EvoCut: Multi-Layer Evolution-Aware Visual Token Compression for Efficient Large Vision-Language Models

Anonymous ACL submission

Abstract

Large vision-language models (LVLMs) achieve strong performance on image and video understanding tasks, but their inference efficiency is constrained by the large number of visual tokens produced by vision encoders. Most existing visual token compression methods estimate token importance from attention scores or representation properties at specific layers, overlooking how visual tokens evolve across the vision encoder. Such layer-specific criteria may provide incomplete importance estimates and limit performance preservation after compression. To address this issue, we analyze layer-wise visual token evolution directions and observe that tokens form multiple group evolution directions across vision-encoder layers. Our analysis further shows that informative tokens tend to exhibit persistent deviations from common group evolution directions. Based on this observation, we propose EvoCut, a training-free and attention-free visual token compression method that estimates token importance from multi-layer evolution deviation. Experimental results show that EvoCut can retain only 11.1% of the visual tokens on LLaVA-1.5-7B while preserving 94.4% of the average performance, demonstrating its effectiveness in balancing efficiency and accuracy.

1 Introduction

Large vision-language models (LVLMs) extend large language models to image and video understanding, and have shown strong performance across multimodal tasks (Alayrac et al., 2022; Li et al., 2023b; Liu et al., 2023a; Bai et al., 2023; Lin et al., 2023). However, processing visual inputs requires the vision encoder to produce a large number of visual tokens. For high-resolution images and videos, the token sequence can easily grow to hundreds or thousands of tokens before it is passed to the language model (Liu et al., 2024; Bai et al., 2025). These visual tokens often dominate the prefilling cost and increase memory consumption, inference latency, and FLOPs. Therefore, reducing redundant visual tokens while preserving multimodal understanding ability is important for efficient LVLM deployment.

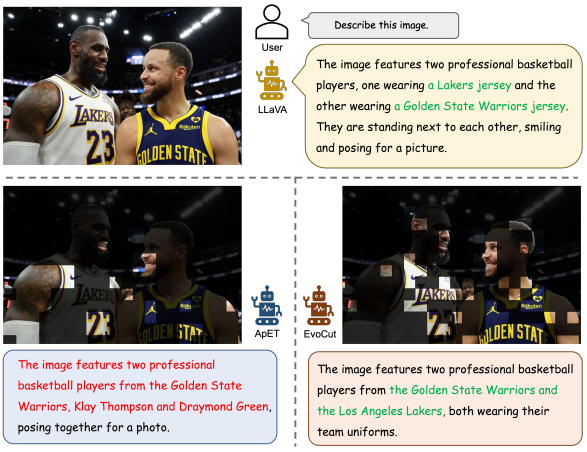

Figure 1: Comparison between ApET and EvoCut. Red annotations indicate incorrect answers, while green annotations indicate correct answers.

To reduce the inference cost of LVLMs, recent studies have explored visual token reduction and compression before or during language-model decoding (Chen et al., 2024a; Zhang et al., 2025; Yang et al., 2025; Shang et al., 2025; Ma et al., 2026; Wen et al., 2025; Jiang et al., 2025; Fan et al., 2025). Existing LVLM compression methods mainly estimate token importance from either attention responses or visual representations. Attention-based methods prune or merge tokens based on attention scores, but these scores can be affected by positional bias and tend to concentrate on later token positions, as shown in Figure 2. They are also difficult to combine with efficient attention implementations such as FlashAttention (Dao et al., 2022; Dao, 2023). Representation-based methods avoid direct dependence on attention maps, but they

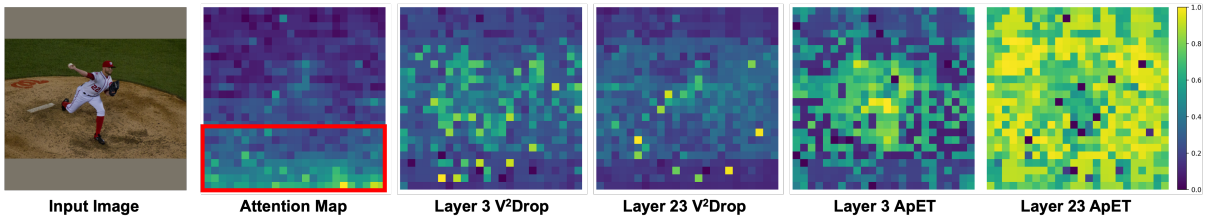

Figure 2: Visualization results show that attention scores are affected by positional bias, causing them to concentrate more on later tokens. Meanwhile, V^2Drop and ApET produce inconsistent importance maps across different layers, suggesting that single-layer criteria can be unstable for identifying important visual tokens.

