Efficient LLaMA-3.2-Vision by Trimming Cross-attended Visual Features

Jewon Lee¹ Ki-Ung Song¹ Seungmin Yang¹ Donguk Lim¹ Jaeyeon Kim¹ Wooksu Shin¹ Bo-Kyeong Kim¹ Yong Jae Lee² Tae-Ho Kim¹*

¹Nota Inc. ²University of Wisconsin-Madison

{jewon.lee, thkim}@nota.ai

Abstract

Visual token reduction lowers inference costs caused by extensive image features in large vision-language models (LVLMs). Unlike relevant studies that prune tokens in self-attention-only LVLMs, our work uniquely addresses cross-attention-based models, which achieve superior performance. We identify that the key-value (KV) cache size for image tokens in cross-attention layers significantly exceeds that of text tokens in self-attention layers, posing a major compute bottleneck. To mitigate this issue, we exploit the sparse nature in cross-attention maps to selectively prune redundant visual features. Our Trimmed Llama effectively reduces KV cache demands without requiring additional training. By benefiting from 50%-reduced visual features, our model can reduce inference latency and memory usage while achieving benchmark parity.

1. Introduction

Large Vision Language Models (LVLMs), such as LLaVA [18], commonly utilize self-attention-only architectures in their large language models (LLMs). These models process visual inputs as sequences of hundreds or thousands of tokens [6, 15] alongside textual prompts (see Figure 1(a)). However, their computational complexity grows quadratically with input length, limiting deployment in high-resolution or feature-rich environments.

In contrast, cross-attention-based architectures, exemplified by Flamingo [1], integrate visual features into LLMs as key-value (KV) pairs in text-visual attention computations (see Figure 1(b)). This design achieves linear computational scaling for image processing, enabling efficient processing of visual inputs. Recent advancements, such as Llama-3.2-Vision [9, 20], demonstrate their capability, positioning them as robust alternatives to self-attention-only models.

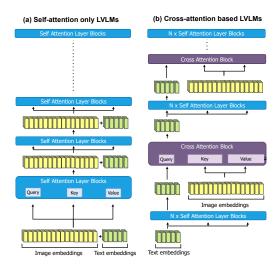


Figure 1. **Comparison of LVLM architectures.** (a) Self-attention-only models process both image and text embeddings in all attention layers. (b) Cross-attention-based models use image features exclusively for KV operations in cross-attention layers, enabling efficient multimodal integration.

Enhancing LVLM efficiency involves optimizing vision token computations by exploiting the characteristics of causal self-attention [5, 11, 12, 19, 26]. However, the study of efficient text-visual cross-attention mechanisms has not been thoroughly investigated.

In this work, we uncover the sparsity in cross-attention maps of LVLMs, revealing a consistent layer-wise pattern where the majority of visual features are selected in earlier layers, with minimal variation in subsequent layers. With these insights, we propose a novel method named **Trimmed Llama** that leverages the *sparsity* and *inter-layer resemblance of cross-attention patterns* to trim out redundant image features during inference (see Figure 2). Our approach is training-free and can achieve a minimal performance tradeoff while reducing KV cache budget and computation cost, resulting in efficient inference. Our contributions are sum-

^{*}Corresponding author.

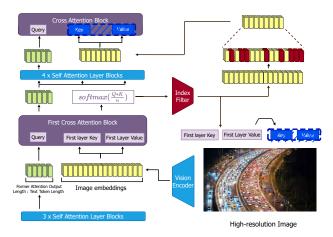


Figure 2. **Proposed method.** Image features are pruned in the first cross-attention block using a criterion derived from attention weights. The features serve as inputs for the keys and values in subsequent cross-attention layers, with the compressed keys and values stored in the KV cache (blue-shaded area).

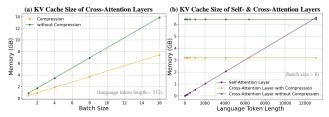


Figure 3. **KV cache memory.** (a) As batch size increases, the KV cache volume from image features grows. (b) As the language token count grows, the KV cache size in cross-attention still dominates that of self-attention, up to a certain number of tokens.

marized as follows.

