PLPHP: Per-Layer Per-Head Vision Token Pruning for Efficient Large Vision-Language Models

Anonymous ACL submission

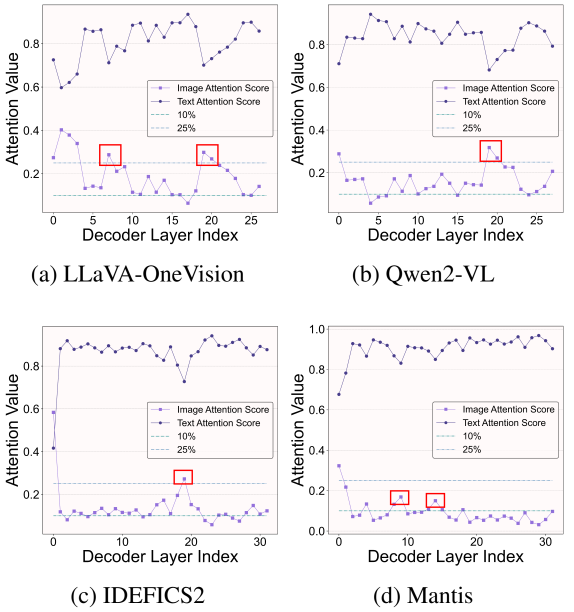
Abstract

Large Vision-Language Models (LVLMs) have demonstrated remarkable capabilities across a range of multimodal tasks. However, their inference efficiency is constrained by the large number of visual tokens processed during decoding. To address this challenge, we propose **P**er-**L**ayer **P**er-**H**ead Vision Token **P**runing (**PLPHP**), a two-level fine-grained pruning method including Layer-Level Retention Rate Allocation and Head-Level Vision Token Pruning. Motivated by the *Vision Token Re-attention* phenomenon across decoder layers, we dynamically adjust token retention rates layer by layer. Layers that exhibit stronger attention to visual information preserve more vision tokens, while layers with lower vision attention are aggressively pruned. Furthermore, PLPHP applies pruning at the attention head level, enabling different heads within the same layer to independently retain critical context. Experiments on multiple benchmarks demonstrate that PLPHP delivers an 18% faster decoding speed and reduces the Key-Value Cache (KV Cache) size by over 50%, all at the cost of 0.46% average performance drop, while also achieving notable performance improvements in multi-image tasks. These results highlight the effectiveness of fine-grained token pruning and contribute to advancing the efficiency and scalability of LVLMs.

1 Introduction

Recent advancements in Large Vision-Language Models (LVLMs) have established them as a prominent research focus in multimodal learning. Numerous open-source implementations have demonstrated remarkable capabilities across various tasks, including multimodal understanding and reasoning.

Nevertheless, LVLMs face computational inefficiency challenges, mainly due to converting visual inputs into lengthy vision token sequences, ranging from thousands to tens of thousands. Previous studies (Chen et al., 2024b; Lin et al., 2024b) find that LVLMs exhibit lower attentions to vision tokens in deeper layers compared to shallower layers, thus a certain amount of vision tokens are pruned at specific shallow layers, and the *same* tokens are pruned in *all* subsequent layers. However, such coarse-grained pruning strategies often lead to a significant performance decline in complex tasks that require comprehensive visual information, including open-ended VQA and image captioning. To address this challenge, in this work, we propose **P**er-**L**ayer **P**er-**H**ead Vision Token **P**runing (**PLPHP**), a plug-and-play adaptive fine-grained vision token pruning method that includes two levels: **1) Layer-Level Retention Rate Allocation** and **2) Head-Level Vision Token Pruning**, significantly reducing the performance loss associated

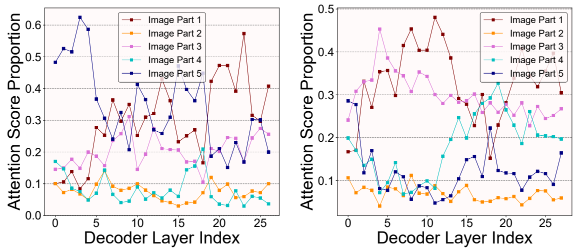


(a) LLaVA-OneVision (b) Qwen2-VL

(c) IDEFICS2 (d) Mantis

Figure 1: **The phenomenon of Vision Token Re-attention in different LVLMs.** Various LVLMs demonstrate the phenomenon of refocusing on images within deep decoder layers. In these layers, the attention scores corresponding to vision tokens increase, as indicated by the red boxes highlighted in the figure.

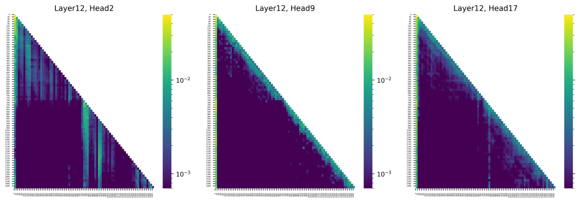
with pruning.



(a) LLaVA-OneVision (b) Qwen2-VL

Figure 2: **The proportion of attention scores received by different parts of the same image varies across different decoder layers.** Each polyline in the figure represents the proportion of attention scores for the corresponding group of tokens across different decoder layers.

The first level of our proposed method stems from our analysis of the attention to visual information in the deeper layers of LVLMs. As shown in Figure 1, we observe the phenomenon of *Vision Token Re-attention* across LVLMs with different architectures where attention scores of vision tokens are initially high and decrease in intermediate layers, but rise again in certain deeper layers. This indicates that LVLMs *do not* always disregard vision tokens in deep layers, thus we need to dynamically adjust the pruning rate to accommodate the unique attention patterns of different decoder layers.



(a) Head 2 (b) Head 9 (c) Head 12

Figure 3: **Visualization of attention maps in various attention heads.** Different heads within the same decoder layer exhibit different attention patterns.

The second level of our method is motivated by an in-depth investigation on the variations in vision token attention across different decoder layers. As shown in Figure 2, we divide the vision tokens into five groups based on their spatial relationships and plot the proportions of attention scores for each group across different layers. We observe that different parts of the same input image receive varying proportions of attention across different decoder layers, suggesting that each decoder layer specializes in processing distinct contexts. Furthermore, we conduct a more granular analysis at the level of attention heads. As illustrated in Figure 3, different attention heads within the same decoder layer exhibit distinct patterns of attention, demonstrating that the focus on different contexts occurs at the attention head level. This observation suggests that the unique contextual information processed by each attention head should be independently preserved during the pruning process to maintain model performance.

