CalibQuant: 1-Bit KV Cache Quantization for Multimodal LLMs

Insu Han [*1] Zeliang Zhang [*2] Zhiyuan Wang [*3] Yifan Zhu [*2] Susan Liang [2] Jiani Liu [2] Haiting Lin [4]
Mingjie Zhao [5] Chenliang Xu [2] Kun Wan [4] Wentian Zhao [4]

Abstract

Multimodal Large Language Models (MLLMs) have demonstrated remarkable performance across diverse applications. However, their computational overhead during deployment remains a critical bottleneck. While Key-Value (KV) caching effectively trades memory for computation to enhance inference efficiency, the growing memory footprint from extensive KV caches significantly reduces throughput and restricts prolonged deployment on memory-constrained GPU devices. To address this challenge, we propose CalibQuant, a simple yet highly effective visual quantization strategy that drastically reduces both memory and computational overhead. Specifically, CalibQuant introduces an extreme 1-bit quantization scheme, complemented by novel post-scaling and calibration techniques tailored to the intrinsic patterns of KV caches, thereby ensuring high efficiency without compromising model performance. Leveraging Triton for runtime optimization, we achieve a **10x** throughput increase on InternVL models. Our method is designed to be plug-and-play, seamlessly integrating with various existing MLLMs without requiring architectural changes. Extensive experiments confirm that our approach significantly reduces memory usage while maintaining computational efficiency and preserving multimodal capabilities.

1. Introduction

Multimodal Large Language Models (MLLMs) have demonstrated strong performance across a wide range of tasks including automated caption generation, interactive storytelling, medical image diagnostics and emotion-driven captioning, to name a few (Tang et al., 2023; Bai et al., 2023; Chen et al., 2024). However, due to the quadratic computation complexity and linear memory complexity of the self-attention mechanism (Bahdanau et al., 2014), Transformer-based MLLMs present significant challenges in terms of memory consumption as the number of visual frames and the image resolution increase (Zhang et al., 2024b). The resulting surge in visual tokens further amplifies the computational burden, making the deployment of MLLMs in real-world applications increasingly difficult. To address these challenges and accelerate MLLMs, various approaches have been proposed to reduce computational costs and improve throughput. These include developing compact multimodal language models (Abdin et al., 2024), applying model pruning (Zhang et al., 2024b;a), leveraging mixture-of-experts strategies (Dai et al., 2024; Jiang et al., 2024), and optimizing KV cache mechanisms (Wan et al., 2024; Zhang et al., 2023; Zandieh et al., 2024a;b; Han et al., 2025a;b).

Among these acceleration methods, KV cache optimization has gained widespread popularity due to its scalability across different models. By storing and reusing intermediate key and value (KV) states during decoding, it allows us to avoid operations running in quadratic time in the number of tokens. The KV cache trades memory for computational efficiency. However, as the size of the KV cache increases linearly in the number of generated tokens, it causes a memory bottleneck.

Our key contributions can be summarized as:

- We introduce a 1-bit visual KV cache quantization method that retains accuracy across multiple multimodal benchmarks (COCO Caption, MMBench-Video, DocVQA).
- We apply channel-wise quantization to both key and value caches and introduce a post-softmax calibration to mitigate quantization-induced outliers.
- We implement our method using a Triton kernel and a post-scaling trick that defers dequantization, achieving up to 11.24× decoding speedup over the 16-bit baseline.

1.1. Related Work

Efficient Inference in MLLMs. Multimodal LLMs contain billions of parameters, making deployment memory-

[*]Equal contribution [1]KAIST [2]University of Rochester [3]UCSB [4]Adobe Inc. [5]Independent Researcher. Correspondence to: Insu Han <insu.han@kaist.ac.kr>.

Proceedings of the 1st Workshop on Long-Context Foundation Models, Vienna, Austria. 2024. Copyright 2024 by the author(s).

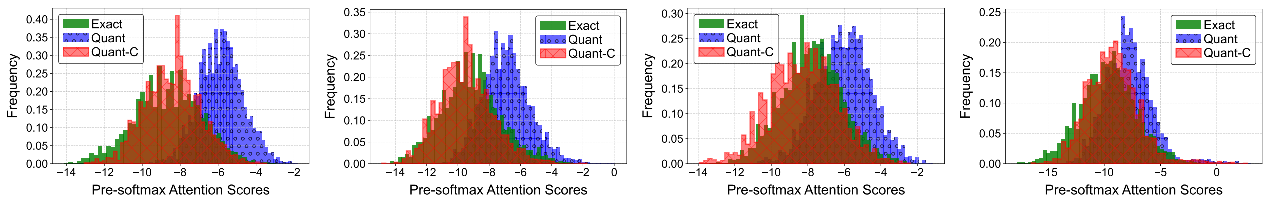


Figure 1. Distribution of entries in $qK^T/\sqrt{d}$ without quantization (Exact, green), with quantization (Quant, blue) and calibration on post-quantization (Quant-C, red) across different layers and heads.

