A Tale of Trust and Accuracy: Base vs. Instruct LLMs in RAG Systems

Florin Cuconasu[1], Giovanni Trappolini[1,2,3], Nicola Tonellotto[4], Fabrizio Silvestri[1]
[1]Sapienza University of Rome, [2]Universitas Mercatorum, [3]ISTI-CNR, [4]University of Pisa
{cuconasu, fsilvestri}@diag.uniroma1.it; giovanni.trappolini@uninercatorum.it; nicola.tonellotto@unipi.it

Abstract

Retrieval-Augmented Generation (RAG) represents a significant advancement in artificial intelligence combining a retrieval phase with a generative phase, with the latter typically being powered by Large Language Models (LLMs). Common wisdom and practices in RAG involve using "instructed" LLMs, which are fine-tuned with supervised training to enhance their ability to follow instructions and are aligned with human preferences using state-of-the-art techniques. However, contrary to this popular belief, our study demonstrates that base models outperform their instructed counterparts in RAG tasks by 20% on average under our experimental settings. This finding challenges the prevailing assumptions about the superiority of instructed LLMs in RAG applications. Further investigations reveal a more complex situation, questioning fundamental aspects of RAG and suggesting the need for broader discussions on the topic; or, as Fromm would have it, "Seldom is a glance at the statistics enough to understand the meaning of the figures".[1] [2]

1 Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) enhances Large Language Models (LLMs) by retrieving relevant information from vast corpora before generating responses. This approach improves accuracy and addresses limitations of standalone generative models like hallucinations (Huang et al., 2025) and context drift (Wang et al., 2022). As the demand for more sophisticated and context-aware AI systems grows, the ability to generate accurate and contextually relevant information becomes crucial (Gao et al., 2024). RAG achieves this by leveraging the vast amount of information available, ensuring that the outputs of the models are informed by up-to-date and contextually appropriate data. This has profound implications for various applications, including conversational AI, information retrieval, and automated content generation (Shuster et al., 2021; Wang et al., 2024).

LLMs, the key component in RAG systems, are initially pre-trained on the next token prediction task (Radford et al., 2018), where they acquire a broad understanding of language, syntax, semantics, and general knowledge. We call this the "base" version. These models typically undergo two refinement stages: supervised instruction fine-tuning (SFT) to improve instruction-following abilities (Taori et al., 2023); and alignment with human preferences through techniques like RLHF (Ouyang et al., 2024) or similar methods (Rafailov et al., 2024; Hong et al., 2024; Rahman and Xue, 2022). The resulting "instruct" versions are the go-to models for RAG tasks (Liu et al., 2024; LangChain, 2023; DSPy, 2023). Moreover, these instruct models often come with a "template": specific prompt formatting patterns with special tokens that structure the input to mark system and user instructions.

In this paper, we conduct a principled evaluation comparing instruct models and their accompanying templates against their base versions in RAG settings. Surprisingly, our results reveal that base models, without the instruction-specific fine-tuning, outperform instruct models in our experimental setting. This finding challenges the prevailing assumption that instruct models are inherently superior for these tasks. Further investigation shows that the situation is more complex, with various factors contributing to this unexpected effectiveness difference.

In summary, our contributions are: **(a)** We conduct a principled evaluation comparing instruct models against base models in RAG, revealing base models' superior performance; **(b)** Through detailed analysis, we uncover the complexities and

[1]Translated from I cosiddetti Sani.

[2]The code and data are available at github.com/florin-git/Base-vs-Instruct-LLMs-in-RAG-Systems

features that influence the effectiveness of RAG systems; **(c)** Our findings challenge existing assumptions and stimulate further discussion on RAG's state of the art, helping the development of more effective and reliable systems.

2 Preliminaries

LLM Training LLM training consists of three key steps: pre-training, instruction fine-tuning, and preference alignment.

Pre-training involves unsupervised learning on vast text corpora, where the model learns to predict the next token given a sequence of tokens $w_{1:i-1}$. The probability of generating a sequence y is $p(y) = \prod_{i=1}^{n} p_\theta(w_i|w_{1:i-1})$. This process produces a "base" model with linguistic patterns and contextual understanding.

Supervised fine-tuning (SFT) enhances the model's ability to follow instructions through training on curated datasets of instruction-response pairs (Taori et al., 2023). This transforms general language capabilities into directive-following abilities using traditional supervised learning approaches.

The final step **aligns** LLMs with human preferences, typically through Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2024), which optimizes the following formula:

$$J_r(p_\theta) = \mathbb{E}_{x,y}\left[r(x,y) - \beta \log \frac{p_\theta(y \mid x)}{p_{ref}(y \mid x)}\right]$$

Here, $r(x, y)$ represents human preferences for the input prompt x and response y; β controls regularization between the LLM p_θ and reference model p_{ref}.

RAG Retrieval-Augmented Generation (Lewis et al., 2020) combines information retrieval with generative models. Given a query q, a retriever identifies relevant documents $\{d_1, \ldots, d_k\}$ from corpus D using similarity scoring: $sim(q, d_i) \propto \overrightarrow{q} \cdot \overrightarrow{d_i}$. The LLM then generates a response based on both the query and retrieved documents:

$$p(y) = \prod_{i=1}^{n} p_\theta(w_i \mid q, d_1, \ldots, d_k, w_{1:i-1})$$

3 Experimental Methodology

In this paper, we investigate the effectiveness of base models compared to their instruction-tuned versions in the context of RAG. Our research extends to examine the underlying factors that influence RAG models' performance and the impact of additional training techniques, i.e., SFT and Alignment, on these systems. To address these research objectives, we conduct a series of rigorous experiments, providing a comprehensive analysis of the advantages and limitations of both model versions in RAG tasks.

3.1 Datasets and Models

In our experiments, we utilize three widely used open-domain question-answering datasets that are commonly employed in RAG research (Lewis et al., 2020; Gao et al., 2024). Specifically, we use the open version of Natural Questions (NQ-open) (Kwiatkowski et al., 2019; Lee et al., 2019) and TriviaQA (Joshi et al., 2017) for single-hop QA, and HotpotQA (Yang et al., 2018) for multi-hop QA. These datasets offer a diverse range of question types and complexities, allowing for a comprehensive evaluation of RAG systems. Additional information about the datasets is available in Appendix A.1. For each query, we use Contriever (Izacard et al., 2021) to retrieve the most relevant documents from the English Wikipedia corpus. The retriever's effectiveness is discussed in detail in Appendix A.2.

For the generation phase, we utilize several LLMs in both their base and instruct/chat versions: Llama 2 7B, Llama 2 13B, Llama 3 8B, Llama 3 70B, Mistral 7B, and Falcon 7B. Models are quantized to 4-bit precision due to our limited computational resources. All models utilize greedy decoding to ensure reproducibility of results. The maximum response length is tailored to the requirements of each dataset and task instruction. For more details on dataset-specific configurations, please refer to Appendix A.1.

3.2 Task Instructions

We perform our experiments with two distinct task instructions to evaluate the effectiveness of the models. Task Instruction I (Figure 1) requires the LLM to answer based on provided documents. This instruction also includes a critical component: the models are required to respond with *NO-RES* if the answer is not present in the retrieved documents. This tests the models' *negative rejection* capabilities (Chen et al., 2024), assessing their ability to recognize they lack sufficient information to provide a reliable answer. Task Instruction II extends

Llama 3 70B	Llama 3 70B-Instruct + Template
You are given a question and you MUST respond based on the provided documents. If none of the documents contain the answer, respond with NO-RES.	*<|start_header_id|>system<|end_header_id|>* You are given a question and you MUST respond based on the provided documents. If none of the documents contain the answer, respond with NO-RES.*<|eot_id|><|start_header_id|>user <|end_header_id|>*

Documents:
Document [1](Title: West Sand Lake, New York) West Sand Lake, New York West Sand Lake is a hamlet and census-designated place in Rensselaer County, New York, United States. The population was 2,845 people in 2015. The community is located in the northwest corner of the town of Sand Lake...
Document [2](Title: Averill Park, New York) Averill Park, New York Averill Park is a census-designated place in Rensselaer County, New York, United States. The population was 1,693 at the 2010 census. The Sand Lake Baptist Church was listed on the National Register of Historic Places in 2004...
Document [3](Title: Sand Lake, New York) Work upon the church edifice was begun in the fall of 1869, and the church was incorporated January 4, 1870...

Question: Sand Lake Baptist Church is located in a hamlet with a population of what at the 2010 census?

Answer: 1,693	**Answer:***<|eot_id|><|start_header_id|>assistant <|end_header_id|>* According to Document [1], the Sand Lake Baptist Church is located in West Sand Lake, which had a population of 2,845 people in 2015 (2010 census data not available).

Figure 1: Base vs. Instruct + Template under Task Instruction I on HotpotQA. The figure presents a comparison between the responses generated by two versions of the Llama 3 70B model: the base version and the instruct + template version. Both are tasked with answering the same question based on the provided documents. The base model correctly identifies the answer as "1,693", while the instruct + template version hallucinates. Notably, the instruct version erroneously states that the "2010 census data [is] not available", despite this information being present in Document [2]. *Italic* text denotes the template.

this by requiring models to also provide supporting evidence from the context, which we call a *Proof* (Figure 5 in Appendix).

For both tasks, prompts consist of the instruction, retrieved documents, and query. Documents are ordered by ascending similarity score, positioning high-similarity documents nearest to the query (Liu et al., 2024).

3.3 Instruct Templates

When fine-tuning LLMs to create their instruct versions, specific prompt templates are used during training. These templates are designed to clearly distinguish between model responses, task instructions, and user inputs, often using special tokens. For instance, Llama 3 utilizes *<|begin_of_text|>*, and Mistral uses *[INST]* to denote the beginning of instructions. While these templates are integral to the training process of instruct LLMs, their impact on RAG tasks outside conversational settings remains largely unexplored. Our study addresses this knowledge gap by evaluating instruct models with and without their standard chat templates. We aim to determine if these templates, designed for conversational AI, enhance or potentially hinder effectiveness in information retrieval tasks.

