Optimizing LLM Based Retrieval Augmented Generation Pipelines in the Financial Domain

Yiyun Zhao[1]¶, Hanoz Bhathena[1]¶, Prateek Singh[1]¶, Saket Sharma[1]¶,
Bernardo Ramos¶, Aviral Joshi¶, Swaroop Gadiyaram¶

¶Machine Learning Center of Excellence, JPMorgan Chase & Co.
hanoz.bhathena@jpmchase.com

Abstract

Retrieval Augmented Generation (RAG) is a prominent approach in real-word applications for grounding large language model (LLM) generations in up-to-date and domain-specific knowledge. However, there is a lack of systematic investigations of the impact of each component (retrieval pipeline, prompts, generation models) on the generation quality of a RAG pipeline in real world scenarios. In this study, we benchmark 6 LLMs in 15 retrieval scenarios, exploring 9 prompts over 2 real world financial domain datasets. We thoroughly discuss the impact of each component in RAG pipeline on answer generation quality and formulate specific recommendations for the design of RAG systems.

1 Introduction

Recent years have seen tremendous improvement in the ability of large language models (LLM) such as GPT-4 (OpenAI et al., 2023) and Llama-2 (Touvron et al., 2023) to address users' questions/queries in diverse domains (medical questions, math problems, code assistants etc). Despite LLMs acquiring immense parametric world knowledge during the pre-training, when adapting to real-world applications, their lack of customized domain-specific knowledge or knowledge of recent events (Kandpal et al., 2023; Sun et al., 2023), frequently results in outdated responses or baseless responses not grounded in the user's domain of interest, also termed *hallucinations* (Bang et al., 2023; Rawte et al., 2023; Li et al., 2023). Hallucinations contribute to a lack of trust with users, and this unreliability is one of the biggest hindrances in the responsible deployment of LLM based systems for critical business applications in the financial domain.

Retrieval Augmented Generation (RAG) is the current go-to approach to connect LLMs to live/updated information sources. Existing works (Lazaridou et al., 2022; Shuster et al., 2021; Ren et al., 2023) show RAG can reduce hallucinations and improve answer quality, without the need for highly expensive and sometimes brittle domain-specific fine-tuning.

Given a user query, a typical RAG system (Figure 1) employs a retriever system to fetch a list of documents likely relevant to the query from an information source (Retrieval). The documents are then fed into the context of the LLM, with users' query / conversation history, and specific instructions / prompts on how to generate a response "grounded" in retrieved information (Generation).

While there is growing number of proposals (Jiang et al., 2023; Siriwardhana et al., 2023) to improve RAG systems (see the survey from Gao et al. (2023)), very few studies (Chen et al., 2023b) systematically investigate the impact of each component (retriever, prompts, models) on answer generation quality and interactions among these various components. Our goal of this paper is to evaluate the efficacy and limits of RAG pipelines for Question Answering (Q&A) systems in the highly specialized financial domain.

In this study, we benchmark LLMs' answer generation quality and explore the following aspects: (i) Comparing different generative LLMs as answer generation models against each other and baseline (purely extractive) models; (ii) Examining how various LLMs handle differences in the quality of information retrieval; (iii) Exploring the impact of varying prompts on answer quality of RAG pipelines.

In line with our objectives, we curated two datasets from the banking sector featuring real user queries. These datasets were used to design test scenarios that mimic the retrieval of information at varying levels of quality. Additionally, we crafted prompts with distinct characteristics (e.g., level of detail in instructions, requirements for citations,

[1] Equal contribution.

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track), pages 279–294
June 16-21, 2024 ©2024 Association for Computational Linguistics

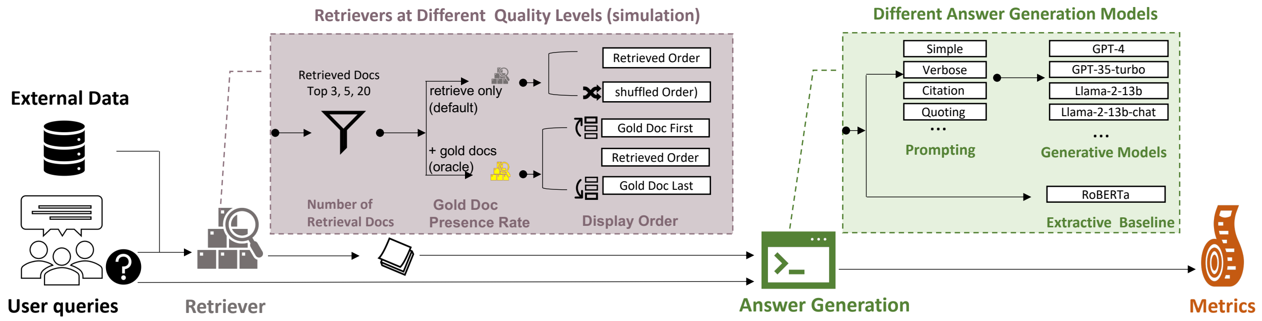


Figure 1: Evaluation Framework of a RAG system

response format) and conducted evaluations on six LLMs. In addition to answer generation quality, we also evaluate LLMs' ability to adhere to instructions on aspects such as answer style, citation output format etc.

Our findings reveal that generative LLMs outperform baseline models in answer quality, even on metrics emphasizing extractiveness which could ideally have given extractive models an advantage. As expected, GPT-4 demonstrates superior performance over GPT-3.5-Turbo, which in turn outperforms the LLama-2 models. We observed that the generation quality is highly influenced by the quality of retrieval: LLMs tend to provide answers even when relevant source information is missing, a form of *pseudo-helpfulness* varying from responding with content present in the context that is somewhat related but not addressing the user question, to hallucinations. The performance of even SOTA LLMs like GPT-4 declines with an increase in number of distractor documents retrieved or when relevant documents are not ranked higher in the retrieval list. Interestingly, we did not find a systematic impact of prompt characteristics on the quality of answer generation especially on OpenAI models: GPT-4 and GPT-3.5-turbo models are resilient to prompt variations, whereas the Llama-2 models exhibit more variability. Finally, GPT-4 and GPT-3.5-turbo exhibit compliance to instructions around structured formatting and language style over 90% of the time, whereas Llama-2 models struggle to follow instructions. To summarize, the main contributions of the current studies are as follows:

- We comprehensively assess various factors that contribute to the answer generation quality in LLM RAG systems, ranging from sensitivity to retrieval quality to the impact of different prompts, conducted across 6 LLMs on two datasets in the financial (banking) sector with real user queries, enabling the testing of RAG systems in realistic scenarios [1]. Our evaluations on internal data absent in the LLMs' pre-training also represents a better estimate of real-world generalization of LLMs.
- We conclude with specific recommendations for the design of RAG systems, grounded in the insights and findings derived from our empirical results.