and compute-intensive. Prior work has explored compact architectures (Lin et al., 2024; Abdin et al., 2024; Chu et al., 2023; Li et al., 2024), pruning (Zhang et al., 2024b; Shang et al., 2024), and co-optimization with hardware (Kwon et al., 2023). Despite these efforts, the self-attention mechanism—quadratic in sequence length—remains a bottleneck, especially during decoding when all past tokens must be reprocessed (Vaswani, 2017).

KV Cache Compression. KV caching avoids recomputation by storing key/value embeddings (Zhang et al., 2023), improving inference speed but increasing memory use. This becomes problematic in long-context or multi-turn settings. To reduce cache size, some methods evict less important tokens using learned importance scores (Zhang et al., 2023; Jin et al., 2024; Cai et al., 2024). Others apply quantization to reduce memory, such as asymmetric (Liu et al., 2024), per-channel (Hooper et al., 2024), or JL-transformed quantization (Zandieh et al., 2024a).

These methods target general LLMs. In contrast, our work addresses MLLMs, where visual tokens dominate cache usage and exhibit distinct statistical properties. We propose a specialized channel-wise quantization with attention-aware calibration to compress the cache while preserving multimodal reasoning.

2. CalibQuant: Low-Bit Quantization of Visual KV Cache via Calibration

Tokens in multimodal large language models (MLLMs) encompass multiple modalities. For instance, vision-language models like InternVL process both textual and visual tokens. In this work, we focus on scenarios where visual tokens dominate, meaning their sequence length exceeds that of textual tokens such as image captioning and video understanding tasks involving text. In these cases, the KV cache for visual tokens becomes a memory bottleneck. To improve the efficiency of token generation, we propose applying the uniform integer quantization to the visual KV cache.

Given a KV cache, we first determine appropriate values α and β that serve as the lower and upper bounds for all entries in the cache. The cache is then encoded into b-bit representations, where the bitwidth b controls the trade-off between memory efficiency and attention accuracy; smaller b values lead to greater information loss. Our primary objective is to quantize visual KV caches to low-bit precision, such as $b = 2$ or 1, while minimizing performance degradation. To improve the accuracy of low-bit quantization in the visual cache, we incorporate two simple yet effective strategies: channel-wise quantization and calibration.

2.1. Channel-wise Quantization with Post-scaling

Rather than using global statistics, we compute per-channel min/max values for quantization. For a key matrix $K \in \mathbb{R}^{n\times d}$, we define:

$$\alpha_i = \min_j K_{j,i}, \quad \beta_i = \max_j K_{j,i}.$$

Quantization is then applied channel-wise to both key and value caches. Unlike KIVI (Liu et al., 2024), which applies token-wise quantization to values, we find channel-wise quantization on both keys and values yields better performance in MLLMs.

2.2. Post-scaling Optimization.

Channel-wise quantization introduces separate scale and bias vectors, increasing overhead during dequantization. We reduce this by algebraically rearranging attention computation:

$$q \cdot k_{\text{deq}} = \left(q \odot \frac{\beta-\alpha}{2^b-1}\right) \cdot k_{\text{dis}} + q \cdot \alpha$$

This defers dequantization, enabling efficient use of integer-only storage and computation in CUDA. The method applies equally to value cache dequantization.

2.3. Calibration of Post-quantization

Low-bit quantization often leads to extreme reconstructed values due to codebook endpoints. This distorts attention outputs. To mitigate this, we propose a calibration step that rescales pre-softmax attention scores.

