The impact of quantization on retrieval-augmented generation: an analysis of small LLMs

Yazan, M.; Verberne, S.; Situmeang, F.; Petroni, F.; Siciliano, F.; Silvestri, F.; Trappolini, G.

The Impact of Quantization on Retrieval-Augmented Generation: An Analysis of Small LLMs

Mert Yazan[1,2,*], Suzan Verberne[2] and Frederik Situmeang[1]

[1]Amsterdam University of Applied Sciences, Fraijlemaborg 133, 1102 CV Amsterdam, Netherlands
[2]Leiden University, Einsteinweg 55, 2333 CC Leiden, Netherlands

Abstract

Post-training quantization reduces the computational demand of Large Language Models (LLMs) but can weaken some of their capabilities. Since LLM abilities emerge with scale, smaller LLMs are more sensitive to quantization. In this paper, we explore how quantization affects smaller LLMs' ability to perform retrieval-augmented generation (RAG), specifically in longer contexts. We chose personalization for evaluation because it is a challenging domain to perform using RAG as it requires long-context reasoning over multiple documents. We compare the original FP16 and the quantized INT4 performance of multiple 7B and 8B LLMs on two tasks while progressively increasing the number of retrieved documents to test how quantized models fare against longer contexts. To better understand the effect of retrieval, we evaluate three retrieval models in our experiments. Our findings reveal that if a 7B LLM performs the task well, quantization does not impair its performance and long-context reasoning capabilities. We conclude that it is possible to utilize RAG with quantized smaller LLMs.

Keywords

Retrieval Augmented Generation, Quantization,, Efficiency, Large Language Models, Personalization

1. Introduction

Large Language Model (LLM) outputs can be enhanced by fetching relevant documents via a retriever and adding them as context for the prompt. The LLM can generate an output grounded with relevant information with the added context. This process is called Retrieval Augmented Generation (RAG). RAG has many benefits such as improving effectiveness in downstream tasks [1, 2, 3, 4], reducing hallucinations [5], increasing factuality [6], by-passing knowledge cut-offs, and presenting proprietary data that is not available to the LLMs.

The performance of RAG depends on the number, quality, and relevance of the retrieved documents [7]. To perform RAG, many tasks demand a lot of passages extracted from multiple, unstructured documents: For question-answering tasks, the answer might be scattered around many documents because of ambiguity or the time-series nature of the question (eg. price change of a stock). For more open-ended tasks like personalization, many documents from different sources might be needed to capture the characteristics of the individual. Therefore to handle RAG in these tasks, an LLM needs to look at multiple sources, identify the relevant parts, and compose the most plausible answer [7].

LLMs do not pay the same attention to their whole context windows, meaning the placement of documents in the prompt directly affects the final output [8]. On top of that, some of the retrieved documents may be unrelated to the task, or they may contain contradictory information compared to the parametric knowledge of the LLM [9]. An LLM has to overcome these challenges to leverage RAG to its advantage. Xu et al. [4] have shown that an open-source 70B LLM [10] equipped with RAG can beat proprietary models, meaning it is not necessary to use an LLM in the caliber of GPT-4 [11] to implement RAG. Still, for many use cases, it might not be feasible to deploy a 70B LLM as it is computationally demanding. To decrease the computational demand of LLMs, post-training quantization can be used. Quantization drastically reduces the required amount of RAM to load a model and can increase the inference speed by more than 3 times [12, 13]. Despite the benefits, quantization affects LLMs differently depending on their size [14]. For capabilities that are important to RAG, such as long-context reasoning, smaller LLMs (<13B) are found to be more sensitive to quantization [14].

In this paper, we investigate the effectiveness of quantization on RAG-enhanced 7B and 8B LLMs. We evaluate the full (FP16) and quantized (INT4) versions of multiple LLMs on two personalization tasks taken from the LaMP [15] benchmark. To better study how quantized LLMs perform in longer contexts, we compared the performance gap between FP16 and INT4 models with an increasing number of retrieved documents. We chose personalization because it is a challenging task to perform with RAG as it demands long-context reasoning over many documents. Contrary to question-answering where the LLM has to find the correct answer from a couple of documents, personalization requires the LLM to carefully study a person's style from all the provided documents. Our findings show that the effect of quantization depends on the model and the task: we find almost no drop in performance for OpenChat while LLaMA2 seems to be more sensitive. Our experiments show that quantized smaller LLMs can be good candidates for RAG pipelines, especially if efficiency is essential.