2 Experiment Framework

This section introduces the design of the evaluation framework (see Figure 1). To summarize, we ran 1620 experiments to assess 6 LLMs in 15 retrieval conditions using 9 prompts over 2 datasets for 2 performance aspects.

2.1 Evaluation Dataset Construction

In our experiments, we developed two RAG datasets from queries against two corpora: (1) **Banking webpages**: Public webpages with general information on banking products, and (2) **Banking policy guides**: internal guides for customer service executives detailing policies and protocols for customer assistance. For both the corpora, we had associated questions, which were either generated by the actual users and gathered from production system logs, or were generated by subject-matter-experts.

We chunked webpages/articles into about 100 word document chunks (also referred as documents) while preserving sentence boundaries. Chunks with majority content in a non-English language or those with fewer than 10 words were dropped. We paired each question to related documents via unifying subject matter experts' coarse

[1] Unfortunately we cannot release this dataset due to confidentiality concerns.

annotation and two-staged models (see details Figure 6 in Appendix). The questions paired with documents were then sent to the human reviewers to (i) assign a binary relevance label to every chunk and (ii) label an answer span within the chunk, served as reference answer.

2.2 Simulated Scenarios of Different Retrieval Quality

The success of highly-customized RAG applications hinges on the quality of the retriever component. Understanding how LLMs are affected by retrieved documents is vital for developing effective enhancement strategies and further research in RAG systems.

We tackle this by designing different test sets. We sampled 800 user queries for each dataset and obtained retrieved document chunks using a retriever(OpenAI Embeddings[2]). We then manipulate the retrieved list to mimic retrievers of different qualities to address scenarios listed below.

Q1. How does absence of retrieved "gold" document influence answer generation? We created two retrieval conditions: Retriever-Only (returning the retrieved set which may or may not contain gold chunk(s)) versus Retriever-W-GT (guaranteed to have gold document to the retrieved list)[3].

Q2. How does the number of documents retrieved influence answer generation? We created three conditions varying in the number of documents displayed to generation models: top_3, top_5, top_20.

Q3. How does order of retrieved documents influence answer generation? We further manipulated the display order of retrieved documents during LLMs' response generation. For Retriever-Only, we added a new condition where we simply shuffled the order of the documents to judge the sensitivity to order in general. For Retriver-W-GT, we added two conditions where we injected the gold document in the first or last position. In summary, we created 15 retrieval conditions. For retrieval_only, we designed 2 (retriever_only, retriever_only_shuffled) x 3 (top 3, 5, 20); for retriever_w_gt, we designed 3 (retriever_w_gt, retriever_w_gt_first, retriever_w_gt_last) x 3 (top 3, 5, 20).

2.3 Answer Generation Models

The quality of responses in Retrieval-Augmented Generation (RAG) systems is significantly influenced by the choice of Answer Generation Models. Here we compare the performance of several LLMs. We also report the performance of a RoBERTa Model, as our baseline.

Baseline Model Since our dataset contains answer spans, for our baselines we use an encoder-only answer-span extraction model (Roberta finetuned on SQuAD2 (Rajpurkar et al., 2016, 2018) and Natural-Questions (Kwiatkowski et al., 2019) datasets).

Generative Large Language Models We assess 6 frequently-used LLMs including gpt-3.5-turbo-0613, GPT-4-0613 (OpenAI et al., 2023), and Llama-2-7B and Llama-2-13B (base and chat)(Touvron et al., 2023). The details of experimental parameters for each model are fully specified in Table 1 in Appendix.

2.4 Prompts

Previous research indicates that the performance of Large Language Models (LLMs) can be affected by the prompts used (Chen et al., 2023a; Zhu et al., 2023). In this work, we investigate the effect of prompting on generation in RAG pipelines. In particular, we created a set of prompts with variations in factors such as the verbosity of instructions, the need for direct quoting, explicit introduction of metrics within prompts, the requirement for citations, and specific response formatting, among other aspects. The full list of prompts experimented in the study can be found in Appendix (Figure 11, Figure 12, Figure 13).

2.5 Evaluation Metrics and Aspects

To assess the performance of RAG system, we evaluate the answer quality, and instruction following ability of the LLMs.

Answer Quality Due to the extractive nature of our tasks (2.1) we followed (Ren et al., 2023) using token F1 scores which show reasonable correlation with human subjects (Adlakha et al., 2023). [4].

[2] https://platform.openai.com/docs/guides/embeddings

[3] in case we need to add we place gold doc in the first position

[4] We did not report Exact Match because it is misleading due to multi-sentence answer responses. Reference-free metrics are not used due to high costs and we found recall can be score hacking shown in section 3.4. Specifically, llama-2 models tend to have long generations by copying many sentences from source, resulting in a high chance to get a high token recall score.

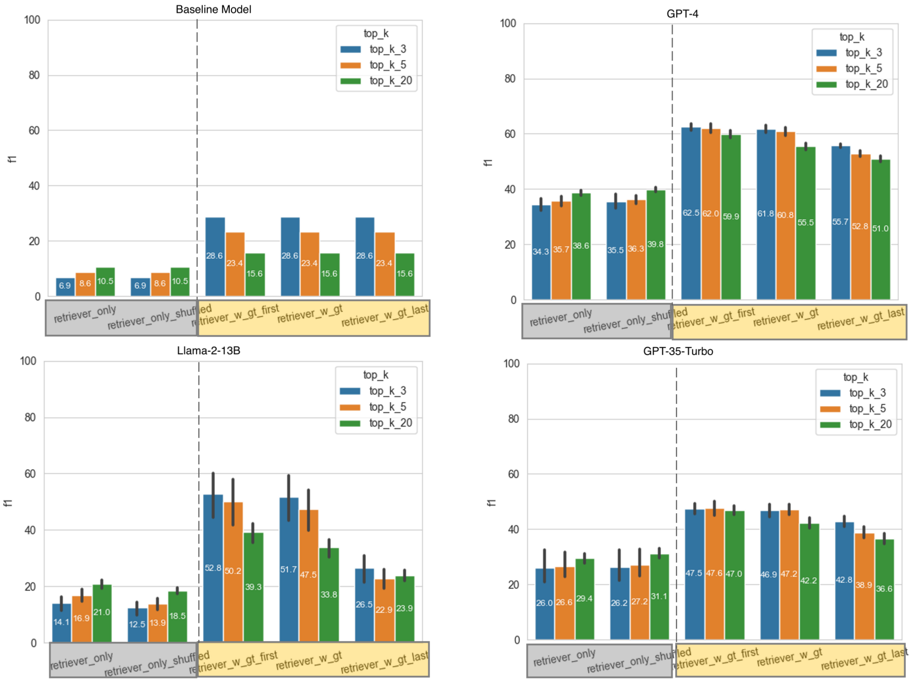


Figure 2: Answer generation quality (Token F1) of baseline model, GPT-4, gpt-3.5-turbo and Llama-2-13b for Banking policy dataset. Full results can be found in Appendix (Figure 9 and Figure 10).

