Know Your RAG: Dataset Taxonomy and Generation Strategies for Evaluating RAG Systems

Rafael Teixeira de Lima[1*], Shubham Gupta[1], Cesar Berrospi[2], Lokesh Mishra[2], Michele Dolfi[2], Peter Staar[2], Panagiotis Vagenas[2]

[1]IBM Research Paris-Saclay, 2 Rue d'Arsonval, Orsay, France;
[2]IBM Research Zurich, Säumerstrasse 4, Rüschlikon, Switzerland;
[*]**Correspondence:** rtdl@ibm.com

Abstract

Retrieval Augmented Generation (RAG) systems are a widespread application of Large Language Models (LLMs) in the industry. While many tools exist empowering developers to build their own systems, measuring their performance locally, with datasets reflective of the system's use cases, is a technological challenge. Solutions to this problem range from non-specific and cheap (most public datasets) to specific and costly (generating data from local documents). In this paper, we show that using public question and answer (Q&A) datasets to assess retrieval performance can lead to non-optimal systems design, and that common tools for RAG dataset generation can lead to unbalanced data. We propose solutions to these issues based on the characterization of RAG datasets through labels and through label-targeted data generation. Finally, we show that fine-tuned small LLMs can efficiently generate Q&A datasets. We believe that these observations are invaluable to the know-your-data step of RAG systems development.

1 Introduction

A Retrieval Augmented Generation (RAG) system pairs a Large Language Model (LLM) with an external knowledge source (Lewis et al., 2020; Guu et al., 2020). Given a user's query, a *retriever* adds relevant information from the knowledge source (context) to the LLM's context window, *augmenting* the LLM's internal knowledge, and helping it *generate* a grounded answer with fewer hallucinations (Petroni et al., 2020). This setup allows LLMs to use information like current news and private enterprise data that was not part of their training data (IBM, 2023), which has prompted rapid adoption across the community (Nakano et al., 2021; Shuster et al., 2022; Semnani et al., 2023; Nvidia, 2023).

This adoption has been accompanied by a growing interest in strategies for evaluating RAG systems. Recent works focus on evaluating the entire system on a downstream task like question-answering (Chen et al., 2017). Others separately measure the retriever's ability to fetch correct information (Karpukhin et al., 2020; Salemi and Zamani, 2024) and the generator's ability to incorporate it in the output (Liu et al., 2023; Chen et al., 2024). Tools like Ragas (Es et al., 2024), ARES (Saad-Falcon et al., 2024), and LlamaIndex (Liu, 2022) have been developed for automated LLM-assisted evaluation of RAG systems. While these approaches focus on evaluation *methods*, we take a step back in this paper and instead focus on the *data* used for evaluation (i.e., a set of (context, query, answer) triplets).

Our **first** contribution is a taxonomy for question-context pairs. We propose labels that identify different ways a user might interface with a RAG system on a given dataset. We show that popular public Q&A datasets can be heavily unbalanced with respect to these labels, and that the performance of popular retrieval strategies can differ significantly across these classes. This can lead to performance measurements that do not reflect how users would interact with a given system, depending on what types of labels are expected in practice.

Our **second** contribution is a demonstration of different strategies to produce diverse Q&A datasets from a collection of contexts. First, by employing prompt engineering and multi-step LLM querying, then by fine-tuning small LLMs. We compare these strategies to common alternatives based on single prompts to big LLMs. This model can provide an easy-to-use tool to the community for generating diverse RAG Q&A datasets without expensive queries to big LLMs[1].

We believe our proposals contribute a crucial *know-your-data* step to RAG evaluation pipelines, even in cases where private data are involved. It also provides RAG developers with strategies to

[1]We will make our model public at the time of publication.

Proceedings of the 31st International Conference on Computational Linguistics: Industry Track, pages 39–57
January 19–24, 2025.

faithfully evaluate their system's performance by building their own testing datasets.

Related work: Gao et al. (2024) provide a thorough review of strategies for developing a RAG system. Our ideas pertain to their evaluation, and are independent of such development strategies. Existing evaluation methods (Ru et al., 2024) focus on LLM-assisted metrics for checking aspects like factuality, faithfulness, groundedness, and robustness of generated answers (Es et al., 2024; Wu et al., 2024; Katranidis and Barany, 2024; Chen et al., 2024; Liu et al., 2023; Thakur et al., 2024; Adlakha et al., 2024). Our approach is complementary to these methods as an accurate measurement of these metrics needs a test dataset that is faithful to the type of questions expected in practice. Our work also relates to synthetic dataset generation methods (Long et al., 2024). Several recent approaches have used LLMs to augment (Møller et al., 2024), label (Gilardi et al., 2023; Ziems et al., 2024), and even generate entirely synthetic datasets (Eldan and Li, 2023). The RAG dataset generation feature offered by Ragas is closest to us (Ragas, 2024). It uses Evol-Instruct (Xu et al., 2023) to morph simple questions into more complex ones. However, we use a different taxonomy for generating examples and we offer a significantly cheaper fine-tuned generation model.

2 Label taxonomy

Unlike general purpose chatbots, enterprise RAG systems have a narrowly defined scope. This allows one to think about the *types* of queries a typical user may ask of the system. Below we introduce a taxonomy over such types or *labels* that can be used by practitioners across application domains. Our experiments show that this taxonomy is applicable to several commonly used RAG evaluation datasets. If needed, domain experts can also refine it for their specific needs before applying our ideas from the subsequent sections for their analysis.

RAG evaluation datasets generally comprise of (context, query, answer) triplets, where the context (a.k.a. *ground-context*) is expected to contain an answer to the associated query. The performance of the retrieval step is based on the system's capability to retrieve the ground-context given a query. Our taxonomy is designed to identify different levels of difficulty for this task. We classify (context, query) pairs based on the *nature* of answer provided by the context to the query. Table 1 describes the four classes in our taxonomy - *fact_single*, *summary*, *reasoning*, and *unanswerable* - along with an example in each case. Classes *fact_single* and *summary* require the context to explicitly provide an answer whereas *reasoning* does not. As the retrieval is done using the contents of the context, it is therefore easier to identify ground-contexts for queries from *fact_single* and *summary* classes. We demonstrate these differences in our experiments.

Queries are not accompanied by a ground-context in practice. However, a RAG developer can likely guess the type of queries expected by the system with respect to the corpus given a narrow enough scope. E.g., a RAG system for referencing specification sheets of electrical sensors would likely get more *fact_single* queries about properties like input voltage of a sensor. Similarly, a RAG system that aims to aid an HR professional might be more often used to query procedures and other types of *summary* information. A system designer would then evaluate their RAG setup on public datasets with an emphasis on its *fact_single* or *summary* performance. Yang et al. (2024) proposed a similar taxonomy based on the question alone, while ours looks at a (context, query) pair. The latter bases the label on the type of answer provided by the context to the query. This distinction makes our taxonomy more suitable for evaluating the retrieval step.

