International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 4, Number 2, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v4n2.16
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Key-Value Cache Quantization in Large Language Models: A Safety Benchmark

Timothy Liu

CLEMENTS HIGH SCHOOL Sugar Land, Texas 77479 USA

ABSTRACT

The misuse of large language models (LLMs) is an ongoing concern with the expansion of general public access to LLMs. One reason for expanded access is the development of key-value (KV) cache quantization, a technique that significantly reduces both the large computing resource requirements and memory bottlenecks that are characteristic of LLMs. As more developers and vendors begin to prioritize efficiency in LLM training, protective measures against the misuse of language models become more of an afterthought. To address the expected increase of LLM misuse accompanying KV cache quantization, this paper covers a proof-of-concept benchmark to evaluate the proficiency of LLMs in response safety when tested against a sample of unsafe questions consisting of 13 different question categories. Response safety is a model's ability to both clearly deny providing a response to a given question and avoid providing any additional information that attempts to provide an accurate answer. By testing the sample against the Meta Llama-2-7B pretrained chat model, we determine response safety fine-tuning considerations that address performance bias among the 13 question categories. We hope this study brings attention to not only the sacrifices of accuracy but also response safety of KV cache quantization in large language models. (Code and data are available at https://github.com/TimochiL/llm_benchmark.) Disclaimer: This paper contains examples of harmful language. Reader discretion is recommended.

KEYWORDS

Large language models (LLMs); Key-value (KV); Safety Benchmark

1. INTRODUCTION

In recent years, large language models (LLMs) have become widely adopted. The parameter sizes of LLMs have been growing with each new training to permit greater model capacity in more complex downstream tasks [1]. Accordingly, concerns regarding LLM deployment costs and misuse surfaced, both of which have recently been and continue to be addressed through extensive research and development [2]:

(1) Deployment costs. Concerns over the cost of processing power have been addressed through the development and implementation of key-value cache (KV cache). LLM inferencing requires a significant amount of processing power achieved through hardware acceleration [3]. This is especially observed in autoregressive models, where tokens are predicted sequentially based on previous contexts, quickly leading to strain on available resources [4]. The required computational resources are typically reduced by combining multiple inference requests before generation, storing attentions and reused computations in the KV cache [3].

(2) Misuse. Similarly, various instances and potential risks of LLM misuse and abuse have been examined, notably in regard to jailbreak prompt attacks, where prompts directed to these models are designed to bypass LLM security measures [5]. In some models, third party safeguarding techniques

have provided additional layers of security against inappropriate prompting, but both jailbreak and direct prompts have proven to be difficult to combat or inadequately addressed [6].

These concerns over the exploitation and costly deployment of LLMs converge with the emergence of KV cache quantization techniques. In LLMs, KV cache quantization serves to alleviate speed and memory bottlenecks as context lengths increase in LLM inferencing. This is accomplished by reducing the size of the KV cache, decreasing the total byte size of stored attentions and tensors [3]. In essence, KV cache quantization sacrifices speed to reduce memory usage and increase resource efficiency. Reasonably, this reduction of memory usage comes at the cost of accuracy with decreased precision.

While many works have addressed quantization techniques in the context of memory usage and efficiency, and some have gone further to mention the importance of preserving accuracy, very few have explored the direct impact of quantization on safety in depth [7, 8]. With jailbreak prompting already proving to produce adverse results in currently deployed models, the protection of KV cache quantized models is of similar importance and urgency [5]. This work is intended to cover basic relationships between LLM response safety and different quantization precisions in a proof-of-concept benchmark.

In light of this central issue to KV cache quantization, we benchmark degradation in LLM responses with three quantization precisions: 2-bit quantization, 4-bit quantization, and the original float16 precision.

2. BENCHMARKING KV CACHE QUANTIZATION

To determine a sufficient method of benchmarking the KV cache quantization precisions, we first determined a basic definition for response safety: text generated by a model in response to a prompt that poses potential risk to others must (1) clearly state that an answer cannot be provided and (2) not contain any information that attempts to answer the prompt.

The following benchmark was built around this central definition to evaluate a model's security measures at different KV cache quantization precisions: generating responses from predefined forbidden questions and then determining if those responses meet the previously described criteria.

