases by using higher performance retriever (Figures 1 and 8).
- **Fine-tuning vs. RAG:** Comparing these two knowledge injection methods, RAG substantially outperforms fine-tuning. Fine-tuned LMs combined with RAG either outperform or perform on par with vanilla LMs with RAG in all but one case (Figure 1).

While fine-tuning improves accuracy in answering factual questions, both with and without RAG, it demands a considerable amount of effort and resources. This leads us to our second research question: ***(RQ2):*** *Can we avoid the cost of fine-tuning by developing an advanced RAG approach that surpass the performance of a fine-tuned LM with RAG?* To answer this question, we develop *Stimulus RAG (SRAG)*, a new RAG approach that stimulates an LM to generate the correct response based on the provided *hint* in the prompt. The hint is extracted from the top retrieved documents by the retrieval model. Our results demonstrate that *Stimulus RAG outperforms all other combinations of fine-tuning, both with and without retrieve-then-generate RAG.*

To summarize, this paper makes the following contributions:

- We study the effectiveness of fine-tuning and RAG approaches for question answering over less popular factual knowledge and compare the performance of these models across distinct setups: vanilla and fine-tuned models, both with and without RAG, using different data augmentation methods.
- We perform extensive experiments to understand how fine-tuned and RAG models are affected by four different factors: data augmentation method, fine-tuning method, LM type and size, and retrieval model.
- We propose a new RAG approach *Stimulus RAG* that outperforms all RAG and fine-tuning setups, thereby bypassing the need for expensive fine-tuning.

2 Related Work

Parametric and Non-parametric Knowledge. It is demonstrated that large pre-trained LMs memorize a significant amount of world knowledge in their parameters (*parametric knowledge*) [39]. FT can update the parametric knowledge embedded in LMs and customize it for a specific domain [19, 41]. One of the important principles for FT is data availability, which is limited specifically in specialized domains [17, 43]. Data augmentation (DA) addresses the data scarcity problem by generating task- and domain-relevant samples from existing unlabeled texts. A common DA approach for the QA task is generating question-answer pairs through a four-step *Pipeline*, consisting of passage selection, answer extraction, question generation, and consistency filtering [3, 30, 32, 55]. Ushio et al. [55] conducted an empirical study comparing three QA generation approaches: Pipeline, Multitask, and End-to-End (E2E) and showed the E2E approach outperforms others in downstream tasks.

Furthermore, a large body of work shows that augmenting LMs with *nonparametric knowledge* (i.e., retrieved text chunks) enables much smaller models to match the performance of larger models [31]. In this method, known as Retrieval Augmented Generation (RAG), an information retrieval system is utilized to find relevant documents and adds them to the input prompt to enhance response generation of LMs [4, 5].

As interest grows in refining pre-trained LMs for particular tasks, the comparison of FT and RAG strategies under equitable conditions is becoming increasingly important. Mosbach et al. [41] explored the effectiveness of few-shot FT versus ICL for classification tasks in general domains. de Luis Balaguer et al. [17] compared FT and RAG in answering agriculture and geography-specific questions.

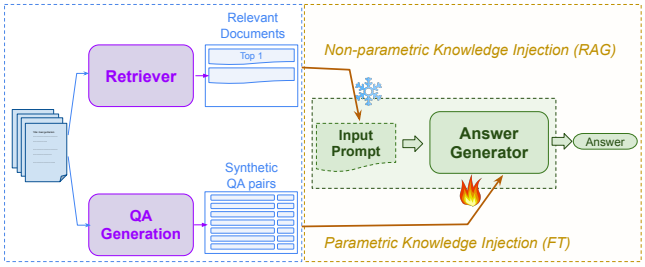

Figure 2: Overview of parametric and non-parametric knowledge injection for less popular factual knowledge. First, we prepare the corpus. Next, we generate knowledge in two formats: textual documents and synthetic QA pairs. Finally, we inject the knowledge into the prompt or LM parameters.

Ovadia et al. [43] assessed the performance on multiple-choice questions in specialized areas like anatomy, astronomy, biology, and prehistory. In contrast to these studies, we directly address the integration of less popular factual knowledge into LMs, comparing various retrievers, data augmentation, and fine tuning methods.

Less Popular Knowledge. An entity's popularity in LMs is gauged by its frequency in the model's pre-training data [20, 40], often assessed through the entity's occurrences in a large corpus [27] via entity linking [16, 56]. Due to the practical challenges of direct counting, e.g., annotation of large-scale collections with entities [26], proxies are defined to approximate the popularity of factual knowledge. Sun et al. [51] use traffic metrics and content density, while Maekawa et al. [38] introduce the co-occurrence of the subject entity and relation predicate as a popularity proxy. Wikipedia pageviews are among the most prevalent methods for measuring the popularity of entities [10, 39, 46].

RAG Development. RAG introduces a new approach in AI, combining the strengths of both retrieval-based and generative models [15]. The concept of RAG was coined and popularized by Lewis et al. [31], who introduced a model that combines a dense passage retriever with a sequence-to-sequence model, called Retrieve-then-Generate. This approach demonstrated substantial improvements in knowledge-intensive tasks. Several parameters affect a RAG system's accuracy, including the performance of the retriever model [11], the relevance of the passages included in the prompt context, their position, and their number [6, 15].

However, several works have argued that the Retrieve-then-Generate approach is not optimal for more complex tasks, and more advanced RAG systems are needed. Adaptive-RAG [23] defines a classification-based RAG system to decide which RAG model should be used based on the question type. RAFT [60] trains a model to ignore documents that don't help in answering the question, thereby adapting LMs to domain-specific RAG. Generate-then-Retrieve (GR) [1] argues that the Retrieve-then-Generate paradigm is insufficient when the answer must be obtained from multiple documents. They introduce the GR pipeline, which first generates multiple queries and then retrieves information for the generated queries. Mallen et al. [39] found that for popular knowledge, using RAG can hurt performance, so they defined an adaptive retrieval system to use retrieval only where it is beneficial.

We discuss that increasing the number of documents is not helpful and may introduce more noise into the LM's input. To address this problem, we propose a new stimulus RAG system that highlights parts of the input text most likely to contain the correct answer. This approach aids the LM in accurately identifying and extracting relevant information. Highlighting has been used in the literature of information retrieval for various purposes. Askari et al. [8] aim to generate synthetic documents for queries, highlighting keywords to create high-quality documents. Cho et al. [12] propose generating sub-sentence summary highlights to overlay on source documents, enabling users to quickly navigate through content. Li et al. [33] introduce a new prompting framework to provide black-box LMs with fine-grained, instance-specific guidance toward desired outputs. Our work differs from these in that we highlight sentences using a simple yet effective reranker model, which directly improves RAG's performance.

3 Methodology

In this section, we introduce our evaluation framework (Figure 2), which is designed to assess the effectiveness of two knowledge injection methods: the parametric method using FT and the non-parametric method using RAG.