3 Public Datasets

We investigate the label composition of Q&A datasets commonly used for RAG performance evaluation. We focus on datasets that contain a well-defined ground-context to help the labelling task and to measure the retrieval performance of the system. The datasets considered are: `HotpotQA` (Yang et al., 2018), `MS MARCO` (Nguyen et al., 2016), `NaturalQ` (Kwiatkowski et al., 2019), `NewsQA` (Trischler et al., 2017), `PubMedQA` (Jin et al., 2019), and `SQuAD2` (Rajpurkar et al., 2018). We use the versions of `HotpotQA` and `MS MARCO` built to train Sentence Transformers (Reimers and Gurevych, 2019), as they contain the question-answer-ground context triplet needed for this study. Details about data processing and subsampling are mentioned in Appendix A.

4 Labelling examples using LLMs

Given the size of typical Q&A datasets, we turn to LLMs for classification. This task typically in-

Class	Description	Example context	Example query
fact_single	Answer is present in the context. It has one unit of information and cannot be partially correct.	A table of a sensor's electrical properties	What supply voltage should I use?
summary	Answer is present in the context. It has multiple units of information. Trading completeness for conciseness yields a partially correct answer.	The conclusion section of a paper	Summarize their key findings for me
reasoning	Answer is not explicitly mentioned in the context but can be inferred from it via simple reasoning	An ESG report section on a company's electricity usage	Has there been a net increase in consumption over 5 years?
unanswerable	Answer is neither present in the context nor can be inferred from it	Claims from a patent on a coffee machine	Is tomato a fruit or a vegetable?

Table 1: Proposed taxonomy for classifying (context, query) pairs based on the nature of the request.

volves describing all the labels to an LLM and prompting it to select the best match for a given example (Es et al., 2024; Chen et al., 2024). We investigate this approach using the prompt detailed in Appendix D with two LLMs - `Llama-2-70b` (Touvron et al., 2023) and `Llama-3-70b` (Meta, 2024). To obtain ground-truth labels, we employed four human annotators to label the same randomly chosen subsets of 100 question-context pairs from each of the six datasets mentioned in Section 3.

The quality of the questions analyzed varies significantly: they can be incomplete, ambiguous or completely unintelligible. This makes the labelling task difficult and can lead to disagreement between annotators. To account for this, we check the level of concordance between the annotators and their majority vote, which is used to define a single label per entry. The majority vote discards entries in which a consensus is not found, avoiding ambiguous or otherwise bad quality questions. To demonstrate the label variability per annotator, we use Fleiss' kappa (Fleiss, 1971) $\kappa(A_i, M - A_i)$ between an annotator A_i and the majority vote excluding A_i ($M - A_i$). We found values of $\kappa(A_i, M - A_i)$ ranging from 0.62 to 0.69, showing a moderate to strong agreement between the annotators and the majority. To contrast this behavior to that of an LLMs labeller, we compare the values of $\kappa(\text{LLM}, M - A_i)$ to $\kappa(A_i, M - A_i)$ for a given A_i. We find that the concordance between $M - A_i$ and `Llama-2-70b` is between 67% to 71% lower than the concordance between $M - A_i$ and A_i. `Llama-3-70b` performs better, with a concordance only 9% and 16% lower than between $M - A_i$ and A_i. We choose the `Llama-3-70b` to generate labels for the following studies, which we will reference as `LLama3` labels.

While we observed that `Llama-3-70b` tends to correctly differentiate between labels, one noted discrepancy was its preference for *fact_single* over other labels, particularly *summary*. This confusion is related to how the information requested by the question is present in its ground context: `Llama-3-70b` tends to label questions as *fact_single* even if they ask for multiple pieces of information, if these pieces are contiguous within the context. For example, given a context that describes a list of devices and their connections, the question "What devices use a USB cable?" is a *summary* question because any subset of devices would still be a correct answer, albeit incomplete. `Llama-3-70b`, however, classifies this example as *fact_single* if the list of devices is presented as a single sentence. The full confusion matrix is presented in Appendix B.

Our LLM-based labelling strategy performs zero-shot classification as the prompt only contains a description of the labels. One could also include (context, query, human label) triplets in the prompt to perform few-shot classification. However, this strategy makes the prompt longer as typical contexts contain several sentences, leading to much higher labelling costs. LLMs also have a finite context window (e.g., 8192 tokens for `Llama-3-70b`), which limits the number of triplets that can be included in the prompt, in turn limiting the accuracy

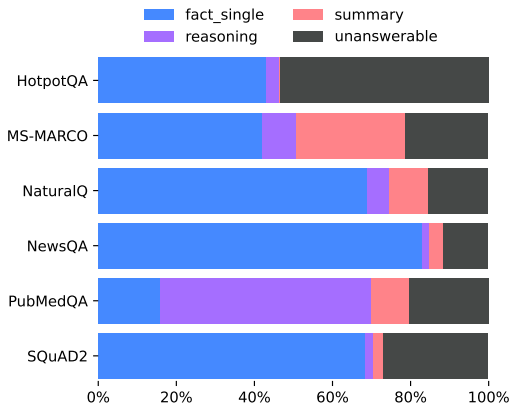

Figure 1: Composition of labels for different datasets.

of predictions. In principle, one could opt for a higher-cost LLM with a longer context window (e.g., newer versions of `Llama`), but we restrict ourselves to lower-cost zero-shot classification with `Llama-3-70b` in this paper.

5 Retrieval performance across classes

We now study the performance of the retrieval step of RAG systems as a function of the proposed labels. We focus on possible differences when tuning retrieval strategies with different dataset compositions. As a testing setup, we use Elasticsearch (Elastic.co, 2024) to store vector embeddings of the public dataset contexts, which are generated with the `bge-small-en-v1.5` model (Xiao et al., 2023). While dense vectors are highly effective for semantic search, recent applications leverage a hybrid approach, adapting the ranking score with a lexical search component (Sawarkar et al., 2024). Elasticsearch provides such a hybrid approach, which can be tuned with a text-weight parameter that varies from 0 (purely vector-based search) to 1 (fully lexical search). Tuning this parameter well is paramount for achieving an optimal performance in a deployed system. However, we show that its optimal value depends not just on the search corpus but also on the types of questions asked.

Recall that Q&As are associated with a unique ground-context in our problem setup. We characterize the performance of the retriever with the Recall@N metric (for this study, N=5). For each public dataset, we perform retrieval experiments four times: for each label individually (except *unanswerable* questions) and once inclusively for all labels. For each round, we perform text weight scans to find their optimal value. The scan is performed in the following steps: 0, 0.05, 0.1, 0.2, 0.5 and 1. The text-weight steps were chosen empirically, based on the initial results of this investigation. We name the text-weight with the highest Recall@5 the best strategy.

A summary of these experiments is presented in Table 2, which shows that the best strategy can vary not only across datasets but also across different labels within a dataset. We see relative variations in the best strategy recall from 4.8% (`NaturalQ`) to 42% (`NewsQA`) between best and worst performing labels. The highest recall is achieved with the *fact_single* label most often, while *reasoning* questions usually achieve the lowest. This is expected, as *fact_single* questions usually contain information directly mentioned in the ground contexts. On the other hand, *reasoning* questions are more difficult to find due to their answers usually being abstractions obtained from their associated contexts.