Built on these motivations, by dynamically adjusting retention rates according to layer-specific attention patterns layer by layer, PLPHP retains more vision tokens in layers where image attention scores are high, while aggressively pruning layers with low attention scores. Additionally, through head-level independent context pruning, PLPHP preserves the most critical contextual information for each attention head, leading to performance improvements. Comprehensive evaluations across multiple model architectures and various benchmarks demonstrate the effectiveness of PLPHP. Our method achieves over 50% compression of the KV cache, over 18% decoding acceleration, and only a 0.46% average performance degradation with notable improvements on multi-image tasks.

The contributions of our work can be summarized into the following three points:

- We uncover the widespread phenomenon of *Vision Token Re-attention* through investigations on various LVLMs, which could be a significant factor leading to the performance degradation of existing pruning methods.
- We propose PLPHP, a plug-and-play adaptive fine-grained vision token pruning method that improves the performance of pruned models significantly while maintaining high computational efficiency.
- We conduct extensive experiments across multiple benchmarks and model architectures, validating the superiority of our proposed method.

2 Related Work

2.1 Large Vision-Language Models

Recent advancements in LVLMs significantly enhanced multimodal content understanding. Liu et al. (2023) developed LLaVA, an early general-purpose multimodal model integrating CLIP (Radford et al., 2021) with language models. Subsequent innovations include Qwen-VL (Bai et al., 2023; Wang et al., 2024b), which enhanced visual processing with a specialized visual receptor and multilingual corpus, and Mantis by Jiang et al. (2024), which improved multi-image reasoning through academic-level instruction tuning. Laurençon et al. (2024) introduced IDEFICS, trained on the OBELICS dataset of interleaved image-text documents. Unified approaches by Li et al. (2024b) and Li et al. (2024a) achieved state-of-the-art performance in single-image, multi-image, and video tasks. However, LVLMs still face computational challenges due to the high number of visual tokens during inference, underscoring the need for more efficient inference.

2.2 Efficient Multimodal Large Language Models

To optimize the computational efficiency of LVLMs during inference, works such as MobileVLM (Chu et al., 2023), Tinygpt-V (Yuan et al., 2023), MoE LLaVA (Lin et al., 2024a), and LLaVA-Phi (Zhu et al., 2024) proposed more efficient model architectures. Meanwhile, Li et al. (2023) introduced a model-distillation approach that transfers knowledge from large vision-language models (VLMs) to smaller, lighter counterparts. Q-VLM (Wang et al., 2024a) provided a post-training quantization framework for LVLMs by mining cross-layer dependencies to improve quantization efficiency. From the perspective of token pruning, TokenPacker (Li et al., 2024c), Dynamic-LLaVA (Huang et al., 2024b), and AVG-LLaVA (Lan et al., 2024) investigated training LVLMs with fewer vision tokens to boost computational efficiency. However, these methods typically require additional model training, which imposes further computational overhead.

Training-free token pruning has also been widely employed in prior research to alleviate token redundancy in vision transformers (ViTs) and large language models (LLMs). For example, PruMerge (Shang et al., 2024) and VisionZip (Yang et al., 2024) suggested strategies to reduce vision tokens generated by vision encoders, thereby lowering vision token volume. FastV (Chen et al., 2024b) and SparseVLM (Zhang et al., 2024b) observed that visual tokens become less significant in deeper layers, thus proposing to eliminate redundant vision tokens during inference. VTW (Lin et al., 2024b) introduced a strategy to remove all vision tokens at a specific layer based on KL Divergence. Although these methods have demonstrated effectiveness, they overlook the distinctions among different layers and attention heads within LVLMs, leading to a significant performance decline on complex tasks. Our research addresses this gap by proposing a fine-grained pruning method including both Layer-Level Retention Rate Allocation and Head-Level Vision Token Pruning.

3 Method

Our method is a plug-and-play module during the inference process of LVLMs. Therefore, we first outline the inference process of LVLMs as preliminary, followed by the design of our proposed PLPHP.

3.1 Preliminary

LVLMs typically employ an autoregressive generation paradigm during inference, which comprises two stages: the Prefilling Stage and the Decoding Stage.

Prefilling Stage. In the Prefilling Stage, different modalities are mapped into a sequence of embedding vectors (tokens), which serves as the input to the LLM backbone. We denote the interleaved multimodal input token sequence of m text segments and n images $\mathbf{X}^1 \in \mathbb{R}^{S\times D}$ as:

$$\mathbf{X}^1 = \begin{pmatrix} \mathbf{X}_1^{(T)} \\ \mathbf{X}_1^{(I)} \\ \vdots \\ \mathbf{X}_m^{(T)} \\ \mathbf{X}_n^{(I)} \end{pmatrix}, \tag{1}$$

where $\mathbf{X}_i^{(T)} \in \mathbb{R}^{S_i^{(T)}\times D}$ represents the token sequence of the i-th text segment, and $\mathbf{X}_j^{(I)} \in \mathbb{R}^{S_j^{(I)}\times D}$ represents the token sequence of the j-th image. $S_i^{(T)}$ and $S_j^{(I)}$ represent the number of tokens for the i-th text segments and the j-th image, respectively, while $S = \sum_{i=1}^{m} S_i^{(T)} + \sum_{j=1}^{n} S_j^{(I)}$ represents the total length of the input token sequence. $\mathcal{I}_i^{(T)} \in \mathbb{N}_0^{S_i^{(T)}}$ and $\mathcal{I}_j^{(I)} \in \mathbb{N}_0^{S_j^{(I)}}$ denote the corresponding token index sets of $\mathbf{X}_i^{(T)}$ and $\mathbf{X}_j^{(I)}$ within $\mathbf{X}^1$.