- Discovery of sparsity in cross-attention: Unlike prior work solely focusing on self-attention-only LVLMs, we target recent cross-attention-based models. We identify that visual attention mechanisms in different crossattention layers exhibit a shared sparse pattern.
- Novel visual token pruning method: Based on the observed sparsity, we leverage head-wise attention scores to filter out unimportant visual features, thereby reducing KV cache overhead.
- Solid empirical validation: We test our approach across diverse benchmarks, ranging from vision-based multiplechoice questions to image grounded open-ended generation task, achieving performance on par with the original model while utilizing only 50% of the image features.

2. Cross-attention Redundancy

2.1. Motivation: Heavy Computation from Cross-Attention KV Cache

Benefit of Cross-Attention Layers in LVLMs. Higher image resolutions generally improve model performance

[7, 17, 27] but result in more image tokens. This poses a computational challenge for self-attention-only models [8, 18, 21], whose complexity grows quadratically with token count. Cross-attention architectures [1, 20], by contrast, mitigate these issues by limiting the handling of image tokens to specific layers, avoiding quadratic scaling.

KV Cache of Cross-Attention Layers. Though cross-attention layers in LVLMs improve efficiency, their KV caches are still heavy. For Llama-3.2-11B-Vision-Instruct [20] as our baseline, visual token length ranges from 1,601 tokens (e.g., 384×384 resolution) to 6,404 tokens (e.g., 720p, 1080p). Figure 3(a) shows that the KV cache memory in cross-attention layers grows significantly with batch size. Figure 3(b) shows that the KV cache size from image features in cross-attention layers surpasses that from text features in self-attention layers, up to a certain number of language tokens. Moreover, the cross-attention KV cache remains constant regardless of generation steps. This analysis emphasizes the cross-attention KV cache as a key bottleneck in model inference.

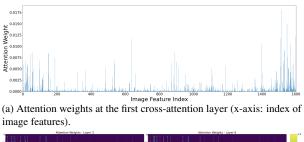
2.2. Insights from Structured and Sparse Cross-Attention Patterns

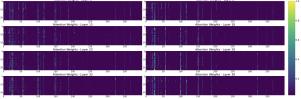
To investigate the processing of image tokens within the model, we analyze the attention patterns across multiple layers of the cross-attention mechanism. For a 384×384 image as the input of Llama-3.2-11B-Vision-Instruct, we aggregate the attention weights by summing them along two key dimensions, head-wise and query-wise. In Figure 4a, we observe that certain image tokens consistently attract attention from query tokens. This suggests that the model is selectively focusing on a relatively small subset of image features while ignoring others, indicating a potential mechanism for feature selection during cross-modal processing.

Figure 4b depicts the attention patterns across different cross-attention layers, which reveal two distinct phenomena. First, a vertically structured attention pattern is evident within each layer, signifying that attention weights are consistently allocated to the same indices, regardless of the specific language query. This behavior suggests the potential to identify globally salient indices. Second, there is an interlayer consistency in attention patterns; the distribution of attention remains remarkably stable across successive layers. This lack of substantial variation implies that the model's cross-attention mechanism might converge to a fixed pattern early, with minimal variation afterward. These findings contribute to our understanding of how cross-modal attention mechanisms operate, particularly in visual-linguistic tasks, and suggest avenues to optimize inference efficiency.