More importantly, the best strategy found by using the inclusive dataset, i.e., without any labelling, is not necessarily the same as with individual labels. For example, for `PubMedQA` the inclusive retrieval prefers a text weight of 0.05 while the *fact_single*-only retrieval prefers a dense-vector only search. On the other hand, for `MS MARCO`, the inclusive evaluation would lead to a text weight of 0.1 while the *fact_single*-only retrieval optimal text weight is 0.05. We have also tested the hypothesis that the labels influence the text weight choice with different embeddings, such as `all-MiniLM-L2-v6` (Reimers and Gurevych, 2019), and after applying re-rankers such as `bge-base` (Xiao et al., 2023). These experiments are documented in Appendix C. These findings show that the performance of RAG systems depends heavily not only on the type of data being searched but also on how the users interact with the system.

6 Generating Balanced Datasets

We now focus on strategies to synthetically generate diverse Q&A datasets for RAG performance testing. Several recipes for synthetic dataset generation are found within RAG frameworks, such as LlamaIndex (Liu, 2022) and the RAG Evaluation recipe in the Hugging Face Cookbook (Roucher, 2024), which use single prompts to generate question-answer pairs from LLMs. As a benchmark, we generated Q&A pairs with `Llama-3-70b` and the prompt suggested by the latter (also doc-

Dataset	Label	Dense	Lexical	Best recall	Best strategy
HotpotQA	Inclusive	0.906	0.904	0.942	0.10
	reasoning	0.890	0.878	*0.924 (-0.076)*	0.10
	fact_single	0.891	0.897	0.930	0.10
	summary	1.000	1.000	**1.000**	0.50
MS MARCO	Inclusive	0.752	0.719	0.804	0.10
	reasoning	0.708	0.706	*0.784 (-0.051)*	0.20
	fact_single	0.790	0.770	**0.835**	0.05
	summary	0.777	0.696	0.820	0.05
NaturalQ	Inclusive	0.686	0.464	*0.686 (-0.033)*	0.00
	reasoning	0.690	0.434	0.690	0.00
	fact_single	0.705	0.493	0.705	0.00
	summary	0.719	0.436	**0.719**	0.00
NewsQA	Inclusive	0.249	0.494	0.500	0.50
	reasoning	0.194	0.379	*0.379 (-0.161)*	1.00
	fact_single	0.262	0.533	**0.540**	0.50
	summary	0.294	0.433	0.465	0.20
PubMedQA	Inclusive	0.949	0.895	*0.935 (-0.052)*	0.05
	reasoning	0.947	0.885	0.947	0.00
	fact_single	0.987	0.952	**0.987**	0.00
	summary	0.985	0.959	0.985	0.00
SQuAD2	Inclusive	0.776	0.831	0.871	0.10
	reasoning	0.757	0.671	*0.789 (-0.104)*	0.10
	fact_single	0.818	0.852	**0.893**	0.10
	summary	0.834	0.751	0.834	0.00

Table 2: Summary of retrieval results on different Q&A labels. Embedding model used is `bge-small-en-v1.5`. The recall accuracy is measured with Recall@5.

umented in Appendix E), on the contexts found in the public datasets described in Section 3. We utilized the labeling strategy defined in Section 4 and found that 95% of generated data falls into the *fact_single* label. As illustrated by the results in Section 5, this can lead to unrealistic performance expectations when dealing with different types of questions.

Advanced techniques, such as the ones employed by Ragas (Ragas, 2024), diversify their generation by sequentially evolving a seed question according to a set of instructions (LLM prompts). While successful in generating datasets with multiple labels, this relies on several LLM queries to generate diverse Q&As. In addition to being costly, the probability of an LLM hallucination grows with each query. These hallucinations can lead to "ungrounding" Q&As from their original contexts. To avoid this, Ragas employs LLM-based critiques at every evolution step to filter out bad examples, which significantly increases the generation cost. We choose to ensure this Q&A-context grounding by inverting the usual generation process: we first build statements based on information from the context and then generate questions that are unambiguously answered by them. This strategy reduces the number of LLM queries, grounds the answers, and reduces hallucinations on the question generation by restricting it to a much smaller scope (answer instead of full context). More information on the Ragas pipeline can be found in appendix J.

Our **statement extraction generation** strategy employs the following steps. **(1)** The input context is summarized into a sentence (theme). **(2) Factual statements** are extracted from the context. For completeness, they can include contextualizing information contained in the theme. **(2.a)** To generate summary questions, we merge the multiple factual statements and the theme into three **summary statements**. **(2.b)** To generate reasoning questions, we derive three **conclusion statements** from the list of factual statements and theme. **(3)** A random

statement is chosen from either the list of factual, summary, or conclusion statements, and a question is generated that is unambiguously answered by it. The theme is once again used to aid with contextualizing information. We used `Llama-3-70b` for generation. See Appendix F for a discussion on different statement strategies and Appendix G for the relevant prompts.

6.1 Model Fine Tuning

The Q&A generation strategies previously outlined rely on querying large, state-of-the-art LLMs multiple times. Most methods also include critique steps, in which the quality of the generated dataset is judged, and bad examples are filtered out. This pipeline is costly and can be significantly inefficient if the generated Q&As are not of good quality. This cost can hinder the performance assessment of RAG systems, particularly for developers with limited access to these large LLMs. To avoid this, we investigate the fine-tuning of small LLMs to generate good quality, diverse Q&A pairs. To limit consumption at both evaluation and training time, we chose to fine-tune `Flan-T5-large` (Chung et al., 2022) with LoRA (Hu et al., 2021). Details on the fine-tuning strategy can be found in Appendix H.

Six *evaluation* trainings were performed by holding out entries from one specific public dataset at a time. After the models were fine tuned, we generated Q&As using each held-out public dataset contexts. With this method, we are able to include the impact of generalizability in the model performance by assessing each evaluation training in an independent dataset. The generation step averaged at 15 minutes for generating 2000 Q&As with a batch size of 64 running on an Apple M1 Max chip.

6.2 Synthetic Datasets Quality Comparisons

We compare the quality of generated Q&As datasets in the three described setups: with the simple prompt described in the Hugging Face Cookbook, with the statement extraction method, and with the fine-tuned model. For the first two cases, `Llama-3-70b` is used to generate the questions and build statements. As previously stated, 95% of the generated questions with simple prompt strategy are labelled as *fact_single*, therefore we consider this full dataset as being of that type.

For the statement extraction method, and the fine-tuned model generation, we find that both are able to produce diverse datasets when prompted with non-*fact_single* labels, as shown in Figure 2.

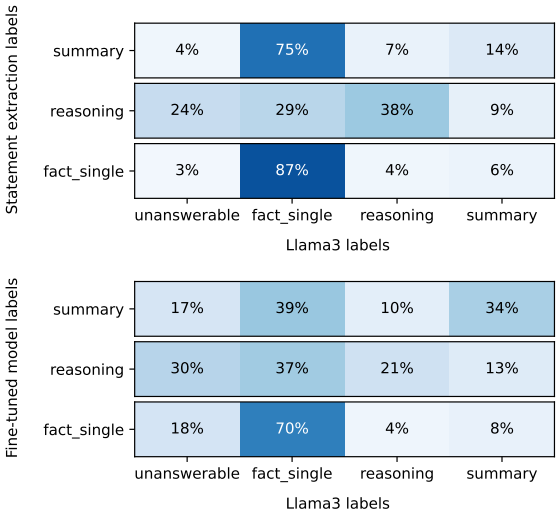

Figure 2: Distribution of `LLama3` labels for statement extraction (top) and fine-tuned model (bottom) per requested label.