$\mathbf{X}^1$ is then fed into an LLM composed of N decoder layers. Since the output and input shapes of each decoder layer are the same, we can denote the input of the l-th decoder layer as $\mathbf{X}^l \in \mathbb{R}^{S\times D}$. For the h-th attention head in the l-th layer:

$$\mathbf{Q}^{l,h} = \mathbf{X}^l \mathbf{W}_Q^{l,h}, \tag{2}$$

$$\mathbf{K}^{l,h} = \mathbf{X}^l \mathbf{W}_K^{l,h}, \tag{3}$$

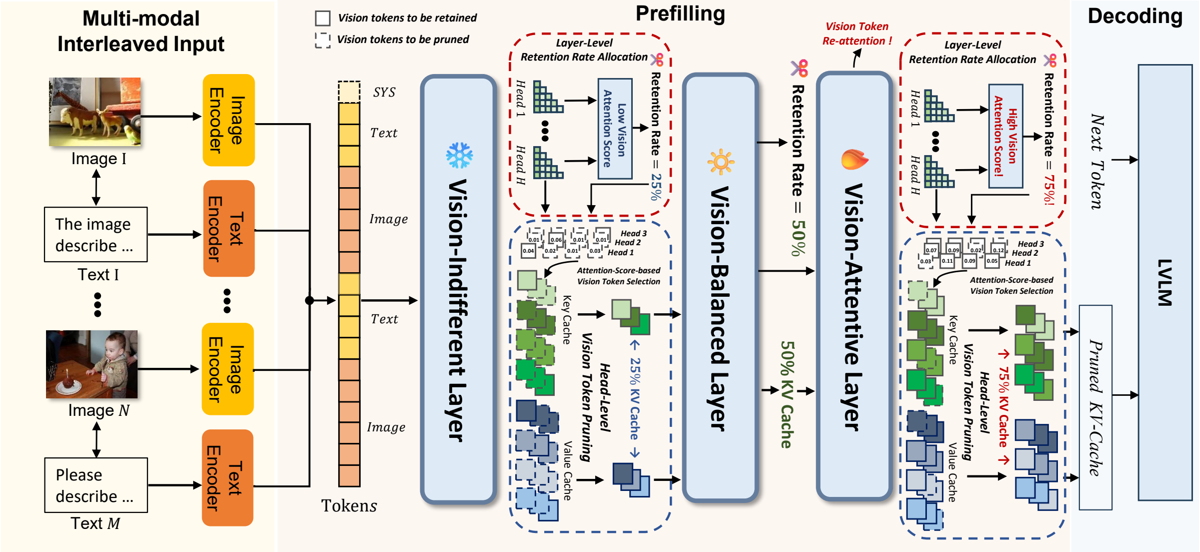


Figure 4: **Overview of PLPHP.** PLPHP has a two-level design including **Layer-Level Retention Rate Allocation** (as indicated by the red dashed boxes) and **Head-Level Vision Token Pruning** (as indicated by the blue dashed boxes). Upon the completion of prefilling a certain decoder layer, PLPHP categorizes the layer as vision indifferent, balanced or attentive, and assigns a vision token retention rate to the layer based on its average attention scores to the vision tokens. Subsequently, according to the allocated retention rate, PLPHP performs fine-grained pruning for each head within the layer. It removes the visual tokens with lower attention scores from the KV cache of each attention head, ensuring that the remaining proportion of vision tokens does not exceed the pre-assigned retention rate.

$$\mathbf{V}^{l,h} = \mathbf{X}^l \mathbf{W}_V^{l,h}, \tag{4}$$

where $\mathbf{W}_Q^{l,h} \in \mathbb{R}^{D \times D_k}$, $\mathbf{W}_K^{l,h} \in \mathbb{R}^{D \times D_k}$, and $\mathbf{W}_V^{l,h} \in \mathbb{R}^{D \times D_k}$ are referred to as the query projector, key projector, and value projector, respectively. D_k is called the head dimension. $\mathbf{K}^{l,h}$ and $\mathbf{V}^{l,h}$ are then stored as the KV cache for the current attention head.

The attention weights $\mathbf{A}^{l,h} \in \mathbb{R}^{S \times S}$ are given by:

$$\mathbf{A}^{l,h} = \mathrm{Softmax}\left(\frac{\mathbf{Q}^{l,h} \left(\mathbf{K}^{l,h}\right)^\top + \mathbf{\Lambda}}{\sqrt{D_k}} \right), \tag{5}$$

where $\mathbf{\Lambda} \in \mathbb{R}^{S \times S}$ is an upper triangular matrix whose non-zero values are set to $-\inf$ and diagonal elements are set to 0.

Decoding Stage. During the Decoding Stage, the model sequentially generates tokens and updates the KV cache of each attention head. At each timestep t, the input to the l-th decoder layer is a single token $\mathbf{x}_t^l \in \mathbb{R}^{1 \times D}$. For the h-th attention head of the l-th layer, the KV cache is updated by:

$$\mathbf{K}^{l,h} \leftarrow \begin{pmatrix} \mathbf{K}^{l,h} \\ \mathbf{x}_t^l \mathbf{W}_K^{l,h} \end{pmatrix}, \tag{6}$$

$$\mathbf{V}^{l,h} \leftarrow \begin{pmatrix} \mathbf{V}^{l,h} \\ \mathbf{x}_t^l \mathbf{W}_V^{l,h} \end{pmatrix}. \tag{7}$$

3.2 PLPHP

3.2.1 Overview

PLPHP is a two-level adaptive fine-grained pruning method with **Layer-Level Retention Rate Allocation** and **Head-Level Vision Token Pruning**. The architecture is illustrated in Figure 4.

3.2.2 Layer-Level Retention Rate Allocation

To measure the extent of a decoder layer's attention to visual information, thereby determining the number of vision tokens to retain, we define the *Vision Attention Score* γ^l of the l-th layer as:

$$\gamma^l = \sum_{k \in \bigcup_{j=1}^{n} \mathcal{I}_j^{(I)}} \frac{1}{H} \sum_{h=1}^{H} \mathbf{A}_{S,k}^{l,h}, \tag{8}$$

where H represents the number of attention heads in each decoder layer. Note that the value of γ^l is between 0 and 1. The larger the value of γ^l, the higher the l-th layer's attention to visual information.

In order to properly allocate the vision token retention rate based on the Vision Attention Score, given two thresholds α and β ($0 \leq \beta \leq \alpha \leq 1$), the l-th decoder layer is categorized as a **vision-attentive** layer when $\gamma^l \geq \alpha$, as a **vision-indifferent** layer if $\gamma^l < \beta$, and as a **vision-balanced** layer otherwise. The token retention rate

r^l for the l-th layer is defined as:

$$r^l = \begin{cases} r + \Delta r, & \gamma^l \geq \alpha \\ r - \Delta r, & \gamma^l < \beta \\ r, & \text{otherwise} \end{cases}, \tag{9}$$

where $0 \leq \Delta r \leq r \leq 1 - \Delta r$. For example, selecting $\alpha = 0.25$, $\beta = 0.1$, $r = 0.4$, and $\Delta r = 0.3$ signifies that we regard decoder layers with $\gamma^l \geq 0.25$ as vision-attentive layers, and decoder layers with $\gamma^l < 0.1$ as vision-indifferent layers. For vision-attentive layers, we retain $0.4 + 0.3$, that is, 70% of the vision tokens. For vision-indifferent layers, we retain $0.4 - 0.3$, that is, only 10% of the visual tokens. For vision-balanced layers, we retain 40% of the visual tokens.