2. Approach

2.1. LLMs

Starting with LLaMA2-7B (Chat-hf) [10] to have a baseline, we experiment with the following LLMs: LLaMA3-8B [16], Zephyr (Beta) [17], OpenChat (3.5) [18], and Starling (LM-alpha) [19]. These models were chosen because they were the highest-ranked 7B and 8B LLMs in the Chatbot Arena Leaderboard [20] according to the Elo ratings at the time of writing. Since all models except LLaMA are finetuned

IR-RAG@SIGIR' 24: The 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, July 14–18, 2024, Washington D.C., USA
*Corresponding author.
m.yazan@hva.nl (M. Yazan); s.verberne@liacs.leidenuniv.nl (S. Verberne); f.b.i.situmeang@uva.nl (F. Situmeang)
0009-0004-3866-597X (M. Yazan); 0000-0002-9609-9505 (S. Verberne); 0000-0002-2156-2083 (F. Situmeang)

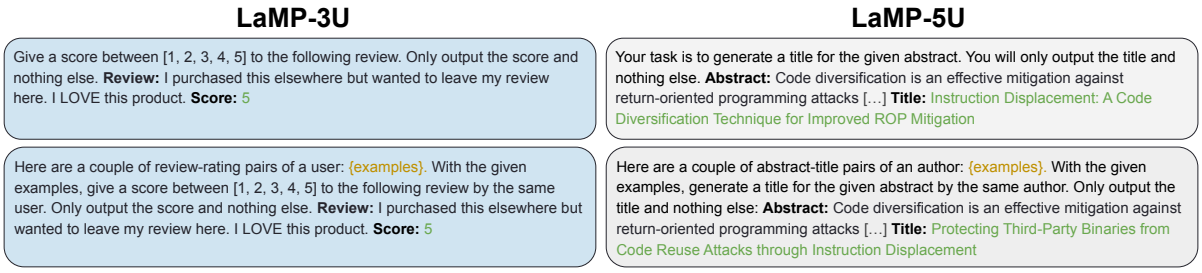


Figure 1: Prompts used for both datasets. The ones on the top represent $k = 0$ (zero-shot, no retrieved documents) and the ones on the bottom are for $k > 0$ settings (RAG). The green text is the model output. Line endings are not shown for space reasons.

variants of Mistral-7B [21], we add Mistral-7B (Instruct-0.1)[1] to our experiments too. We use Activation-aware Weight Quantization (AWQ) as it outperforms other methods [22].

2.2. Tasks and Datasets

We use the LaMP benchmark that offers 7 personalization datasets with either a classification or a generation task [15]. To represent both types of tasks, we chose one dataset from each: LaMP-3 ("Personalized Product Rating") and LaMP-5 ("Personalized Scholarly Title Generation"). LaMP-3 is composed of product reviews and their corresponding scores. For each user, one of the review–score pairs is chosen as the target and other pairs become the user profile. The LLM's task, in this case, is to predict the score given a review using the other review–score pairs of the same user. LaMP-5 aims to generate a title for an academic paper based on the abstract. In this case, the user profile consists of abstract–title pairs that demonstrate the writing style of the user (scholar). The task of the LLM is to generate a title for the given abstract by incorporating the writing style of the scholar. Those datasets were chosen because compared to the other ones, on average, they had more samples in their user profiles, and the samples were longer. Therefore, they represented a better opportunity to evaluate RAG effectiveness as the retrieval part would be trickier.

We work with the user-based splits (LaMP-3U, LaMP-5U) where the user appears only in one of the data splits [15]. The labels for the test sets are not publicly available (results can be obtained by submitting the predictions to the leaderboard) and since we did not fine-tune our models, we chose to use the validation sets for evaluation. For both datasets, we noticed that some samples do not fit in the context windows. After analyzing the overall length of the samples, we concluded that those cases only represent a tiny minority and removed data points that are not in the 0.995th percentile. For LaMP-5U, we also removed abstracts that consisted only of the text "no abstract available". There are 2500 samples in the validation sets, and we have 2487 samples left after the preprocessing steps for both datasets.

2.2.1. Evaluation

We used mean absolute error (MAE) for LaMP-3 and Rouge-L [23] for LaMP-5, following the LaMP paper [15]. Their experiments also include root mean square error (RMSE) and Rouge-1 scores, but we found that the correlation between MAE and RMSE is 0.94, and between Rouge-1 and Rouge-L is 0.99. Therefore, we do not include those metrics in our results. The prompts we use are shown in Figure 1. Even though the LLMs are instructed to output only the score or the title, we notice that some are prone to give lengthy answers such as "Sure, here is the title for the given abstract, Title: (generated title)". We apply a post-processing step on the LLM outputs to extract only the score or the title before evaluation.