Instruction Following Ability Practical RAG deployments typically require the LLM returns answer in a specific language style and may require citations in certain structured format that can be extracted from the model response. Thus, it is important to measure instruction-following performance.

- For structured output, we designed prompts that require pipe format (**<answer_span>|||<Document_ID>**) and JSON format ({ **"text": <answer_span>, "source_id": <Document_ID>**}). We calculated the proportion of the output that correctly produced the expected formatting for each model.
- For language-style output, we designed a prompt that requires direct quoting from the source. Therefore, we calculated the the proportion of the sentences of responses that are directly taken from the source retrieved documents for each model.

3 Results

3.1 Influence of Retrieval Quality on Answer Generation Quality

Figure 2 displays different models' performance on 5 retrieval conditions with 3 different number of retrieved documents for Banking Policy guides dataset (Complete results can be found in figure 9 and 10 in Appendix).

Does the presence of gold document matter? To address the question, we compare performance in Retriever-Only Conditions with that in Retriever-W-GT-X Conditions for each model in Figure 2. Across models, we observe that performance generally improves when a source verified to contain answer is retrieved. In other words, even powerful LLMs such as GPT-4, are imperfect at rejecting to answer ("No Answer Found") in cases of retrieval failure[5]. However, GPT LLMs are better at gracefully handling retrieval failures compared to other models we consider. This behaviour represents a form of *pseudo-helpfulness* in LLM based RAG systems, wherein LLMs try to be *helpful* even when relevant information is missing in their context, overriding the typical expectation injected in RAG systems to only use information in the context that addresses the question. This phenomenon manifests as responses containing related content not addressing the user's question, and hallucinations.

[5] We consider "No answer found" or equivalent to be correct behavior in our metrics for cases where gold document is not retrieved

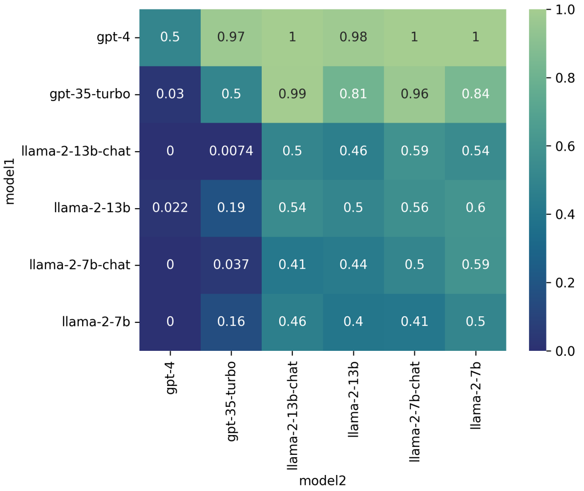


Figure 3: Head-to-head pairwise win rate in Token F1 for Banking Policy guides. Results for Banking webpages can be found in appendix (Figure 7).

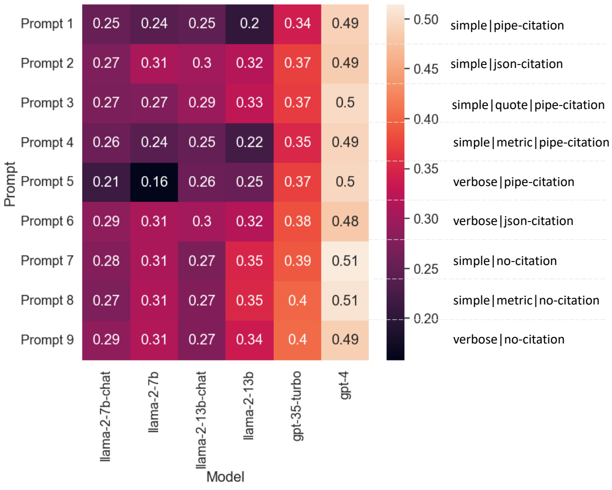


Figure 4: Heatmap of Token F1 for 9 prompts X 6 Models on Banking Policy guides. Results for Banking webpages can be found in appendix (Figure 8).

Does the number of retrieved docs matter? To answer this question, we compare how models' performance varies across different top-k conditions (3,5,20). If the number of retrieved documents is irrelevant to answer generation quality for a model, we would expect the models' performance to not vary across different top-k scenarios. From Figure 2, we observe across models the influence of top-K varies between Retriever-Only Conditions and Retriever-W-GT-X Conditions.

- For Retriever-W-GT-X Conditions, models' performance tends to decline when the number of retrieved documents increases: TOP-K-3 > TOP-K-5 > TOP-K-20. This decline is less pronounced when the gold document appears near the top of the retrieved list for GPT-4 and GPT-3.5-turbo. This indicates that models are in general sensitive to noisy documents in the retrieved list but GPT models are less distracted if the good document is placed in the top position.

- For Retriever-Only Conditions, models' performance increases with the number of documents retrieved. This increase potentially is due to the improvement of gold document recall as LLMs are better at finding gold spans when they exist than rejecting to answer in absence of gold information.

Overall, our results indicate that we cannot reduce our retrieval optimization objective to maximize recall due to LLMs' sensitivity to retrieval noise.

Does the order of retrieved documents matter? To assess the influence of order, for each top-k, we compare model performance bar (that is bar with the same color) among retriever_w_gt_first v.s. retriever_w_gt t v.s. retriever_w_gt_last. From Figure 2 we observe across different LLMs (We ignore the baseline model whose pairwise implementation is insensitive to the display order by design), the performance varies noticeably across the different ordering conditions. Specifically, placing gold document in the first position leads to a better performance than in the last position. This discrepancy is more obvious when a higher number of documents are retrieved. For retriever_only conditions, where we compare against retriever_only_shuffled, the order does not show much influence presumably due to presence of retrieval failure cases that reduces the gap between naive retrieved ordering versus shuffled ordering. Overall, our results indicate that investing in a re-ranking system as part of retrieval optimization is still necessary with LLM based pipelines.