Given dequantized keys K_{deq}, we apply a linear transforma-

Methods	Bitwidth b	Evaluation Metrics (↑)							
		SPICE	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
					Model: `internvl-2.5-8b`				
Baseline	16	0.235	0.795	0.629	0.477	0.352	0.292	0.580	1.257
KIVI	4	0.232	0.8	0.635	0.481	0.355	0.289	0.580	1.255
VLCache	4	0.237	0.793	0.628	0.475	0.350	0.291	0.579	1.252
Ours	4	0.236	0.795	0.630	0.477	0.352	0.292	0.580	**1.259**
KIVI	2	0.233	0.801	0.635	0.480	0.354	0.290	0.581	1.252
VLCache	2	0.235	0.793	0.628	0.474	0.349	0.290	0.577	1.250
Ours	2	0.232	0.798	0.632	0.477	0.351	0.289	0.579	**1.254**
KIVI	1	0.230	0.784	0.617	0.464	0.339	0.285	0.572	1.194
VLCache	1	0.232	0.792	0.625	0.472	0.347	0.288	0.574	**1.236**
Ours	1	0.231	0.792	0.625	0.471	0.346	0.287	0.577	1.231
					Model: `internvl-2.5-26b`				
Baseline	16	0.244	0.813	0.653	0.499	0.374	0.300	0.594	1.321
KIVI	4	0.239	0.808	0.644	0.489	0.361	0.296	0.588	1.289
VLCache	4	0.241	0.813	0.653	0.501	0.375	0.298	0.591	1.319
Ours	4	0.243	0.813	0.654	0.500	0.374	0.300	0.594	**1.320**
KIVI	2	0.239	0.806	0.643	0.488	0.362	0.296	0.588	1.284
VLCache	2	0.238	0.809	0.648	0.495	0.371	0.295	0.587	1.302
Ours	2	0.243	0.812	0.651	0.497	0.371	0.299	0.592	**1.313**
KIVI	1	0.237	0.794	0.631	0.479	0.355	0.292	0.579	1.261
VLCache	1	0.234	0.802	0.640	0.488	0.364	0.291	0.582	**1.282**
Ours	1	0.238	0.802	0.640	0.486	0.360	0.293	0.586	1.280

Table 1. Performance evaluations on COCO Caption (Chen et al., 2015) of various KV cache compression methods for different models. Among all metrics, the CIDEr score is the most conclusive metric for image captioning, aligning closely with human judgment.

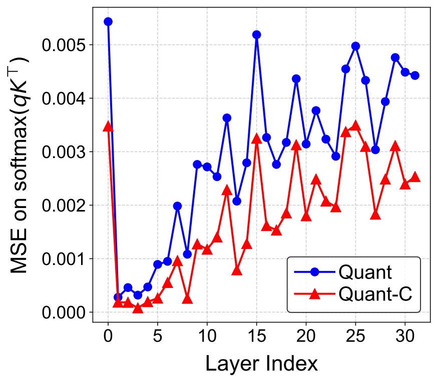


Figure 2. Mean squared error (MSE) for $\mathrm{softmax}(qK^\top/\sqrt{d})$ across multiple layers. The quantization with calibration (Quant-C, red) shows lower errors than the quantization only (Quant, blue).

tion g to the scores $qK_{\text{deq}}^T/\sqrt{d}$:

$$g(x) = \frac{\delta - \gamma + \tau_1 - \tau_2}{\delta - \gamma}(x - \gamma) + \gamma - \tau_1$$

for $x \in [\gamma, \delta]$ and parameters $\tau_1, \tau_2 \in \{0, 1, 2, 3\}$. This adjustment (Quant-C) aligns the score distribution more closely with the original (Exact), as shown in Figure 1, and substantially reduces MSE in attention scores (See Figure 2). We use fixed calibration parameters across all layers and heads. Experiments in Section 3 confirm significant performance gains over uncalibrated quantization.

3. Experiment

All experiments were performed using NVIDIA H100 GPUs with 80GB VRAM.

3.1. Image Captioning

We evaluate our quantization method on the COCO Caption dataset (Chen et al., 2015) using `llava-1.5` (7b, 13b) and `internvl-2.5` (8b, 26b). Prompts include system text and image tokens, and outputs are assessed with standard captioning metrics (BLEU, METEOR, ROUGE-L, SPICE, CIDEr). We compare against KIVI (Liu et al., 2024) and VLCache (Tu et al., 2024), adjusting VLCache's compression ratio to match memory budgets. Each method is tested at bitwidths from 8 to 1.

Table 1 shows our method consistently matches or outperforms baselines and prior work across all settings. Notably, for `llava-1.5-7b`, we match the baseline CIDEr score (1.105) at 8 bits and surpass VLCache at 1 bit (1.109 vs. 1.053). For `internvl-2.5-26b`, our method achieves top CIDEr scores of 1.32 (4-bit) and 1.313 (2-bit), outperforming both baselines.