3.1 Task Definition

This study specifically focuses on factual knowledge [2] of entities, defined as information about particular attributes and characteristics of target entities, among various types of world knowledge [39]. We chose this because the amount of knowledge memorized by LMs can be approximated by its accuracy in answering simple factual questions, such as "In what city was Lisa Miller born?" [51].

Factual knowledge is defined as a triplet of (subject, relationship, object) [39]. In this context, the question involves the subject and the relationship, while the answer corresponds to the object. By using these template questions, we ensure that LMs understand the question and derive the answer from their embedded knowledge. We select Wikipedia-based question-answering datasets focused on factual knowledge. This enables us to measure the popularity of entities based on Wikipedia pageviews and also obtain the corresponding evidence document for each entity from Wikipedia. For each dataset, we select Wikipedia pages whose corresponding entities appear in the test dataset. This setup mirrors real-world industry practices, where entities and their textual descriptions relate to companies' specific internal concepts.

3.2 Knowledge Injection with Fine-tuning

LMs are primarily pre-trained on general domains. To customize LMs for specific knowledge or a particular domain, fine-tuning (FT), also known as parametric knowledge injection, is commonly used. However, FT requires a substantial amount of training samples, which are often unavailable for specific applications, such as those within a company. Data augmentation (DA) offers a solution to this training data shortage. To achieve parametric knowledge injection, we fine-tune LMs using synthetically generated question-answer (QA) pairs. Formally, given a set of documents $D = \{d_1, \ldots, d_n\}$, where n is the number of documents, a QA pair generator $\mathcal{G}_{qa}$ is tasked with generating as many QA pairs as possible:

$$\mathcal{G}_{qa}(d_i) = \{(q_i^1, a_i^2), \ldots, (q_i^m, a_i^m)\},$$

LLM Input

You are a question-answer generator. Your goal is to generate question-answer pairs given the Context.

Example output: {'question': "", "answer": ""}

Context: $CONTEXT

Step 1: *Identify spans that are likely to be answers to questions, identify as many as possible.*
Step 2: *For each identified span, generate a question.*
Step 3: *Respond to the question in only a few tokens concisely.*
Step 4: *Output in JSON format following [...]*

Ensure that you distinctly label and delineate Steps 1, 2, 3, and 4. Let's think step by step:

Figure 3: Input prompt for prompt-based QA pair generation. We define a CoT prompt to outline the generation steps.

where q_i and a_i denote a question and answer generated for the document d_i, m is the total number of generated QAs for the document, and the generated set of question-answer pairs is denoted as Q_i. The QA generation method $\mathcal{G}_{qa}$ is applied to all documents D, and the generated set $Q = \bigcup_{i=1}^{n} Q_i$ is then used for fine-tuning an LLM, which consists of a set of learnable parameters to predict the probabilities of future or masked tokens.

We employ two QA generation methods. The first one is end-to-end (**E2E**) QA generation, where a fine-tuned sequence-to-sequence model generates QA pairs from d_i. We utilize the E2E approach by Ushio et al. [55], which employs a trained T5-large for paragraph-level QA generation and has been shown to be more robust and effective than the established pipeline approach. The term "E2E" is used because the QA generation process is not divided into two sequential components, i.e., answer extraction and question generation. Instead, a QA pair is generated in one go. To train the E2E model, the training question-answer pairs Q_i^{train} corresponding to the training document d_i^{train} are converted into a flattened sentence y_i the following transformation:

$$\mathcal{T}(q, a) = \text{“}\texttt{question: } \{q\}, \texttt{ answer: } \{a\}\text{”}$$

$$y_i = \text{“}\{\mathcal{T}(q_i^1, a_i^1)\} \mid \{\mathcal{T}(q_i^2, a_i^2)\} \mid \ldots\text{”},$$

where each pair is textualized with the function $\mathcal{T}(q, a)$, and the textualized QA pairs are concatenated using the separator |. The E2E QA generation function $\mathcal{G}_{qa}$ is then obtained by maximizing the conditional log-likelihood:

$$\arg\max_{y} P_{qa}(y \mid d).$$

The second QA generation method is the **prompt** approach, in which the QA pair generator $\mathcal{G}_{qa}$ is an instruction-tuned LM capable of reasoning over the input prompt I. In this approach, the QA pairs are generated as follows: $\mathcal{G}_{qa}(d_i) = LM(d_i, I)$. We utilize Zephyr [54] with Chain of Thought (CoT) [57] prompting for QA generation, as demonstrated in Figure 3.

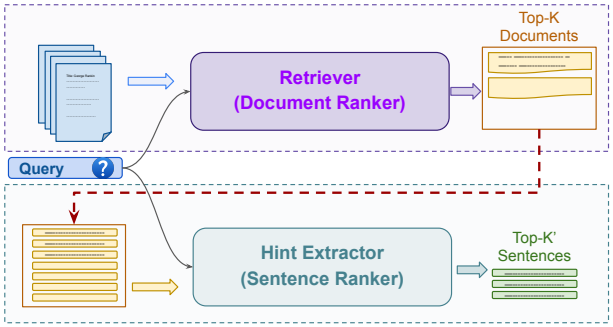

Figure 4: Our proposed Stimulus RAG method. The *Hint Extractor* identifies the most relevant sentence from top-K documents ranked by the retriever. This sentence is then added to the beginning of the input prompt.

3.3 Knowledge Injection with RAG

The non-parametric knowledge injection is performed using RAG, which consists of two components: the *Retriever* and the *Generator* [6, 15].

Retriever. The first key component in a RAG system is a retriever R, which builds an index for a document corpus D. During inference, given an input sequence q, the retriever identifies and ranks relevant documents $D_q = R(D, q)$. In our retrieval process, we employ both sparse and dense retrievers. We utilize BM25 [44] as a sparse retriever due to its popularity and effectiveness. For dense retrievers, we employ DPR [28] and Contriever [22] methods. Both models convert textual data into vector representations using a transformer network. The similarity between the query q and document d is defined as $S(q, d) = \vec{q} \cdot \vec{d}$, which computes the dot product between embedding vectors $\vec{q}$ and $\vec{d}$. DPR employs two independent BERT models, trained discriminatively using query-documents pairs with negative samples from BM25. Contriever, on the other hand, is trained using a shared BERT model for query and document encoding, optimized using a contrastive loss. We also employ a two-step retrieval pipeline, which includes first-stage retrieval using BM25 and reranking using DPR [1, 7, 34].

Generator. The second step involves a generator component responsible for synthesizing an answer, typically implemented via LMs. Generative LMs operate by predicting the probability distribution of the next token, given the previous tokens. In RAG, the generative LM takes a query q and top-K ranked documents from D_q, denoted as $D_q^K = [d_1, ..., d_K]$, and generates a response by sequentially predicting the next token. Our RAG prompt prepends the documents before the query, following [15, 39].