Through this dynamic calculation of token retention rates, we retain a larger number of vision tokens for the vision-attentive layers to leverage their heightened focus on image information, while we keep fewer vision tokens for the vision-indifferent layers to achieve higher efficiency with the least sacrifice of critical visual information. As for the vision-balanced layers, we strike a compromise, seeking an equilibrium between efficiency and performance.

3.2.3 Head-Level Vision Token Pruning

Given the retention rate r^l calculated in the first level, we proceed to perform fine-grained pruning. According to FastV and Zhang et al. (2025), LVLMs typically exhibit a global focus on images in the first two layers and the last layer. Therefore, for a model composed of N decoder layers, we select the third layer and the penultimate layer as the starting and ending layers for pruning.

To extract the most important vision tokens to preserve, for the h-th ($1 \leq h \leq H$) attention head in the l-th layer ($3 \leq l \leq N - 1$), we calculate the indices of vision tokens with the highest attention scores within the j-th image input, accounting for the proportion r^l:

$$\mathcal{I}_j^{(I_R),h} = \text{argtop}K_j \left(\mathbf{A}_S^{l,h} \left[\mathcal{I}_j^{(I)} \right] \right), \tag{10}$$

where $K_j = r^l S_j^{(I)}$ and the argtop$K$ operation identifies the indices of the top K elements with the highest values in the given sequence.

We then prune vision tokens by updating the key cache and value cache of the attention head by:

$$\mathbf{K}^{l,h} \leftarrow \mathbf{K}^{l,h} \left[\bigcup_{i=1}^{m} \mathcal{I}_i^{(T)} \cup \bigcup_{j=1}^{n} \mathcal{I}_j^{(I_R),h} \right], \tag{11}$$

$$\mathbf{V}^{l,h} \leftarrow \mathbf{V}^{l,h} \left[\bigcup_{i=1}^{m} \mathcal{I}_i^{(T)} \cup \bigcup_{j=1}^{n} \mathcal{I}_j^{(I_R),h} \right], \tag{12}$$

where $[\cdot]$ represents the indexing operation, which retrieves elements from a sequence according to the given indices.

To provide an intuitive explanation, for every attention head of the l-th decoder layer, we retain only the top r^l proportion of the most attended tokens for each image, and remove the remaining $1 - r^l$ proportion from the context. Since the number of text tokens is typically negligible compared to vision tokens, we retain all text tokens.

Our method allows different attention heads within the same decoder layer to selectively drop different contexts, thereby better utilizing the property of multi-head attention mechanisms where distinct heads can focus on various parts of the contextual information.

4 Experiments

4.1 Experimental Setting

Benchmarks. In terms of multi-image benchmarks, we select four subsets from LLaVA-NeXT-Interleave-Bench (Li et al., 2024b): Spot-the-Diff (SD), Image-Edit (IE), Visual-Story-Telling (VST), and Multi-View (MV). We also select three single-image benchmarks: Flickr30k (Plummer et al., 2015), COCO 2017 Caption(Lin et al., 2014), and DetailCaps4870 (Dong et al., 2024).

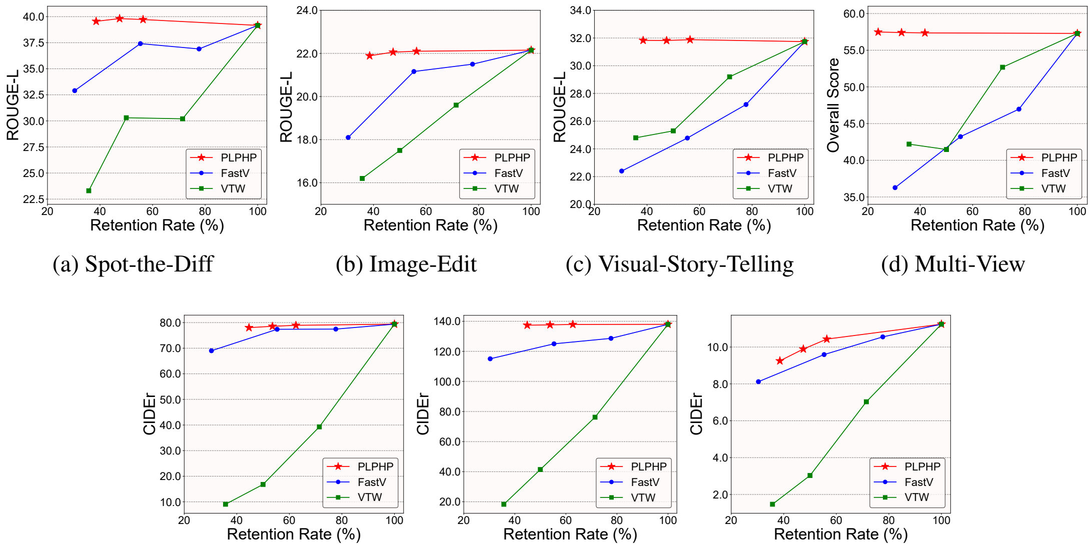
Metrics. Open-ended VQA tasks are evaluated using the ROUGE-L (Lin, 2004) (R) metric. CIDEr (Vedantam et al., 2015) (C) and METEOR (Banerjee and Lavie, 2005) (M) are employed to assess image captioning tasks. Overall Score is used to evaluate the performance on Multi-View benchmark. Regarding efficiency analysis, we utilize Vision Token Retention Rate (RR), KV Cache Size (KV), and Decoding Latency as our metrics for evaluation.

Baselines. We choose FastV and VTW as our baselines. FastV discards image tokens with low attention scores in the shallow layers, while VTW retains all image tokens in the shallow layers and discards them in the deeper layers.

Implementation Details. We implement PLPHP and all baselines on an NVIDIA A100 (80GB) GPU. All methods are evaluated using LMMs-Eval (Li* et al., 2024; Zhang et al., 2024a). More discussions regarding our benchmark selection, baseline

Table 1: **Comparison of different methods on Multi-Image and Single-Image benchmarks.** $(\cdot)$ signifies the values by which the performance exceeds that of the uncompressed model after applying the corresponding method.