3.2. Document Visual Question Answering

We evaluate our method on the DocVQA dataset (Mathew et al., 2021) using `internvl-2.5` models, measuring performance via Average Normalized Levenshtein Similarity

Methods	Bitwidth b	ANLS (↑)			
		`internvl-2.5`		`llava-1.5`	
		8b	26b	7b	13b
Baseline	16	0.9135	0.9242	0.2131	0.2368
KIVI	4	0.9074	0.9074	0.2127	0.2367
VLCache	4	0.9048	0.9091	0.1955	0.2196
Ours	4	**0.9138**	**0.9237**	**0.2133**	**0.2368**
KIVI	2	0.8877	0.9056	0.2116	0.2379
VLCache	2	0.8881	0.8869	0.1865	0.2089
Ours	2	**0.8937**	**0.9037**	**0.2133**	**0.2368**
KIVI	1	0.8023	0.8617	-	-
VLCache	1	**0.8558**	0.8700	0.1783	0.1960
Ours	1	0.8455	**0.8894**	**0.1927**	**0.2161**

Table 2. Performance evaluations on DocVQA (Mathew et al., 2021) by Average Normalized Levenshtein Similarity (ANLS) for different methods using LLaVA and InternVL models.

(ANLS). As shown in Table 2, our method maintains strong accuracy across 4, 2, and 1-bit settings, outperforming or matching the 16-bit baseline and competing methods like KIVI and VLCache. At 4 bits, we achieve ANLS scores of 0.9138 and 0.9237—on par with the baseline. At 2 bits, our scores remain competitive (0.8937, 0.9037), outperforming both KIVI and VLCache. Even at 1 bit, our method retains solid performance (0.8455, 0.8894), exceeding KIVI and rivaling VLCache. These results confirm the robustness of our quantization under low-bit settings.

3.3. Video Understanding

We evaluate quantization methods on the video understanding task using the MMBench-Video dataset (Fang et al., 2024), with results for `internvl-2.5` models shown in Table 3. The 16-bit full-precision baseline achieves the highest scores and serves as an upper bound.

Across all bitwidths, our method consistently outperforms KIVI and VLCache. At 8-bit and 4-bit, it nearly matches the baseline, indicating minimal degradation. Even at 2-bit, it preserves superior performance, especially for the 26b model. At 1-bit, while accuracy drops as expected, our method still achieves a higher overall score (0.8894) than VLCache (0.87) and KIVI (0.8617), showing better resilience under extreme quantization.

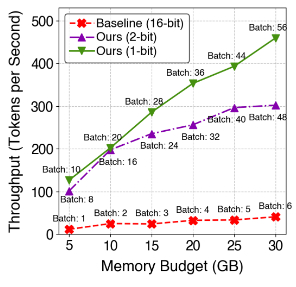
Specifically, at 8-bit, our method matches the baseline with perception/overall scores of 1.53/1.5 for the 8b model and 1.68/1.68 for the 26b model. At 4-bit, it leads all methods with scores of 1.54/1.51 (8b) and 1.69/1.68 (26b), outperforming VLCache and KIVI. At 2-bit, it maintains strong results (1.67/1.66 for 26b), exceeding VLCache (1.64/1.64) and KIVI (1.64/1.63). Performance slightly dips at 1-bit but remains competitive.

Methods	Bitwidth b	Perception (↑)		Overall (↑)	
		`internvl-2.5`			
		8b	26b	8b	26b
Baseline	16	1.53	1.68	1.50	1.68
KIVI	4	1.50	1.66	1.47	1.65
VLCache	4	1.53	1.68	1.50	1.67
Ours	4	**1.54**	**1.69**	**1.51**	**1.68**
KIVI	2	1.50	1.64	1.47	1.63
VLCache	2	**1.51**	1.64	**1.49**	1.64
Ours	2	1.50	**1.67**	1.47	**1.66**
KIVI	1	1.39	1.52	1.31	1.51
VLCache	1	**1.50**	**1.65**	**1.48**	**1.64**
Ours	1	1.49	1.63	1.45	1.62

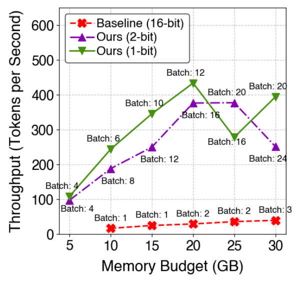
Table 3. Performance evaluations on MMBench-Video (Fang et al., 2024) by perception and overall scores for different methods using InternVL models.