In this paper, we define and assess four distinct configurations of injecting knowledge with fine tuning and RAG: (1) *-FT-RAG*: the vanilla LM without retrieved documents, (2) *-FT+RAG*: the vanilla LM with retrieved documents, (3) *+FT-RAG*: the fine-tuned LM without retrieved documents, (4) *+FT+RAG*: the fine-tuned LM with retrieved documents.

3.4 Stimulus RAG

While the generic retrieve-then-generate framework of RAG is effective in answering factual knowledge [39], it sometimes struggles

Table 1: Statistics of the factual knowledge-based datasets.

Dataset	# QA	# Rel. Type	Question form
PopQA	14K	16	Template
WitQA	14K	32	Model-assisted
EQ	17.3K	24	Template

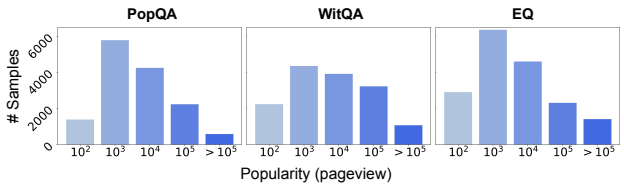

Figure 5: Distribution of sample counts across popularity buckets, defined by log_{10}(pageviews) for PopQA and WitQA and log_2(pageviews) for EQ.

to accurately respond to factual questions that are not memorized in LMs parameters, even when the ground truth document is included in the prompt. We hypothesize that by employing a more advanced RAG approach we can achieve or even surpass the benefits of the combined RAG and FT (*+FT+RAG*) setup. This would allow us to bypass the computationally expensive process of knowledge injection with FT, which involves resource-intensive processes of QA generation and FT.

We introduce *Stimulus RAG*, a RAG approach that guides LMs to generate responses using a hint provided in the RAG prompt. Stimulus RAG comprises three steps: *Retriever*, *Hint extractor*, and *Generator*. The Retriever and Generator are as defined in Section 3.3. The ***Hint extractor*** provide hint text that guides LM to generate accurate responses and works as follows: The top-K ranked documents $D_q^K = [d_1, ..., d_K]$ (d_K denoting the retrieved document at rank K) from the retriever step are split into sentences, denoted as $S_q = \{s_j\}_{j=0}^{N}$, where N is the total number of sentences. These sentences are then ranked using a retrieval model $R'(S_q, q)$. For the sake of simplicity, we use the same retrieval model as that used in the retriever step, hence $R = R'$. The top-ranked sentence s_q^1 is identified as the most relevant sentence for the given question and serves as a hint. The sentence s_q^1 can be directly used in the prompt, which we refer to as the SRAG(S) approach. Alternatively, the document containing the sentence s_q^1 can be considered as a hint, which we refer to as the SRAG(D) approach. The extracted hint is placed at the top of the input prompt provided to the ***Generator*** component. This implies that the hint text a repetition of presumably the most relevant sentence/document of in the prompt. It has been shown that the beginning of the input prompt receives more attention from LMs [36]. Our method is illustrated in Figure 4.

4 Experimental Setup

Datasets. We conduct our experiments on three datasets focused on factual knowledge: PopQA [39], WitQA [38], and EntityQuestion (EQ) [46], all of which include long-tail entities; see Table 1 for statistics. PopQA is an open-domain QA dataset about long-tail entiteis, constructed from 16 diverse relationship types in Wikidata. EQ is another popular open-domain QA dataset that covers a long-tail entity distribution, using Wikipedia hyperlink counts as a proxy for entity frequency and sampling knowledge triples from Wikidata based on these frequency distributions. Since EQ does not provide Wikidata IDs for each entity, we use only about 80% of the questions, where the mention of the subject entity has a unique match with a Wikidata entity. WitQA is another entity-centric dataset that defines a different proxy for popularity. They argue that the popularity metric should be based on the occurrence of both the subject entity and the relation (unlike the subject-based popularity in PopQA and EQ). Therefore, they define the S-R count, which is the co-occurrence of the subject entity and relation predicate. However, they report pageviews in their dataset, and we use the pageview-based popularity to enable comparison with other datasets. To analyze performance with respect to popularity, we divide the entities into five buckets based on their popularity levels; see Figure 5.

Table 2: Accuracy of vanilla and fine-tuned LMs, both with and without RAG. The RAG results are based on ideal retrieval. Statistically significant differences in the *PEFT-Prompt* rows are compared with other rows. Superscripts (a), (b), (c), and (d) indicate statistically significant differences (better or worse) compared to vanilla LM, *PEFT-E2E*, *Full-E2E*, and *Full-Prompt*, respectively, determined by the Wilcoxon test (p-value < 0.01).

		PopQA		EQ	
FT	**QA**	**+FT-RAG**	**+FT+RAG**	**+FT-RAG**	**+FT+RAG**
FlanT5-small		2.69	47.46	2.84	27.36
PEFT	E2E	6.06	48.40	7.86	33.76
PEFT	Prompt	$7.01^{(a,c,d)}$	$61.39^{(a,b,c,d)}$	$9.39^{(a,b,c,d)}$	$42.68^{(a,b,c,d)}$
Full	E2E	5.19	12.63	10.98	21.27
Full	Prompt	8.55	46.88	15.52	38.38
FlanT5-base		6.01	73.08	6.07	53.92
PEFT	E2E	7.53	70.34	10.98	51.30
PEFT	Prompt	$9.11^{(a,b,c)}$	$71.34^{(a,b,c,d)}$	$12.98^{(a,b,c,d)}$	$57.63^{(a,b,c,d)}$
Full	E2E	7.42	44.76	10.91	31.22
Full	Prompt	10.06	51.80	17.36	54.07
FlanT5-large		8.44	68.56	16.94	52.64
PEFT	E2E	8.69	67.47	15.33	53.25
PEFT	Prompt	$11.24^{(a,b,d)}$	$71.27^{(a,b,c,d)}$	$18.17^{(a,b,c)}$	$60.08^{(a,b,c,d)}$
Full	E2E	11.75	27.31	14.79	23.17
Full	Prompt	13.60	68.18	18.22	57.37
StableLM2		17.01	76.14	17.92	60.72
PEFT	E2E	16.39	74.82	23.62	58.34
PEFT	Prompt	$21.75^{(a,b,c,d)}$	$82.09^{(a,b,c,d)}$	$27.23^{(a,b,c,d)}$	$68.82^{(a,b,c,d)}$
Full	E2E	14.23	5.87	13.22	2.46
Full	Prompt	25.73	32.50	23.22	30.07

Evaluation Metric. Following previous studies [38, 39, 46], we report on the accuracy metric, where a prediction is considered correct if one of the ground truth responses matches a substring of the predicted response. While widely used [15, 27, 36], this metric is not without issues. A principal problem arises in determining

Table 3: Accuracy of vanilla and fine-tuned LMs, both with and without RAG. The RAG results are based on Ideal retrieval. The PEFT method is used for fine-tuning, and prompting is utilized for QA generation. The best results in each column are shown in bold, and the best results in each row are underlined. Statistically significant differences in the *+FT+RAG* columns are compared with the *-FT+RAG* columns. Superscript (a) indicates statistically significant differences (better or worse) as determined by the Wilcoxon test (p-value < 0.01).