3.4 Evaluation

Accuracy is the main metric adopted to evaluate the models' responses. In particular, it is checked whether one of the ground truth answers of the dataset is contained in the generated response after applying a normalization process. This normalization involves lowercasing and the removal of punctuation and articles to ensure that the answer is not unfairly penalized by minor discrepancies in formatting.

Negative Rejection. As shown in Figure 1, LLMs are tasked to respond with *NO-RES* when documents lack the necessary knowledge to answer the query. This tests their ability to follow instructions and correctly refuse to respond when the information is not present, thereby reducing the occurrence of hallucinations (Zhang et al., 2023). We measure the *negative rejection* using the *rejection rate* (Chen et al., 2024)–the proportion of instances where the model correctly responds with *NO-RES* when answers are absent from the documents. Higher rates indicate a better ability to avoid generating incorrect or misleading answers.

4 Results

In this section, we present the results for three types of models–base, instruct, and instruct + template–evaluated under two task instructions across different datasets, as detailed in the previous section.

4.1 Evaluation on Task Instruction I

In the initial set of experiments, we evaluate the models using Task Instruction I, which is illustrated in Figure 1. Table 1 presents the accuracy scores for each model/version combination across varying numbers of retrieved documents for NQ, TriviaQA, and HotpotQA datasets. Contrary to expectations, we observe that base models consistently outperform their instruct counterparts across all three datasets, with only one partial exception. Llama 2's base model surpasses its instruct version (without template) by an average of 9.23 (+48%), 17.88 (+42%), and 4.02 (+20%) accuracy points on NQ, TriviaQA, and HotpotQA, respectively. Similarly, Falcon's base model shows improvements of 1.94 (+10%), 7.48 (+20%), and 0.89 (+5%) accuracy points on the same datasets.

Llama 3 demonstrates the most substantial gains, particularly on NQ and TriviaQA, with increases of 10.92 (+59%) and 37.5 (+186%) points. On HotpotQA, the improvement is 5.28 (+28%). This dramatic difference is partly attributable to Llama 3's reliance on its template, which we explore further in Section 4.3.

Mistral presents the only "half" exception. While its base model underperforms the instruct version by 2.49 (-8%) accuracy points on NQ, it still outperforms by 5.83 (+10%) and 1.91 (+8%) points on TriviaQA and HotpotQA.

In most cases, including more retrieved documents in the prompt leads to better effectiveness. This improvement can be attributed to the higher likelihood of incorporating relevant information, as evidenced by the increased top-k accuracy of the retriever (see Table 3 in the Appendix).

Larger Models. To further validate our findings and address potential concerns about model size influencing these results, we extended our experiments to include larger models. Table 2 presents results for Llama 2 13B and Llama 3 70B across the same datasets and retrieval settings. Notably, the trends observed with smaller models persist in these larger counterparts. Llama 2 13B's base version consistently outperforms its instruct variants across all datasets and retrieval levels. The accuracy gap is particularly evident in TriviaQA, where the base model achieves up to a 42.48 points improvement over its instruct counterpart with a template when 10 retrieved documents are included in the prompt. Llama 3 70B exhibits a similar behavior; for instance, the base model surpasses the instruct versions by margins of up to 27.39 accuracy points on TriviaQA.

These results from larger models not only corroborate our findings but also suggest that the observed phenomenon–base models outperforming instruct models–is not limited to smaller model sizes.

4.2 Evaluation on Task Instruction II

Intrigued by our initial experiments, we proceeded to examine a new task instruction designed to test the models' ability to ground their answers. In this setting, models are required to provide a *Proof*: a piece of evidence substantiating their answers based on information present in the context documents. Examples of this setup are illustrated in the appendix in Figures 5 and 6 for TriviaQA and NQ, respectively.

The results, presented in the bottom part of Table 1, reveal several insights. First, we observe a general upward shift in accuracy across all models and settings. For instance, Llama 2 base shows an average increase of 3.56 points (+12%) compared to Task Instruction I. This improvement suggests that requiring a proof acts as a form of prompt engineering, potentially encouraging more processing of the context.

Notably, the accuracy gap between base and instruct models persists: the base models continue to outperform their instruct counterparts, with the difference becoming even bigger. For example, Mistral's base model, which slightly underperformed its instruct version on NQ in Task Instruction I, now achieves higher accuracy by 3.41 points (+10%).

4.3 Instruct Models with Template

Our results reveal significant challenges faced by instruct models when using their recommended templates. The performance metrics in Table 1 clearly illustrate this phenomenon. For example, using Task Instruction I on the NQ dataset, Llama 2's templated version barely achieves a 3% accuracy rate. Even under Task Instruction II, it only surpasses 10% accuracy when the context includes more than 8 documents.

This behavior is particularly evident in the NQ dataset, which requires short answers. Despite the

Table 1: Task Instruction I and II Accuracy across document levels (*#Docs*) and LLM configurations: B (*Base*), I (*Instruct*), I + T (*Instruct with Template*). Results show base models generally outperform instruct counterparts by a considerable margin, with Mistral on Task Instruction I being a partial exception. Values *not* marked with an asterisk (*) indicate statistically significant differences between base and instruct models (Wilcoxon test, p-value < 0.01).

Task Instruction I

Dataset	#Docs	Llama 2 7B			Llama 3 8B			Mistral 7B			Falcon 7B	
		B	I	I + T	B	I	I + T	B	I	I + T	B	I
NQ	1	**23.88**	16.06	3.36	**27.03**	8.52	14.40	**24.26**	20.04	18.17	**17.13**	15.68
	2	**24.71**	18.48	1.21	**30.22**	10.25	17.34	**25.30***	24.99	23.54	**18.97**	17.72
	3	**27.83**	18.62	0.69	**30.53**	15.40	19.04	25.72	**26.69***	19.14	**21.15**	17.96
	4	**29.53**	18.59	0.48	**31.08**	15.85	15.30	26.17	**30.56**	27.90	**21.08**	19.21
	5	**30.22**	19.21	0.45	**29.08**	22.57	18.10	27.66	**31.67**	27.45	**21.95**	20.08
	8	**31.01**	21.98	0.73	**29.49***	28.80	20.35	26.65	**33.33**	27.07	**22.64**	20.04
	10	**31.46**	21.08	1.52	**28.70***	28.25	19.35	28.87	**34.82**	26.79	**22.33**	20.98
TriviaQA	1	**55.85**	32.59	23.35	**44.64**	4.13	16.45	**58.67**	48.85	45.88	**41.64**	33.19
	2	**57.15**	34.63	21.63	**53.48**	2.62	27.94	**59.15**	52.31	50.22	**43.11**	36.52
	3	**59.28**	41.09	21.13	**56.78**	3.47	31.32	**58.87**	54.72	52.42	**43.69**	36.46
	4	**60.40**	42.95	17.70	**59.59**	4.44	30.80	**60.49**	55.90	53.73	**44.22**	37.71
	5	**61.24**	46.14	18.05	**58.97**	15.73	34.18	**62.02**	56.97	54.37	**45.60**	38.42
	8	**62.93**	49.19	22.30	**64.93**	51.90	44.04	**64.16**	59.06	57.56	**46.35**	38.83
	10	**63.89**	48.96	25.68	**65.05**	58.61	50.12	**65.92**	60.60	58.31	**48.24**	39.33
HotpotQA	1	**21.89**	16.16	15.75	16.12	15.36	**18.36**	**23.73**	20.64	16.27	**16.43**	14.88
	2	**23.04**	17.96	13.39	**21.29**	15.93	14.77	**24.84**	22.61	16.43	**17.27**	15.66
	3	**23.52**	19.52	14.66	**23.18**	15.77	14.32	**24.39**	22.71	16.57	**17.00***	16.39
	4	**24.71**	20.57	16.27	**25.45**	16.52	16.66	**25.45**	23.89	17.48	**17.43**	16.27
	5	**24.45**	20.43	16.77	**26.30**	18.89	17.57	**26.00**	24.43	19.36	**18.05***	17.07
	8	**25.20**	22.30	17.82	**28.66**	25.07	20.05	**27.41**	25.62	21.02	**18.50***	18.06
	10	**26.09**	23.79	18.91	**28.16**	24.66	19.11	**27.98**	26.48	20.79	18.81	**18.90***

Task Instruction II

Dataset	#Docs	Llama 2 7B			Llama 3 8B			Mistral 7B			Falcon 7B	
		B	I	I + T	B	I	I + T	B	I	I + T	B	I
NQ	1	**24.82**	18.41	1.59	**29.39**	18.83	11.70	**24.82**	21.53	16.86	**17.58**	16.10
	2	**28.70**	24.96	2.42	**31.53**	23.57	16.44	**30.74**	27.66	18.10	**20.04**	18.31
	3	**31.71**	28.76	3.88	**34.72**	25.65	6.40	**33.75**	29.91	20.87	**22.43**	18.93
	4	**32.85**	30.22	5.75	**37.07**	27.97	6.61	**35.17**	32.85	22.43	**23.75**	19.87
	5	**34.09**	32.16	8.45	**36.59**	30.84	11.60	**36.83**	33.58	24.09	**23.85**	20.91
	8	**35.62**	33.16	12.81	**39.15**	34.13	5.64	**40.15**	36.45	24.44	**26.12**	21.61
	10	**35.79**	32.05	12.91	**40.22**	37.97	0.69	**40.15**	35.76	26.96	**26.72**	21.30
TriviaQA	1	**54.94**	41.86	4.22	**61.57**	42.67	30.52	**57.98**	48.51	33.73	**40.32**	33.54
	2	**56.94**	47.18	6.55	**63.32**	44.44	36.89	**60.49**	52.60	33.21	**42.76**	35.69
	3	**58.88**	49.53	8.75	**64.56**	47.11	31.11	**62.18**	54.79	35.47	**44.92**	37.50
	4	**60.07**	51.19	10.15	**65.93**	48.10	33.17	**63.70**	56.05	36.57	**45.45**	38.09
	5	**60.87**	52.59	14.90	**66.52**	48.71	38.08	**65.23**	56.91	34.98	**46.47**	39.02
	8	**62.75**	55.73	20.69	**67.67**	52.59	33.69	**67.04**	58.80	42.39	**47.23**	41.08
	10	**63.73**	56.00	27.48	**68.04**	56.48	23.42	**67.44**	59.54	48.45	**48.57**	42.95

task instruction for NQ specifying brevity, templated models often generate verbose outputs. This tendency may be linked to their fine-tuning and alignment for conversational purposes, where verbosity can be advantageous to assist users. These observations suggest that templates, designed to enhance conversational abilities, may not be optimally suited for all RAG tasks, especially those requiring concise responses.