2.3. Retrieval

We conduct the experiments with the following number of retrieved documents: $k \in \{0, 1, 3, 5, max\_4K, max\_8K\}$. 0 refers to zero-shot without any retrieval and *max* is the maximum number of documents that can be put into the prompt, given the context window of the LLM. LLaMA2 has a context window of 4096 tokens while other models have 8192 tokens. To make it fair, we include two options for the *max* setting: 4K and 8K. For *max_4K*, we assume that all models have a 4096 token context window, we use the original 8192 token context windows for *max_8K*. Consequently, LLaMA2-7B is not included in the *max_8K* experiments. To put it into perspective, the number of retrieved documents varies between 15-18 for *max_4K*, and between 25-28 for *max_8K* in LaMP-5U, depending on the average length of documents in the user profile. As retrievers, we evaluate BM25 [24] (BM25 Okapi) [2], Contriever [25] (finetuned on MS-Marco), and DPR [26] (finetuned on Natural Questions)[3]. Since we focus on efficiency by reducing the computational load, the retrievers are not finetuned on the datasets.

3. Results

3.1. LLMs

The results are shown in Table 1. We see the dominance of OpenChat in both datasets. Zephyr performs very close to OpenChat in LaMP-3U but falls far behind in LaMP-5U. The same can be said for Starling but reversed. Mistral-7B performs stable in both datasets albeit being slightly behind OpenChat. Overall, LLaMA2 performs the worst as it is below average in both datasets. Despite being the dominant small LLM currently, LLaMA3 is not the best for both tasks, despite performing reasonably well in LaMP-3U. Interestingly, LLaMA3 has the best zero-shot score in

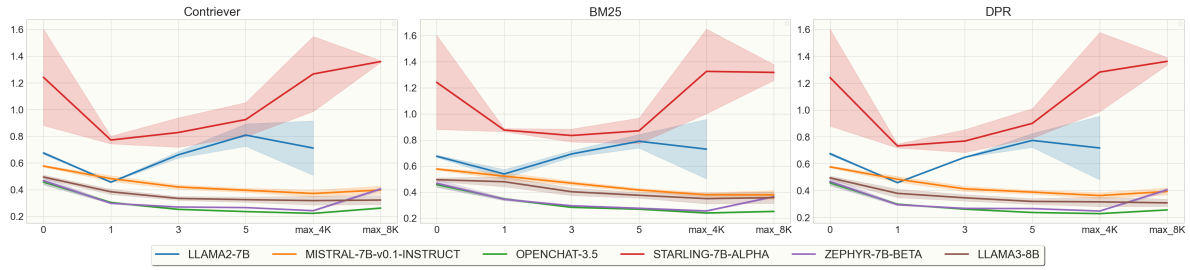
[1] Although there is an updated v0.2 version of Mistral-7B, we used v0.1 to match the other LLMs that are finetuned on it

[2] https://pypi.org/project/rank-bm25/

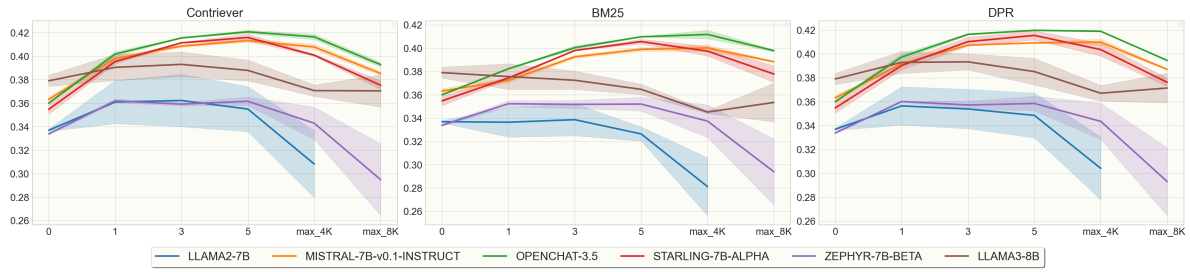
[3] https://huggingface.co/facebook/dpr-question_encoder-single-nq-base

Table 1
The absolute percentage change between FP16 and INT4 scores, using Contriever. More than a 5% drop in performance is highlighted in red. For MAE, the lower is better while the inverse is true for Rouge-L.