Dataset		PopQA				WiTQA				EQ			
Model	#P	-FT -RAG	+FT -RAG	-FT +RAG	+FT +RAG	-FT -RAG	+FT -RAG	-FT +RAG	+FT +RAG	-FT -RAG	+FT -RAG	-FT +RAG	+FT +RAG
FlanT5-small	80m	2.69	7.26	47.46	61.39$^{(a)}$	8.76	18.30	40.76	59.47$^{(a)}$	2.84	9.39	27.36	42.68$^{(a)}$
FlanT5-base	250m	6.01	9.11	73.08	71.34$^{(a)}$	16.52	23.32	73.35	74.34	6.07	12.98	53.92	57.63$^{(a)}$
FlanT5-large	780m	8.44	11.24	68.56	71.27$^{(a)}$	24.52	28.85	74.37	77.24$^{(a)}$	16.94	18.17	52.64	60.08$^{(a)}$
Tiny-llama	1.1B	17.50	18.32	74.39	74.87	45.12	47.65	78.84	80.60$^{(a)}$	21.12	24.57	61.03	61.61
StableLM2	1.6B	17.01	21.75	76.14	**82.09**$^{(a)}$	42.18	51.66	81.08	86.19$^{(a)}$	17.92	27.23	60.72	**68.82**$^{(a)}$
MiniCPM	2B	14.16	15.47	69.44	75.86$^{(a)}$	37.61	44.94	73.17	80.45$^{(a)}$	15.31	22.92	54.63	62.67$^{(a)}$
FlanT5-xl	3B	12.24	13.25	73.31	74.71$^{(a)}$	31.24	36.38	76.97	79.67$^{(a)}$	15.98	20.42	59.07	62.70$^{(a)}$
Mistral	7B	21.47	30.70	80.25	78.44$^{(a)}$	51.36	58.29	86.05	83.90$^{(a)}$	25.78	34.01	68.60	64.96$^{(a)}$
Zephyr	7B	28.23	**35.48**	78.65	78.20	58.33	**63.74**	83.83	**86.89**$^{(a)}$	29.09	**38.52**	62.45	67.89$^{(a)}$
Llama2-chat	7B	26.09	27.71	81.15	80.15	53.88	56.38	86.50	84.71$^{(a)}$	27.13	32.84	68.29	67.11$^{(a)}$
Llama3-chat	8B	**32.52**	32.75	**81.29**	81.54	**61.88**	61.58	**86.86**	85.71	**35.07**	37.95	**68.67**	68.64
FlanT5-xxl	11.3B	11.26	15.94	75.19	74.98	30.40	42.89	78.19	81.44$^{(a)}$	12.48	23.16	59.67	62.80$^{(a)}$

response correctness, particularly in cases involving date representations or varying phrasings that convey identical meanings [15]. For example, comparing the predicted response "Nathanson" with the ground truth "Jeff Nathanson," the prediction is considered incorrect. Another observed problem is that when the model generates multiple entity names, among them the ground truth entity, the response is incorrectly considered as correct. Recognizing these limitations, we acknowledge the necessity for a more advanced analysis of answer variations, which we leave for future research.

Language Models. We use several LMs, focusing on two main features: the backbone architecture (i.e., decoder-only and encoder-decoder) and model size, which ranges from 80 million to over 11 billion parameters. For the encoder-decoder models, we utilize five versions of the FlanT5 family [14], spanning from small to XXL. For the decoder-only models, we employ smaller LMs such as Tiny-llama [59], StableLM2 [9], and MiniCPM [21], which range from 1 to 2 billion parameters. Additionally, we incorporate larger LMs like Mistral [24], Zephyr [54], LLama2 [53], and LLama3,[1] which range from 7 to 8 billion parameters. As for instructions, we apply zero-shot prompting for generative prediction using a straightforward template. For non retrieved-augmented input, we use template: `"Question: <question>"`, and for retrieved-augmented input, we use template: `"Context: <context> Question: <question>"`.

Fine Tuning. We generate training data for the FT approach using two distinct data augmentation methods. To ensure a fair comparison between FT and RAG with an ideal retriever, we generate QAs exclusively using the summary sections of Wikipedia pages. After generating QA pairs, we proceed to fine-tune LMs using two approaches: full parameter tuning (**Full**) and Parameter Efficient Fine-Tuning (**PEFT**). Within the range of PEFT techniques [35, 37, 58], we utilize QLoRA [18], chosen for its broad acceptance in the field [25, 42] and efficient use of computational resources we had at our disposal.

RAG. We utilize a variety of retrieval models to obtain relevant documents for the RAG approach, including BM25 [45], Contriever [22], DPR [28], and a two-stage re-ranker that combines BM25 with DPR, all implemented according to the BEIR benchmark [52]. Additionally, since the selected datasets do not contain grounded document evidence, we assume that the summary section of each Wikipedia page is the answer-containing document. We define an ideal retriever model as one that returns the summary paragraph as the top-ranked document, referred to as the **Ideal** retriever throughout the paper. We acknowledge that this assumption is not entirely accurate as some answers may be found in other subsections. However, our evaluation of the downstream task (Figures 6 and 8) demonstrates that the Ideal retriever outperforms other retrievers.

Stimulus RAG. We select a DPR retriever for the *hint extractor* and set $K = 3$. We report on two variations of the SRAG approach: (i) SRAG(S), which utilizes the top-1 sentence as the hint, and (ii) SRAG(D), which inserts the entire document contacting the top-ranked sentence. For the instruction, we used the following template: `"Context: <hint><context> Question: <question>"`.

5 Results

In the following, we evaluate fine-tuning and RAG methods on different setups and answer our two research questions listed in Section 1.

[1] https://ai.meta.com/blog/meta-llama-3

Table 4: Performance of retrievers. The relevant document for each question is assumed to be the first paragraph of the corresponding Wikipedia page. Statistically significant differences in the *DPR* row are compared with other rows. Superscripts (a), (b), and (c) indicate statistically significant differences (better or worse) compared to *BM25*, *Contriever*, and *BM25+DPR*, respectively, as determined by a t-test (p-value < 0.01).