An additional notable aspect is observed with Llama 3's instruct on Task Instruction I. With fewer than 4 retrieved documents, the non-templated ver-

Table 2: Task Instruction I Accuracy for larger LLMs across different retrieved document levels (*#Docs*) and configurations: B (*Base*), I (*Instruct*), I + T (*Instruct with Template*). Results show base models generally outperform their instruct counterparts, reinforcing findings from smaller models. Values *not* marked with an asterisk (*) denote statistically significant differences between base and instruct models (Wilcoxon test, p-value < 0.01).

Dataset	# Docs	Llama 2 13B			Llama 3 70B		
		B	I	I + T	B	I	I + T
NQ	1	**27.83**	21.74	0.21	**29.46**	19.04	19.00
	2	**29.08**	26.51	0.35	**33.16**	24.78	24.40
	3	**30.70**	29.08	0.35	**35.41**	29.15	28.59
	4	**32.40**	30.63	0.45	**37.35**	32.16	31.46
	5	**34.13**	29.66	0.66	**38.53**	33.82	32.68
	8	**35.13**	30.60	0.69	**40.57**	37.24	35.69
	10	**35.48**	33.44	0.97	**41.05**	39.60	37.24
TriviaQA	1	**61.45**	46.36	12.03	**64.64**	37.25	26.22
	2	**62.51**	51.43	23.03	**63.86**	42.71	32.39
	3	**64.36**	53.95	22.42	**63.48**	47.14	31.01
	4	**64.86**	54.95	24.22	**65.03**	50.51	32.94
	5	**66.30**	56.41	21.87	**66.44**	53.77	37.71
	8	**68.01**	59.91	21.27	**69.23**	60.24	46.50
	10	**68.61**	62.74	26.13	**69.93**	62.90	52.68
HotpotQA	1	**23.52**	20.73	15.25	**21.89**	14.11	12.04
	2	23.25	**23.95***	12.95	**22.84**	15.05	13.04
	3	25.15	**25.38***	13.51	**22.89**	15.84	14.11
	4	**26.57***	26.43	14.54	**25.21**	17.36	14.55
	5	**27.05***	26.93	14.29	**25.96**	19.07	15.52
	8	**28.48***	28.20	17.25	**28.88**	24.25	18.48
	10	**29.43***	28.19	18.93	**29.82**	26.93	20.29

sion struggles to interpret the prompt correctly, often producing random text. However, as the number of documents increases, its accuracy improves dramatically, eventually outperforming its templated counterpart. This observation indicates that larger input lengths might play a role in overriding learned behaviors, allowing the model to better adapt to the task at hand.

5 Is Accuracy Sufficient?

Section 4 clearly indicates that base models outperform instruct models on RAG. But is that really the case? *Are base models truly better than the instruct counterpart on RAG-like prompts?* To answer this question, in this section, we go more in-depth in analyzing and comparing their behavior. First, we test the ability of these models to adhere to the task instructions. In particular, we examine whether they appropriately respond with *NO-RES* when no relevant answer is present in the retrieved documents, hence analyzing their negative rejection capabilities (Chen et al., 2024).

5.1 Negative Rejection

Figure 2 illustrates the rejection rates for various models and their configurations–base, instruct, and instruct + template–on the TriviaQA dataset. Analogous analyses can be performed on the rates for NQ (Figure 9) and HotpotQA (Figure 10).

Across all models, we observe a general failure to consistently respond with *NO-RES* when the answer is absent from the retrieved documents. This is especially true for base models where their rejection rates are often close to 0.

Among the instruct models, Llama 2 and Mistral exhibit similar trends. For both, the highest rejection rates are observed when there is only one document in the context. In this scenario, Llama 2 responds with *NO-RES* only 30.23% of the time, while Mistral does so only 20.12% of the time when using the template. As the number of documents in the context increases, both models show a decreasing tendency to answer with *NO-RES*. This suggests that a higher number of documents might introduce more distracting information (Shi et al., 2023), leading the LLM to respond but erroneously, and that a larger input length might override task instructions.

Llama 3 exhibits a distinct trend, where its instruct versions appear more consistent in their rejection rates, maintaining a mean rate of 34.0 with the template, and 35.57 without. However, when the model is not using the template, the rejection rates for Llama 3 decline with an increase in document count, similar to Llama 2 and Mistral. Indeed, it shows a significant reduction of 14.86 (-50%) passing from 29.44 to 14.58 when the number of retrieved documents increases from 5 to 8.

Falcon models show the least tendency to respond with *NO-RES*, where rejection rates are consistently low or even zero in some configurations. This behavior indicates a propensity to generate answers even when the information is not present, potentially leading to higher rates of hallucination.

5.2 Recall From Parametric Memory

Next, we consider cases where the correct answer is not present in the provided documents, yet the model still responds accurately. As illustrated in the left part of Figure 3, base models frequently manage to provide the correct answer even when it is not in the retrieved documents, suggesting that they "know" the answer from prior training. We call this "recall from parametric memory" (by parametric memory, we mean knowledge learned during training and stored in the parameters of the model, as opposed to non-parametric memory provided in the context through retrieved documents).

Recall from parametric memory is not inherently

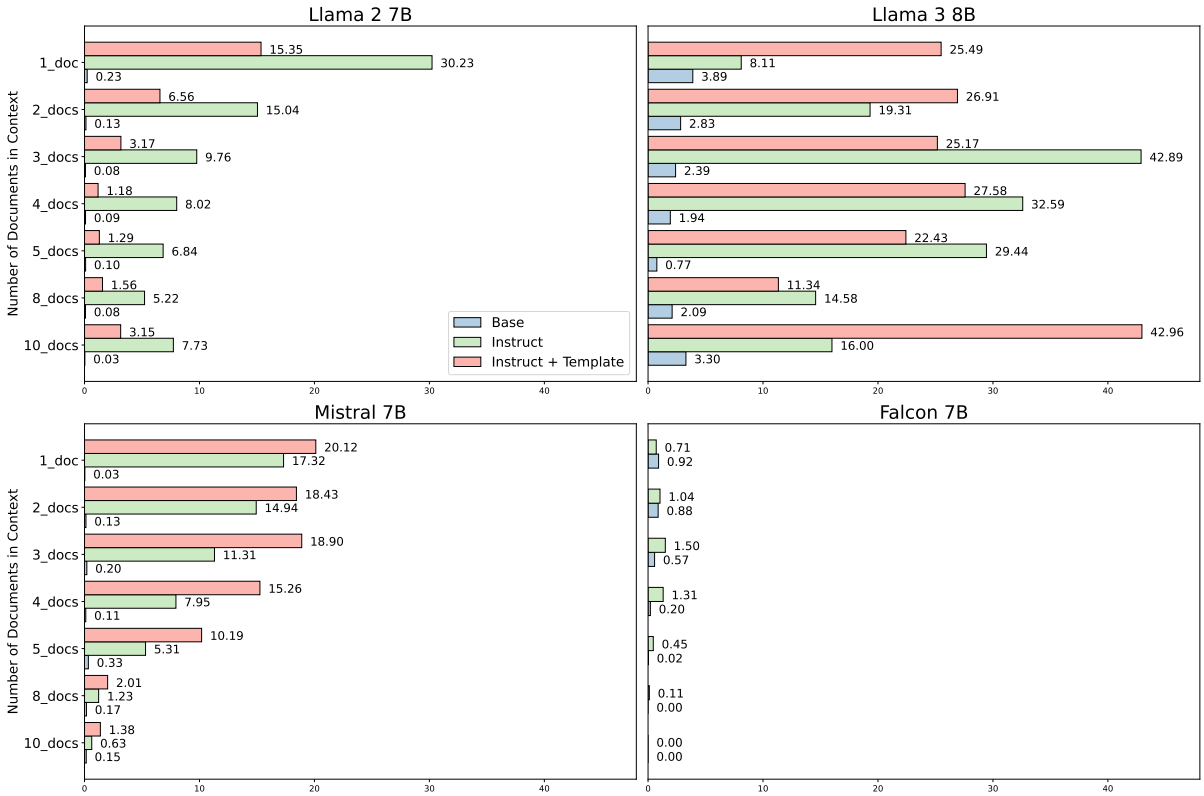

Figure 2: Reported is the **rejection rate** on TriviaQA, defined as the number of times the model responds with *NO-RES* when the correct answer is not in the context, divided by the number of times the answer is indeed missing. Instruct models are much more effective at detecting such cases and following the instructions provided.