Model	Method	SEED-Bench Image	MME	MME-cog.	MME-per.	MMVP	LLaVA- Bench
Llama-3.2-V Inst. 11B	Original	72.6	1685.9	307.1	1378.7	46.7	88.3
	$K_{\rm ratio} = 0.25$	72.3	1687.3	312.5	1374.8	47.3	88.1
		61.5%	65.1%	65.1%	65.1%	61.9%	72.7%
	$K_{\rm ratio} = 0.20$	72.1	1682.8	307.9	1374.9	47.3	87.3
		51.7%	53.7%	53.7%	53.7%	46.7%	61.6%
	$K_{\rm ratio} = 0.15$	71.4	1669.1	305.7	1363.4	45.3	88.3
		40.7%	41.6%	41.6%	41.6%	37.2%	49.1%
	$K_{\rm ratio} = 0.10$	69.8	1675.8	297.86	1378.0	39.3	84.9
		28.2%	28.9%	28.9%	28.9%	26.7%	35.3%
	$K_{\rm ratio} = 0.05$	62.3	1586.1	297.5	1288.6	33.0	83.5
		14.2%	15.5%	15.5%	15.5%	14.2%	20.5%
Llama-3.2-V Inst. 90B	Original	76.3	2029.2	423.9	1605.3	56.7	92.0
	$K_{\text{ratio}} = 0.25$	75.9	2034.2	444.6	1589.6	54.7	93.9
		74.2%	71.6%	71.6%	71.6%	71.3%	75.2%
	$K_{\rm ratio} = 0.15$	75.4	2065.2	423.9	1641.3	56.6	92.2
		51.0%	48.4%	48.4%	48.4%	45.7%	52.4%

Table 1. **Performance of Llama-3.2-Vision-Instruct on various benchmarks.** The value in grey denotes the mean percentage of remaining image features for each K_{ratio} . Bold values denote performance comparable to or better than the full-cache baseline.





(b) Cross-attention weight patterns across different layers (x-axis: index of image features; y-axis: index of text query features).

Figure 4. **Aggregated cross-attention weights.** (a) The attention weights at the first cross-attention layer are summed over attention heads and text queries. (b) The attention weights for each cross-attention layer are summed over heads and visualized with the sequence length clipped to 400 for better visibility. Over different layers, specific image tokens consistently attract more attention from query tokens, indicating a structured sparse pattern.

3. Trimming Visual Features in Cross-Attention-Based LVLMs

With insights from Section 2.2, our method leverages *head-wise* attention scores accumulated across language sequences to remove unimportant image features. For each attention head of the first cross-attention layer, the top-k most salient image features are identified based on their attention scores. Then, the union of these top-k sets across all heads is merged to determine the final selection of important features, ensur-

ing a focused representation of the image.

Precisely, the sum of query-wise attention weights is computed for the cumulative importance score $p_i^h = \sum_{j=0}^{m-1} \alpha_i^{j,h}$ (m: input query tokens, i: image feature index, j: query token index, h: head index, L: set of image feature indices, H: set of head indices, and $\alpha_i^{j,h}$: attention score of image feature). The importance scores p_i^h are aggregated into P_h , where $p_i^h \in P_h$, $\forall i \in L$, $\forall h \in H$.

Each image feature f_i^h at head h belongs to $\mathbb{F}_h = \{f_i^h \mid i=0,1,2,\ldots,|L|-1\}$. Critical tokens are selected by evaluating the importance of image tokens uniquely per head, such that $\mathbb{T}_h = \{i \mid p_i^h \in \text{top-k}(P_h, \text{top-k} = K_{ratio} \cdot |L|)\}$. The selected features for each head are $\mathbb{F}_{select}^h = \{f_i^h \mid \mathbb{1}_{\mathbb{T}_h} = 1\}$, and the total set of selected feature is $\mathbb{F}_{select} = \bigcup_{h \in H} \mathbb{F}_{select}^h$. Here, K_{ratio} is an input parameter that determines the fraction of the feature space selected based on top-k criteria for each attention head.

4. Experimental Setup

Models. We used the Llama-3.2-Vision-{11B, 90B}-Instruct models [9] in our experiments. Compared to other cross-attention-based LVLMs like Open-Flamingo [2] and Otter [14], the Llama-3.2-Vision family exhibits superior capabilities by leveraging a significantly larger amount of visual tokens. Due to the limited visual tokens and lower performance of earlier models, we focused on the recent Llama-3.2-Vision for better compression results.