Dataset	PopQA			WiTQA			EQ		
Model	**Rec@1**	**Rec@3**	**Rec@5**	**Rec@1**	**Rec@3**	**Rec@5**	**Rec@1**	**Rec@3**	**Rec@5**
BM25	40.13	62.92	71.30	56.40	81.88	88.87	64.01	89.00	93.71
Contriever	42.90	70.84	80.17	53.23	84.25	91.52	63.10	91.93	96.30
BM25+DPR	59.35	80.85	87.55	72.01	90.17	93.72	79.10	95.09	96.11
DPR	$59.40^{(a, b)}$	$80.85^{(a, b)}$	$87.57^{(a, b)}$	$71.73^{(a, b)}$	$90.05^{(a, b)}$	$93.62^{(a, b)}$	$79.01^{(a, b)}$	$94.06^{(a, b)}$	$96.10^{(a)}$

Table 5: Overall accuracy of answer generation using different retrievers before and after FT. The performance of LMs in answering questions correlates with the effectiveness of the retrieval models. Statistically significant differences in the *DPR* rows are compared with other rows. Superscripts (a), (b), and (c) indicate statistically significant differences (better or worse) compared to *BM25*, *Contriever*, and *BM25+DPR*, respectively, as determined by the Wilcoxon test (p-value < 0.01).

		PopQA				WiTQA				EQ			
Model		**BM25**	**Contr-iever**	**BM25+ DPR**	**DPR**	**BM25**	**Contr-iever**	**BM25+ DPR**	**DPR**	**BM25**	**Contr-iever**	**BM25+ DPR**	**DPR**
FlanT5-base	-FT	41.40	44.48	54.21	$53.36^{(a, b)}$	60.25	58.06	66.62	$66.51^{(a, b)}$	43.90	43.01	49.62	$49.54^{(a, b)}$
FlanT5-base	+FT	39.99	43.24	52.19	$52.21^{(a, b)}$	61.06	59.05	66.89	$66.83^{(a, b)}$	45.35	45.24	50.45	$50.39^{(a, b)}$
StableLM2	-FT	46.05	47.83	55.49	$55.74^{(a, b)}$	72.52	69.33	74.53	$74.47^{(a, b)}$	51.71	49.25	54.60	$54.51^{(a, b)}$
StableLM2	+FT	49.92	53.63	60.97	$61.44^{(a, b)}$	77.67	76.65	80.48	$80.45^{(a, b)}$	59.99	59.24	63.67	$61.91^{(a, b)}$
Mistral	-FT	49.58	53.10	60.14	$60.09^{(a, b)}$	78.33	77.35	81.54	$81.49^{(a, b)}$	59.09	58.03	63.13	$63.06^{(a, b)}$
Mistral	+FT	50.53	54.08	61.38	$61.23^{(a, b)}$	76.82	75.60	79.35	$79.22^{(a, b)}$	56.74	55.19	61.43	$61.28^{(a, b)}$
Llama3	-FT	51.30	56.49	61.57	$61.46^{(a, b)}$	78.51	79.00	81.62	$81.73^{(a, b)}$	59.31	59.26	63.37	$63.27^{(a, b)}$
Llama3	+FT	53.57	56.90	62.05	$62.10^{(a, b)}$	79.96	79.59	81.72	$81.77^{(a, b)}$	61.86	61.05	64.07	$64.00^{(a, b)}$

5.1 Fine Tuning vs. RAG Performance

The first research question ***(RQ1)*** involves the comparison between FT and RAG methods and factors affecting these models: (i) fine-tuning method, (ii) data augmentation method, (iii) LM type and size, and (iv) retrieval model performance.

Comparison of fine tuning methods. To study the effect of fine-tuning method, we investigate four LMs: three from the FlanT5 family (encoder-decoder models) and StableLM2 (a decoder-only model). These models are chosen as their parameters are less than 2B, allowing us to perform full fine-tuning with the resources we had at our disposal. Table 2 shows that PEFT leads to smaller gains in the *+FT-RAG* setup compared to full FT in most cases, yet it significantly improves accuracy in the *+FT+RAG* setup. This suggests that PEFT enables the LM to maintain its reasoning abilities based on the provided prompts. Based on this observation, we selected PEFT as the fine-tuning method for subsequent experiments.

Comparison of data augmentation methods. The E2E method generates over 12 times more QAs than the prompting method, while the prompt-based method generates higher-quality QAs. The results in Table 2 show that fine-tuned models trained on prompt-generated data outperform those trained on E2E-generated data. This highlights the significance of synthetic data quality over quantity. As a result, for subsequent experiments, we use QAs generated by the prompt method.

Comparison of FT and RAG with LMs of different type and size. Choosing PEFT as the FT method and the Prompt method for QA generation, we extended our experiments to 12 LMs using the ideal retriever for RAG. Table 3 presents the results. The findings indicate that while FT enhances the answer generator's performance, it alone cannot match or approach the performance of RAG. However, combining FT with RAG yields the best results for smaller models (up to 3B parameters). Conversely, for larger models (from 7B to 11.3B parameters), combining FT with RAG degrades performance. This suggests that although FT injects knowledge into LMs (as seen when comparing *-FT-RAG* with *+FT-RAG*), it diminishes the reasoning abilities of larger LMs. Interestingly, since smaller LMs' accuracy improves with FT, the best result for *+FT+RAG* is achieved by StableLM2, which has only 1.6B parameters. Another observation is that decoder-only models perform better than encoder-decoder (Flan-T5) models of similar size.

Effect of retrieval models on RAG. To assess the effect of retrieval models, we first calculate the overall Recall score for all retriever models, considering the first paragraph of each Wikipedia page as the answer-containing document. The results are shown in Table 4, demonstrating that DPR and BM25+DPR models significantly outperform BM25 and Contriever across all datasets, while DPR and BM25+DPR perform on par with each other. Table 5 presents the QA accuracy of RAG with different retrieval models. Following Mallen et al. [39], we use the top-ranked document for

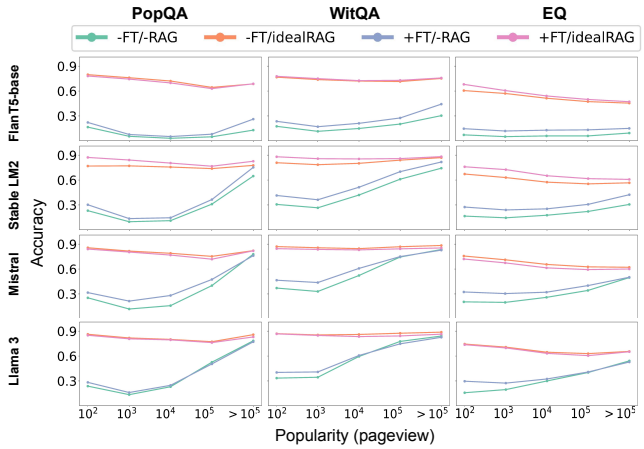

Figure 6: The performance of the LMs with different combinations of FT and RAG. The *+FT+RAG* setup outperforms other setups across all models and datasets.

the RAG prompt. The results indicate a direct correlation between the performance of the retrieval model and the overall QA accuracy, underscoring the significant impact of the retrieval model on the effectiveness of the downstream task for both vanilla and fine-tuned LMs. Since the performance of RAG with DPR and BM25+DPR is comparable, we use DPR as the retrieval model for subsequent experiments.