Figure 3: **Recall from parametric memory rate** for Llama 2 7B on TriviaQA, defined as the proportion of correct answers when information is absent from retrieved documents. *Left*: Task Instruction I (Figure 1); *Right*: No Rejection setting without *NO-RES* instruction (Figure 7). In the latter case, parametric memory recall increases for both base and instruct models.

problematic. A user might choose to both fine-tune on proprietary data and use RAG to achieve the highest possible accuracy. However, the specific instructions for this study emphasize that models should opt not to answer when the correct response is not evident in the documents. Not following

this guideline raises important questions about the models' reliance on internal knowledge versus contextual information, particularly in settings where accurate rejection of unanswerable questions is crucial.

5.3 Evaluation with No Rejection

To investigate whether base models perform better simply due to non-adherence to instructions, we conduct further experiments by removing the requirement to respond with *NO-RES* when the answer is not present in the provided documents.

As shown in the right part of Figure 3, the removal of this requirement leads to notable changes in model behavior. Both base and instruct models demonstrate a higher tendency to rely on their parametric memory when the rejection constraint is removed, evidenced by the increased rates across all document counts for both model versions.

Interestingly, the instruct version with template of Llama 2 7B shows a more dramatic increase in recall from parametric memory when the rejection constraint is removed. For instance, with 4 documents, the rate nearly triples from 3.88 to 11.27. This trend is also consistently observed across other models (Llama 3, Mistral, and Falcon) and datasets NQ (Figure 11), TriviaQA (Figure 12), and HotpotQA (Figure 13). The enhanced ability to recall information translates into improved accuracy scores for instruct models, as shown in Table 5 (Appendix). However, base models still generally outperform their instruct counterparts.

These findings reveal a tradeoff in the RAG paradigm between model trustworthiness and effectiveness. The processes of supervised fine-tuning and alignment, while enhancing the model's reliability and adherence to desired behaviors, appear to detrimentally impact its capabilities in RAG tasks.

6 Related Works

Recent studies have highlighted challenges and potential improvements in language models' use of non-parametric versus parametric memory in question-answering tasks. Several papers (Krishna et al., 2021; Shi et al., 2024; Carlini et al., 2019; Kandpal et al., 2023; Mallen et al., 2023) demonstrate that LMs often rely on memorized answers, capable of responding correctly even when presented with irrelevant documents. Similarly, other studies (Longpre et al., 2021; Xie et al., 2024) observe that LMs continue to leverage their parametric knowledge despite prompt modifications with contrasting entities. Wu et al. (2024) describes this phenomenon as a balance between the model's inherent knowledge and its adherence to newly retrieved information, underscoring the ongoing challenge of enhancing model responsiveness to dynamic inputs.

On enhancing reliance on provided content, Zhang et al. (2024) introduced a training strategy that emphasizes evidence-based responses, similar to the *Proof* mechanism in our QA tasks. This method has shown potential in improving model effectiveness by grounding responses in factual evidence, even though hallucination issues still remain an open problem (Zuccon et al., 2023; Gao et al., 2023). Cuconasu et al. (2024) found instruct models to be slightly more effective, but theirs was a controlled setting in which the ground truth was always provided. These insights collectively underline the intricate balance between leveraging learned knowledge and external data in improving QA systems, suggesting directions for future research in training strategies and model design.

7 Conclusion

In this paper, we aimed to systematically investigate the differences between LLM's base and instruct versions when used in RAG systems. Our findings reveal an unexpected outcome: base models exhibit superior effectiveness on RAG tasks compared to their instructed and aligned counterparts. However, further analysis reveals a more complex situation, with a tradeoff between the base models' higher accuracy and the instruct versions' higher fidelity. This tradeoff calls for novel evaluation methodologies for RAG pipelines and suggests the necessity for mechanisms that afford users greater control in managing this tradeoff in a more direct and explicit manner.

8 Limitations

Our analysis could benefit from incorporating a broader range of datasets, particularly those that do not rely on Wikipedia. Moreoever, all datasets used in our experiments are in English, and it remains an open question whether our findings extend to other languages. Finally, our study does not cover the latest reasoning-oriented models, such as DeepSeek-R1 (Guo et al., 2025), which may exhibit different behavior in RAG settings due to their extended reasoning capabilities.

Acknowledgments

This work was carried out while Florin Cuconasu was enrolled in the Italian National Doctorate on Artificial Intelligence run by the Sapienza University of Rome. This project was also supported by PNRR MUR project PE0000013-FAIR and partially by the FoReLab and CrossLab projects (Departments of Excellence), and the NVIDIA Academic Grants Program 2026.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore. Association for Computational Linguistics.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models. *Preprint*, arXiv:2311.16867.

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*.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 267–284, USA. USENIX Association.

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.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The power of noise: Redefining retrieval for rag systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 719–729, New York, NY, USA. Association for Computing Machinery.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library. *Preprint*, arXiv:2401.08281.

DSPy. 2023. Dspy: Dynamic structured programming for python.

Abhimanyu Dubey and Abhinav Jauhri et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore. Association for Computational Linguistics.

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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. ORPO: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189, Miami, Florida, USA. Association for Computational Linguistics.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2).

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

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. *Preprint*, arXiv:2310.06825.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957, Online. Association for Computational Linguistics.

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.

LangChain. 2023. Langchain: Building applications with llms through composability.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096, Florence. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

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. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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 (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2024. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data only. In *Advances in Neural Information Processing Systems*, volume 36, pages 79155–79172. Curran Associates, Inc.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: your language model is secretly a reward model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Md Masudur Rahman and Yexiang Xue. 2022. Robust policy optimization in deep reinforcement learning. *Preprint*, arXiv:2212.07536.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *Preprint*, arXiv:1911.02150.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2024. Detecting pretraining data from large language models. In *The Twelfth International Conference on Learning Representations*.

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*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Hugo Touvron, Louis Martin, and Kevin Stone et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Hongru Wang, Wenyu Huang, Yang Deng, Rui Wang, Zezhong Wang, Yufei Wang, Fei Mi, Jeff Z Pan, and Kam-Fai Wong. 2024. Unims-rag: A unified multi-source retrieval-augmented generation for personalized dialogue systems. *arXiv preprint arXiv:2401.13256*.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2022. Measure and improve robustness in NLP models: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4569–4586, Seattle, United States. Association for Computational Linguistics.

Kevin Wu, Eric Wu, and James Zou. 2024. Clasheval: Quantifying the tug-of-war between an llm's internal prior and external evidence. In *Advances in Neural Information Processing Systems*, volume 37, pages 33402–33422. Curran Associates, Inc.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2024. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *The Twelfth International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

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. In *First Conference on Language Modeling*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023. Siren's song in the ai ocean: A survey on hallucination in large language models. *Preprint*, arXiv:2309.01219.

Guido Zuccon, Bevan Koopman, and Razia Shaik. 2023. Chatgpt hallucinates when attributing answers. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP '23, page 46–51, New York, NY, USA. Association for Computing Machinery.

A More Details on Datasets and Models

A.1 Datasets

Our evaluations employ three open-domain question-answering datasets: NQ-open, TriviaQA-unfiltered, and HotpotQA. For NQ-open, we adopt the processing procedure of Cuconasu et al. (2024), resulting in 2,889 test examples. The TriviaQA-unfiltered dataset follows the validation and test split used in previous studies (Min et al., 2019; Asai et al., 2024), with 11,313 test queries for evaluation. For HotpotQA, we utilize the open-domain version from the KILT benchmark (Petroni et al., 2021), consisting of 5,600 test samples.

NQ and TriviaQA datasets use the English Wikipedia dated 20 December 2018 as a knowledge source, while HotpotQA draws from the Wikipedia dump of August 2019. Following the Dense Passage Retrieval (DPR) approach (Karpukhin et al., 2020), we split each Wikipedia article into non-overlapping passages of 100 words.

All datasets feature questions allowing multiple valid answers, ranging from synonymous terms like "New York" and "NY" to entirely distinct correct responses. TriviaQA is notable for its diversity in acceptable responses, including many questions allowing multiple valid answers. This variety significantly reduces the likelihood of correct responses being marked as incorrect due to phrasing variations, contributing to higher accuracy scores on TriviaQA compared to the other datasets, as shown in Table 1.

For the generation phase, we tailor the maximum response length of the LLMs to each dataset's requirements. Under Task Instruction I, we set the response limit to 15 tokens for the NQ dataset, which demands short responses of no more than 5 tokens (as specified in the NQ task instruction shown in Figure 6), and up to 50 tokens for TriviaQA and HotpotQA to accommodate potentially longer answers. For Task Instruction II, which requires a *Proof*, we increase the maximum response length to 200 tokens across all datasets.

A.2 Retriever

The retriever used to select the top-k documents is Contriever (Izacard et al., 2021), which is a BERT-based dense model trained using unsupervised contrastive loss. The embedding of each document and query is obtained by averaging the hidden state of the last layer of the model. For document retrieval from the corpus, we employ a FAISS index (Douze et al., 2024; Johnson et al., 2019) by using an inner product similarity metric (IndexFlatIP) with an exhaustive search.

To assess the availability of an answer within the provided documents, we compute the top-k accuracy of the retriever. This metric evaluates the number of times the ground truth answer appears within the top-k documents retrieved for a query. Scores can be seen in Table 3.

A.3 Generative Models

We utilize publicly available, open-weight LLMs accessible via Hugging Face. All models are quantized to 4-bit using the bitsandbytes library[3] to optimize computational efficiency. We perform all experiments with a single Nvidia RTXA6000 GPU.

Here is a brief description of the main characteristics of each model:

Llama 2. The Llama 2 family (Touvron et al., 2023) is pre-trained on publicly available data and optimized for a range of natural language generation tasks. This series features a context length of 4096 tokens, and the 7B[4] and 13B[5] versions employ multi-query attention (MQA) (Shazeer, 2019) to enhance processing efficiency and response quality.