3.4. Runtime Analysis

To show the impact of our quantization on decoding efficiency, we evaluated the throughput (i.e., the number of generated tokens per second) of the proposed 1-bit quantization method against a 16-bit baseline using the `internvl-2.5` models. We consider two scenarios where the lengths of the visual tokens are $n = 3328$ and 8192. We vary the maximum GPU memory from 5 GB to 30 GB and for each memory constraint we find the maximum number of batch size to fit in and measure throughput of the decoding stage. Figure 4 demonstrates that our 1-bit quantization method consistently outperforms the baseline in all memory budgets. For example, when $n = 3329$ for 8B parameter model, we achieve 126.582 tokens/s at 5 GB (versus 11.628 tokens/s for the baseline) and it scales to 459.016 tokens/s at 30 GB (versus 40.816 tokens/s for the baseline). This represents a throughput boost of approximately $9.88\times$ to $11.24\times$ over the baseline, showing the efficacy of our approach in enhancing decoding performance under constrained memory conditions. We have attached detailed throughput data as an appendix to the paper.



(a) `internvl2_5-26b`, $n = 3328$



(b) `internvl2_5-26b`, $n = 8192$

Figure 3. Throughputs of our 2-bit, 1-bit quantization and the baseline (16-bit) across various memory budgets (5 to 30 GB). The visual token lengths are $n = 3328$ and 8192. The annotated texts indicate the maximum batch size within each memory budget.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, et al. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*, 2024.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yinuo Liu, Yaochen Wang, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. *arXiv preprint arXiv:2402.04788*, 2024.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

Xiangxiang Chu, Limeng Qiao, Xinyang Lin, Shuang Xu, Yang Yang, Yiming Hu, Fei Wei, Xinyu Zhang, Bo Zhang, Xiaolin Wei, et al. Mobilevlm: A fast, strong and open vision language assistant for mobile devices. *arXiv preprint arXiv:2312.16886*, 2023.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

Xinyu Fang, Kangrui Mao, Haodong Duan, Xiangyu Zhao, Yining Li, Dahua Lin, and Kai Chen. Mmbench-video: A long-form multi-shot benchmark for holistic video understanding. *arXiv preprint arXiv:2406.14515*, 2024.

Insu Han, Praneeth Kacham, Amin Karbasi, Vahab Mirrokni, and Amir Zandieh. Polarquant: Quantizing kv caches with polar transformation. *arXiv preprint arXiv:2502.02617*, 2025a.

Insu Han, Michael Kapralov, Ekaterina Kochetkova, Kshiteej Sheth, and Amir Zandieh. Balancekv: Kv cache compression through discrepancy theory. *arXiv preprint arXiv:2502.07861*, 2025b.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *Advances in Neural Information Processing Systems*, 37:1270–1303, 2024.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. Llm maybe longlm: Self-extend llm context window without tuning. *arXiv preprint arXiv:2401.01325*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023.

Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. Mini-gemini: Mining the potential of multi-modality vision language models. *arXiv preprint arXiv:2403.18814*, 2024.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Jinfa Huang, Junwu Zhang, Yatian Pang, Munan Ning, et al. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*, 2024.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. In *International Conference on Machine Learning*, 2024.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*, 2024.

Yunlong Tang, Jing Bi, Siting Xu, Luchuan Song, Susan Liang, Teng Wang, Daoan Zhang, Jie An, Jingyang Lin, Rongyi Zhu, et al. Video understanding with large language models: A survey. *arXiv preprint arXiv:2312.17432*, 2023.

Dezhan Tu, Danylo Vashchilenko, Yuzhe Lu, and Panpan Xu. Vl-cache: Sparsity and modality-aware kv cache compression for vision-language model inference acceleration. *arXiv preprint arXiv:2410.23317*, 2024.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. *arXiv preprint arXiv:2406.18139*, 2024.

Amir Zandieh, Majid Daliri, and Insu Han. Qjl: 1-bit quantized jl transform for kv cache quantization with zero overhead. *arXiv preprint arXiv:2406.03482*, 2024a.

Amir Zandieh, Insu Han, Vahab Mirrokni, and Amin Karbasi. Subgen: Token generation in sublinear time and memory. *arXiv preprint arXiv:2402.06082*, 2024b.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

Zeliang Zhang, Xiaodong Liu, Hao Cheng, Chenliang Xu, and Jianfeng Gao. Diversifying the expert knowledge for task-agnostic pruning in sparse mixture-of-experts. *arXiv preprint arXiv:2407.09590*, 2024a.

Zeliang Zhang, Phu Pham, Wentian Zhao, Kun Wan, Yu-Jhe Li, Jianing Zhou, Daniel Miranda, Ajinkya Kale, and Chenliang Xu. Treat visual tokens as text? but your mllm only needs fewer efforts to see. *arXiv preprint arXiv:2410.06169*, 2024b.