Benchmark Datasets. We used various benchmark datasets [28] to assess its performance in vision-language tasks. Specifically, we conducted experiments on MMVP [24] for binary classification question answering focusing on CLIP [22] blind pairs, MME [10] for fine-grained task-driven benchmark, SEED-Bench [13] as vision-grounded multiple

Feature Util.	Batch 1	Batch 4	Batch 8	Batch 16	Batch 32
100% (Orig.)	95.1ms	358.4ms	751.7ms	1648.6ms	3940.0ms
50.9%	91.2ms (4.1%)	332.9ms (7.1%)	660.8ms (12.1%)	1414.7ms (14.2%)	3165.5ms (19.7%)
39.6%	91.0ms (4.3%)	317.5ms (11.4%)	646.3ms (14.0%)	1347.7ms (18.3%)	2916.7ms (26.0%)

Table 2. Inference latency of the backbone LLM evaluated across different feature utilization ratios. Parenthetical values indicate the relative latency reduction compared to the baseline model. The experiment was conducted using Llama-3.2-11B-Vision-Instruct on an A100 80GB GPU.

Method	SEED-Bench Image	MME	MMVP	LLaVA-Bench
Ours (budget < 0.50)	71.4	1669.1	47.3	88.3
Ours (budget < 0.50)	40.7%	41.6%	46.7%	49.1%
Random (0.50)	67.00	1537.1	44.7	83.2
Spatial (0.50)	71.8	1627.7	46.0	85.9

Table 3. Comparison of visual token pruning methods at a compression ratio of 50%. The value in grey denotes the mean ratio of remaining image features used during generation.

choice question answering benchmark and LLaVA-Bench [18] for open-ended vision-grounded generation.

5. Results

Main Results. Table 1 demonstrates that our method consistently outperforms or achieves comparable performance while leveraging 40~50% of the image features. Notably, the pruning ratios are *adaptively allocated* for each task, as evidenced by LLaVA-Bench, an open-ended generation task utilizing more image features compared to other benchmarks. Figure 5 shows that our approach effectively maintains performance across benchmarks, even as the compression ratio increases. Figure 6 shows that our method effectively preserves salient visual information (e.g., text cues or everyday objects) while pruning unimportant features.

Latency Reduction. Table 2 shows the inference speedup for the first token when utilizing $40{\sim}50\%$ of the image features. Our method reduces latency by pruning key and value inputs in the cross-attention layers. Since image features are pruned after the first cross-attention layer, both the key-value projections and the attention operations are consequently reduced. Furthermore, the impact of the reduction grows more significant with larger batch sizes.

KV Cache Memory Reduction. By removing image features after the initial cross-attention layer, we achieve optimal computational efficiency in reducing FLOPs. Figure 3 shows the impact of our approach on KV cache memory (indicated with 'Compression'). The amount of reduced cache size is amplified with larger batch sizes, highlighting the efficiency of our method under high-throughput conditions. Ablation Study. We evaluate the impact of using attention weights as a visual feature pruning criterion. As shown in Table 3, random sampling—where image features are selected randomly—fails to achieve consistent performance across all benchmarks. Additionally, we investigated spatial

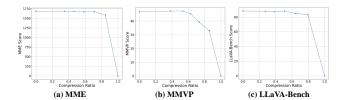


Figure 5. **Results under different compression ratios.** Even with up to 50% reduction of visual features, our method retains the performance of the original model.







Figure 6. **Visualization of compression.** Purple patches indicate features trimmed by our method.

sampling, a structured approach that selects image tokens in a fixed pattern—every alternate feature index—to approximate holistic image representations. While spatial sampling showed competitive performance on multiple-choice question benchmarks, which we consider less challenging due to the availability of explicit answer choices, it underperformed in more demanding task-driven evaluations (e.g., MME) and open-ended generative tasks (e.g., LLaVA-Bench).