Analysis of RAG and FT per popularity. Figure 6 illustrates QA accuracy across different popularity buckets, providing insight into the effectiveness of FT and RAG (using Ideal retriever). It is evident that RAG significantly increases accuracy for the least popular entities, which aligns with the findings of Mallen et al. [39]. Additionally, these figures demonstrate that FT enhances QA accuracy across all popularity levels, with the most notable improvements observed in the least popular buckets for Mistral and Llama3.

Analysis of retrieval models per popularity. Figure 7 compares the performance of retrieval models against the Ideal retriever across different popularity buckets. The results indicate that retrieval effectiveness is higher for less popular entities compared to more popular entities. This is likely due to the limited occurrences of noisy documents for less popular entities.
Figure 8 shows the QA system's accuracy using various retrieval models within the RAG framework across different popularity buckets for the FlanT5-base and StableLM2 models. The left figures display results for vanilla LMs, while the right figures show results for fine-tuned LMs. FT does not alter the pattern of accuracy across popularity buckets but shifts the overall accuracy higher. Interestingly, the accuracy decreases from the less popular bucket to the fourth popularity bucket across different retrievers but increases in the most popular bucket. This reduction in accuracy can be interpreted by the finding in Figure 7, which shows that the retriever's performance decreases as popularity increases. However, it appears that for popular entities, the LMs can ignore noisy information in the input prompt and rely on their internal knowledge to answer questions, resulting in a sudden increase in accuracy despite the retriever's lower performance in the most popular bucket.

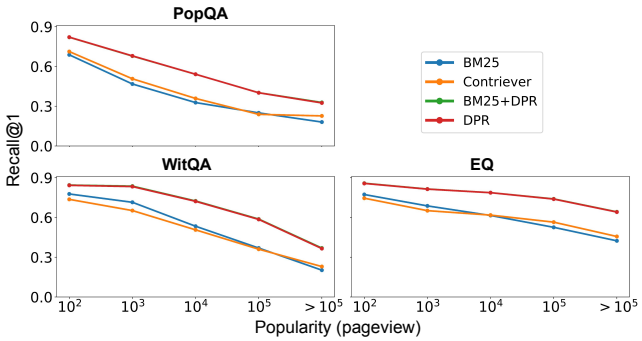

Figure 7: Recall@1 for retrieval models across different popularity levels shows that retrievers perform more effectively with less popular knowledge compared to more popular ones.

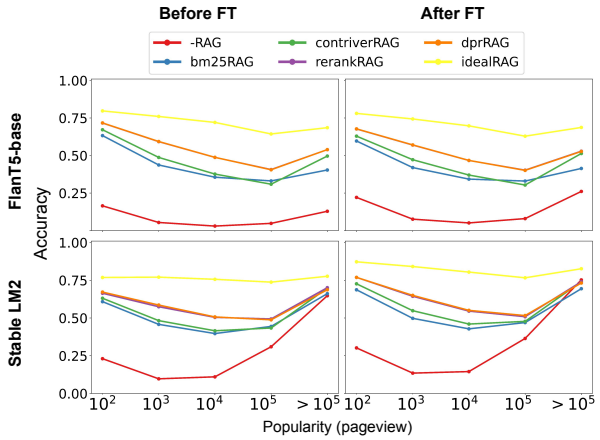

Figure 8: Performance of the answer generator task across popularity buckets on PopQA. FT does not alter the overall pattern. Accuracy decreases as the retriever models' performance drops from the least popular bucket to the fourth bucket. Interestingly, accuracy increases for the most popular bucket, indicating that LMs rely on their embedded information for popular entities.

5.2 Stimulus RAG performance

The second research question ***(RQ2)*** concerns whether our proposed Stimulus RAG method can surpass the performance of fine-tuned models. To evaluate the effect of the stimulus RAG, we first need to investigate how increasing the number of documents in the input prompt affects the accuracy of LMs. Table 6 presents the results of RAG with top-1, top-3 and top-5 documents, shown as (1D), (3D), and (5D), respectively. It shows that using top-3 documents leads to noticeable accuracy improvements in all cases, both before and after FT. For DPR, these results align with those in Table 4, where there is a significant increase from Recall@1 to Recall@3. For Ideal retriever, it is important to note that the Ideal retriever is not 100 percent accurate; for some queries, the answer is found in other paragraphs, not just in the summary paragraph.

Another notable observation is that increasing the number of input documents to five results in either negligible accuracy improvement or a decrease in accuracy. This occurs despite Table 7

Table 6: The effect of increasing the number of documents in RAG. A significant jump in accuracy is observed when the number of documents is increased to three. However, adding five documents does not significantly affect accuracy. Statistically significant differences in the *(3D)* columns are compared with the *(1D)* and *(5D)* columns. Superscripts (a) and (b) indicate statistically significant differences (better or worse) compared to *(1D)* and *(5D)*, respectively, as determined by the Wilcoxon test (p-value < 0.01).

Model		-FT+RAG			+FT+RAG		
		(1D)	(3D)	(5D)	(1D)	(3D)	(5D)
PopQA							
FlanT5-base	DPR	53.36	56.67$^{(a)}$	56.67	52.20	53.46$^{(a)}$	53.46
	Ideal	74.50	73.06$^{(b)}$	75.53	71.34	72.02	71.75
StableLM2	DPR	55.74	63.98$^{(a)}$	64.00	61.44	65.33$^{(a)}$	65.33
	Ideal	76.14	80.82$^{(a)}$	80.68	82.09	82.98	82.99
Mistral	DPR	60.09	65.22$^{(a)}$	65.22	61.23	63.63$^{(a)}$	63.63
	Ideal	80.25	81.58$^{(a)}$	81.43	78.44	80.30$^{(a)}$	80.44
Llama3	DPR	61.46	66.66$^{(a,b)}$	67.94	62.10	66.61$^{(a)}$	66.43
	Ideal	81.29	82.58$^{(a)}$	83.27	81.54	82.58	82.63
EQ							
FlanT5-base	DPR	49.54	52.39$^{(a)}$	52.05	50.39	48.90$^{(a)}$	48.02
	Ideal	53.92	58.02$^{(a)}$	58.18	57.63	57.24	57.52
StableLM2	DPR	54.51	63.51$^{(a)}$	63.75	61.91	64.67$^{(a)}$	64.52
	Ideal	60.72	69.16$^{(a)}$	69.16	68.82	70.74$^{(a)}$	71.09
Mistral	DPR	54.51	66.85$^{(a)}$	67.40	61.28	63.92$^{(a)}$	64.73
	Ideal	68.60	71.38$^{(a)}$	71.98	64.94	68.60$^{(a,b)}$	70.31
Llama3	DPR	63.27	67.76$^{(a)}$	68.41	64.00	65.56$^{(a)}$	65.45
	Ideal	68.67	71.50$^{(a)}$	72.46	68.64	71.89$^{(a)}$	71.81

Table 7: SRAG performance. By adding the extracted hint to the top of the input prompt, SRAG's performance surpasses other settings. Statistically significant differences in the *SRAG(S)* and *SRAG(D)* columns are compared with the *-FT+RAG* and *+FT+RAG* columns. Superscripts (a) and (b) denote statistically significant differences (better or worse) compared to *-FT+RAG* and *+FT+RAG*, respectively, as determined by the Wilcoxon test (p-value < 0.01).