A. Additional Notes on KV Cache/Quantization

A.1. KV Cache for Efficient Token Generation

In autoregressive sequence generation with Transformers (Vaswani, 2017), the key-value (KV) cache improves efficiency by eliminating redundant computation in self-attention. The token generation process consists of two stages: **prefill** and **decoding**.

In the **prefill** stage, the model processes a prompt of length n and computes key and value for all tokens in a form of matrices $K \in \mathbb{R}^{n\times d}, V \in \mathbb{R}^{n\times d}$ where d is the embedding dimension. These caches store past key and value embeddings, enabling efficient self-attention without recomputation in the next token generation steps.

In the **decoding** stage, the model generates one token at a time. Given a query vector $q_{\text{new}} \in \mathbb{R}^{1\times d}$ and its corresponding key-value pair $(k_{\text{new}}, v_{\text{new}})$, the KV cache updates by appending the new key and value:

$$K \leftarrow [K; k_{\text{new}}], \quad V \leftarrow [V; v_{\text{new}}]. \tag{1}$$

The attention output is then computed as:

$$\text{softmax}\left(\frac{q_{\text{new}}K^T}{\sqrt{d}}\right)V. \tag{2}$$

This incremental update reduces the time complexity of self-attention from $O(n^2 d)$ to $O(nd)$ per step. While this explanation considers a single layer and head, the same mechanism applies across multiple layers and heads in practical Transformer models.

A.2. Uniform Integer Quantization

Quantization is a process that reduces high-precision floating-point values (e.g., 32-bit floating point) to lower-precision integer representations (e.g., 8-bit integer). This compression not only reduces memory space but also accelerates computation speed, which is particularly useful in resource-constrained environments like edge devices and accelerators.

A widely used approach is the uniform integer quantization. Given a bitwidth $b > 0$ and an input value x within a range $[\alpha, \beta]$ for some $\beta > \alpha$, it is mapped to a discrete integer $x_{\text{dis}} \in \{0, 1, \ldots, 2^b - 1\}$ computed as

$$x_{\text{dis}} = \left\lfloor (x - \alpha) \cdot \frac{2^b - 1}{\beta - \alpha} \right\rceil, \tag{3}$$

where $\lfloor\cdot\rceil$ denotes the rounding operator. Representing x_{dis} requires b bits and it can reduce memory requirements significantly when b is smaller than the full precision.

To recover an approximate floating-point representation, the dequantization process converts it back as follows:

$$x_{\text{deq}} = x_{\text{dis}} \cdot \frac{\beta - \alpha}{2^b - 1} + \alpha. \tag{4}$$

This can be naturally extended to vectors or matrices by applying entry-wise.

B. Full Experiments

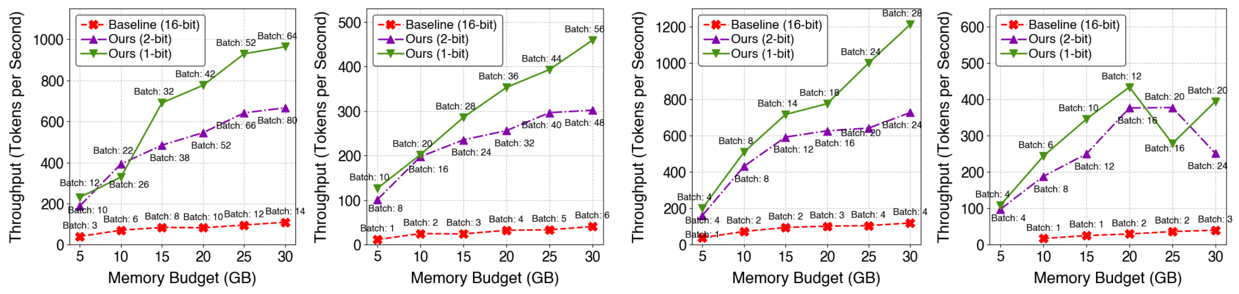
B.1. Image Captioning

We test our quantization method on the image captioning task using the COCO Caption dataset (Chen et al., 2015). We use the following models: `llava-1.5-7b`, `13b`, and `internvl-2.5-8b`, `26b`. An input prompt is constructed with a system prompt and the image tokens andevaluateses the output generation using standard captioning metrics, including BLEU, METEOR, ROUGE-L, SPICE, and CIDEr, to assess both lexical similarity and semantic coherence with ground-truth captions. We compare our quantization with other KV cache quantization or compression methods including KIVI (Liu et al., 2024) and VLCache (Tu et al., 2024). For VLCache, we adjust its hyperparameter such as compression ratio so that the total memory budget matches that in others. For each model, we evaluate the quality of generations where the bidwidth b of each method is changing from 8 to 1.