6. Related Work

LVLMs. LLaVA [18] and its relevant models [8, 21] combine an LLM with a vision encoder to integrate visual modality features. Similarly, also using an LLM backbone, the recent LLaMA-3.2-Vision [20] leverages visual features through cross-attention layers, a design inspired by Flamingo [1]. This design replaces compute-heavy self-attention layers with cross-modality interaction. We aim to optimize cross-attention-based LVLMs, a relatively under-explored. Visual Token Reduction for Efficient LVLMs. Processing visual features efficiently in LVLMs remains a key challenge. Strategies such as token compression [3, 23] and sparse attention [4, 16, 26, 29] optimize visual inputs for the LLM backbones. Examples include FastV [5], which exploits sparsity in higher-layer visual attention. ElasticCache and LOOK-M [19, 25], which merge KV caches to reduce overhead and ZipVL [12], employing mixed-precision KV caching and importance-based sparse attention for computational gains. However, these advances predominantly target self-attentionbased architectures, leaving cross-attention mechanisms underexplored. Moreover, the non-causal relationship between visual inputs and language queries renders direct application of these methods in cross-attention infeasible. Our approach targets effective reducing of cross-attention KV cache without compromising model performance.

7. Conclusion

We introduce Trimmed-Llama, a plug-and-play inference optimization method for cross-attention-based LVLMs, leveraging insights from cross-attention weight patterns. By identifying and exploiting inter-layer repetitive cross-attention patterns, our method trims redundant KV caches and reduces computational overhead without additional training.

Acknowledgement

This research was supported by Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT (MSIT, Korea) & Gwangju Metropolitan City.

References

- [1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022. 1, 2, 4
- [2] Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. Openflamingo: An opensource framework for training large autoregressive visionlanguage models. arXiv preprint arXiv:2308.01390, 2023.
- [3] Mu Cai, Jianwei Yang, Jianfeng Gao, and Yong Jae Lee. Matryoshka multimodal models. *arXiv preprint arXiv:2405.17430*, 2024. 4
- [4] 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. 4
- [5] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pages 19–35. Springer, 2025. 1, 4
- [6] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. arXiv preprint arXiv:2412.05271, 2024. 1
- [7] Wenliang Dai, Nayeon Lee, Boxin Wang, Zhuoling Yang, Zihan Liu, Jon Barker, Tuomas Rintamaki, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Nvlm: Open frontier-class multimodal llms. arXiv preprint arXiv:2409.11402, 2024. 2
- [8] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models. arXiv preprint arXiv:2409.17146, 2024. 2,

- [9] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024. 1, 3
- [10] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Meng-dan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. Mme: A comprehensive evaluation benchmark for multimodal large language models, 2024. 3
- [11] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023. 1
- [12] Yefei He, Feng Chen, Jing Liu, Wenqi Shao, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Zipvl: Efficient large vision-language models with dynamic token sparsification and kv cache compression. arXiv preprint arXiv:2410.08584, 2024. 1, 4
- [13] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint* arXiv:2307.16125, 2023. 3
- [14] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Mimic-it: Multi-modal in-context instruction tuning. arXiv preprint arXiv:2306.05425, 2023. 3
- [15] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. arXiv preprint arXiv:2408.03326, 2024.
- [16] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. arXiv preprint arXiv:2404.14469, 2024. 4
- [17] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26689–26699, 2024. 2
- [18] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. Advances in neural information processing systems, 36, 2024. 1, 2, 4
- [19] Zuyan Liu, Benlin Liu, Jiahui Wang, Yuhao Dong, Guangyi Chen, Yongming Rao, Ranjay Krishna, and Jiwen Lu. Efficient inference of vision instruction-following models with elastic cache. *arXiv preprint arXiv:2407.18121*, 2024. 1, 4
- [20] Meta. Llama-3.2-11b-vision-instruct, 2024. Available at https://huggingface.co/meta-llama/Llama-3.2-11B-Vision-Instruct.1,2,4
- [21] OpenGVLab. Internvl2-8b, 2024. Available at https:// huggingface.co/OpenGVLab/InternVL2-8B. 2,
- [22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervi-