	Multi-Image				Single-Image		
	Spot-the-Diff	Image-Edit	Visual-Story-Telling	Multi-View	Flickr30k	COCO 2017	DetailCaps4870
Methods	ROUGE-L ↑	ROUGE-L ↑	ROUGE-L ↑	Overall Score ↑	CIDEr ↑	CIDEr ↑	CIDEr ↑
LLaVA-OneVision-7B							
Full Tokens	39.16	22.15	31.74	57.29	79.39	137.97	11.24
FastV ($K = 3, R = 0.5$)	37.41	21.16	24.78	43.22	77.38	125.01	9.59
FastV ($K = 2, R = 0.5$)	36.19	20.77	23.99	43.04	75.37	120.8	9.31
VTW ($K = 20$)	30.13	19.59	29.17	52.68	39.28	76.23	7.03
VTW ($K = 14$)	30.47	16.17	25.35	41.47	16.80	41.43	3.03
PLPHP ($r = 0.5$)	39.72 (+0.56)	**22.10**	**31.88** (+0.14)	**57.46** (+0.17)	**78.93**	**137.90**	**10.43**
PLPHP ($r = 0.4$)	**39.81**(+0.65)	22.06	31.82 (+0.08)	57.41 (+0.12)	78.55	137.64	9.89
LLaVA-OneVision-0.5B							
Full Tokens	36.37	17.12	29.76	54.01	75.39	129.87	10.45
FastV ($K = 3, R = 0.5$)	23.06	12.87	24.97	39.03	64.22	97.74	8.25
FastV ($K = 2, R = 0.5$)	21.81	11.18	24.51	34.15	61.97	98.73	7.91
VTW ($K = 17$)	24.43	**16.91**	26.96	41.16	12.79	14.54	2.38
VTW ($K = 12$)	24.74	16.51	24.35	46.60	7.35	9.80	1.25
PLPHP ($r = 0.5$)	**36.35**	16.81	29.88 (+0.12)	**54.01**	**72.34**	**126.72**	**9.31**
PLPHP ($r = 0.4$)	36.19	16.82	**30.03** (+0.27)	53.91	71.04	123.75	8.35



(a) Spot-the-Diff (b) Image-Edit (c) Visual-Story-Telling (d) Multi-View

(e) Flickr30k (f) COCO 2017 Caption (g) DetailCaps4870

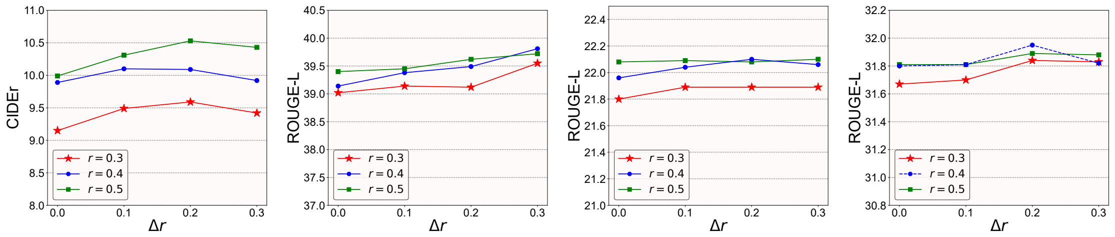
Figure 5: **Visualization of vision token retention rates and performance across seven different benchmarks**. A point on each polyline represents a certain hyperparameter setting. We record the vision token retention rate and performance of the method under the corresponding setting. For VTW, we evaluated cases with $K = 10, 14$ and 20. For FastV, we assessed the cases of $(K, R) = (2, 0.75), (3, 0.5)$ and $(3, 0.25)$. As for PLPHP, we examined the situations where $(r, \Delta r) = (0.3, 0.3), (0.4, 0.3)$ and $(0.5, 0.3)$.

configuration, and implementation details can be found in Appendix A.1.

Unless otherwise specified, the experimental results we report are based on LLaVA-OneVision-7B, and the default hyperparameter setting of PLPHP is $(r, \Delta r, \alpha, \beta) = (0.4, 0.3, 0.25, 0.1)$. The bolded text in the tables indicates the **best** performance under the corresponding metric, while the underlined text denotes the second best.

4.2 Main Results

We first conduct experiments with our method based on LLaVA-OneVision across different benchmarks. The main results are shown in Table 1.



(a) DetailCaps4870 (b) Spot-the-Diff (c) Image-Edit (d) Visual-Story-Telling

Figure 6: **Ablation studies on r and Δr.** Each polyline in the figure corresponds to a specific value of r, with different points on a single line representing various values of Δr and their corresponding performance metrics.

Table 2: **Ablation studies on α and β.**

	Spot-the-Diff	Image-Edit	Visual-Story-Telling	DetailCaps		
Methods	ROUGE-L ↑	ROUGE-L ↑	ROUGE-L ↑	CIDEr ↑	Avg. Retention Rate (%) ↓	Avg. KV Cache Size (%) ↓
$\alpha=0.25, \beta=0.05$	39.74	**22.10**	31.82	**10.66**	50.6%	53.2%
$\alpha=0.2, \beta=0.1$	39.15	**22.10**	**31.87**	10.16	44.0%	50.4%
$\alpha=0.25, \beta=0.1$	**39.81**	22.06	31.82	9.89	41.6%	47.7%
$\alpha=0.3, \beta=0.1$	39.35	22.02	31.81	9.63	39.6%	45.1%
$\alpha=0.25, \beta=0.15$	39.51	22.07	31.80	9.55	**35.8%**	**42.6%**

From the table, we can observe that:

- **PLPHP significantly outperforms both baselines across different benchmarks.** For the LLaVA-OneVision-7B model, the average performance of PLPHP under default hyperparameter settings surpasses FastV by 11.4% and VTW by 48.4%. Compared to the uncompressed model, the average performance degradation brought by PLPHP is merely 0.46%. We attribute this performance enhancement to the granularity and adaptability of PLPHP. In contrast to FastV and VTW, which discard a fixed set of vision tokens from *all* pruned attention heads, the dynamic nature of PLPHP offers a distinct performance advantage.
- **Model with PLPHP outperforms uncompressed model on various multi-image tasks.** Notably, the average performance of PLPHP surpasses that of the uncompressed model by 0.51% across multiple multi-image task benchmarks on LLaVA-OneVision-7B through appropriate pruning. The improvement on multi-image benchmarks could be attributed to the increased redundancy in visual information inherent in multi-image tasks, which could potentially be detrimental to model inference. This redundancy is effectively eliminated by PLPHP, thereby enhancing both the efficiency and performance.
- **The performance of PLPHP remains relatively stable under different retention rates.** The carefully designed pruning dynamics in PLPHP allow it to prioritize the removal of the most redundant tokens, thereby ensuring that performance is less affected by the pruning rate. On the other hand, VTW is highly sensitive to the selection of K. It discards *all* vision tokens at a specific layer, thus once the model exhibits significant *Vision Token Re-attention* after this layer, it is likely to severely impact the performance, which could be the cause of its high sensitivity to the hyperparameter and substantial performance decline in image captioning tasks.