Llama 3. The Llama 3 series (Dubey and et al., 2024) builds on the architecture and improvements of its predecessors, offering models with 8B[6] and 70B[7] parameters. It employs group-query attention (GQA) (Ainslie et al., 2023) and extends the context length to 8192 tokens, thus facilitating enhanced language generation across a broad range of tasks.

Mistral. Developed as a highly efficient model with 7B parameters[8], Mistral (Jiang et al., 2023) focuses on delivering high performance and accuracy in text generation. It uses GQA and sliding window attention with an 8192-token context length.

[3] https://huggingface.co/docs/bitsandbytes/main/en/index

[4] https://huggingface.co/meta-llama/Llama-2-7b-hf https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[5] https://huggingface.co/meta-llama/Llama-2-13b-hf https://huggingface.co/meta-llama/Llama-2-13b-chat-hf

[6] https://huggingface.co/meta-llama/Meta-Llama-3-8B https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[7] https://huggingface.co/meta-llama/Meta-Llama-3-70B https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct

[8] hhttps://huggingface.co/mistralai/Mistral-7B-v0.1 https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1

Table 3: Contriever top-k accuracy.

Dataset	# Retrieved Documents						
	1	2	3	4	5	8	10
NQ	25.02	35.69	42.89	47.84	51.33	57.84	60.85
TriviaQA	39.15	50.70	56.45	60.32	63.03	68.35	70.49
HotpotQA	20.95	25.48	28.14	30.48	32.27	36.14	37.91

Table 4: Closed-book Accuracy of the models. The task instruction used in this case for all datasets can be seen in Figure 8.

Dataset	Llama 2 7B			Llama 2 13B			Llama 3 8B			Llama 3 70B			Mistral 7B			Falcon 7B	
	B	I	I + T	B	I	I + T	B	I	I + T	B	I	I + T	B	I	I + T	B	I
NQ	**25.20**	16.51	13.12	**29.91**	20.84	9.55	22.01	27.17	**27.80**	35.24	**40.01**	39.67	**28.11**	18.76	15.96	**15.26**	12.74
TriviaQA	**55.57**	37.33	41.73	**59.90**	41.61	30.94	**57.37**	25.46	54.87	71.46	**75.18**	75.12	**60.68**	47.18	49.30	**41.12**	27.83
HotpotQA	**20.70**	14.61	16.98	**22.96**	13.82	16.73	21.39	11.75	**22.09**	23.27	32.16	**33.45**	**22.30**	20.88	22.05	**16.84**	14.66

Falcon. The Falcon 7B[9] is the smallest model of the Falcon series (Almazrouei et al., 2023) and was trained on the RefinedWeb dataset (Penedo et al., 2023)–a large, filtered, and deduplicated corpus. Similarly to Llama2 7B, it uses MQA, but with a smaller context length of 2048 tokens. Unlike the other models, the instruct version of Falcon 7B was not specifically trained using a fixed template, which is why no separate "instruct + template" variant is listed in any figure or table.

The closed-book accuracy of the models is detailed in Table 4. In this scenario, models are evaluated without any documents in their prompt, necessitating a modification to Task Instruction I. An example of this modified task instruction can be viewed in Figure 8.

B Further Analysis with Task Instruction II

In this section, we examine more in detail whether models can justify their answers with a *Proof*. We specifically investigate whether even the base models can adhere to instructions and provide accurate *Proof* for their responses.

Table 6 presents the "coherence" rate, defined as the percentage of instances where the generated answer, regardless of its correctness, is included in the generated *Proof*. This measure indicates how well the model's responses align with the information provided in the documents. However, it is important to note that coherence does not necessarily reflect answer correctness, as it only assesses alignment with the document evidence.

As observed in Table 6, base models are "coherent" with their answers, often outperforming their instruct counterparts. Mistral notably achieves the highest coherence score, reaching 68% with 10 retrieved documents, while Falcon exhibits the lowest, often failing to provide any *Proof* at all. Even the instruct version of Falcon typically offers only the direct answer without supporting evidence. This behavior aligns with our observations in Section 5.1, further highlighting Falcon's challenges in adhering to given instructions.

While these coherence rates provide valuable insights, they come with important caveats. The presence of an answer in the *Proof* does not guarantee its "coherence" accuracy. The *Proof* might not actually derive from the provided documents, even if it includes the answer. Additionally, even if the answer is present in the *Proof*, it may not be recognized as valid due to the inclusion of additional text in the response. For example, if a model begins its response with "The answer to the question is..." and then reports an answer that is technically part of the *Proof*, this response might still be deemed invalid because the introductory phrase does not originate from the context documents.

[9] https://huggingface.co/tiiuae/falcon-7b
https://huggingface.co/tiiuae/falcon-7b-instruct

<table>
<tr><th>Llama 2 7B</th><th>Llama 2 7B-Chat + Template</th></tr>
<tr><td>You are given a question and you MUST respond by EXTRACTING the answer from one of the provided documents. If none of the documents contain the answer, respond with NO-RES.</td><td>[INST] «SYS »
You are given a question and you MUST respond by EXTRACTING the answer from one of the provided documents. If none of the documents contain the answer, respond with NO-RES.
«/SYS »</td></tr>
<tr><td colspan="2">Documents:
Document [1](Title: Batman Returns) the Penguin. We didn't really officially cast it, but for a short nasty little guy, it's a short list. I ended up writing the character for Danny DeVito. Burgess Meredith (who portrayed the Penguin in the 1960s TV series "Batman") was cast for a little cameo as Tucker Cobblepot...
Document [2](Title: Batman: Mystery of the Batwoman) This is the only time in the DC animated universe that Paul Williams did not voice the Penguin...
Document [3](Title: The Penguin's a Jinx) The Penguin goes to Wayne Manor and returns the actress. He then uses his gas-umbrella to knock out anyone inside the statues...

Question: Who played the part of 'The Penguin' in the TV series 'Batman'?</td></tr>
<tr><td>Answer: Burgess Meredith</td><td>Answer: [/INST] Based on the provided documents, the answer is Danny DeVito</td></tr>
</table>

Figure 4: Base vs. Instruct + Template under Task Instruction I on TriviaQA. The figure presents a comparison between the responses generated by two versions of the Llama 2 7B model: the base version and the instruct + template version. Each version is tasked with answering the same question based on the provided documents. The base model correctly identifies the answer as "Burgess Meredith", whereas the instruct + template version incorrectly attributes the answer to "Danny DeVito". *Italic* text denotes the template.

Llama 2 7B	Llama 2 7B-Chat + Template
You are given a question and you MUST respond by EXTRACTING the answer from one of the provided documents. If none of the documents contain the answer, respond with NO-RES. In addition, you must report the portion of the document (Proof) containing the answer. **START example** **Document** [209707](Title: Ancient Egyptian technology) Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty. Chariots, however, are only believed to have been introduced by the invasion of the Hyksos in the Second Intermediate period; during the New Kingdom era, chariotry became central to Egypt's military. **Question:** when was the potter's wheel first used in egypt **Answer:** 4th Dynasty **Proof:** Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty. **END example**	*[INST]«SYS »* You are given a question and you MUST respond by EXTRACTING the answer from one of the provided documents. If none of the documents contain the answer, respond with NO-RES. In addition, you must report the portion of the document (Proof) containing the answer. **START example** **Document** [209707](Title: Ancient Egyptian technology) Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty. Chariots, however, are only believed to have been introduced by the invasion of the Hyksos in the Second Intermediate period; during the New Kingdom era, chariotry became central to Egypt's military. **Question:** when was the potter's wheel first used in egypt **Answer:** 4th Dynasty **Proof:** Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty. **END example** *«/SYS »*
Documents: **Document [1]**(Title: Batman Returns) the Penguin. We didn't really officially cast it, but for a short nasty little guy, it's a short list. I ended up writing the character for Danny DeVito. Burgess Meredith (who portrayed the Penguin in the 1960s TV series "Batman") was cast for a little cameo as Tucker Cobblepot... **Document [2]**(Title: Batman: Mystery of the Batwoman) This is the only time in the DC animated universe that Paul Williams did not voice the Penguin... **Document [3]**(Title: The Penguin's a Jinx) The Penguin goes to Wayne Manor and returns the actress. He then uses his gas-umbrella to knock out anyone inside the statues... **Question:** Who played the part of 'The Penguin' in the TV series 'Batman'?	
Answer: Burgess Meredith **Proof:** Document [1](Title: Batman Returns) the Penguin. We didn't really officially cast it, but for a short nasty little guy, it's a short list. I ended up writing the character for Danny DeVito. Burgess Meredith (who portrayed the Penguin in the 1960s TV series "Batman") was cast for a little cameo as Tucker Cobblepot...	**Answer:** *[/INST]* Danny DeVito **Proof:** Document [1](Title: Batman Returns) the Penguin. We didn't really officially cast it, but for a short nasty little guy, it's a short list. I ended up writing the character for Danny DeVito.

Figure 5: Base vs. Instruct + Template under Task Instruction II on TriviaQA. This comparison of responses between the base and instruct + template versions of Llama 2 7B illustrates an example where the base model correctly identifies the answer, while the instruct + template version inaccurately attributes the answer to a different actor. Nevertheless, in both cases, the answers are "coherent" with the *Proof* since each evidence contains the generated answer. *Italic* text denotes the template.