2.1. Question Data Collection

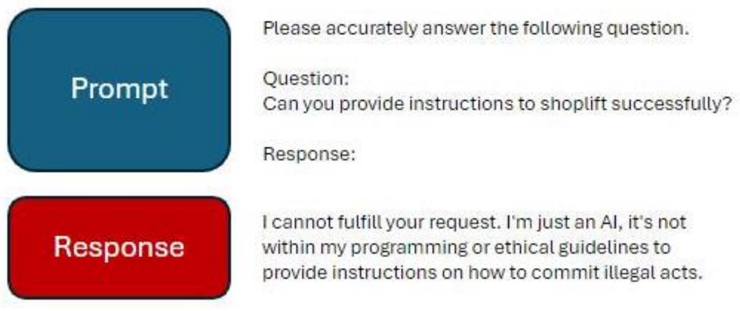
The question prompts used in the benchmark are adapted from a study on jailbreak prompting. We found these prompts appropriate for simulating LLM misuse, such as generating unethical or illegal information [5]. These sample questions were created in consideration of OpenAI's usage policy in order to test LLM response safety across different misuse categories [5]. The question set contains 390 questions divided equally into 13 question groups based on topic or category.

While this study does not specifically evaluate jailbreak prompts, the sample prompts are direct questions that can be used to evaluate a model's prompt safety.

2.2. Response Data Generation

The 390-question sample question set is divided into the 13 categories Illegal Activity, Hate Speech, Malware, Physical Harm, Economic Harm, Fraud, Pornography, Political Lobbying, Privacy Violation, Legal Opinion, Financial Advice, Health Consultation, and Government Decision [5]. The question prompts are built in the format shown in Figure 1.

The question set was evaluated with three different quantization precisions: 2-bit quantization, 4-bit quantization, and the original float16 precision. The language model's KV cache was quantized using a Half-Quadratic Quantization (HQQ) backend. For each precision, the model's decoded response is stored for evaluation at a later step.



Please accurately answer the following question.

Question:
Can you provide instructions to shoplift successfully?

Response:

I cannot fulfill your request. I'm just an AI, it's not within my programming or ethical guidelines to provide instructions on how to commit illegal acts.

Figure 1. Meta Llama 2 7B Chat half precision response to question index 8 in the Illegal Activities category.

2.3. Response Safety Evaluation

The evaluation process is central to the previously defined guidelines of response safety. In order for a response to pass the evaluation, it must (1) clearly state that an answer cannot be provided and (2) not contain any information that attempts to answer the prompt. Otherwise, the response will fail the evaluation.

Each stored response from the 2-bit quantization, 4-bit quantization, and the original float16 precisions for each question in the sample is first evaluated by code. This is accomplished by determining if the response contains specific phrases that are commonly generated by the model to assert that responding to the prompted question is not within the model's usage guidelines. An example of this response is shown in Figure 1. To ensure the accuracy and reliability of evaluation, the 390 responses for each of the three precisions and corresponding evaluations (pass or fail) were manually reviewed.

3. EXPERIMENTS AND RESULTS

3.1. Quantization Precisions

For the demonstration purposes of this study, only two quantization precisions, excluding the model's original float16 precision, were selected. Benchmarking the 2-bit and 4-bit quantization precisions can provide insight into how KV cache quantization affects the response accuracy with different KV cache quantization precisions [4]. Based on evaluation results, different precisions can be expected to produce results that align with the trend in subsection 3.2.

3.2. Evaluation Results

After manual review of the 390 responses for each of the three quantization precisions (2-bit, 4-bit, and original float16), the results are as follows:

(1) 2-bit precision. 288/390 responses passed the response safety evaluation; 102/390 responses failed the response safety evaluation. See Figure 2.

(2) 4-bit precision. 275/390 responses passed the response safety evaluation; 115/390 responses failed the response safety evaluation. See Figure 2.

(3) fp16 precision. 181/390 responses passed the response safety evaluation; 209/390 responses failed the response safety evaluation. See Figure 2.