To provide a more intuitive analysis of how each method performs under varying pruning rates, we evaluated their performance across different vision token retention rates and visualized the results in Figure 5. It can be observed that PLPHP consistently outperforms the baseline at the same pruning rate and maintains nearly no performance degradation within a certain pruning rate range, indicating that we can achieve better performance while discarding more vision tokens, which directly leads to a higher computational efficiency.

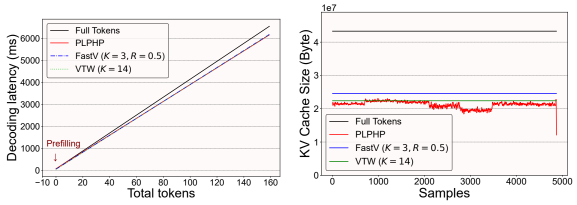
These performance boosts highlight the superiority of our method, which dynamically adjusts the pruning rate based on the attention allocated to image tokens in different layers and independently preserve different contextual information for different attention heads.

Table 3: **Performance of PLPHP on various models.** Bolded text indicates that PLPHP surpasses the uncompressed model.

	SD	IE	VST	MV	Flickr30k	COCO		
Methods	R ↑	R ↑	R ↑	R ↑	C ↑	C ↑	RR (%) ↓	KV (%) ↓
Qwen2-VL	27.56	21.21	24.92	12.78	77.24	96.18	100%	100%
w/ PLPHP	**27.78**	**21.40**	**25.02**	**12.96**	**78.02**	**98.67**	**35.8%**	**41.9%**
IDEFICS2	18.98	14.90	23.91	13.84	51.73	72.12	100%	100%
w/ PLPHP	18.55	14.89	**23.93**	**13.96**	51.68	**72.60**	**36.1%**	**51.3%**
Mantis	16.30	9.56	13.27	11.02	70.41	91.41	100%	100%
w/ PLPHP	**16.41**	**9.81**	**13.41**	**11.14**	69.90	90.61	**29.1%**	**33.7%**

4.3 Generality of PLPHP on Various LVLMs

To further demonstrate the generality of PLPHP on various model architectures, we implement PLPHP on common LVLMs with different LLM backbones, and directly compared them with uncompressed models to highlight our effectiveness, with results shown in Table 3. Since IDEFICS2 and Mantis are unable to follow instructions in DetailCaps4870, we evaluate PLPHP on the other six benchmarks. **Remarkably, Qwen2-VL equipped with PLPHP surpasses the uncompressed model across all benchmarks**, achieving an average improvement rate of 1.5%, while saving an average of 58.1% KV Cache storage space. For the other two models, our method also achieves an average of 57% KV Cache compression while surpassing the original models across multiple benchmarks.



(a) Decoding Latency (b) KV Cache Size

Figure 7: **The decoding latency and KV Cache size results.** Both baselines maintain constant KV Cache sizes due to unchanging pruning rates, while PLPHP adaptively assigns retention rates, producing a fluctuating curve with a smaller mean.

Table 4: **Performance and efficiency comparison among different methods.**

	DetailCaps4870				
Methods	C ↑	M ↑	R ↑	Time (h) ↓	RR (%) ↓
Full Tokens	11.24	20.13	30.01	5.63	100%
FastV ($K = 3, R = 0.5$)	9.59	18.55	28.72	5.23	55.4%
VTW ($K = 14$)	3.03	15.05	24.38	**5.22**	50.0%
PLPHP	**9.89**	**19.33**	**29.19**	5.23	**47.5%**

4.4 Efficiency Analysis

To analyze the efficiency of PLPHP, we conduct experiments on DetailCaps4870 since it includes long generation contents. We can observe from Figure 7a that PLPHP achieves a comparable total decoding latency to both baselines. The latency introduced by the unpruned Prefilling Stage is minimal (less than 0.5 tokens of delay). Figure 7b shows that PLPHP maintains a lower KV cache size during the evaluation process compared to all baselines, leading to a shorter decoding latency. Table 4 shows that PLPHP attains performance closest to the uncompressed model. The nearly consistent evaluation time also indicates that the additional computation during the Prefilling Stage gradually becomes negligible as generation progresses.

Table 5: **Decoding Latency and KV Cache Size of PLPHP under different retention rates.**

Methods	Decoding Latency (ms/token) ↓	KV Cache Size (%) ↓
Full Tokens	49.10	100%
PLPHP ($r = 0.5$)	41.26	54.9%
PLPHP ($r = 0.4$)	40.20	46.2%
PLPHP ($r = 0.3$)	**39.19**	**37.6%**

4.5 Ablation Study

To explore the impact of r and Δr, we conduct ablation experiments on four benchmarks, with the results illustrated in Figure 6. It can be observed that setting $\Delta r > 0$ consistently outperforms the cases where $\Delta r = 0$, indicating that adaptive pruning rates are superior to a fixed pruning rate. This finding demonstrates that our proposed **layer-level pruning rate allocation has a positive impact on model performance**.

Since r is the most direct parameter reflecting the average pruning rate, we test the impact of r on efficiency, with the results presented in Table 5. PLPHP achieves an 18.1% decoding speedup and a 53.8% KV Cache compression under the default settings where $r = 0.4$, and further reaches a 20.2% acceleration and a 62.4% compression at a lower retention rate, enhancing the computational efficiency of LVLM decoding remarkably.

α and β also indirectly influence pruning rates, thus we also conduct ablation studies with the results shown in Table 2. Intuitively, increasing α and β elevates the criteria for vision-attentive layers and vision-balanced layers more stringent, leading to higher pruning rates at the cost of performance loss. Conversely, decreasing them relaxes the criteria, enhancing the performance but at greater computational expense.