Alignment between requested label at generation time and `LLama3` labels is not observed, however, potentially due to two causes: first, as previously stated, `LLama3` prefers *fact_single* over the other labels due to how the answers are present in the context. Second, not every context equally supports all question types. Some contain mostly factual information statements, for example, the introduction paragraphs of Wikipedia articles. Other contexts can be very small, without enough information to generate independent *reasoning* or *summary* statements. In addition, it is important to note that even though the fraction of *unanswerable* questions generated by the fine-tuned model is higher with respect to the statement extraction, the low cost of the former allows users to generate much bigger datasets which can then be cleaned with these labels.

After selecting Q&As with valid labels (excluding *unanswerable*), we employ LLM-based critiques to further gauge their quality. We choose to apply the following criteria, which are commonly used for this application (Liu, 2022; Roucher, 2024). **Stand Alone:** whether the question makes sense by itself or if it needs its context to be understood (e.g., questions that mention the word *context* should score low). **Question Specificity:** how specific the question is to the context (those that are too general, even if answerable by the context, should score low as they are not useful to assess RAG performance). **Question-Context Grounding:** how well the information requested can be found in the

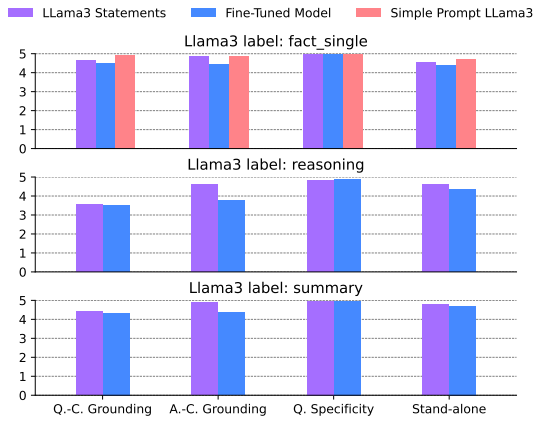

Figure 3: Average critique ratings per question label for different Q&A generation strategies, for all datasets.

context (questions that cannot be answered should score low). **Answer-Context Grounding:** how well the information contained in the actual answer can be found in the grounding context. The prompts used to perform these critiques can be found in Appendix I. The LLM used to obtain the critiques was `Llama-3-70b`.

The critique results are shown in Figure 3. First, we observe the similarly high scores of both the simple prompt and the statement extraction strategies for *fact_single* questions. This is consistent with the previous observation that these questions are usually simple statements, containing less information than other labels, thus simpler to label and generate. For the other labels, which the simple prompt is unable to generate, we also see generally high ratings for the statement extraction method. Question-context grounding ratings are slightly lower, which we believe is due to the nature of these questions: this critique is more likely to rate the question "What is the population of Paris?" (*fact_single*) higher than "What is the role of Paris in EU?" (*reasoning*), even though both are answerable with the first paragraphs of the Paris Wikipedia article, because the first question is partly contained in the text, while the second needs to be inferred.

Finally, we see good agreement between the statement extraction and the fine-tuned model, particularly for *fact_single* and *reasoning*. For *summary* questions, the low question-context grounding is consistent with the lower statement extraction rating. Here, the rate of a possible hallucination is similar because, for both cases, the question is generated by an LLM (fine-tuned or `Llama-3-70b`). On the other hand, for the answer-context grounding, the statement extraction strategy answer is less likely to be affected by hallucinations because it is based on a statement present in context. While for the fine-tuned model, the LLM needs to construct the answer from the context, conditioned on the question it just generated. We believe the fine-tuned model to be of high value: it is cheaper to generate many examples with it, even if they have to be discarded with LLM-based critiques, than to generate examples with multi-step LLM querying that also need to be filtered.

7 Conclusion

In this study, we present tools to build synthetic datasets aimed at evaluating RAG systems and strategies to characterize these datasets in terms of information request labels. We show that public Q&A datasets, and synthetic datasets generated with simple LLM prompts, can be highly unbalanced in terms of these labels, and that the retrieval performance of common RAG strategies depend on them. The combination of these two observations can lead to non-optimal design choices when building a RAG system if the type of user interactions is not reflected in the evaluation dataset. To mitigate this issue, we present strategies to generate diverse synthetic data. First, we propose a statement extraction strategy to generate grounded and labelled Q&As, and then we fine-tune a small LLM to perform the Q&A generation. Both strategies are successful in generating high quality, diverse Q&A datasets. While these strategies still require a second step of quality evaluation and cleaning, we believe they are more efficient in terms of cost and performance than current available solutions. These proposals constitute an important step in empowering RAG developers to properly evaluate and optimize their own systems. Even though our study focuses on the impact of the labeling strategy on the retrieval performance, further experiments on the response generation step may also be of interest.

References

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. 2024. Evaluating correctness and faithfulness of instruction-following models for question answering. *Transactions of the Association for Computational Linguistics*, 12:681–699.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *Proceedings of the 55th Annual*

Meeting of the Association for Computational Linguistics, 1: Long papers:1870–1879.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking large language models in retrieval-augmented generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17754–17762.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Elastic.co. 2024. Elasticsearch.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *Preprint*, arXiv:2305.07759.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated evaluation of retrieval augmented generation. *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158.

J. L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76:378–382.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. *Proceedings of the 37th International Conference on Machine Learning*, 119:3929–3938.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

IBM. 2023. What is retrieval-augmented generation? https://research.ibm.com/blog/retrieval-augmented-generation-RAG.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 2567–2577.

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. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Vasileios Katranidis and Gabor Barany. 2024. Faaf: Facts as a function for the evaluation of generated text. *Preprint*, arXiv:2403.03888.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. *Preprint*, arXiv:2405.01535.

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*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Jerry Liu. 2022. LlamaIndex.

Yi Liu, Lianzhe Huang, Shicheng Li, Sishuo Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. RECALL: A benchmark for llms robustness against external counterfactual knowledge. *arXiv*, 2311.08147.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On LLMs-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 11065–11082, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Meta. 2024. Llama-3. https://llama.meta.com/llama3/.

Anders Giovanni Møller, Jacob Aarup Dalsgaard, Arianna Pera, and Luca Maria Aiello. 2024. The parrot dilemma: Human-labeled vs. LLM-augmented data in classification tasks. *Preprint*, arXiv:2304.13861.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. WebGPT: Browser-assisted question-answering with human feedback. *arXiv*, 2112.09332.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268.

Nvidia. 2023. NVIDIA ChatRTX - your personalized ai chatbot. https://www.nvidia.com/en-us/ai-on-rtx/chatrtx/.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *Automated Knowledge Base Construction*.

Ragas. 2024. Ragas: Synthetic data generation.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2: Short Papers:784–789.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992.