3.2 Influence of Choice of Generation Models

This section compares different LLMs on answer generation quality. Figure 2 has shown that baseline models under-perform the other generative LLMs counterparts by a large margin. We compare the 6 LLMs using a head-to-head win rate across all experiments. Figure 3 demonstrates the win rate of a model when compared to another model across all the experiments, which shows the overall tendency: GPT-4 » GPT-3.5-turbo » Llama-2-13b > llama-2-13b-chat, llama-2-7b, llama-2-7b-chat.

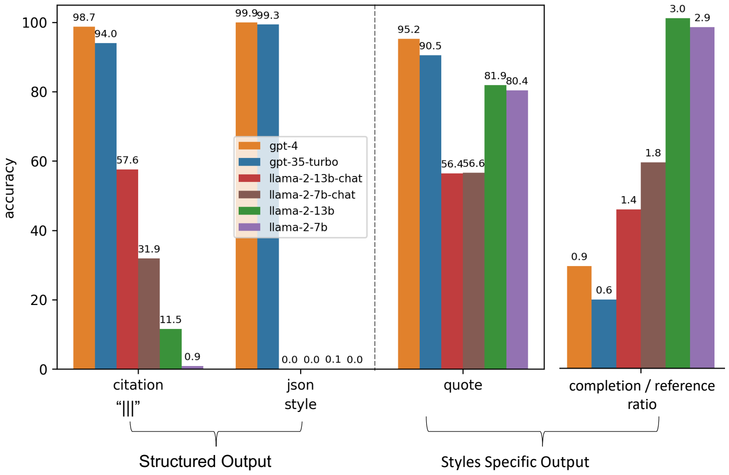


Figure 5: Accuracy of instruction following for structural formatting and language-style.

3.3 Influence of Prompts

This section investigates model robustness to different prompts. Figure 4 captures the answer quality of different models (horizontal axis) with different prompts (vertical axis) for the Banking Policy dataset. We observe that GPT-4 is most robust to the prompts with max difference less than 4% followed by GPT-3.5-turbo about 6%. The two Llama-2 base models we tested show huge variations (around 15%) compared to the chat version. Though there is consistency in prompting trends across two datasets (Figure 8 in Appendix), we did not observe any salient effect of prompt features (such as verbosity, quoting or citation requirement etc) on answer generation quality.

3.4 Instruction-following Accuracy

Structured Output We assess the proportion of models' output that followed correct output format (||| and JSON). From Figure 5, we observe that both GPT-4 and GPT-3.5-turbo follow instructions (> 94%) in both styles (||| or JSON). By contrast, for Llama-2 models, the chat models outperform base models but are far behind the OpenAI models in the "|||" citation instruction. All Llama-2 models tested barely produced a parseable JSON format with required fields.

Quoting as Answer Style Figure 5 indicates that both GPT-4 and GPT-3.5-turbo quote more than 90% of the times. The Llama-2-base models also show high level of quoting (80%) compared to the chat versions (50%). However, qualitative analysis reveals that base models tend to copy line by line from the source text regardless of its relevance to the question, thereby, *cheating* the metric. This can also be observed by comparing the completion to reference ratio (Figure 5): A high value of 2.9 and 3 indicates that Llama2 base models repeat up to 3 times of the expected completion.

4 Conclusion

We conducted thorough investigation of the influence of several components of a RAG pipeline on the overall generation quality. Based on our findings, we find retrieval optimization is an important part of the RAG pipeline design, even with high quality LLMs like GPT-4. Firstly, we recommend prioritizing retrieval recall, while tuning retrieval systems, as LLMs exhibit *pseudo-helpfulness* when relevant gold document(s) are not retrieved. Additionally, improving precision of retrieval, either by using re-rankers or fine-tuned retrievers, will likely improve performance as they improve the gold document(s) rank in the retrieved list. We also find RAG systems to be sensitive to the presence of distractors in the context. We find a big delta between vendor LLMs (OpenAI) and smaller scale open-source alternatives in our experiments, with respect to sensitivity to prompting, overall quality, and instruction following on a domain specific use cases. Finally, for smaller-sized LLama-2 models we recommend simple instructions as they often fail to follow longer or more complicated instructions.

Ethics Statement

All the work done and discussed in this paper meets and upholds the ACL Code of Ethics. User data wherever used was anonymized.

References

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. 2023. Evaluating correctness and faithfulness of instruction-following models for question answering. *arXiv preprint arXiv:2307.16877*.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Banghao Chen, Zhaofeng Zhang, Nicolas Langrené, and Shengxin Zhu. 2023a. Unleashing the potential of prompt engineering in large language models: a comprehensive review.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2023b. Benchmarking large language models in retrieval-augmented generation.

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

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. 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 *International Conference on Machine Learning*, pages 15696–15707. PMLR.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore. Association for Computational Linguistics.

OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. Gpt-4 technical report.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, S.M Towhidul Islam Tonmoy, Aman Chadha, Amit Sheth, and Amitava Das. 2023. The troubling emergence of hallucination in large language models - an extensive definition, quantification, and prescriptive remediations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2541–2573, Singapore. Association for Computational Linguistics.

Ruiyang Ren, Yuhao Wang, Yingqi Qu, Wayne Xin Zhao, Jing Liu, Hao Tian, Hua Wu, Ji-Rong Wen, and Haifeng Wang. 2023. Investigating the factual knowledge boundary of large language models with retrieval augmentation. *arXiv preprint arXiv:2307.11019*.

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.

Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. 2023. Improving the domain adaptation of retrieval augmented generation (rag) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11:1–17.

Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2023. Head-to-tail: How knowledgeable are large language models (llm)? aka will llms replace knowledge graphs? *arXiv preprint arXiv:2308.10168*.

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

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, et al. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528*.