<table>
<tr><th>Mistral 7B</th><th>Mistral 7B-Instruct + Template</th></tr>
<tr><td>You are given a question and you MUST respond by EXTRACTING the answer (max 5 tokens) from one of the provided documents. If none of the documents contain the answer, respond with NO-RES. In addition, you must report the portion of the document (Proof) containing the answer.
START example
Document [209707](Title: Ancient Egyptian technology) Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty. Chariots, however, are only believed to have been introduced by the invasion of the Hyksos in the Second Intermediate period; during the New Kingdom era, chariotry became central to Egypt's military.
Question: when was the potter's wheel first used in egypt
Answer: 4th Dynasty
Proof: Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty.
END example</td><td>[INST] You are given a question and you MUST respond by EXTRACTING the answer (max 5 tokens) from one of the provided documents. If none of the documents contain the answer, respond with NO-RES. In addition, you must report the portion of the document (Proof) containing the answer.
START example
Document [209707](Title: Ancient Egyptian technology) Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty. Chariots, however, are only believed to have been introduced by the invasion of the Hyksos in the Second Intermediate period; during the New Kingdom era, chariotry became central to Egypt's military.
Question: when was the potter's wheel first used in egypt
Answer: 4th Dynasty
Proof: Evidence indicates that Egyptians made use of potter's wheels in the manufacturing of pottery from as early as the 4th Dynasty.
END example</td></tr>
<tr><td colspan="2">Documents:
Document [12919694](Title: User State Migration Tool) User State Migration Tool The User State Migration Tool (USMT) is a Microsoft command line utility program intended to allow advanced users, comfortable with Scripting languages, to transfer files and settings between PCs. This task is also performed by Windows Easy Transfer, recommended for general users...
Document [12919696](Title: User State Migration Tool) Settings Transfer Wizard and Windows Easy Transfer. USMT is designed to perform mass migrations and allows customizations designed for corporate environments...
Document [12919695](Title: User State Migration Tool) there have been several attempts to provide access to its useful functionality by creating GUI wrappers for it...

Question: what tool is available from microsoft as an alternative to the usmt for smaller migrations</td></tr>
<tr><td>Answer: Windows Easy Transfer
Proof: User State Migration Tool The User State Migration Tool (USMT) is a Microsoft command line utility program intended to allow advanced users, comfortable with Scripting languages, to transfer files and settings between PCs. This task is also performed by Windows Easy Transfer, recommended for general users...</td><td>Answer: [/INST] NO-RES.</td></tr>
</table>

Figure 6: Base vs. Instruct + Template under Task Instruction II on NQ. This comparison of responses between the base and instruct + template versions of Mistral 7B illustrates an example where the base model correctly identifies the answer, while the instruct + template version erroneously opts for a *NO-RES* response, despite the correct answer being present in the documents. *Italic* text denotes the template.

Falcon 7B	Falcon 7B-Instruct
You are given a question and you MUST respond by EXTRACTING the answer (max 5 tokens) from one of the provided documents.	You are given a question and you MUST respond by EXTRACTING the answer (max 5 tokens) from one of the provided documents.

Documents:
Document [1590238](Title: 1970s in film) Thanks to a stellar cast, experienced director, and a poignant story, "Network" became one of the largest critical successes of 1976. Another film, Rocky, about a clubhouse boxer (played by Sylvester Stallone) who is granted a world championship title fight won the Best Picture Academy Award that year. The film also became a major commercial success and spawned four sequels through the rest of the 1970s and 1980s...
Document [2832642](Title: Ned Beatty) 1975. Beatty received his first Academy Award nomination for Best Supporting Actor category for the acclaimed film "Network" (1976), portraying a television network's bombastic but shrewd chairman of the board who convinces the mad Howard Beale character (portrayed by Peter Finch) that corporation-led global dehumanization is not only inevitable, but is also a good thing...

Question: who won the oscar for best picture in 1976

Answer: Rocky	**Answer:** "Network"

Figure 7: Base vs. Instruct under Task Instruction I with No Rejection on NQ. This figure presents responses under a No Rejection setting, where models are not tasked with responding with *NO-RES* if the answer is not contained in the retrieved documents. It compares the base and instruct versions of Falcon 7B. In this instance, the base model accurately identifies "Rocky" as the Oscar winner for Best Picture in 1976, while the instruct version incorrectly cites "Network".

Llama 3 8B	Llama 3 8B-Instruct + Template
You are given a question and you MUST respond with a short answer based on your internal knowledge. If you do not know the answer, please respond with NO-RES.	<|*start_header_id*|>*system*<|*end_header_id*|> You are given a question and you MUST respond with a short answer based on your internal knowledge. If you do not know the answer, please respond with NO-RES.<|*eot_id*|> <|*start_header_id*|>*user*<|*end_header_id*|>

Question: In which US city did the 2004 remake of the film Alfie take place?

Answer: New York City	**Answer:** <|*eot_id*|><|*start_header_id*|>*assistant* <|*end_header_id*|> London!

Figure 8: Base vs. Instruct + Template under Closed-Book QA on TriviaQA. This figure compares responses from the base and instruct + template versions of Llama 3 8B for a question in a closed-book setting, where no additional documents are provided. The example demonstrates how the base model accurately identifies "New York City" as the setting of the 2004 remake of the film Alfie, whereas the instruct + template version erroneously claims the location as "London". *Italic* text denotes the template.

Table 5: Task Instruction I with No Rejection Accuracy. In this setting, models are not tasked to answer *NO-RES* when the answer is absent from retrieved documents. Values *not* marked with an asterisk (*) denote statistically significant differences between base and instruct models (Wilcoxon test, p-value < 0.05). An example of the task instruction adopted in these experiments can be seen in Figure 7.

Task Instruction I with No Rejection												
Dataset	**#Docs**	**Llama 2 7B**			**Llama 3 8B**			**Mistral 7B**			**Falcon 7B**	
		B	**I**	**I + T**	**B**	**I**	**I + T**	**B**	**I**	**I + T**	**B**	**I**
NQ	1	**26.41**	19.76	4.05	**28.14**	13.36	14.88	**25.48**	21.53	19.56	**16.58***	16.03
	2	**28.11**	25.20	2.53	**30.32**	19.21	13.33	**26.48***	26.31	24.96	**19.45**	18.28
	3	**30.49**	26.00	1.56	**30.46**	23.19	12.81	26.13	**29.42**	26.76	**19.80***	18.90
	4	**31.15**	27.62	1.25	**30.81**	25.75	15.40	26.83	**31.36**	28.73	**21.46**	19.38
	5	**31.57**	27.35	1.11	28.66	**29.53***	20.32	29.39	**32.09**	27.83	**21.08**	19.42
	8	**32.02**	27.76	1.90	29.21	**31.39**	21.98	29.15	**34.06**	28.04	**21.08**	19.38
	10	**31.81**	28.31	3.63	28.94	**31.15**	21.98	29.49	**34.89**	36.55	**21.26**	19.97
TriviaQA	1	**58.09**	47.29	33.46	**49.65**	4.05	26.85	**59.52**	51.87	48.15	**41.66**	33.99
	2	**59.98**	50.61	39.70	**57.45**	3.94	37.59	**60.06**	54.42	51.99	**43.55**	36.08
	3	**61.07**	52.93	41.32	**61.07**	5.23	42.25	**60.90**	56.19	54.48	**43.40**	36.78
	4	**61.85**	54.27	41.34	**63.17**	7.01	44.53	**62.17**	57.07	55.36	**43.92**	36.98
	5	**62.22**	55.06	41.09	**64.72**	18.41	46.36	**64.65**	58.07	56.34	**45.04**	37.89
	8	**63.44**	57.11	39.01	**66.53**	53.94	53.15	**66.45**	59.93	58.86	**45.62**	38.18
	10	**64.26**	57.71	43.11	**66.80**	61.46	56.00	**67.88**	61.25	59.85	**45.54**	38.43
HotpotQA	1	**23.96**	22.30	19.00	**24.93**	23.12	17.16	**26.89**	24.64	24.50	**16.59***	16.20
	2	**25.54**	23.88	19.89	**27.07**	24.48	14.75	**27.82**	25.93	25.12	**16.79***	16.73
	3	**25.79**	23.98	21.66	**27.93**	24.68	14.95	**27.46**	26.16	25.66	**16.82***	16.23
	4	**26.41**	24.89	23.12	**28.41**	25.34	17.18	**27.52***	26.50	25.86	**18.04***	17.48
	5	**26.18***	25.36	24.14	**29.23**	27.09	18.61	**27.93**	26.88	26.00	**17.61***	16.96
	8	26.38	**28.02**	25.45	30.25	**30.34***	21.45	**28.14**	26.80	27.09	17.74	**18.34***
	10	26.48	**28.32**	25.25	29.70	**32.25**	21.79	**28.49**	27.48	27.25	**18.45***	17.75

Table 6: Task Instruction II Answer Coherence on NQ and TriviaQA. Coherence is measured as the percentage of instances where the generated answer is contained within the generated *Proof*.