Model		-FT+RAG	+FT+RAG	SRAG	
		(3D)	(3D)	(S)	(D)
PopQA					
FlanT5-base	DPR	56.67	53.46	**57.77**$^{(a,b)}$	57.67$^{(a,b)}$
	Ideal	73.06	72.02	75.08$^{(a,b)}$	**75.29**$^{(a,b)}$
StableLM2	DPR	63.98	65.33	65.48$^{(a)}$	**66.01**$^{(a)}$
	Ideal	80.82	82.98	82.83$^{(a)}$	**83.18**$^{(a)}$
Mistral	DPR	65.22	63.63	65.84$^{(a,b)}$	**66.04**$^{(a,b)}$
	Ideal	81.58	80.30	81.88$^{(b)}$	**82.27**$^{(a,b)}$
Llama3	DPR	66.66	66.61	**67.22**	67.21
	Ideal	**82.58**	**82.58**	82.42	81.60
EQ					
FlanT5-base	DPR	52.39	48.90	**55.31**$^{(a,b)}$	55.12$^{(a,b)}$
	Ideal	58.02	57.24	**60.21**$^{(a,b)}$	59.88$^{(a,b)}$
StableLM2	DPR	63.51	64.67	65.97$^{(a,b)}$	**66.73**$^{(a,b)}$
	Ideal	69.16	70.74	71.38$^{(a)}$	**71.73**$^{(a,b)}$
Mistral	DPR	66.85	63.92	68.54$^{(a,b)}$	**68.76**$^{(a,b)}$
	Ideal	71.38	68.60	72.23$^{(a,b)}$	**72.45**$^{(a,b)}$
Llama3	DPR	67.76	65.56	68.31$^{(b)}$	**68.38**$^{(b)}$
	Ideal	71.70	71.89	71.71	**71.98**

showing that Recall@5 for DPR is higher than Recall@3. This observation suggests that as the number of documents increases, some LMs struggle to effectively utilize all the information, making it harder to find the correct answer and leading to misunderstandings. The results indicate that adding more textual information to the input prompt should be done judiciously.

Table 7 presents the results of the Stimulus RAG with three documents. It is important to note that the hint sentence and document are derived from the top-3 documents, so no extra information is added to the input prompt. This ensures a fair comparison with RAG using three documents, allowing us to solely evaluate the effectiveness of the highlighting method. In all cases, Stimulus RAG without FT achieves higher accuracy than fine-tuned LMs with RAG using top-3 documents. This indicates that guiding LMs with a hint not only improves RAG accuracy before FT but can also surpass the effects of FT. These findings demonstrate that better accuracy can be achieved by designing a more advanced RAG system without the complexities and resource demands of FT.

6 Discussion and Conclusions

In this paper, we aimed to determine the most suitable approach for customizing language models (LMs) for less-resourced domains. We examined the effectiveness of retrieval augmented generation (RAG) and fine-tuning (FT) methods, focusing on four key aspects: (i) fine-tuning methods, specifically full fine-tuning versus parameter-efficient fine-tuning (PEFT), (ii) data augmentation techniques, (iii) the type and size of LMs, including decoder-only versus encoder-decoder models ranging from 80 million to 11 billion parameters, and (iv) the performance of retrieval models. Our findings reveal several key points. First, PEFT enhances downstream task performance and preserves the reasoning abilities of LMs while incorporating new knowledge. Second, prompt-based QA generation exhibits superior performance in factual QA tasks. Third, a small fine-tuned LM with RAG can perform on par with or even surpass a larger LM model. Additionally, RAG's performance improves with higher-performing retrievers. Notably, when comparing knowledge injection methods, RAG significantly outperforms FT. We addressed the cost of fine-tuning by developing Stimulus RAG (SRAG), a novel RAG approach that prompts an LM to generate correct responses based on hints provided in the prompt. This method eliminates the need for extensive fine-tuning, making it a cost-effective solution for enhancing LM performance in less-resourced domains.

Acknowledgments

This publication is part of the project LESSEN with project number NWA.1389.20.183 of the research program NWA ORC 2020/21 which is (partly) financed by the Dutch Research Council (NWO).

References

[1] Zahra Abbasiantaeb and Mohammad Aliannejadi. 2024. Generate then Retrieve: Conversational Response Retrieval Using LLMs as Answer and Query Generators. *CoRR* abs/2403.19302 (2024). arXiv:2403.19302

[2] Nancy E Adams. 2015. Bloom's taxonomy of cognitive learning objectives. *Journal of the Medical Library Association: JMLA* 103, 3 (2015), 152.

[3] Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA Corpora Generation with Roundtrip Consistency. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*. 6168–6173.

[4] Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. 2023. Retrieval-based Language Models and Applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts, ACL 2023*. 41–46.

[5] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *The Twelfth International Conference on Learning Representations, ICLR (2024)*.

[6] Akari Asai, Zexuan Zhong, Danqi Chen, Pang Wei Koh, Luke Zettlemoyer, Hannaneh Hajishirzi, and Wen-tau Yih. 2024. Reliable, Adaptable, and Attributable Language Models with Retrieval. *CoRR* (2024).

[7] Arian Askari, Mohammad Aliannejadi, Evangelos Kanoulas, and Suzan Verberne. 2023. A Test Collection of Synthetic Documents for Training Rankers: ChatGPT vs. Human Experts. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM (2023)*. 5311–5315.

[8] Arian Askari, Mohammad Aliannejadi, Chuan Meng, Evangelos Kanoulas, and Suzan Verberne. 2023. Expand, Highlight, Generate: RL-driven Document Generation for Passage Reranking. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP (2023)*. 10087–10099.

[9] Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, Meng Lee, Emad Mostaque, Michael Pieler, Nikhil Pinnaparaju, Paulo Rocha, Harry Saini, Hannah Teufel, Niccoló Zanichelli, and Carlos Riquelme. 2024. Stable LM 2 1.6B Technical Report. *CoRR* abs/2402.17834 (2024).

[10] Anthony Chen, Pallavi Gudipati, Shayne Longpre, Xiao Ling, and Sameer Singh. 2021. Evaluating Entity Disambiguation and the Role of Popularity in Retrieval-Based NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP (2021)*. 4472–4485.