Aymeric Roucher. 2024. RAG evaluation: Hugging Face cookbook.

Dongyu Ru, Lin Qiu, Xiangkun Hu, Tianhang Zhang, Peng Shi, Shuaichen Chang, Cheng Jiayang, Cunxiang Wang, Shichao Sun, Huanyu Li, Zizhao Zhang, Binjie Wang, Jiarong Jiang, Tong He, Zhiguo Wang, Pengfei Liu, Yue Zhang, and Zheng Zhang. 2024. Ragchecker: A fine-grained framework for diagnosing retrieval-augmented generation. *Preprint*, arXiv:2408.08067.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2024. ARES: An automated evaluation framework for retrieval-augmented generation systems. *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1: Long Papers:338–354.

Alireza Salemi and Hamed Zamani. 2024. Evaluating retrieval quality in retrieval-augmented generation. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2395–2400.

Kunal Sawarkar, Abhilasha Mangal, and Shivam Raj Solanki. 2024. Blended RAG: Improving RAG (retriever-augmented generation) accuracy with semantic search and hybrid query-based retrievers. In *2024 IEEE 7th International Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 155–161.

Sina Semnani, Violet Yao, Heidi Zhang, and Monica Lam. 2023. Wikichat: Stopping the hallucination of large language model chatbots by few-shot grounding on wikipedia. *Findings of the Association for Computational Linguistics: EMNLP*, pages 2387–2413.

Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, Morteza Behrooz, William Ngan, Spencer Poff, Naman Goyal, Arthur Szlam, Y-Lan Boureau, Melanie Kambadur, and Jason Weston. 2022. BlenderBot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv*, 2208.03188.

Nandan Thakur, Luiz Bonifacio, Xinyu Zhang, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Boxing Chen, Mehdi Rezagholizadeh, and Jimmy Lin. 2024. Nomiracl: Knowing when you don't know for robust multilingual retrieval-augmented generation. *Preprint*, arXiv:2312.11361.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *arXiv*, 2302.13971.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

Kevin Wu, Eric Wu, and James Zou. 2024. Clasheval: Quantifying the tug-of-war between an LLM's internal prior and external evidence. *Preprint*, arXiv:2404.10198.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *Preprint*, arXiv:2304.12244.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang,

Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen tau Yih, and Xin Luna Dong. 2024. Crag – comprehensive RAG benchmark. *Preprint*, arXiv:2406.04744.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *CoRR*, abs/1809.09600.

Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. Can large language models transform computational social science? *Computational Linguistics*, 50(1):237–291.

A Public Datasets

Table 3 lists the public datasets we use. The preprocessing for each dataset is described below. In each case, we only consider contexts with at most 10 000 characters.

SQuAD2: We use the training set of SQuAD2 (Rajpurkar et al., 2018). In this dataset, question-answer pairs are associated with specific paragraphs from Wikipedia articles. We treat each paragraph as a separate context, resulting in 19 029 unique contexts. We merge the question-answer pairs from all paragraphs into a list and randomly sample 6910 pairs along with their associated contexts for labeling.

NewsQA: This dataset consists of QA pairs, each associated with a news story (Trischler et al., 2017). We use entire news stories as individual contexts and randomly sample 6890 context-question-answer triplets for labeling.

PubMedQA: We use the unlabelled subset of this dataset that has 61 243 eligible contexts, each corresponding to the abstract of a research article (Jin et al., 2019). Each context is accompanied by a question and an answer. We retain a subset of 68 47 randomly sampled examples for labeling.

HotpotQA: The original HotpotQA dataset was designed to test the ability of QA systems to iteratively combine information across multiple contexts (**?**). Consequently, each question in this dataset is associated with several contexts. While this is an interesting use case, we are primarily concerned with the one context, one question setting. Therefore, we use the version of the dataset used in Reimers and Gurevych (2019) for training a sentence similarity model. In this version, each question is associated with a relevant *positive* context and an irrelevant *negative* context. We randomly sampled 5000 of the available 65 489 (question, positive context) pairs for labeling. Due to the nature of the dataset, most of the sampled questions cannot be answered using the single *positive* context alone. This is evident from Table 3, which shows that our labeller marks a majority of the HotpotQA questions as unanswerable.

MS MARCO: We obtained v2.1 of this dataset from Hugging Face (Nguyen et al., 2016). Each question has 10 passages (top-10 hits on Bing) associated with it. We concatenated these passages to get one context per question. We then randomly selected 5000 examples. Of these, 28 contained characters that `Llama-3-70b` was unable to process. This left us with 4972 labelled (context, question, answer) triplets.

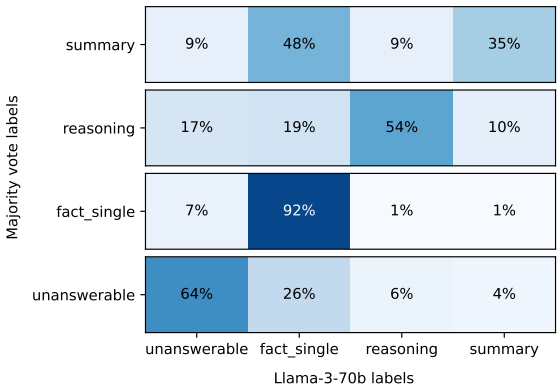

Figure 4: Confusion matrix between labels given by `Llama-3-70b` and the annotators' majority vote. Each row shows the distribution of `Llama-3-70b` labels given a majority vote label.

NaturalQ: We obtained the simplified training set of the natural questions dataset (Kwiatkowski et al., 2019). Each row contains a question along with a *long answer* comprising paragraphs from Wikipedia that contain an answer to the question. We concatenate all long answers associated with a question to get the corresponding context and then randomly sample 5000 out of 111 388 examples for labeling.

B `Llama-3-70b` confusion matrix

Figure 4 compares the labels assigned by `Llama-3-70b` to the labels selected by a majority of the human annotators.

C Retrieval Experiments

Tables 2, 4, 5 and 6 show the dependence of the retrieval performance on proposed taxonomy. We quantify the results with the use of the Recall@5 metric. Tables 2 and 4 employ the embedding model `bge-small-en-v1.5`, with the latter also re-ranking the retrieval results with the `bge-base` re-ranker. Tables 5 and 6 employ the embedding model `all-minilm-l6-v2`, with the latter also re-ranking the retrieval results with the `bge-base` re-ranker. We do not show the results for the *unanswerable* label as that is not a label of interest for our study. However, these questions are included in the "Inclusive" evaluation, which reflects the

Dataset	Contexts	Label Q&As	fact_single	reasoning	summary	unanswerable
HotpotQA	65 489	5000	42.9%	3.4%	0.2%	**53.4%**
MS MARCO	808 712	4972	**41.8%**	8.8%	27.9%	21.5%
NaturalQ	111 388	5000	**68.9%**	5.6%	9.9%	15.6%
NewsQA	89 481	6890	**83.0%**	1.8%	3.6%	11.7%
PubMedQA	61 243	6847	15.8%	**53.9%**	9.9%	20.3%
SQuAD2	19 029	6910	**68.2%**	2.2%	2.4%	27.1%

Table 3: Datasets considered in the study.

standard use of these datasets by the public. For this reason, the retrieval performance of the "Inclusive" evaluation can be lower than any of the displayed individual labels. The results for the `HotpotQA` *summary* label are statistically limited by the number of Q&As present in the analyzed dataset after labelling – we leave them in the results tables for completeness.