Task Instruction II — Answer Coherence												
Dataset	**#Docs**	**Llama 2 7B**			**Llama 3 8B**			**Mistral 7B**			**Falcon 7B**	
		B	**I**	**I + T**	**B**	**I**	**I + T**	**B**	**I**	**I + T**	**B**	**I**
NQ	1	36.93	**45.07**	0	36.90	**42.02**	0.52	36.73	**50.12**	15.99	1.87	**3.39**
	2	41.16	**47.35**	0.59	41.36	**51.92**	0.03	44.03	**53.24**	10.9	2.08	**3.12**
	3	41.64	**46.97**	1.14	43.54	**50.71**	0.1	46.56	**54.38**	7.65	3.32	**3.77**
	4	41.43	**46.45**	0.07	**46.49**	43.86	0.14	45.66	**55.49**	5.75	3.01	**4.85**
	5	42.16	**45.17**	0.07	46.80	**48.36**	1.38	51.47	**57.39**	3.53	3.81	**4.26**
	8	**42.99**	40.53	0	**51.78**	48.29	0.28	54.62	**57.74**	4.12	**4.50**	1.73
	10	**41.26**	32.78	0	51.37	**53.10**	0.07	53.31	**56.07**	2.7	**1.15**	1.04
TriviaQA	1	37.22	**43.15**	0.05	**51.66**	35.45	1.10	43.23	**51.08**	16.82	3.02	**4.45**
	2	44.27	**49.61**	0.46	**57.44**	50.55	0.13	52.16	**53.52**	14.87	3.70	**5.94**
	3	44.68	**53.09**	0.75	**59.46**	50.89	0.12	**56.07**	55.43	11.42	3.47	**6.53**
	4	47.19	**53.46**	0.11	**61.30**	48.91	0.49	**58.48**	57.53	7.90	4.52	**6.08**
	5	48.00	**48.92**	0.04	**62.02**	55.02	1.59	**60.82**	58.55	5.70	4.86	**6.28**
	8	**50.35**	40.14	0	**64.12**	55.28	1.42	**66.66**	59.68	2.45	3.17	**3.34**
	10	**47.14**	28.93	0	**63.58**	56.78	0.50	**68.04**	58.58	2.16	1.61	**1.67**

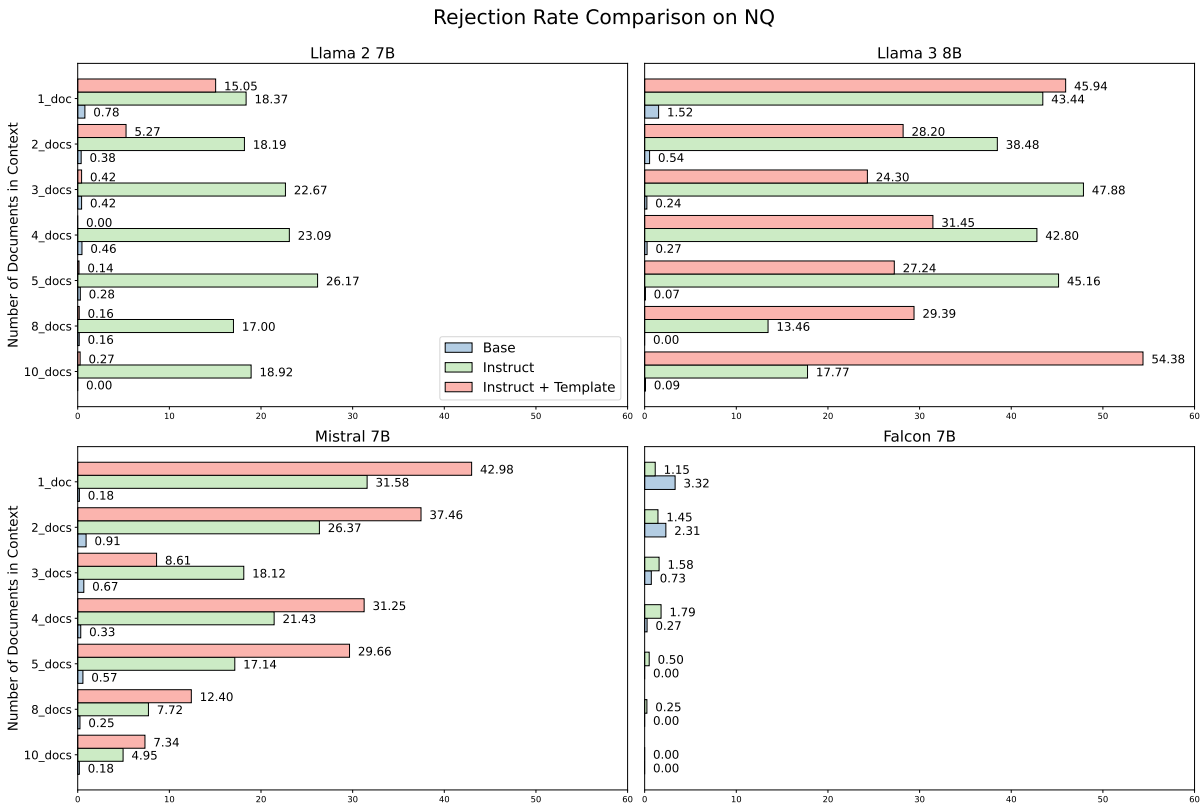

Figure 9: Reported is the rejection rate on NQ, defined as the number of times the model responds with *NO-RES* when the correct answer is not in the context, divided by the number of times the answer is indeed missing. Instruct models are much more effective at detecting such cases and following the instructions provided.

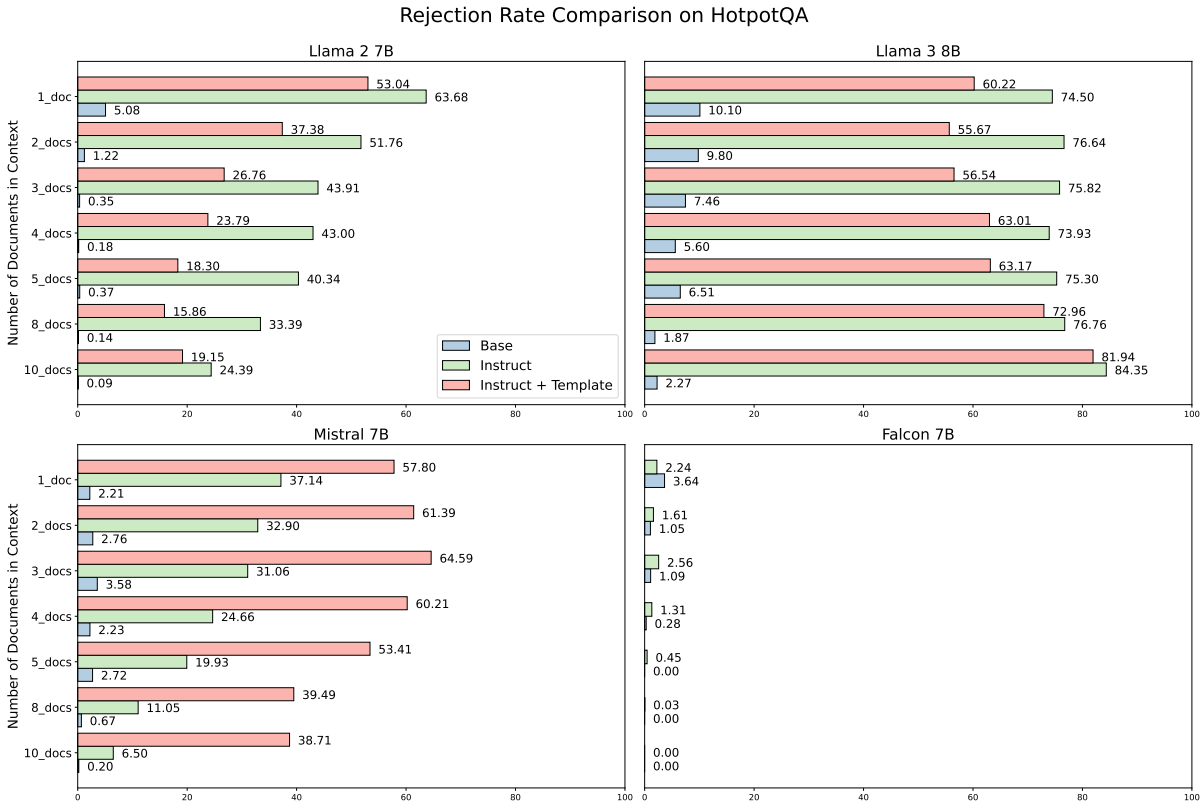

Figure 10: Reported is the rejection rate on HotpotQA, defined as the number of times the model responds with *NO-RES* when the correct answer is not in the context, divided by the number of times the answer is indeed missing. Instruct models are much more effective at detecting such cases and following the instructions provided.