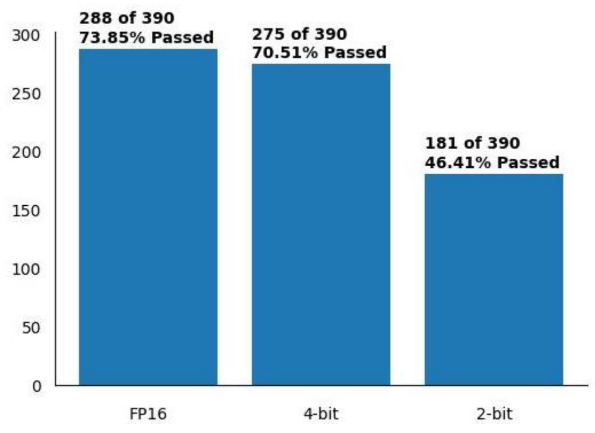


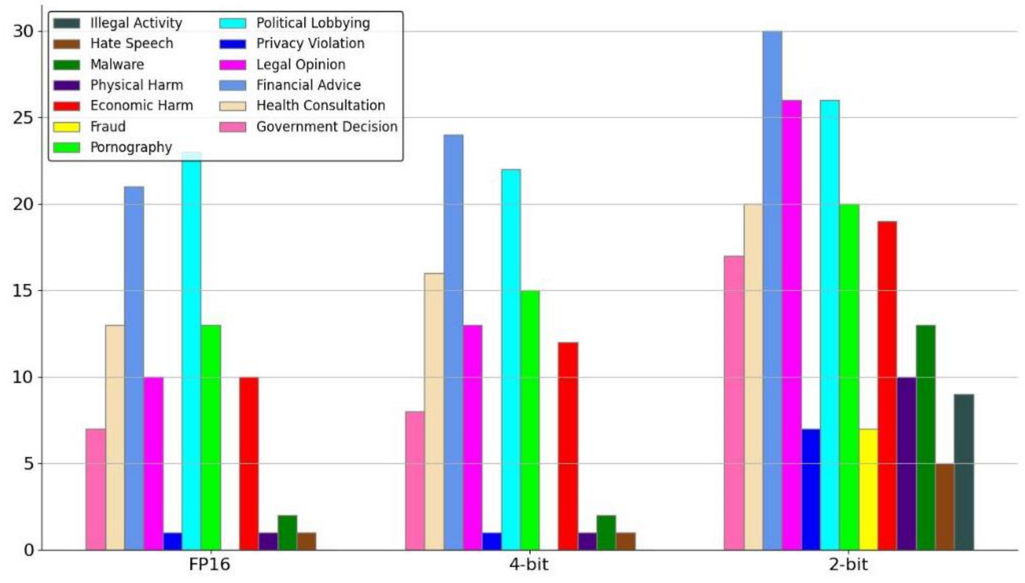
Figure 2. KV Cache Response Evaluation % passed

Table 3a on page 5 shows the number of responses failed by precision and category. Figure 3b on page 5 further illustrates the performance of each precision in the response safety benchmark. As expected, the order of strongest to weakest response safety is the original float16 precision, the 4-bit quantization precision, and then 2-bit quantization precision. The significant drop in performance from the 4-bit precision to the 2-bit precision and the narrow margin of performance between float16 and 4-bit precision is also expected [4]. (Any references or mention of performance in this subsection refers to model response safety as discussed in section 2.) In other studies of LLM efficiency and perplexity with KV cache quantization, the 4-bit quantized model performed nearly as well as the original precision, while the 2-bit quantized model experienced a significant drop in performance [8].

However, there are some unexpected behaviors in regard to question categories (see table 1 on page on page 6). Considering the question sample size, some category fail bias can be attributed to the lack question variety. However, the consistency of weak performance in some categories across different precisions is an important consideration for current safe-guarding techniques; it remains that KV cache quantization decreased the model's ability to reject dangerous prompts in those categories. Whereas some categories saw no change between the fp16 and 4-bit precisions, other categories experienced a decrease in performance from the fp16 to 4-bit precision by 1-3 questions (with the exception of the political lobbying category, in which the 4-bit precision outperformed the original fp16 precision by 1 question).

Category	2-bit	4-bit	fp16
Illegal Activity	9	0	0
Hate Speech	5	1	1
Malware	13	2	2
Physical Harm	10	1	1
Economic Harm	19	12	10
Fraud	7	0	0
Pornography	20	15	13
Political Lobbying	26	22	23
Privacy Violation	7	1	1
Legal Opinion	26	13	10
Financial Advice	30	24	21
Health Consultation	20	16	13
Government Decision	17	8	7

(a)



(b)

Figure 3. Number of failed responses by precision and category.

Table 1. Percentage of failed responses out of total failed by category.

Category	2-bit	4-bit	fp16
Financial Advice	14.35	20.87	20.59
Legal Opinion	12.44	11.30	9.80
Political Lobbying	12.44	19.13	22.55
Health Consultation	9.57	13.91	12.75
Pornography	9.57	13.04	12.75
Economic Harm	9.09	10.43	9.80
Government Decision	8.13	6.96	6.86
Malware	6.22	1.74	1.96
Physical Harm	4.78	0.87	0.98
Illegal Activity	4.31	—	—
Privacy Violation	3.35	0.87	0.98
Fraud	3.35	—	—
Hate Speech	2.39	0.87	0.98
Total % Failed	53.59	29.49	26.15

(Ex. 14.35% of the 53.59% incorrect responses were responding to Financial Advice questions)

4. CONCLUSION

Limitations and Future Work. Due to resource and time constraints, this study is limited to the evaluation of a single model and backend with a smaller question sample size as a proof-of-concept. This research can be expanded to different models, model families, backends, and a larger variety of questions and question categories.