D Labelling Prompt

```
Consider the following context
    information and a related question.

-- Context start --

[[{context}]]

-- Context end --

-- Question start --

[[{question}]]

-- Question end --

Select the most suitable label from
    the list below:

{label_name: fact_single,
    label_description: A complete
    answer to this question is
    explicitly mentioned in the
    context and is a single simple
    value}
{label_name: summary,
    label_description: A complete
    answer to this question is
    explicitly mentioned in the
    context and is more like a summary,
     a procedure for doing something,
    or a composite of multiple parts}
{label_name: reasoning,
    label_description: A complete
    answer to this question is not
    explicitly mentioned in the
    context but can be inferred from
    the information given in it}
{label_name: unanswerable,
    label_description: A complete
    answer to this question is neither
     explicitly mentioned in the
    context nor can be inferred from
    the information given in it}

Return your response in the following
    JSON format: {"label_name": "
    selected_label_name", "reason": "
    reason_for_your_choice"}

You must select exactly one label from
     the list above. Do not select
    anything that is not in the list.
    Do not return anything other than
    the JSON format requested above.
```

E Simple Prompt

The simple prompt used in this study was obtained from the Hugging Face RAG Evaluation Cookbook (Roucher, 2024) with a small modification to generate a python dictionary. We found this generation style worked well with `Llama-3-70b` and avoided missing questions and/or answers.

```
Your task is to write a factoid
    question and an answer given a
    context.
Your factoid question should be
    answerable with a specific,
    concise piece of factual
    information from the context.
Your factoid question should be
    formulated in the same style as
    questions users could ask in a
    search engine.
This means that your factoid question
    MUST NOT mention something like "
    according to the passage" or "
    context".

Provide your answer as a JSON
    dictionary as follows:

Output:::
{{"question": "your factoid question",
     "answer": "your answer to the
    factoid question"}}

Now here is the context.

Context:
[[{context}]]
Output:::
```

Dataset	Label	Dense	Lexical	Best recall	Best strategy
HotpotQA	Inclusive	0.933	0.946	0.965	0.10
	reasoning	0.907	0.919	*0.948 (-0.052)*	0.10
	fact_single	0.926	0.938	0.954	0.10
	summary	1.000	1.000	**1.000**	0.10
MS MARCO	Inclusive	0.790	0.780	0.795	0.50
	reasoning	0.749	0.763	*0.749 (-0.093)*	0.00
	fact_single	0.825	0.822	**0.842**	0.20
	summary	0.822	0.788	0.822	0.00
NaturalQ	Inclusive	0.695	0.580	0.695	0.00
	reasoning	0.673	0.562	*0.687 (-0.056)*	0.10
	fact_single	0.723	0.614	**0.743**	0.05
	summary	0.715	0.560	0.715	0.00
NewsQA	Inclusive	0.300	0.461	0.463	0.50
	reasoning	0.210	0.315	*0.323 (-0.178)*	0.20
	fact_single	0.320	0.498	**0.501**	0.50
	summary	0.302	0.384	0.388	0.50
PubMedQA	Inclusive	0.892	0.908	0.896	0.05
	reasoning	0.887	0.907	*0.887 (-0.078)*	0.00
	fact_single	0.941	0.961	0.965	0.50
	summary	0.965	0.977	**0.965**	0.00
SQuAD2	Inclusive	0.659	0.828	0.830	0.50
	reasoning	0.618	0.757	*0.763 (-0.106)*	0.20
	fact_single	0.717	0.867	**0.869**	0.50
	summary	0.716	0.834	0.852	0.20

Table 4: Embedding model: `bge-small-en-v1.5`; Re-ranking model: `BGE-base`

F Discussion on Statements

The final generated question depends on how its source statement, i.e., answer, was generated. Factual statements focus on unitary pieces of factual information directly contained in the context. For example, parsing the first couple of sentences from the Wikipedia article on Paris, the generated factual statements would be such as "Paris is the capital of France" (which would answer the question "What is the capital of France?"), "The population of Paris is estimated to be 2,102,650 residents as of January 2023" ("What is the population of Paris?"), "The Paris Region had a GDP of 765 billion euros in 2021." ("What is the GDP of Paris?"), etc. To generate a summary statement, information is combined into composite sentences. In the previous example, a summary statement would be "Paris, the capital and largest city of France, has a population of approximately 2.1 million residents as of 2023." (which would answer the composite question "What is the capital of France and what is its population?" or the indirect question "What is the population of the capital of France?"). For conclusion statements, we ask the LLM to infer statements that are not directly included in the original factual statements list. In this case, one possible conclusion statement would be "Paris is a significant economic hub in the European Union, given its large population and high GDP.", which answers the question "What is the role of Paris in the European Union?".

The usage of themes to ground both the extracted statements and question generation comes from the observed difference between *corpus-level* questions and *document-level* questions. This differentiation is related to a broad categorization of RAG applications as corpus-level or document-level. Corpus-level RAG involves multiple documents which can include multiple themes, while document-level RAG generally contains a narrower scope. In the previous example, we could expect users to ask questions in a different manner if performing RAG over the entire Wikipedia collection of articles, as opposed to directly querying

Dataset	Label	Dense	Lexical	Best recall	Best strategy
HotpotQA	Inclusive	0.830	0.904	0.929	0.10
	reasoning	0.767	0.878	*0.907 (-0.093)*	0.10
	fact_single	0.813	0.897	0.914	0.10
	summary	0.818	1.000	**1.000**	0.10
MS MARCO	Inclusive	0.697	0.719	0.799	0.10
	reasoning	0.661	0.706	*0.781 (-0.053)*	0.20
	fact_single	0.740	0.770	**0.834**	0.10
	summary	0.711	0.696	0.801	0.05
NaturalQ	Inclusive	0.625	0.464	0.625	0.00
	reasoning	0.623	0.434	*0.623 (-0.018)*	0.00
	fact_single	0.641	0.493	**0.641**	0.00
	summary	0.640	0.436	0.640	0.00
NewsQA	Inclusive	0.186	0.494	0.496	0.50
	reasoning	0.177	0.379	*0.379 (-0.156)*	1.00
	fact_single	0.195	0.533	**0.535**	0.50
	summary	0.229	0.433	0.441	0.50
PubMedQA	Inclusive	0.886	0.895	*0.902 (-0.075)*	0.50
	reasoning	0.877	0.885	0.922	0.10
	fact_single	0.944	0.952	0.969	0.05
	summary	0.935	0.959	**0.977**	0.10
SQuAD2	Inclusive	0.773	0.831	0.878	0.10
	reasoning	0.743	0.671	*0.803 (-0.096)*	0.10
	fact_single	0.816	0.852	**0.899**	0.10
	summary	0.852	0.751	0.852	0.05

Table 5: Embedding model: `all-minilm-l6-v2`; Rerank: False

the article on Paris. In the former case, the user would more likely craft a more specific question ("What is the population *of Paris*?"), while in the latter, we can expect less specification ("What's the city's population?"). We observed that the usage of themes favored the more specific corpus-level questions, while omitting it led to less specific document-level questions.