- sion. In *International conference on machine learning*, pages 8748–8763. PMLR. 2021. 3
- [23] 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. 4
- [24] Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9568–9578, 2024. 3
- [25] Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. Look-m: Lookonce optimization in kv cache for efficient multimodal longcontext inference. arXiv preprint arXiv:2406.18139, 2024.
- [26] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. arXiv preprint arXiv:2309.17453, 2023. 1, 4
- [27] Ruyi Xu, Yuan Yao, Zonghao Guo, Junbo Cui, Zanlin Ni, Chunjiang Ge, Tat-Seng Chua, Zhiyuan Liu, Maosong Sun, and Gao Huang. Llava-uhd: an lmm perceiving any aspect ratio and high-resolution images. *arXiv preprint arXiv:2403.11703*, 2024. 2
- [28] Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Lmmseval: Reality check on the evaluation of large multimodal models, 2024. 3
- [29] 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. 4

A. Appendix

A.1. Cross-attention Weight Patterns

Figure 7 presents the vertical patterns observed in the cross-attention layers and inter-layer similarities. The attention weights are extracted from samples of LLaVA-Bench's image resized to 384×384 and corresponding instruction, offering a visualization of attention distributions across layers.

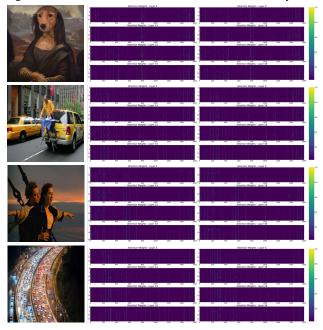


Figure 7. Additional results of cross-attention weights. (Left) Images utilized for the extraction of attention weights. (Right) Cross-attention weight patterns of different layers from corresponding image (x-axis: the index of image features; y-axis: the index of text query features).

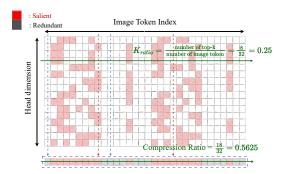


Figure 8. **Visualization of our token pruning algorithm.** The 2D grid represents image token indices (x-axis) and attention heads (y-axis). The final compression ratio is determined by the fraction of tokens not selected as salient.

A.2. Visual Token Pruning Algorithm

Figure 8 illustrates our proposed algorithm using a conceptual example. Here, the attention map is reduced to two

dimensions by summing along the language query dimension. The hyperparameter K_{ratio} denotes the proportion of salient image tokens retained per attention head. In this example, $K_{ratio} = 0.25$ is used, meaning that each head selects the top-k (k = 0.25 * 32 = 8) most attended image tokens.

Based on these selections, an image token is deemed salient if selected by any attention head (indicated by vertical red arrows in the figure), whereas redundant tokens are those not selected by any head (represented by vertical blue arrows). This algorithm effectively captures head-specific token importance, ensuring adaptive attention token filtering across different attention heads.

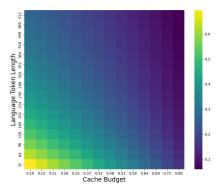


Figure 9. **Visualization of computational cost reduction.** The heatmap illustrating the theoretical FLOPs reduction ratio is presented with varying KV cache budgets and input sequence lengths.

A.3. Computational Cost Estimation

The reduction of computational cost achieved by our approach is presented below. The computation covers multihead self-attention and cross-attention modules along with feed-forward networks. The image feature is pruned after the first layer with dynamic budget ratio R produced by the compression method. For estimation, n denotes the language token length, m and d denote the feature dimension of the MLP and attention module, and n_k denotes the length of the image feature. The number of cross-attention and self-attention layers is indicated as C and S, respectively.

FLOPs_{self}:
$$4nd^2 + 2n^2d + 2ndm$$

FLOPs_{cross}: $2nd^2 + 2n_kd^2 + 2nn_kd + 2ndm$

FLOPs_{prune}: $2nd^2 + 2n_kRd^2 + 2nn_kRd + 2ndm$

The theoretical reduction ratio is then calculated as follows. Figure 9 shows a heatmap visualization with different budget ratios R and input sequence lengths n.

$$1 - \frac{S*FLOPs_{self} + FLOPs_{cross} + (C-1)*FLOPs_{prune}}{S*FLOPs_{self} + C*FLOPs_{cross}}$$