Table 4 summarizes the results. The proposed method ("Ours") consistently demonstrate competitive or superior performance across different bitwidths (8, 4, 2, and 1 bits) compared to the baseline (16-bit) and other methods such as VLCache and KIVI. In particular, for `llava-1.5-7b`, our method achieves the highest CIDEr score of 1.105 at 8 bits, matching the baseline, and improved to 1.109 at 1 bit, surpassing VLCache (1.053). Similarly, for `internvl-2.5-26b`, our method yielded the highest CIDEr score of 1.32 at 4 bits and 1.313 at 2 bits, outperforming both VLCache and KIVI. These results highlight the efficacy of our approach in maintaining or enhancing performance under reduced bitwidths, demonstrating its robustness across diverse model architectures and quantization levels.

B.2. Runtime Analysis

To show the impact of our quantization on decoding efficiency, we evaluated the throughput (i.e., the number of generated tokens per second) of the proposed 1-bit quantization method against a 16-bit baseline using the `internvl-2.5` models. We consider two scenarios where the lengths of the visual tokens are $n = 3328$ and 8192. We vary the maximum GPU memory from 5 GB to 30 GB and for each memory constraint we find the maximum number of batch size to fit in and measure throughput of the decoding stage. Figure 4 demonstrates that our 1-bit quantization method consistently outperforms the baseline in all memory budgets. For example, when $n = 3329$ for 8B parameter model, we achieve 126.582 tokens/s at 5 GB (versus 11.628 tokens/s for the baseline) and it scales to 459.016 tokens/s at 30 GB (versus 40.816 tokens/s for the baseline). This represents a throughput boost of approximately $9.88\times$ to $11.24\times$ over the baseline, showing the efficacy of our approach in enhancing decoding performance under constrained memory conditions. We have attached detailed throughput data as an appendix to the paper.



(a) `internvl2_5-8b`, $n = 3328$ (b) `internvl2_5-26b`, $n = 3328$ (c) `internvl2_5-8b`, $n = 8192$ (d) `internvl2_5-26b`, $n = 8192$

Figure 4. Throughputs of our 2-bit, 1-bit quantization and the baseline (16-bit) across various memory budgets (5 to 30 GB). We use 2 models: `internvl2_5-8b` and `internvl2_5-26b`, and the the visual token lengths are $n = 3328$ and 8192. The annotated texts indicate the maximum batch size accommodated within each memory budget.

B.3. Ablation Study

We conduct two ablation studies to validate our key contributions: the calibration technique and channel-wise quantization applied to the value cache. We replicate the image captioning task outlined in Section 3.1 using `internvl-2.5-8b` model.

Calibration of Post-quantization. In order to investigate the impact of calibration on pre-softmax attention scores discussed in Section 2.3, we fix all hyperparameters of quantization with $b = 1$ and compare evaluation metrics of quantizations with and without calibration. As presented Table 5, calibration (Quant-C) substantially outperforms the uncalibration (Quant) for all evaluation metrics. This justifies the crucial role of calibration to achieve promising performances.

Channel-wise Quantization on Value Cache. We additionally compare different quantization approaches for the value cache. In particular, we compare channel-wise to the non-channel-wise, which finds the global minimum and maximum values. In Table 5, we observe that non-channel-wise quantization performs worse than the channel-wise one. This supports our approach for the value cache.