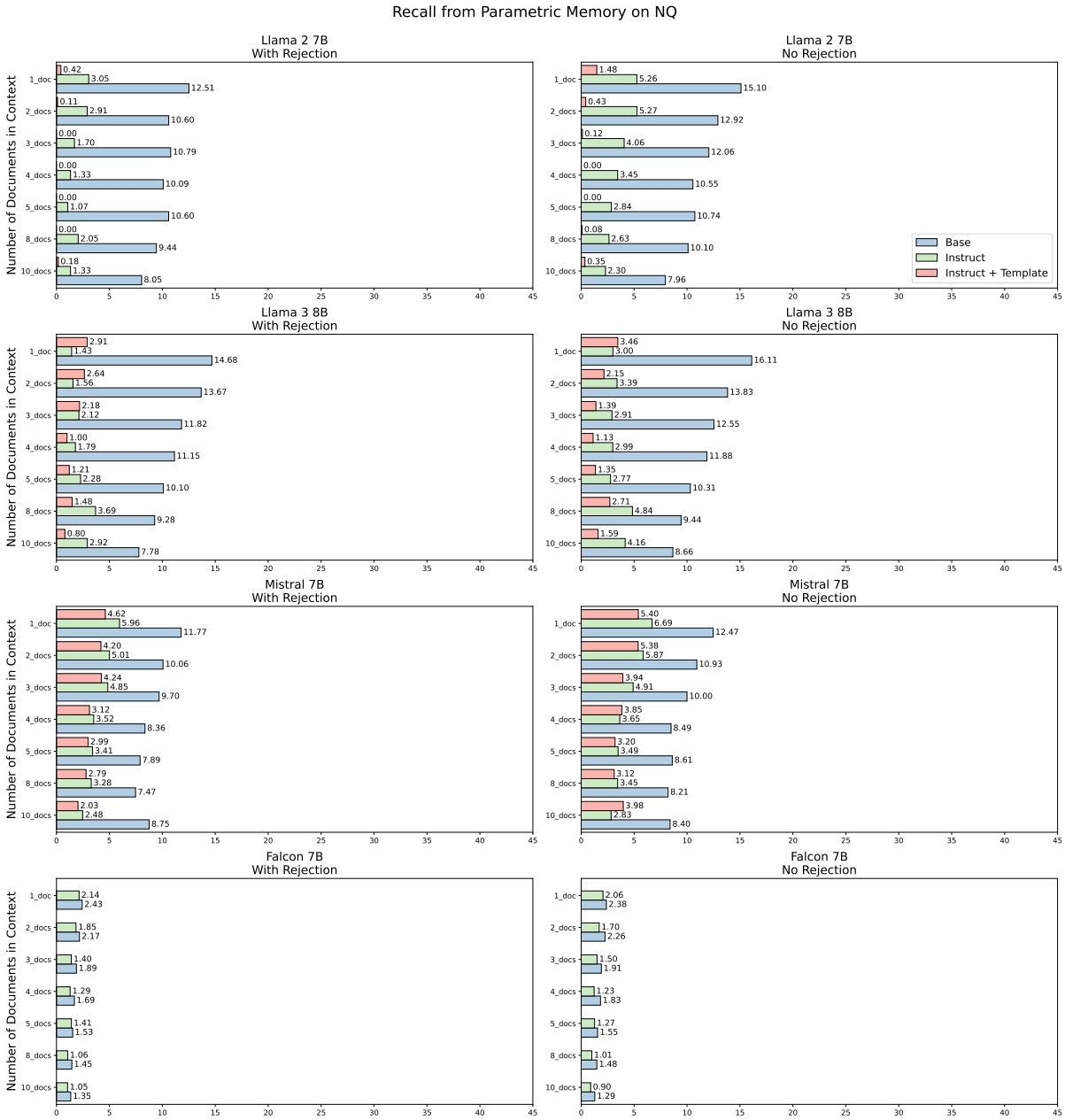

Figure 11: Comparison of Recall from Parametric Memory on NQ between the case where we specify to answer with *NO-RES* when the answer is not contained in the retrieved documents (*left*) and the No Rejection setting (*right*).

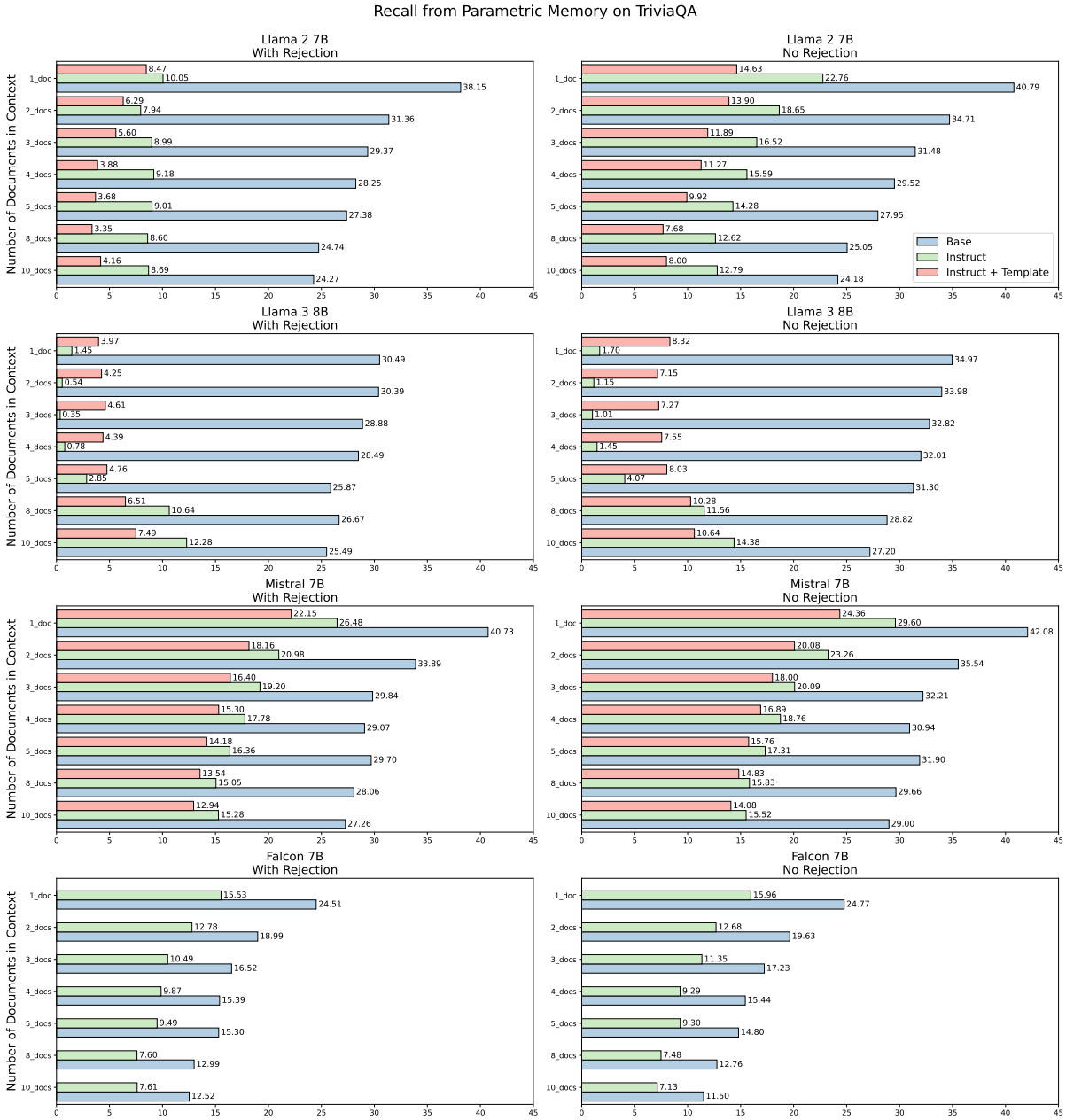

Figure 12: Comparison of Recall from Parametric Memory on TriviaQA between the case where we specify to answer with *NO-RES* when the answer is not contained in the retrieved documents (*left*) and the No Rejection setting (*right*).

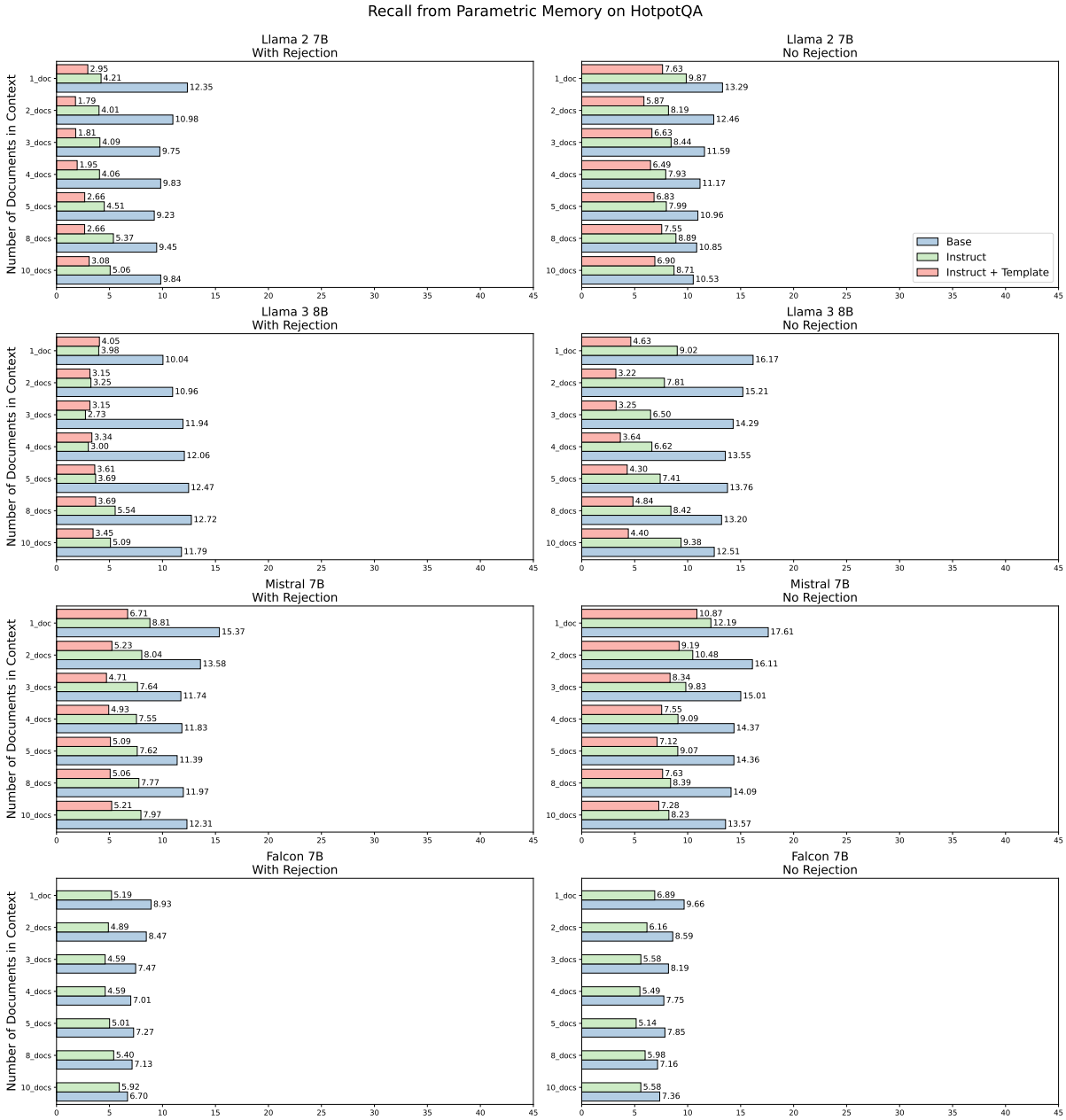

Figure 13: Comparison of Recall from Parametric Memory on HotpotQA between the case where we specify to answer with *NO-RES* when the answer is not contained in the retrieved documents (*left*) and the No Rejection setting (*right*).