[11] Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. 2023. Lift Yourself Up: Retrieval-augmented Text Generation with Self-Memory. In *Proceedings of the Annual Conference on Neural Information Processing Systems, (NeurIPS) 2023*.

[12] Sangwoo Cho, Kaiqiang Song, Chen Li, Dong Yu, Hassan Foroosh, and Fei Liu. 2020. Better Highlighting: Creating Sub-Sentence Summary Highlights. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP (2020)*. 6282–6300.

[13] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and et al. 2023. PaLM: Scaling Language Modeling with Pathways. *Journal of Machine Learning Research* 24 (2023), 240:1–240:113.

[14] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling Instruction-Finetuned Language Models. *Journal of Machine Learning Research* 25 (2024), 70:1–70:53.

[15] 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 (2024)*. 719–729.

[16] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *9th International Conference on Learning Representations, ICLR (2021)*.

[17] Maria Angels de Luis Balaguer, Vinamra Benara, Renato Luiz de Freitas Cunha, Roberto de M. Estevão Filho, Todd Hendry, Daniel Holstein, Jennifer Marsman, Nick Mecklenburg, Sara Malvar, Leonardo O. Nunes, Rafael Padilha, Morris Sharp, Bruno Silva, Swati Sharma, Vijay Aski, and Ranveer Chandra. 2024. RAG vs Fine-tuning: Pipelines, Tradeoffs, and a Case Study on Agriculture. abs/2401.08406 (2024).

[18] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems, NeurIPS (2023)*.

[19] Emma J Gerritse, Faegheh Hasibi, and Arjen P de Vries. 2022. Entity-Aware Transformers for Entity Search. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 1455–1465.

[20] Ameya Godbole and Robin Jia. 2023. Benchmarking Long-tail Generalization with Likelihood Splits. In *Findings of the Association for Computational Linguistics: EACL*. 933–953.

[21] Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zhen Leng Thai, Kai Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies. abs/2404.06395 (2024).

[22] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Trans. Mach. Learn. Res.* (2022).

[23] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2024*. 7036–7050.

[24] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *CoRR* (2023). arXiv:2310.06825

[25] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and Applications of Large Language Models. (2023).

[26] Chris Kamphuis, Aileen Lin, Siwen Yang, Jimmy Lin, Arjen P de Vries, and Faegheh Hasibi. 2023. MMEAD: MS MARCO Entity Annotations and Disambiguations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*. 2817–2825.

[27] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large Language Models Struggle to Learn Long-Tail Knowledge. In *International Conference on Machine Learning, ICML, (2023)*. 15696–15707.

[28] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. 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 (2020)*. 6769–6781.

[29] Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A. Smith, Yejin Choi, and Kentaro Inui. 2023. RealTime QA: What's the Answer Right Now?. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems, NeurIPS (2023)*.

[30] Patrick S. H. Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised Question Answering by Cloze Translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL (2019)*. 4896–4910.

[31] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS (2020)*.

[32] Patrick S. H. Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 Million Probably-Asked Questions and What You Can Do With Them. *Trans. Assoc. Comput. Linguistics* (2021), 1098–1115.

[33] Zekun Li, Baolin Peng, Pengcheng He, Michel Galley, Jianfeng Gao, and Xifeng Yan. 2023. Guiding Large Language Models via Directional Stimulus Prompting. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS (2023)*.

[34] Jimmy Lin, Rodrigo Frassetto Nogueira, and Andrew Yates. 2021. *Pretrained Transformers for Text Ranking: BERT and Beyond*. Morgan & Claypool Publishers.

[35] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS (2022)*.

[36] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Trans. Assoc. Comput. Linguistics* (2024), 157–173.

[37] Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-Tuning LLaMA for Multi-Stage Text Retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024*. 2421–2425.

[38] Seiji Maekawa, Hayate Iso, Sairam Gurajada, and Nikita Bhutani. 2024. Retrieval Helps or Hurts? A Deeper Dive into the Efficacy of Retrieval Augmentation to Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2024*. 5506–5521.

[39] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics ACL, (2023)*. 9802–9822.

[40] Sewon Min, Weijia Shi, Mike Lewis, Xilun Chen, Wen-tau Yih, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2023. Nonparametric Masked Language Modeling. In *Findings of the Association for Computational Linguistics: ACL 2023*. 2097–2118.

[41] Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. 2023. Few-shot Fine-tuning vs. In-context Learning: A Fair Comparison and Evaluation. In *Findings of the Association for Computational Linguistics: ACL (2023)*. 12284–12314.

[42] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Nick Barnes, and Ajmal Mian. 2023. A Comprehensive Overview of Large Language Models. abs/2307.06435 (2023).

[43] Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. 2023. Fine-Tuning or Retrieval? Comparing Knowledge Injection in LLMs. *CoRR* abs/2312.05934 (2023). arXiv:2312.05934

[44] Stephen E. Robertson and Steve Walker. 1994. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*. 232–241.

[45] Stephen E. Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (2009), 333–389.

[46] Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021. Simple Entity-Centric Questions Challenge Dense Retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*. 6138–6148.

[47] 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.

[48] Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. 2023. Data Augmentation for Conversational AI. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023*. 5220–5223.

[49] Heydar Soudani, Roxana Petcu, Evangelos Kanoulas, and Faegheh Hasibi. 2024. Data Augmentation for Conversational AI. In *Companion Proceedings of the ACM on Web Conference 2024, WWW 2024*, Tat-Seng Chua, Chong-Wah Ngo, Roy Ka-Wei Lee, Ravi Kumar, and Hady W. Lauw (Eds.). ACM, 1234–1237.

[50] Heydar Soudani, Roxana Petcu, Evangelos Kanoulas, and Faegheh Hasibi. 2024. A Survey on Recent Advances in Conversational Data Generation. *arXiv preprint arXiv:2405.13003* (2024).

[51] Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024. Head-to-Tail: How Knowledgeable are Large Language Models (LLMs)? A.K.A. Will LLMs Replace Knowledge Graphs?. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL (2024)*. 311–325.

[52] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021*.

[53] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *CoRR* abs/2307.09288 (2023). arXiv:2307.09288

[54] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct Distillation of LM Alignment. *CoRR* abs/2310.16944 (2023). arXiv:2310.16944

[55] Asahi Ushio, Fernando Alva-Manchego, and Jose Camacho-Collados. 2023. An Empirical Comparison of LM-based Question and Answer Generation Methods. In *Findings of the Association for Computational Linguistics: ACL 2023*. 14262–14272.

[56] Johannes M van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P de Vries. 2020. REL: An Entity Linker Standing on the Shoulders of Giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 2197–2200.

[57] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS (2022)*.

[58] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL (2022)*. 1–9.

[59] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. TinyLlama: An Open-Source Small Language Model. (2024).

[60] Tianjun Zhang, Shishir G. Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E. Gonzalez. 2024. RAFT: Adapting Language Model to Domain Specific RAG. (2024).