G Statement Extraction Prompts

G.1 Theme

```
In a few words, extract the main theme
    behind the following passage: [[{
    context}]]
```

G.2 Factual statements

```
Extract at most five factual
    statements based on the following
    passage and its theme. You need to
     strictly comply with the
    following guidelines:
- Each statement must contain a single
     unit of factual information.
- Each statement must be written in
    the style of an answer to a
    factual question.
- Each statement must be
    understandable without the aid of
    any other source of information.
- Each statement must include
    contextual information derived
    from the passage theme.
- Each statement must only contain
    information that exists in the
    original passage and theme.
- Each statement must be independent
    from the other statements.

Generate the statements as a bullet
    list with the following format:
> Statement
> Statement
etc

Theme: [[{theme}]]
Passage: [[{context}]]
```

G.3 Summary statements

```
Merge the following sentences into
    three summary statements.
Each summary statement must summarise
    information contained in more than
```

Dataset	Label	Dense	Lexical	Best recall	Best strategy
HotpotQA	Inclusive	0.881	0.946	0.959	0.10
	reasoning	0.814	0.919	*0.814 (-0.186)*	0.00
	fact_single	0.868	0.938	0.868	0.00
	summary	1.000	1.000	**1.000**	0.50
MS MARCO	Inclusive	0.773	0.780	0.804	0.20
	reasoning	0.724	0.763	*0.784 (-0.051)*	0.50
	fact_single	0.814	0.822	0.814	0.00
	summary	0.810	0.788	**0.835**	0.20
NaturalQ	Inclusive	0.668	0.580	0.668	0.00
	reasoning	0.630	0.562	0.658	0.05
	fact_single	0.697	0.614	**0.722**	0.05
	summary	0.655	0.560	*0.655 (-0.067)*	0.00
NewsQA	Inclusive	0.239	0.461	0.462	0.50
	reasoning	0.218	0.315	*0.331 (-0.169)*	0.10
	fact_single	0.254	0.498	**0.500**	0.50
	summary	0.229	0.384	0.388	0.10
PubMedQA	Inclusive	0.878	0.908	*0.899 (-0.078)*	0.05
	reasoning	0.869	0.907	0.913	0.20
	fact_single	0.938	0.961	0.938	0.00
	summary	0.956	0.977	**0.977**	1.00
SQuAD2	Inclusive	0.678	0.828	0.830	0.50
	reasoning	0.625	0.757	0.757	0.50
	fact_single	0.738	0.867	*0.738 (-0.108)*	0.00
	summary	0.722	0.834	**0.846**	0.10

Table 6: Embedding model: `all-minilm-l6-v2`; Re-ranking model: `BGE-base`

```
     one sentence.
Each summary statement must be
    independent and non-overlapping.
Each summary statement should be a
    complete sentence.
Each summary statement can include
    contextual information contained
    in the theme below.
Each summary statement must be
    understandable without the aid of
    any other source of information.

Generate the statements as a bullet
    list with the following format:
> Summary statement
> Summary statement
> Summary statement

Theme: [[{theme}]]

Sentences:[[
{statements}
]]
```

G.4 Reasoning statements

```
Generate three reasoning conclusions
    that can be drawn from the
    following statements.

A reasoning conclusion is an inferred
    piece of information obtained from
     critically analysing a group of
    multiple statements.
Reasoning conclusions do not contain
    information directly contained on
    any statements.
Each conclusion must be independent
    and non-overlapping.
Each conclusion should be a complete
    sentence.
Each conclusion must be understandable
     without the aid of any other
    source of information.
Each conclusion can include contextual
     information contained in the
    theme below.

Generate the conclusions as a bullet
    list with the following format:
> conclusion
> conclusion
> conclusion
etc

Theme: [[{theme}]]

Statements:[[
{statements}
]]
```

G.5 Question

```
I have a paragraph with the following
    theme:
[[{theme}]]

From this paragraph, I extracted the
    following statement:
[[{statement}]]

Generate one question which is
    answered only by the statement
    above.
In order to avoid generic questions,
    use contextual information from
    the theme to formulate the
    question.
The question should be concise and in
    the style of a user asking
    questions to a search engine.

Generate the question as a bullet list
     with the following format:
> Question
Do not output anything else other than
     the question.
```

H Model Fine-Tuning

Our fine-tuning strategy starts with the FLAN-T5 family of models (Chung et al., 2022). We found that the `small` and `base` model sizes were not perceptive enough to extract interesting information from the contexts used, with the `large` model size being the smallest model that achieved that goal. Keeping in mind the objective of providing a low-resources strategy, we employ LoRA (Hu et al., 2021) in the fine-tuning step, training 30% of the 785M parameters in the `Flan-T5-large` model.

The fine-tuning training data contains the contexts extracted from the public datasets described in Section 3 as inputs. The outputs are the Q&As generated through the answer-first statement extraction method described previously. The final dataset contained 2000[2] context-Q&As per type, per public dataset to a total of 36k entries, from which 20% was held out for validation.

In order to allow for the generation of multiple question types with the same fine-tuned model, we add a question type flag (`<<fact_single>>`, `<<summary>>` or `<<reasoning>>`) to the beginning of each context to identify which Q&A type will be used as target. The Q&A is represented by a single string separated by the token "⟨a⟩", which is added to the T5 model tokenizer. Therefore, the fine-tuning step sees each context three times, each time with a different question type flag and a different associated Q&A. In summary, the inputs and outputs used for the fine-tuning are as follows.

```
Input: <<question_type>> Ground truth
    context
Output: <<question_type>> Statement
    extraction question <a> Statement
    extraction answer
```

I Critique Prompts

The prompts described here are adapted from (Roucher, 2024). The ratings obtained range from 1 to 5. For visualization purposes, they are scaled to range from 0 to 5.

```
In a few words, extract the main theme
     behind the following passage: [[{
    context}]]


q_to_c_groundedness:
You will be given a context and a
    sentence that should be a question.

Your task is to provide a 'total
    rating' scoring how well one can
    answer the given question
    unambiguously with the given
    context.
Give your answer on a scale of 1 to 5,
     where 1 means that the question
    is not answerable at all given the
     context, and 5 means that the
    question is clearly and
    unambiguously answerable with the
    context.
If the sentence provided is not
    actually a question, rate it as 1.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here are the question and context.

Question: "{question}"

Context: "{context}"

Answer:::


a_to_c_groundedness:
You will be given a context, and a
    passage.
```