A Appendix

A.1 Limitations

Our research opens up various avenues for future investigation. Firstly, the RAG pipeline's inherent nature makes it challenging to comprehensively identify all relevant documents for a query. We aim to develop algorithms that can enhance automatic identification with greater precision. Secondly, our study presupposes that answers are derived from a single passage, which is not always true in practical scenarios. We plan to broaden this assumption to more realistic applications. Thirdly, our research is currently limited to six models from two families (OpenAI GPT and Llama); it would be intriguing to evaluate additional models like Claude, Mistral, etc., across a wider range of datasets. Finally, our study mimics the retrieval quality via manipulation of retrieved documents. It would be advantageous to examine various real-world retrieval systems to determine how improvements at the retrieval stage can translate into enhanced answer quality.

A.2 Model Parameters

Model	Version	Context Length	Completion Length	Decoding Strategy
GPT-3.5-turbo	0613	15.7k	700	temperature 1
GPT-4	0613	7.5K	700	temperature 1
Llama-2-13b	-	3.9K	200	greedy
Llama-2-13b-chat	-	3.9K	200	greedy
Llama-2-7b	-	3.9K	200	greedy
Llama-2-7b-chat	-	3.9K	200	greedy

Table 1: LLM Model Parameters. We used the set of parameters to balance the context length for retrieved document list and also completion length required to generate responses. Based on our estimates of reference answers, two OpenAI models can keep more than 99% for both inputs and outputs from being chunked and Llama-2 models keep 99% (both inputs and outputs) for the public wepage dataset from being chunked and 70% (inputs) and 90% (outputs) for the internal service dataset.

A.3 Evaluation Dataset Generation

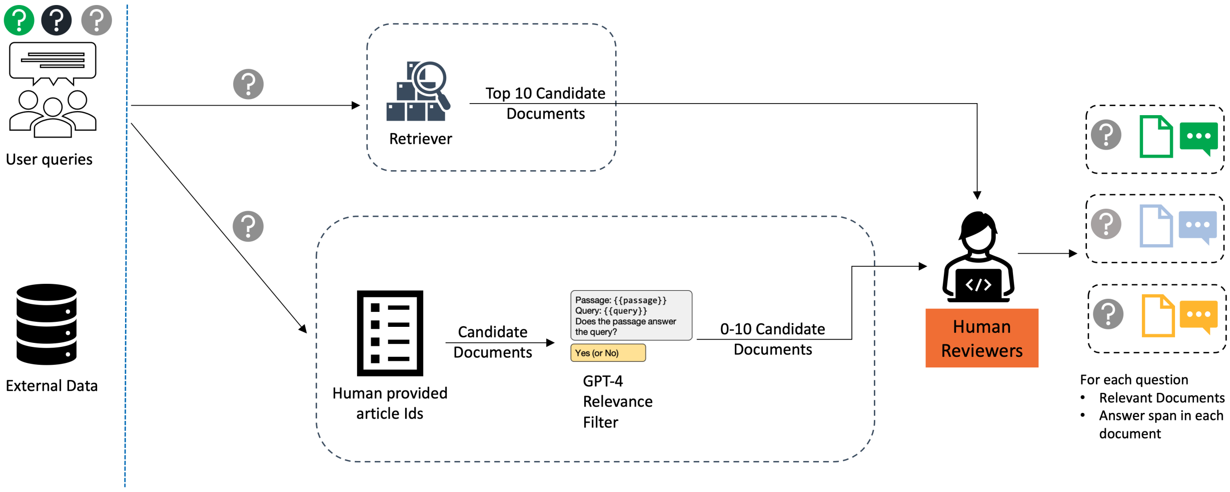


Figure 6: Evaluation dataset creation pipeline: We paired each question to related documents via a two-staged pipeline. First subject matter experts (SMEs) annotated coarse grained document labels for certain queries. Given our requirement for more fine-grained query, chunk alignment pairs, a two stage model based approach was applied to get an initial mapping and then review it with human reviewers: (1) We took chunks from the SME provided higher level articles and use GPT4 as a pairwise evaluator of relevance. Each chunk in the SME annotated articles was paired with the query and GPT4 was instructed to return a binary label for relevance of the chunk against the query. (2) Since our SME annotations are not comprehensive, there could be articles in the corpus which they did not tag but could contain the answer. So we used a SOTA dense retriever to get top-10 chunks for a query from the entire document corpus. The union of the above selected chunks was provided to a team of human reviewers who (i) assigned a binary relevance label to every chunk and (ii) selected an answer span within the chunk which answered the question.Our datasets are used purely for evaluation purposes, not for any fine-tuning.

A.4 Head-to-head comparison on Banking Webpage

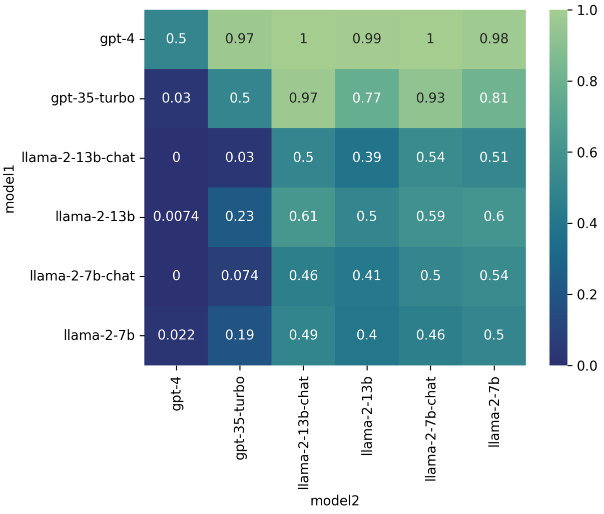


Figure 7: Head-to-heand pairwise winrate in Answer generation quality (Token F1) for Banking webpage dataset.