			LLaMA2		OpenChat		Starling		Zephyr		Mistral		LLaMA3	
Dataset	**Metric**	**k**	**FP16**	**INT4**	**FP16**	**INT4**	**FP16**	**INT4**	**FP16**	**INT4**	**FP16**	**INT4**	**FP16**	**INT4**
LaMP-3U	MAE ↓	0	0.684	+2.9%	0.440	-7.8%	1.603	+45%	0.435	-14.7%	0.569	-2.5%	0.481	-5.9%
		1	0.453	-1.1%	0.312	+5.5%	0.800	+7.1%	0.300	+1.9%	0.461	-9.3%	0.364	-10.8%
		3	0.637	-7.6%	0.256	+2.8%	0.718	-30.0%	0.273	+2.6%	0.404	-8.0%	0.320	-9.2%
		5	0.724	-23.3%	0.238	+1.8%	0.797	-32.0%	0.266	+0.8%	0.380	-8.1%	0.305	-13.0%
		max_4K	0.508	-80.2%	**0.224**	+1.8%	0.985	-57.1%	0.237	-4.4%	0.346	-14.7%	0.285	-23.1%
		max_8K	-	-	0.257	-3.9%	1.352	-1.1%	0.392	-6.3%	0.368	-16.1%	0.288	-23.4%
LaMP-5U	Rouge-L ↑	0	0.338	-0.6%	0.361	-0.5%	0.359	-2.3%	0.335	-0.4%	0.361	+1.2%	0.384	-2.5%
		1	0.380	-9.7%	0.404	-1.0%	0.397	-1.0%	0.360	+0.9%	0.400	-0.9%	0.402	-5.6%
		3	0.385	-11.6%	0.415	0.0%	0.412	-0.2%	0.360	-0.8%	0.410	-0.5%	0.404	-5.2%
		5	0.374	-10.3%	**0.422**	-0.7%	0.419	-1.4%	0.365	-2.1%	0.415	-0.7%	0.397	-4.5%
		max_4K	0.337	-16.8%	0.419	-1.1%	0.402	-0.5%	0.357	-7.7%	0.410	-1.1%	0.376	-2.6%
		max_8K	-	-	0.395	-1.0%	0.379	-1.9%	0.326	-18.7%	0.387	-0.8%	0.384	-7.2%



(a) MAE results for LaMP-3U, the lower the better



(b) Rouge-L results for LaMP-5U, the higher the better

Figure 2: Results for both datasets. The upper and lower borders of each colored area represent the quantized and not-quantized performances of the models, and the corresponding lines are the mean of both.

LaMP-5U but it struggles to improve itself with retrieval.

3.2. Quantization

How much an LLM gets affected by quantization seems to be related to how well it performs the task. OpenChat suffers almost no performance degradation from quantization. On the contrary, LLaMA2 seems very sensitive, especially when the number of retrieved documents is increased. Starling suffers no significant consequence from quantization in LaMP-5U where it performs well, but it does suffer in LaMP-3U. There also seems to be a disparity between the tasks as quantized LLMs perform much worse in LaMP-3U than in LaMP-5U.

3.3. Number of retrieved documents

Figure 2 shows that LLM performance is saturated with a couple of documents, and the improvement obtained from more is marginal. In LaMP-5U, adding more than 5 documents starts to hurt the performance: Figure 2b shows an inverse-U-shaped distribution for all LLMs except LLaMA3. For some models, performance even drops below the zero-shot setting when all the available context window is filled with retrieved documents. The LaMP-3U results (Figure 2a) continue to improve after adding more than 5 documents as can be observed from *max_4K* scores, but MAE also starts to get worse for *max_8K*.

We analyze whether a longer context window hurts the quantized variants more and find that there seems to be a peculiar relationship. INT4 LLaMA2 suffers from longer contexts, while INT4 OpenChat performs well and acts almost the same as its FP16 counterpart. INT4 Mistral and LLaMA3 act very similar to their FP16 counterparts in LaMP-5U but in LaMP-3U, they get progressively worse with more documents. Overall, quantization can increase the risk of worsened long-context capabilities but there is not a direct relationship as it is highly task and context-dependent.