Methods	Bitwidth b	Evaluation Metrics (↑)							
		SPICE	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
Model: `llava-1.5-7b`									
Baseline	16	0.235	0.731	0.563	0.413	0.295	0.292	0.558	1.105
VLCache	8	0.233	0.732	0.563	0.413	0.295	0.291	0.556	1.103
Ours	8	0.235	0.731	0.563	0.413	0.295	0.293	0.558	**1.105**
KIVI	4	0.235	0.730	0.563	0.413	0.296	0.293	0.558	**1.106**
VLCache	4	0.230	0.731	0.560	0.410	0.293	0.288	0.554	1.091
Ours	4	0.235	0.730	0.563	0.413	0.296	0.293	0.558	1.105
KIVI	2	0.235	0.728	0.560	0.410	0.293	0.292	0.556	**1.099**
VLCache	2	0.225	0.729	0.557	0.406	0.289	0.285	0.550	1.074
Ours	2	0.235	0.729	0.561	0.412	0.295	0.292	0.557	**1.099**
VLCache	1	0.218	0.723	0.551	0.401	0.284	0.281	0.545	1.053
Ours	1	0.227	0.739	0.571	0.419	0.300	0.287	0.558	**1.109**
Model: `llava-1.5-13b`									
Baseline	16	0.239	0.747	0.582	0.434	0.316	0.296	0.564	1.159
VLCache	8	0.236	0.752	0.585	0.436	0.316	0.294	0.565	**1.163**
Ours	8	0.239	0.747	0.582	0.434	0.316	0.296	0.565	1.159
KIVI	4	0.240	0.747	0.583	0.435	0.316	0.296	0.565	1.160
VLCache	4	0.233	0.752	0.586	0.436	0.317	0.292	0.563	**1.166**
Ours	4	0.239	0.747	0.582	0.434	0.316	0.296	0.564	1.159
KIVI	2	0.240	0.742	0.578	0.430	0.313	0.296	0.563	1.149
VLCache	2	0.227	0.753	0.586	0.436	0.316	0.288	0.559	1.150
Ours	2	0.239	0.746	0.581	0.433	0.314	0.295	0.564	**1.155**
VLCache	1	0.223	0.751	0.584	0.434	0.314	0.284	0.556	1.137
Ours	1	0.230	0.764	0.597	0.445	0.323	0.288	0.566	**1.168**
Model: `internvl-2.5-8b`									
Baseline	16	0.235	0.795	0.629	0.477	0.352	0.292	0.580	1.257
VLCache	8	0.236	0.794	0.628	0.476	0.351	0.291	0.580	1.252
Ours	8	0.236	0.795	0.630	0.476	0.351	0.292	0.580	**1.257**
KIVI	4	0.232	0.8	0.635	0.481	0.355	0.289	0.580	1.255
VLCache	4	0.237	0.793	0.628	0.475	0.350	0.291	0.579	1.252
Ours	4	0.236	0.795	0.630	0.477	0.352	0.292	0.580	**1.259**
KIVI	2	0.233	0.801	0.635	0.480	0.354	0.290	0.581	1.252
VLCache	2	0.235	0.793	0.628	0.474	0.349	0.290	0.577	1.250
Ours	2	0.232	0.798	0.632	0.477	0.351	0.289	0.579	**1.254**
KIVI	1	0.230	0.784	0.617	0.464	0.339	0.285	0.572	1.194
VLCache	1	0.232	0.792	0.625	0.472	0.347	0.288	0.574	**1.236**
Ours	1	0.231	0.792	0.625	0.471	0.346	0.287	0.577	1.231
Model: `internvl-2.5-26b`									
Baseline	16	0.244	0.813	0.653	0.499	0.374	0.300	0.594	1.321
VLCache	8	0.242	0.813	0.654	0.501	0.375	0.299	0.593	1.321
Ours	8	0.244	0.813	0.654	0.499	0.374	0.300	0.593	1.321
KIVI	4	0.239	0.808	0.644	0.489	0.361	0.296	0.588	1.289
VLCache	4	0.241	0.813	0.653	0.501	0.375	0.298	0.591	1.319
Ours	4	0.243	0.813	0.654	0.500	0.374	0.300	0.594	**1.320**
KIVI	2	0.239	0.806	0.643	0.488	0.362	0.296	0.588	1.284
VLCache	2	0.238	0.809	0.648	0.495	0.371	0.295	0.587	1.302
Ours	2	0.243	0.812	0.651	0.497	0.371	0.299	0.592	**1.313**
KIVI	1	0.237	0.794	0.631	0.479	0.355	0.292	0.579	1.261
VLCache	1	0.234	0.802	0.640	0.488	0.364	0.291	0.582	**1.282**
Ours	1	0.238	0.802	0.640	0.486	0.360	0.293	0.586	1.280

Table 4. Performance evaluations on COCO Caption (Chen et al., 2015) of various KV cache compression methods for different models. Among all metrics, the CIDEr score is the most conclusive metric for image captioning, aligning closely with human judgment.

Metrics (↑)	Channel-wise + Calibration	Without Calibration	Without Channel-wise
SPICE	0.231	0.230	0.235
BLEU_1	0.792	0.784	0.792
BLEU_2	0.625	0.617	0.609
BLEU_3	0.471	0.464	0.457
BLEU_4	0.346	0.339	0.334
METEOR	0.287	0.285	0.288
ROUGE_L	0.577	0.572	0.571
CIDEr	**1.231**	1.194	1.200

Table 5. Comparison of calibration on pre-softmax attentions and channel-wise quantization on the value cache on the COCO Caption dataset.