A.5 Prompt Variance heatmap

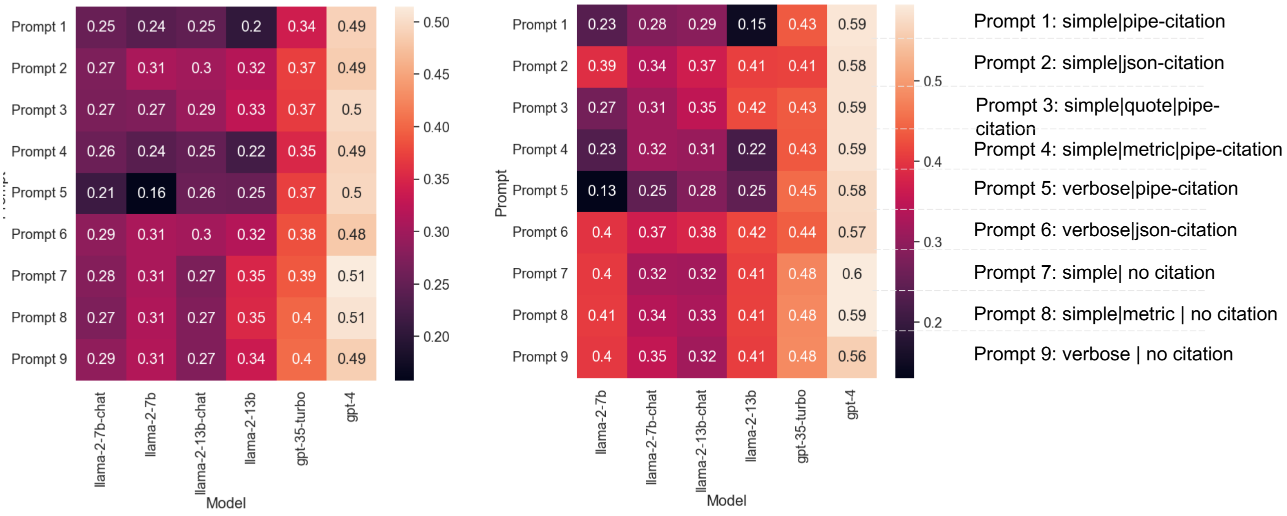


Figure 8: Head-to-head pairwise winrate in Answer generation quality (Token F1). Left figure shows the results for Banking Policy Guides (Left) and the Banking Public webpages (Right).