3.4. Retrievers

Figure 2 shows that the three retrievers gave almost identical results, albeit BM25 being marginally behind the others. Also in LaMP-5U, the gap between the FP16 and INT4 LLaMA2 varies slightly. Other than that, the retriever model does not have a noticeable impact on the personalization tasks we experimented with. The patterns we found regarding LLMs, quantization, and the number of retrieved

Table 2
Our results compared with LaMP. Results indicated with * are not significantly lower than the reported best result (FlanT5-XXL). A quantized 7B LLM can perform on par with a larger model while being much more efficient.

	LaMP-3U (MAE) ↓	LaMP-5U (Rouge-L) ↑	Required VRAM ↓
ChatGPT [15]	0.658	0.336	?
FlanT5-XXL [15]	0.282	**0.424**	43 GB
OpenChat (FP16)	0.238	0.423*	28 GB
OpenChat (INT4)	**0.234**	0.419*	**4.2 GB**

documents are the same for all the retrievers.

3.5. Benchmark comparison

Finally, we compared our findings with the RAG results from LaMP [15]. Table 2 shows that OpenChat (Contriever, $k = 5$) can beat FlanT5-XXL in LaMP-3U and performs very close in LaMP-5U. More importantly, its quantized counterpart has very similar results. Since we do not have the per-sample scores for the baseline models from LaMP, we perform a one-sample t-test on the Rouge-L scores. The corresponding p value of 0.29 shows a non-significant difference between the results. Moreover, the results from the LaMP paper are with finetuned retrievers while our results are with non-finetuned retrievers. This indicates that a quantized-7B LLM can compete and even outperform a bigger model on personalization with RAG.

Table 2 shows how much GPU VRAM is needed to deploy each model. With this comparison, the benefit of quantization becomes more pronounced: multiple high-level consumer GPUs or an A100 is necessary for running even a 7B LLM while a mid-level consumer GPU (eg. RTX 3060) would be enough to run it with quantization. According to the scores taken from LaMP, both FlanT5-XXL and OpenChat decisively beat ChatGPT, but the authors warn that the prompts used for ChatGPT may not be ideal and may contribute to a sub-optimal performance. Therefore, our results should not be used to make a comparison with ChatGPT.

4. Discussion

Our results show that some LLMs (in particular OpenChat) can be successful in RAG pipelines, even after quantization, but the performance is LLM- and task-dependent. The method of quantization affects LLMs differently [14]. Thus, the relationship between quantization and RAG performance is not straightforward and can be studied more extensively. Still, our results indicate that when a small LLM performs the task well, its AWQ-quantized counterpart performs on par.

The differing performance of some LLMs between the datasets may be partly due to prompting. LLMs are sensitive to prompts, and a prompt that works for one LLM may not work for another one [27]. The most peculiar result is the lackluster performance of LLaMA3 in LaMP-5U. LLaMA3 is a recently released model trained with an extensive pretraining corpus [16]. It has a higher chance of seeing the abstracts presented in the LaMP-5U in its pretraining data. This may explain its superior zero-shot performance. LLMs suffer from a knowledge conflict between their parametric information and the contextual information presented through retrieval [9]. If LLaMA3 had already memorized some of the titles of the abstracts in LaMP-5U, it might result in a knowledge conflict when similar abstract-title pairs of the same author are presented. This may explain the reduced improvement in its performance with retrieval.

LLMs have been shown to struggle with too many retrieved documents [8], and our findings are in accordance. Our results indicate that more than > 5 documents do not help and can even hurt performance. From prior studies, we know that LLMs focus more on the bottom and the top of their context window [8]. We progressively put the most relevant documents starting from the bottom to the top. Therefore especially for $k = max$ settings, the less relevant documents are put on the top. This situation might hurt the LLM performance as it would focus on the most and the least related information in this case. That being said, state-of-the-art LLMs with more than 7B parameters also suffer from the same phenomenon even when not quantized [8]. Although quantization increases the risk of worsened long-context performance, we cannot conclude that it is the sole perpetrator, as this is an inherent problem for all LLMs.

5. Conclusion

We have shown that quantized smaller LLMs can use RAG to perform complex tasks such as personalization. Even though quantization might decrease the ability of LLMs to analyze long contexts, it is task- and LLM-dependent. An LLM that performs well on a task does not lose much of its long-context abilities when quantized. Thus, we conclude that quantized 7B LLMs can be the backbones of RAG with long contexts. The reduced computational load obtained from quantization would make it possible to run RAG applications with more affordable and accessible hardware. For future work, more quantization methods can be included in the experiments to see if the findings can be replicated across different methods. We can also extend the number set of k, especially between $k = 5$ and $k = max\_4K$, and change the order of the documents to better understand how quantized LLMs use their context windows.