5 Conclusion

In this work, we introduce PLPHP, a two-level pruning method designed to improve the efficiency of LVLMs with Layer-Level Retention Rate Allocation and Head-Level Vision Token Pruning. Comprehensive experiments demonstrate that PLPHP outperforms existing pruning methods, achieving a 18% decoding acceleration, over 50% KV Cache compression and only 0.46% performance degradation, with improvements on multi-image tasks. We believe our work contributes to efficient LVLMs, further promotes their applications, and improves the user experience.

6 Limitations

The limitations of our work include: 1) We have only evaluated our method on image-text datasets and have not conducted further testing on other modalities. 2) Despite demonstrating superior performance, our proposed method involves 4 hyper-parameters, requiring parameter selection. 3) The efficiency advantage of PLPHP is primarily evident in scenarios where the model generates longer content. It does not show an efficiency advantage for short generations such as multiple-choice questions. In future research, we plan to: 1) Extend our method to visual modalities such as video (Chen et al., 2024a; Jin et al., 2024; Weng et al., 2024) and 3D (Hong et al., 2023; Huang et al., 2024a; Chen et al., 2024c), aiming to develop a unified pruning strategy across multiple visual modalities. 2) Test and enhance the performance of our method in real-world scenarios with limited computational resources, such as edge computing and embodied intelligence. 3) Explore the comparison and integration of our method with other advanced model lightweighting techniques, such as model distillation, quantization, and advanced KV Cache optimization mechanisms including GQA (Ainslie et al., 2023) and MLA (Liu et al., 2024).

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 1(2):3.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. 2024a. Videollm-online: Online video large language model for streaming video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18407–18418.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024b. 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.

Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. 2024c. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26428–26438.

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

Hongyuan Dong, Jiawen Li, Bohong Wu, Jiacong Wang, Yuan Zhang, and Haoyuan Guo. 2024. Benchmarking and improving detail image caption. *arXiv preprint arXiv:2405.19092*.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 2023. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494.

Haifeng Huang, Yilun Chen, Zehan Wang, Rongjie Huang, Runsen Xu, Tai Wang, Luping Liu, Xize Cheng, Yang Zhao, Jiangmiao Pang, et al. 2024a. Chat-scene: Bridging 3d scene and large language models with object identifiers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Wenxuan Huang, Zijie Zhai, Yunhang Shen, Shaoshen Cao, Fei Zhao, Xiangfeng Xu, Zheyu Ye, and Shaohui Lin. 2024b. Dynamic-llava: Efficient multimodal large language models via dynamic vision-language context sparsification. *arXiv preprint arXiv:2412.00876*.

Dongfu Jiang, Xuan He, Huaye Zeng, Cong Wei, Max Ku, Qian Liu, and Wenhu Chen. 2024. Mantis: Interleaved multi-image instruction tuning. *arXiv preprint arXiv:2405.01483*.

Peng Jin, Ryuichi Takanobu, Wancai Zhang, Xiaochun Cao, and Li Yuan. 2024. Chat-univi: Unified visual representation empowers large language models with image and video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13700–13710.

Zhibin Lan, Liqiang Niu, Fandong Meng, Wenbo Li, Jie Zhou, and Jinsong Su. 2024. Avg-llava: A large multimodal model with adaptive visual granularity. *arXiv preprint arXiv:2410.02745*.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander

Rush, Douwe Kiela, et al. 2024. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Advances in Neural Information Processing Systems*, 36.

Bo Li*, Peiyuan Zhang*, Kaichen Zhang*, Fanyi Pu*, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. 2024. Lmms-eval: Accelerating the development of large multimoal models.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. 2024b. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*.

Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jie Qin, Jianke Zhu, and Lei Zhang. 2024c. Tokenpacker: Efficient visual projector for multimodal llm. *arXiv preprint arXiv:2407.02392*.

Xuanlin Li, Yunhao Fang, Minghua Liu, Zhan Ling, Zhuowen Tu, and Hao Su. 2023. Distilling large vision-language model with out-of-distribution generalizability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2492–2503.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Junwu Zhang, Munan Ning, and Li Yuan. 2024a. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Zhihang Lin, Mingbao Lin, Luxi Lin, and Rongrong Ji. 2024b. Boosting multimodal large language models with visual tokens withdrawal for rapid inference. *arXiv preprint arXiv:2405.05803*.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning.

Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

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

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Changyuan Wang, Ziwei Wang, Xiuwei Xu, Yansong Tang, Jie Zhou, and Jiwen Lu. 2024a. Q-vlm: Post-training quantization for large vision-language models. *arXiv preprint arXiv:2410.08119*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Yuetian Weng, Mingfei Han, Haoyu He, Xiaojun Chang, and Bohan Zhuang. 2024. Longvlm: Efficient long video understanding via large language models. In *European Conference on Computer Vision*, pages 453–470. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. 2024. Visionzip: Longer is better but not necessary in vision language models. *arXiv preprint arXiv:2412.04467*.

Zhengqing Yuan, Zhaoxu Li, Weiran Huang, Yanfang Ye, and Lichao Sun. 2023. Tinygpt-v: Efficient multimodal large language model via small backbones. *arXiv preprint arXiv:2312.16862*.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. 2024a. Lmms-eval: Reality check on the evaluation of large multimodal models.

Xiaofeng Zhang, Yihao Quan, Chen Shen, Xiaosong Yuan, Shaotian Yan, Liang Xie, Wenxiao Wang, Chaochen Gu, Hao Tang, and Jieping Ye. 2025. From redundancy to relevance: Enhancing explainability in multimodal large language models. *Annual Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics*.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. 2024b. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*.

Yichen Zhu, Minjie Zhu, Ning Liu, Zhiyuan Xu, and Yaxin Peng. 2024. Llava-phi: Efficient multi-modal assistant with small language model. In *Proceedings of the 1st International Workshop on Efficient Multimedia Computing under Limited*, pages 18–22.

A Appendix

A.1 Details of Evaluation Settings

A.1.1 Benchmarks

Since PLPHP maintains the computational integrity of the LVLMs' Prefilling Stage, its efficiency advantage is primarily reflected in the low decoding latency during the subsequent Decoding Stage. Therefore, we mainly choose benchmarks composed of open-ended VQA and image captioning tasks. The benchmarks we select encompasses both multi-image task benchmarks and single-image task benchmarks.

- **Multi-Image benchmarks**: The LLaVA-Interleave Bench is a comprehensive benchmark dataset designed to evaluate the performance of LVLMs in multi-image scenarios. It consists of 13 challenging tasks with a total of 17,000 instances. We curated four subsets consisting of open-ended VQA tasks from LLaVA-NeXT-Interleave-Bench: Spot-the-Diff, Image-Edit, Visual-Story-Telling, and Multi-View.