A.6 F1 Performance for all datasets, baselines and conditions

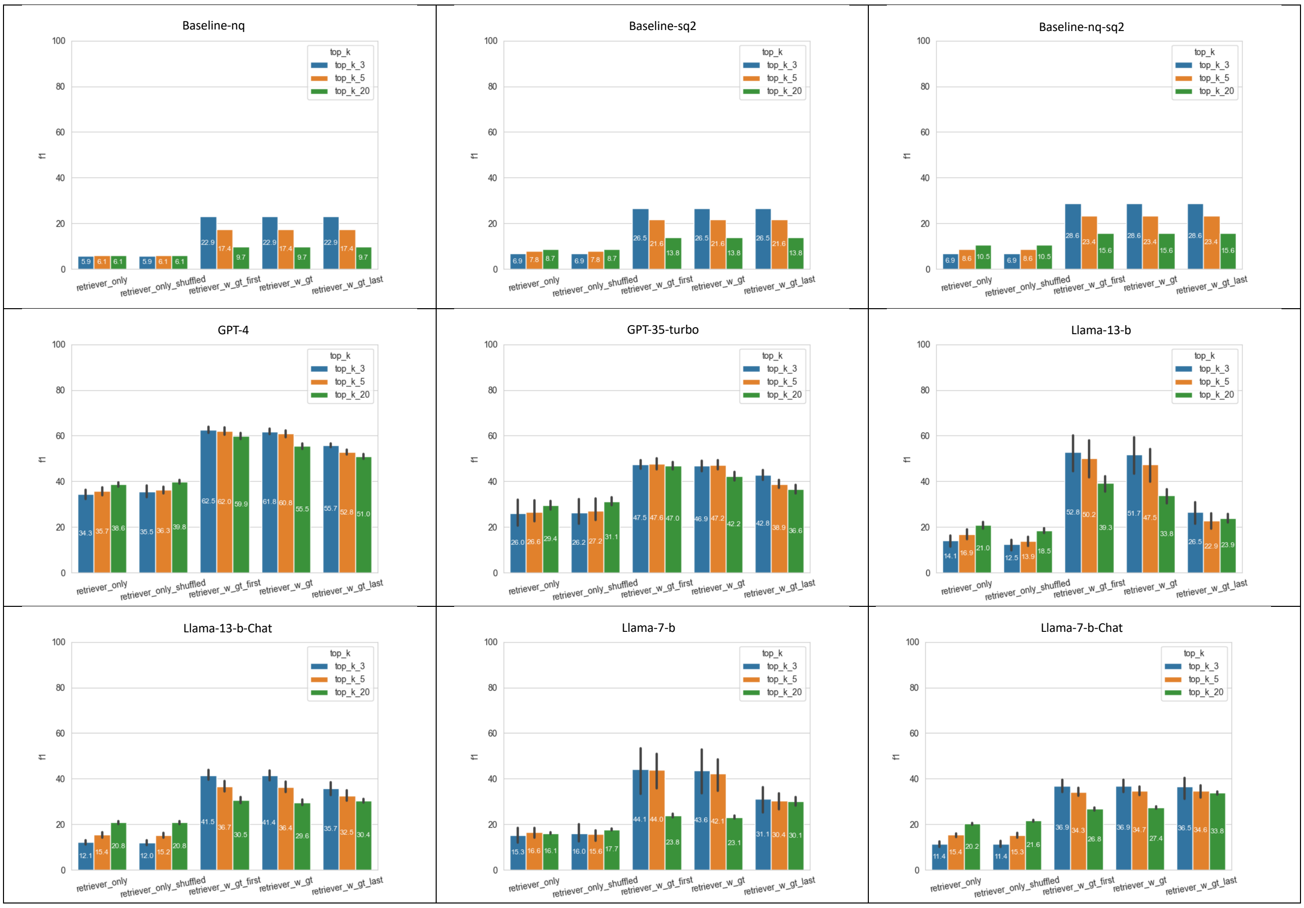


Figure 9: F1 score for all models on the Banking Policy Guides dataset

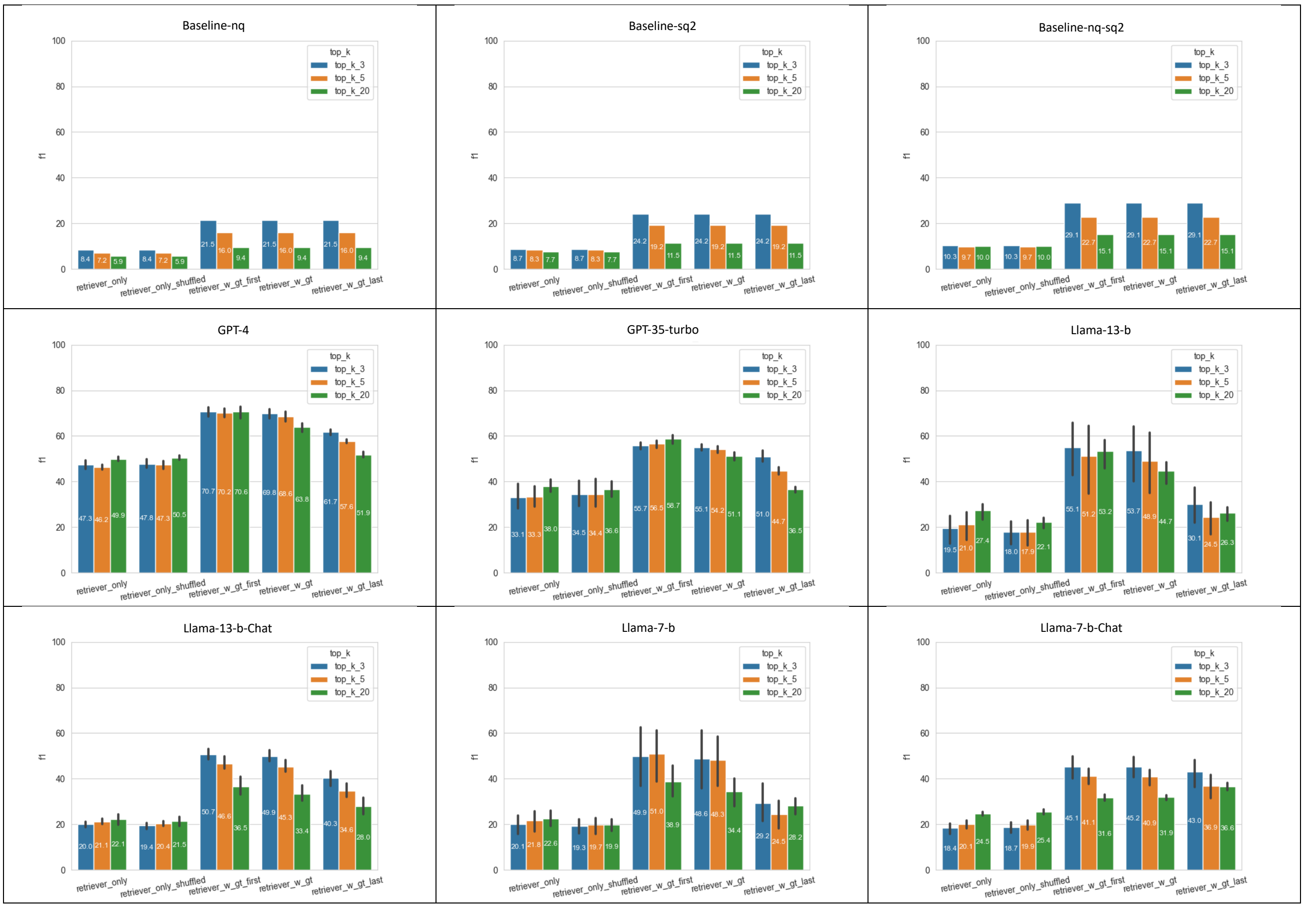
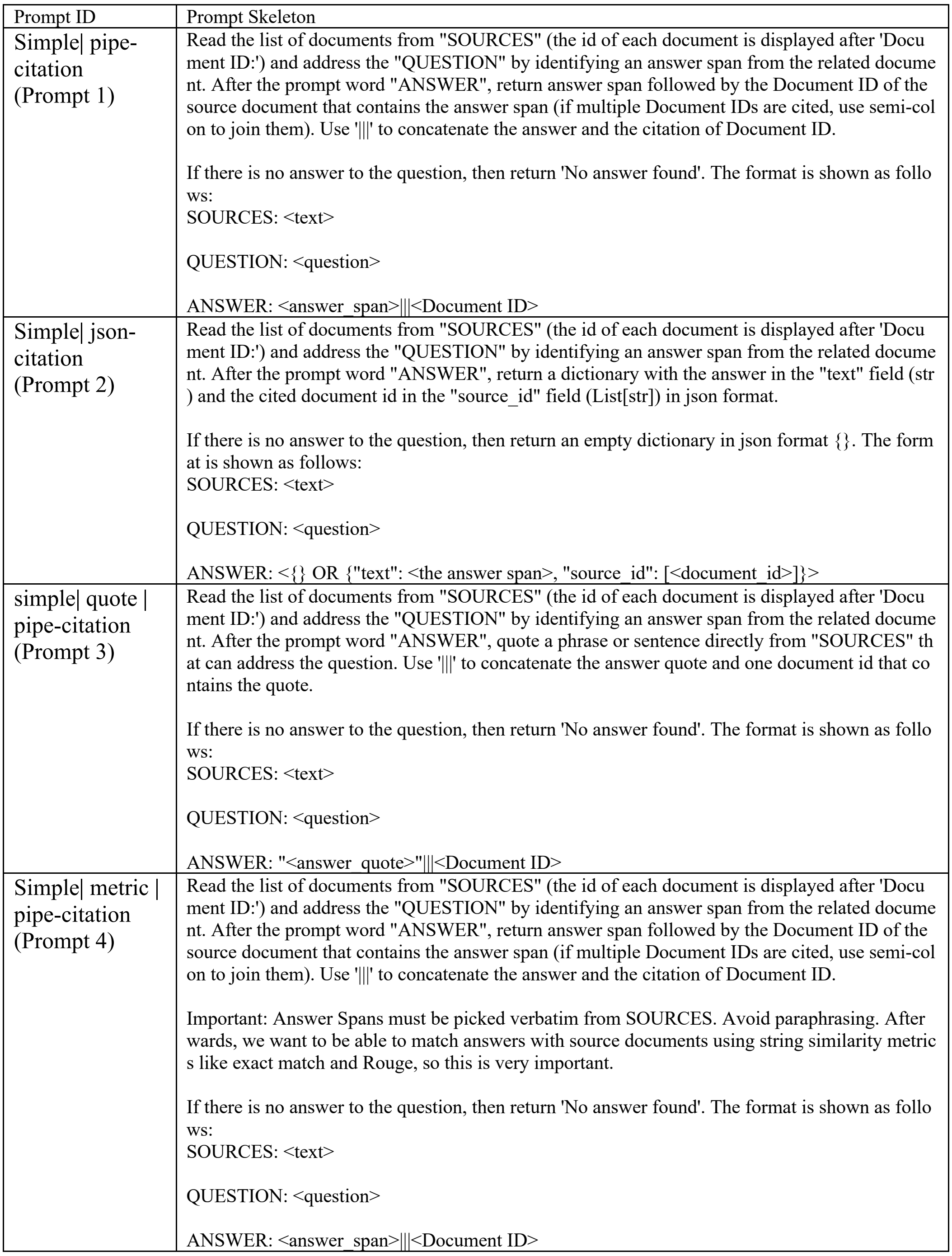