Acknowledgments

This research is part of the project LESSEN with project number NWA.1389.20.183 of the research program NWA_ORC 202021, which is (partly) funded by the Dutch Research Council (NWO).

References

[1] J. Huang, W. Ping, P. Xu, M. Shoeybi, K. C.-C. Chang, B. Catanzaro, Raven: In-context learning with retrieval augmented encoder-decoder language models, 2023. arXiv:2308.07922.

[2] X. Ma, Y. Gong, P. He, H. Zhao, N. Duan, Query rewriting for retrieval-augmented large language models, 2023. arXiv:2305.14283.

[3] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, W. tau Yih, Replug:

Retrieval-augmented black-box language models, 2023. arXiv:2301.12652.

[4] P. Xu, W. Ping, X. Wu, L. McAfee, C. Zhu, Z. Liu, S. Subramanian, E. Bakhturina, M. Shoeybi, B. Catanzaro, Retrieval meets long context large language models, 2023. arXiv:2310.03025.

[5] Z. Proser, Retrieval augmented generation (rag): Reducing hallucinations in genai applications, 2023. URL: https://www.pinecone.io/learn/retrieval-augmented-generation/.

[6] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders, X. Jiang, K. Cobbe, T. Eloundou, G. Krueger, K. Button, M. Knight, B. Chess, J. Schulman, Webgpt: Browser-assisted question-answering with human feedback, 2022. arXiv:2112.09332.

[7] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, H. Wang, Retrieval-augmented generation for large language models: A survey, 2024. arXiv:2312.10997.

[8] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, 2023. arXiv:2307.03172.

[9] R. Xu, Z. Qi, C. Wang, H. Wang, Y. Zhang, W. Xu, Knowledge conflicts for llms: A survey, 2024. arXiv:2403.08319.

[10] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, T. Scialom, Llama 2: Open foundation and fine-tuned chat models, 2023. arXiv:2307.09288.

[11] OpenAI, Gpt-4 technical report, 2023. arXiv:2303.08774.

[12] T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer, Qlora: Efficient finetuning of quantized llms, 2023. arXiv:2305.14314.

[13] E. Frantar, S. Ashkboos, T. Hoefler, D. Alistarh, Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. arXiv:2210.17323.

[14] S. Li, X. Ning, L. Wang, T. Liu, X. Shi, S. Yan, G. Dai, H. Yang, Y. Wang, Evaluating quantized large language models, 2024. arXiv:2402.18158.

[15] A. Salemi, S. Mysore, M. Bendersky, H. Zamani, Lamp: When large language models meet personalization, 2023. arXiv:2304.11406.

[16] Meta, Introducing meta llama 3: The most capable openly available llm to date, 2024. URL: https://ai.meta.com/blog/meta-llama-3/.

[17] L. Tunstall, E. Beeching, N. Lambert, N. Rajani, K. Rasul, Y. Belkada, S. Huang, L. von Werra, C. Fourrier, N. Habib, N. Sarrazin, O. Sanseviero, A. M. Rush, T. Wolf, Zephyr: Direct distillation of lm alignment, 2023. arXiv:2310.16944.

[18] G. Wang, S. Cheng, X. Zhan, X. Li, S. Song, Y. Liu, Openchat: Advancing open-source language models with mixed-quality data, 2023. arXiv:2309.11235.

[19] B. Zhu, E. Frick, T. Wu, H. Zhu, J. Jiao, Starling-7b: Improving llm helpfulness & harmlessness with rlaif, 2023.

[20] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, I. Stoica, Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. arXiv:2306.05685.

[21] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, W. E. Sayed, Mistral 7b, 2023. arXiv:2310.06825.

[22] J. Lin, J. Tang, H. Tang, S. Yang, X. Dang, C. Gan, S. Han, Awq: Activation-aware weight quantization for llm compression and acceleration, 2023. arXiv:2306.00978.

[23] C.-Y. Lin, ROUGE: A package for automatic evaluation of summaries, in: Text Summarization Branches Out, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013.

[24] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford, Okapi at trec-3., 1994, pp. 0–.

[25] G. Izacard, M. Caron, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, E. Grave, Unsupervised dense information retrieval with contrastive learning, 2022. arXiv:2112.09118.

[26] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W. tau Yih, Dense passage retrieval for open-domain question answering, 2020. arXiv:2004.04906.

[27] M. Sclar, Y. Choi, Y. Tsvetkov, A. Suhr, Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting, 2023. arXiv:2310.11324.