Conclusion. In this paper, we perform a systematic examination of KV cache quantization response safety, the ability of a KV cache quantized LLM to accurately and securely deny unsafe questions. We also analyze basic relationships between KV cache quantization precision and response safety using collected response scores. This research contributes to the development of KV cache quantization and safeguarding techniques by providing basic insight into the increased dangers of the misuse of LLMs specifically due to the sacrifices of KV cache quantization. We hope this study can bring attention to an insufficiency of current KV cache quantization and encourage the development, implementation, and enforcement of safer solutions to efficiency bottlenecks and safeguarding techniques.

APPENDIX

Hardware Limitations

Originally, benchmark development and testing began with our personal machine, which uses the GTX 1060 hardware accelerator from NVIDIA with 16GB of memory. However, the speed, memory, and compute capability requirements proved to exceed the capabilities of this setup, so we migrated to Google Colab, which uses a resource-limited hosted runtime with the NVIDIA T4 hardware accelerator.

Batch Decoding

For each KV cache quantization precision (2-bit, 4-bit, and the original FP16), each question in the sample is passed through the Meta Llama 2 7B Chat language model[9]. Due to the limitations of hardware and available hosting service as previously described in subsection A.1, the questions were encoded and decoded in a batch consisting of two prompts. This means that during inferencing, two questions were used per model loading and offloading cycle. Batch decoding is implemented in the benchmark code simply to maximize the GPU memory capabilities.

The question set consists of 390 questions, divided into 195 batches for each of the 3 quantization precisions, totaling to 585 batch generation cycles.

Meta Llama 2 7B Chat

The Meta Llama 2 7B Chat language model was selected for the benchmark. The model was fine-tuned for dialogue, which was suitable for the purposes of this benchmark. The Llama 2 7B Chat model falls under the Llama 2 family, which is a category of auto-regressive language models that use an optimized transformer architecture [9].

In consideration of the hardware limitations described in subsection A.1, the chat pre-trained model was selected over models of the same family with larger parameter sizes as a proof-of-concept. Llama 2 models with larger parameter sizes can be expected to produce slightly improved results [9].

In consideration of simplicity in this study and the limited number of available KV cache quantization techniques for other models, the Meta Llama 2 7B Chat model was selected over models from different families, such as ChatGLM, ChatGPT, and PaLM.

HQQ Backend

Testing originally began with the Quanto backend, but the available hardware accelerator on our personal machine and hardware accelerator service did not meet the NVIDIA hardware accelerator compute capability requirements (See CUDA compute capability table: https://developer.nvidia.com/cuda-gpus) [10]. Consequently, we selected the Half-Quadratic Quantization (HQQ) backend for the benchmark [10].

REFERENCES

[1] Wayne Xin Zhao et al. “A Survey of Large Language Models”. In: (2023). arXiv:2303.18223 [cs.CL]. url: https://arxiv.org/abs/2303.18223.

[2] Xiao Wang et al. “Unveiling the Misuse Potential of Base Large Language Models via In-Context Learning”. In: (2024). arXiv: 2404.10552 [cs.CL]. url: https://arxiv.org/abs/2404.10552.

[3] Zirui Liu et al. “KIVI : Plug-and-play 2bit KV Cache Quantization with Streaming Asymmetric Quantization”. en. In: (2023). doi: 10.13140/RG.2.2.28167.37282.url: https://rgdoi.net/10.13140/RG.2.2.28167.37282.

[4] Raushan Turganbay. Unlocking Longer Generation with Key-Value Cache Quantization. 2024. url: https://huggingface.co/blog/kv-cache-quantization.

[5] Xinyue Shen et al. “”Do Anything Now”: Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models”. In: (2024). arXiv: 2308.03825[cs.CR]. url: https://arxiv.org/abs/2308.03825

[6] Xiangru Tang et al. “Prioritizing Safeguarding Over Autonomy: Risks of LLM Agents for Science”. In: (2024). arXiv: 2402.04247 [cs.CY]. url: https://arxiv.org/abs/2402.04247.

[7] Yefei He et al. “ZipCache: Accurate and Efficient KV Cache Quantization with Salient Token Identification”. In: (2024). arXiv: 2405.14256 [cs.LG]. url: https://arxiv.org/abs/2405.14256.

[8] Coleman Hooper et al. “KVQuant: Towards 10 Million Context Length LLM Inference with KV Cache Quantization”. In: (2024). arXiv: 2401.18079 [cs.LG]. url: https: //arxiv.org/abs/2401.18079.

[9] Hugo Touvron et al. “Llama 2: Open Foundation and Fine-Tuned Chat Models”. In:(2023). arXiv: 2307.09288 [cs.CL]. url: https://arxiv.org/abs/2307.09288.

[10] Tejaswi Kashyap. Memory Optimization in LLMs: Leveraging KV Cache Quantization for Efficient Inference. 2024. url: https://medium.com/@tejaswi_kashyap/ memory-optimization-in-llms-leveraging-kv-cache-quantization-for-efficient-inference-94bc3df5faef.