Figure 10: F1 score for all models on the Banking Public Webpages dataset

A.7 Prompt List



Prompt ID	Prompt Skeleton
Simple| pipe-citation (Prompt 1)	Read the list of documents from "SOURCES" (the id of each document is displayed after 'Document ID:') and address the "QUESTION" by identifying an answer span from the related document. After the prompt word "ANSWER", return answer span followed by the Document ID of the source document that contains the answer span (if multiple Document IDs are cited, use semi-colon to join them). Use '|||' to concatenate the answer and the citation of Document ID. If there is no answer to the question, then return 'No answer found'. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <answer_span>|||<Document ID>
Simple| json-citation (Prompt 2)	Read the list of documents from "SOURCES" (the id of each document is displayed after 'Document ID:') and address the "QUESTION" by identifying an answer span from the related document. After the prompt word "ANSWER", return a dictionary with the answer in the "text" field (str) and the cited document id in the "source_id" field (List[str]) in json format. If there is no answer to the question, then return an empty dictionary in json format {}. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <{} OR {"text": <the answer span>, "source_id": [<document_id>]}>
simple| quote | pipe-citation (Prompt 3)	Read the list of documents from "SOURCES" (the id of each document is displayed after 'Document ID:') and address the "QUESTION" by identifying an answer span from the related document. After the prompt word "ANSWER", quote a phrase or sentence directly from "SOURCES" that can address the question. Use '|||' to concatenate the answer quote and one document id that contains the quote. If there is no answer to the question, then return 'No answer found'. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: "<answer_quote>"|||<Document ID>
Simple| metric | pipe-citation (Prompt 4)	Read the list of documents from "SOURCES" (the id of each document is displayed after 'Document ID:') and address the "QUESTION" by identifying an answer span from the related document. After the prompt word "ANSWER", return answer span followed by the Document ID of the source document that contains the answer span (if multiple Document IDs are cited, use semi-colon to join them). Use '|||' to concatenate the answer and the citation of Document ID. Important: Answer Spans must be picked verbatim from SOURCES. Avoid paraphrasing. Afterwards, we want to be able to match answers with source documents using string similarity metrics like exact match and Rouge, so this is very important. If there is no answer to the question, then return 'No answer found'. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <answer_span>|||<Document ID>

Figure 11: Prompts List 1

verbose| pipe-citation (Prompt 5)	In this task you are provided with some "SOURCES" and asked a "QUESTION". Please answer the "QUESTION" based on information present in the "SOURCES" and provide corresponding citations. The specific guidelines are as follows: Guidelines: - Source documents are listed under in the "SOURCES" section and separated by '---'. The ID of each document is provided after "Document ID:". - You can extract <answer_span> from ONLY the sources defined in the "SOURCES" section below. Do not use any other sources or create new ones. - <answer_span> must be picked verbatim from "SOURCES". Avoid paraphrasing. Afterwards, we want to be able to match answers with source documents using string similarity metrics like exact match and Rouge, so this is very important. - If there isn't enough information in the "SOURCES", say "No answer found". Do not generate answers that don't use the sources below. - Always add <citation> by extracting the document ID that corresponds the source of the answer span. If multiple Document IDs are cited, use semi-colon to join them. If "No answer found", then <citation> is not needed. - Report the numbers and key facts in the sources below without modification. - After prompt 'ANSWER:' provide your answer in the following format: <answer_span>|||<Document ID> The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <answer_span>|||<Document ID>
verbose| json-citation (Prompt 6)	In this task you are provided with some "SOURCES" and asked a "QUESTION". Please answer the "QUESTION" based on information present in the "SOURCES" and provide corresponding citations. The specific guidelines are as follows: Guidelines: - Source documents are listed under in the "SOURCES" section and separated by '---'. The ID of each document is provided after "Document ID:". - You can extract <answer_span> from ONLY the sources defined in the "SOURCES" section below. Do not use any other sources or create new ones. - The <answer_span> must be extracted verbatim from the "SOURCES". DO NOT paraphrase the answer. Extract it word for word from the "SOURCES". - If there isn't enough information in the "SOURCES", return {}. Do not generate answers that don't use the sources below. - Always add <citation> by extracting the document ID that corresponds the source of the answer span. - Report the numbers and key facts in the sources below without modification. - After the prompt word "ANSWER", return a dictionary with the answer in the "text" field (str) and the cited document id in the "source_id" field (List[str]) in json format. If there is no answer to the question, then return an empty dictionary in json format {}. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <{} OR {"text": <the answer span>, "source_id": [<document_id>]}>

Figure 12: Prompts List 2

simple| no-citation (Prompt 7)	Read the list of documents from "SOURCES" (the id of each document is displayed after 'Document ID:') and address the "QUESTION" by identifying an answer span from the related document. If there is no answer to the question, then return 'No answer found'. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <answer_span>
simple| metric| no citation (prompt 8)	Read the list of documents from "SOURCES" (the id of each document is displayed after 'Document ID:') and address the "QUESTION" by identifying an answer span from the related document. Important: Answer Spans must be picked verbatim from SOURCES. Avoid paraphrasing. Afterwards, we want to be able to match answers with source documents using string similarity metrics like exact match and Rouge, so this is very important. If there is no answer to the question, then return 'No answer found'. The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <answer_span>
Verbose | no-citation (Prompt 9)	In this task you are provided with some "SOURCES" and asked a "QUESTION". Please answer the "QUESTION" based on information present in the "SOURCES" and following guidelines: Guidelines: - Source documents are listed under in the "SOURCES" section and separated by '---'. The ID of each document is provided after "Document ID:". - You can extract <answer_span> from ONLY the sources defined in the "SOURCES" section below. Do not use any other sources or create new ones. - <answer_span> must be picked verbatim from "SOURCES". Avoid paraphrasing. Afterwards, we want to be able to match answers with source documents using string similarity metrics like exact match and Rouge, so this is very important. - If there isn't enough information in the "SOURCES", say "No answer found". Do not generate answers that don't use the sources below. - Report the numbers and key facts in the sources below without modification. - After prompt 'ANSWER:' provide your answer in the following format: <answer_span> The format is shown as follows: SOURCES: <text> QUESTION: <question> ANSWER: <answer_span>

Figure 13: Prompts List 3