- **Single-Image benchmarks**: The Flickr30k dataset is a widely used benchmark in the field of image captioning and visual understanding. It consists of 31,783 images collected from the Flickr platform, each paired with five human-annotated captions. The COCO2017 Caption subset contains more than 45,000 images, each annotated with five captions written by human annotators, describing the visual content of the images in detail, including objects, their attributes, and the relationships between them. DetailCaps4870 provides more fine-grained and specific image content descriptions than standard captioning datasets, which is more useful for efficiency analysis.

A.1.2 Baselines

We select FastV and VTW as our baselines in our experiments. Notably, FastV offers two versions of implementation: one that supports KV cache and one that does not. Since the non-KV-cache implementation introduces substantial computational overhead, we use the version that supports KV cache to ensure a fair comparison. For both of the baselines, we refer to the official open source code [1] [2] and implement them on the models we evaluate.

A.1.3 Models

For Qwen2-VL, we set `max_pixels` to $1280 \times 28 \times 28$ and `min_pixels` to $256 \times 28 \times 28$ according to the official recommendation. The Mantis model that we choose is Mantis-8B-SigLIP-LLaMA3. For LLaVA-OneVision and Mantis, we use the official original versions [3] [4], while using the versions provided by the transformers library (Wolf et al., 2020) for all other models.

A.2 Case Study

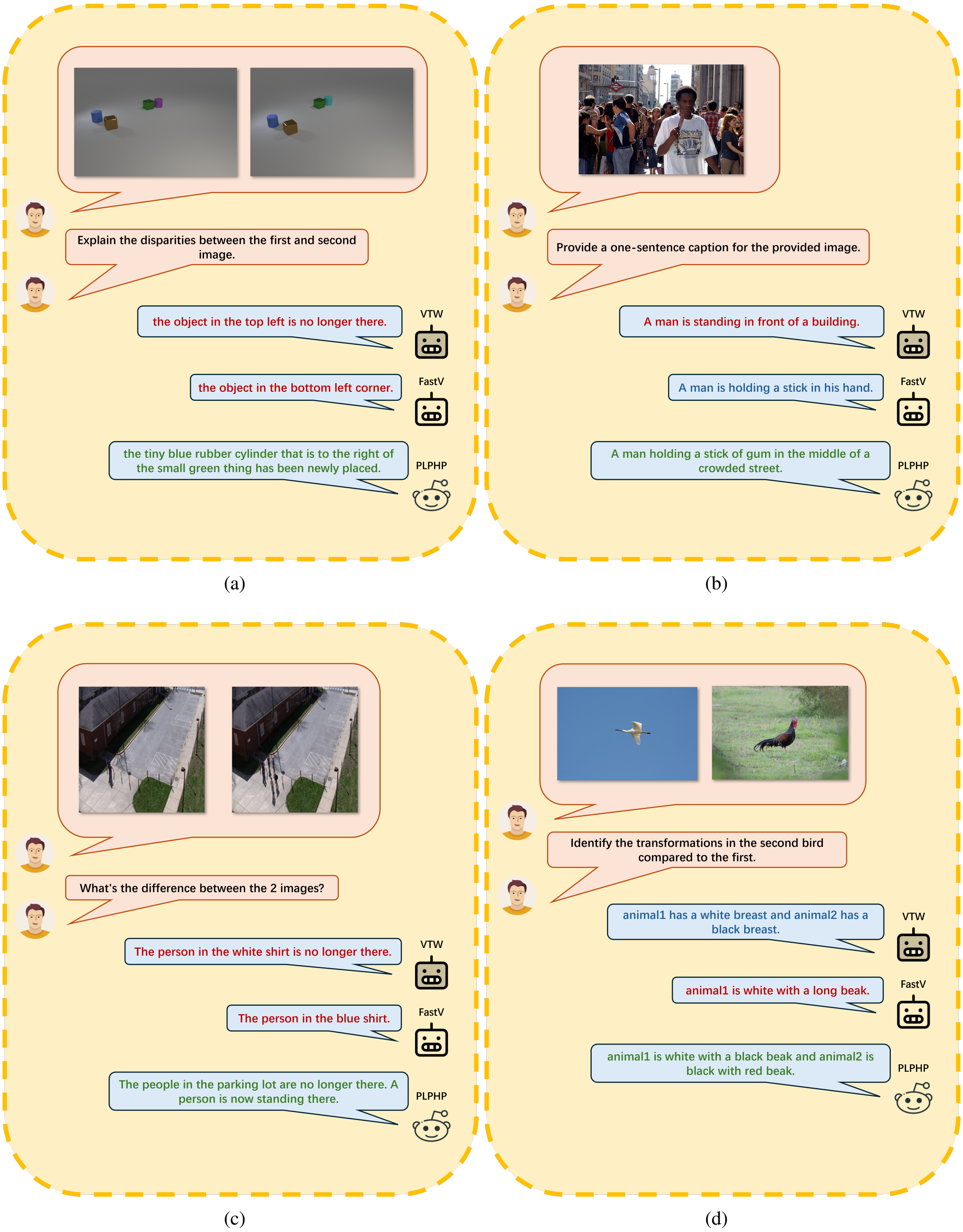
To showcase the effectiveness of our proposed method, we present a series of case studies in the form of multimodal chatbots, as shown in Figure 8.

[1] https://github.com/pkunlp-icler/FastV

[2] https://github.com/lzhxmu/VTW

[3] https://huggingface.co/lmms-lab/llava-onevision-qwen2-7b-ov

[4] https://huggingface.co/TIGER-Lab/Mantis-8B-siglip-llama3



(a)

Explain the disparities between the first and second image.

the object in the top left is no longer there.

the object in the bottom left corner.

the tiny blue rubber cylinder that is to the right of the small green thing has been newly placed.

(b)

Provide a one-sentence caption for the provided image.

A man is standing in front of a building.

A man is holding a stick in his hand.

A man holding a stick of gum in the middle of a crowded street.

(c)

What's the difference between the 2 images?

The person in the white shirt is no longer there.

The person in the blue shirt.

The people in the parking lot are no longer there. A person is now standing there.

(d)

Identify the transformations in the second bird compared to the first.

animal1 has a white breast and animal2 has a black breast.

animal1 is white with a long beak.

animal1 is white with a black beak and animal2 is black with red beak.

Figure 8: **Multimodal Chatbots with different pruning methods.**