[2]The number of training examples is limited by the generation of Q&As with the standard, multi-step methods.

```
Your task is to provide a 'total
    rating' scoring how well the
    statements in the provided passage
     can be infered from the provided
    context.
Give your rating on a scale of 1 to 5,
     where 1 means that none of the
    statements in the passage can be
    inferred from the provided context,
     while 5 means that all of the
    statements in the passage can be
    unambiguously and entirely
    obtained from the context.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here are the context and statement.


Context: "{context}"

Passage: "{answer}"

Answer:::


q_feasibility:
You will be given a context and a
    question.
This context is extracted from a
    collection of passages, and the
    question will be used to find it.
Your task is to provide a 'total
    rating' scoring how well this
    context can be retrieved based on
    the specificity and pertinence of
    the question.
Give your answer on a scale of 1 to 5,
     where 1 means that it will be
    difficult to find this context
    from this question due to lack of
    specificity or pertinence, and 5
    means that the context can clearly
     be found with information
    contained in the question.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here are the question and context.

Question: "{question}"

Context: "{context}"

Answer:::

stand_alone:
You will be given a question.
Your task is to provide a 'total
    rating' representing how context-
    independent this question is.
Give your answer on a scale of 1 to 5,
     where 1 means that the question
    depends on additional information
    to be understood, and 5 means that
     the question makes sense by
    itself.
For instance, if the question refers
    to a particular setting, like 'in
    the context' or 'in the document',
     the rating must be 1.
The questions can contain obscure
    technical nouns or acronyms and
    still be a 5: it must simply be
    clear to an operator with access
    to documentation what the question
     is about.

For instance, "What is the name of the
     checkpoint from which the ViT
    model is imported?" should receive
     a 1, since there is an implicit
    mention of a context, thus the
    question is not independent from
    the context.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here is the question.

Question: "{question}"

Answer:::

q_usefulness:
You will be given a question.
This question is to be used to find
    information in a collection of
    documents.
Your task is to provide a 'total
    rating' representing how useful
    this question can be to a user
    with domain knowledge on the
    subject covered by the document
    collection.
```

```
Give your answer on a scale of 1 to 5,
     where 1 means that the question
    is not useful at all, and 5 means
    that the question is extremely
    useful.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here is the question.

Question: "{question}"

Answer:::


c_usefulness:
You will be given a context.
This context is a part of a collection
     of contexts that users can query.
Your task is to provide a 'total
    rating' representing how useful
    this context can be to extract
    statements for a user with domain
    knowledge on the subject covered
    by the context collection.
Give your answer on a scale of 1 to 5,
     where 1 means that the context
    does not contain any useful
    statements, and 5 means that the
    context contains multiple
    statements that provide the user
    with different pieces of
    information.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here is the context.

Context::: "{context}"

Answer:::


c_clarity:
You will be given a context.
This context is a part of a collection
     of contexts that users can query.
Your task is to provide a 'total
    rating' representing the clarity
    of the information contained in
    the context.
Give your answer on a scale of 1 to 5,
     where 1 means that the context
    contains incomplete, unclear or
    poorly formatted information, and
    5 means that the context contains
    only complete, clear and well
    formatted statements.

Provide your answer as a python
    dictionary as follows:

Answer:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here is the context.

Context::: "{context}"

Answer:::



qa_tautology:
You will be given a question and
    passage its answer.
Your question is to judge whether this
     question and answer pair form a
    tautological exchange.
Give your answer on a scale of 1 to 5,
     where 1 means that the question
    and answer repeat the same
    information, and 5 means that the
    answer is made of entirely new
    information.

Provide your output as a python
    dictionary as follows:

Output:::
{{"evaluation": "Your rationale for
    the rating, as a brief and concise
     text", "rating": "your rating, as
     a number between 1 and 5"}}

You MUST provide values for '
    evaluation' and 'rating' in your
    answer. Provide ONLY the python
    dictionary as your answer.

Now here are the question and its
    answer.

Question::: "{question}"

Answer::: "{answer}"

Output:::
```

J Dataset generation using Ragas

We discussed our approach for generating RAG evaluation datasets in Section 6. Ragas offers a similar feature (Ragas, 2024) based on the Evol-Instruct framework (Xu et al., 2023). Evol-Instruct was originally developed to generate complex questions by *evolving* a set of simpler *seed* questions. For example, starting with the seed question "what is the boiling point of water?" the so called *add-constraint* evolution asks an LLM to make the question more complex by adding a constraint to it. This, for instance, would lead to an output like "what is the boiling point of water at 5 atm pressure?" Evol-Instruct defines many such evolution prompts. Ragas adapts three of them to the RAG setting - *simple*, *multi-context* and *reasoning*.

Ragas begins by generating a seed question that can be answered by the given context. There are no additional requirements on the type of this seed question. The *simple* evolution simply returns this seed question. *Multi-context* combines two contexts and generates a question that can only be answered by reading both contexts. The *reasoning* evolution is similar to our reasoning class in Table 1, and requires a logical chain of reasoning to infer an answer to the question. Users can specify the relative proportion of these three evolutions in the generated dataset.

Two differences between our taxonomy in Table 1 and Ragas' evolutions are immediately obvious. First, Ragas lacks a counterpart for our *summary* class. One might assume that the *multi-context* evolution is similar to *summary*. However, this is not true as *multi-context* evolution only requires the answer to combine information from multiple contexts. This answer need not contain multiple facts, as required by *summary*. Second, the *simple* evolution is not a pure class as per our taxonomy. This evolution just returns the seed question, which was generated without any requirement on the expected answer type. Owing to these differences, it is not possible to directly use Ragas to generate questions according to our taxonomy.

We generated 600 (context, question, answer) triplets for each public dataset in Table 3 using Ragas. In each case, we used the *simple* and *reasoning* evolutions in equal proportion. The *multi-context* evolution associates more than one context per question as explained above, and hence is outside our scope. Our first attempt at generating these examples using `Llama-3-70b` failed as a significant number of questions returned by Ragas included part of the question-generation prompt used by the library. We then switched to `Llama-2-70b` but, despite multiple attempts, this lead to unresolved `AssertionError` while generating the dataset. Eventually, we were able to successfully generate the required examples using `kaist-ai/prometheus-8x7b-v2` (Kim et al., 2024).

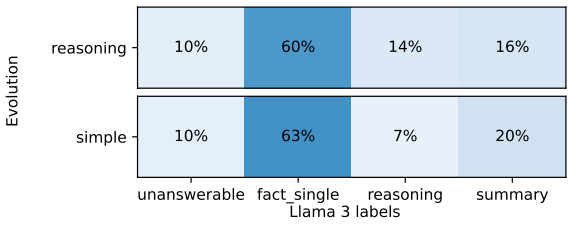

Figure 5: Distribution of `Llama-3-70b` labels for questions generated by Ragas using *simple* and *reasoning* evolutions

Figure 5 shows the result of passing the generated context-question pairs through our `Llama-3-70b`-based labeller described in Section 4. Note that *fact_single* is over-represented in the output of both evolutions. In contrast, our models generate a more significant fraction of reasoning questions when asked to do so (see Figure 2). Additionally, as expected, the *simple* evolution produces a sizable portion of both *fact_single* and *summary* questions instead of being a *pure* class with respect to our taxonomy. One can use our significantly cheaper fine-tuned model to generate a more balanced dataset than Ragas, as is evident from Figure 2.

We also critiqued the Q&A pairs generated by Ragas using our critiques and found the scores to be similar to our statement extraction method.