often rely on single-layer features or local token relations. As shown in Figure 2, such layer-specific criteria can produce inconsistent importance estimates across different layers. These limitations suggest that many existing criteria provide only a partial view of token importance, making it difficult to consistently preserve informative visual tokens after compression.

We argue that visual token importance should be estimated from its layer-wise evolution, rather than from a single-layer snapshot. To examine this hypothesis, we analyze token movements between adjacent vision-encoder layers and observe that visual tokens do not follow a single global evolution direction. Instead, they form multiple group evolution directions, and tokens that consistently deviate from these shared directions often correspond to visually informative regions. This observation suggests that persistent deviation from group evolution directions provides a useful signal for identifying important visual tokens. At the same time, single-layer deviation can be noisy, especially in deeper layers where token representations become highly mixed. Therefore, token importance should be modeled from cross-layer evolution behavior rather than from isolated layer-wise scores.

Motivated by these observations, we propose EvoCut, a training-free and attention-free visual token compression method for LVLMs. EvoCut estimates token importance from layer-wise token evolution inside the vision encoder. For each layer transition, it measures the deviation between each token's evolution direction and the group evolution directions, and accumulates the resulting scores across layers with a history-aware update. This multi-layer scoring allows EvoCut to retain tokens that consistently provide distinctive visual information, as shown in Figure 1. Since EvoCut does not rely on attention maps, it can be integrated into existing LVLMs and remains compatible with efficient attention operators. Our main contributions are summarized as follows:

- We investigate visual-token evolution across vision-encoder layers and observe that token movements follow multiple group evolution directions. We further observe that tokens with persistent deviations from these shared directions are more likely to correspond to informative visual regions.
- Based on these observations, we propose EvoCut, a training-free and attention-free visual token compression method for LVLMs. EvoCut estimates token importance by measuring deviation from multiple group evolution directions and accumulating this signal across layers, avoiding reliance on single-layer importance estimates.
- Experiments on image and video understanding tasks show that EvoCut consistently improves the performance-efficiency trade-off across multiple LVLM backbones. For example, EvoCut preserves 94.4% of the average performance on LLaVA-1.5-7B with only 64 visual tokens while achieving a 1.44× total-time speedup.

2 Related Work

2.1 Large Vision-Language Models

Large Vision-Language Models (LVLMs) extend large language models (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2023) to multimodal scenarios by incorporating visual encoders and projection modules. Most LVLMs build on Transformer-based (Vaswani et al., 2017) pretrained visual encoders such as ViT or CLIP to convert visual inputs into token sequences (Dosovitskiy et al., 2021; Radford et al., 2021). Typically, visual tokens are then mapped into the language model space, enabling the model to perform tasks such as visual question answering, image captioning, and multimodal reasoning. Models such as BLIP-2, In-

structBLIP, MiniGPT-4, LLaVA, Qwen-VL, and InternVL have demonstrated the effectiveness of this paradigm across various vision-language tasks (Li et al., 2023b; Dai et al., 2023; Zhu et al., 2023; Liu et al., 2023a; Bai et al., 2023; Chen et al., 2024b). Recent LVLMs further improve fine-grained visual understanding by supporting high-resolution images and long videos (Liu et al., 2024; Bai et al., 2025; Zhang et al., 2023b,a; Lin et al., 2023). However, the resulting visual tokens are often much more numerous than textual tokens, introducing substantial computational and memory overhead. This makes reducing redundant visual tokens while preserving multimodal reasoning ability a key problem for efficient LVLMs.

2.2 Visual Token Compression for LVLMs

Existing visual token compression methods for LVLMs mainly estimate token importance from either attention responses or visual representations. Attention-based methods use visual self-attention, CLS-token attention, or text-to-vision attention to prune or merge visual tokens. For example, VisionZip compresses visual tokens based on CLS-token attention scores from the selected vision-encoder layer used by the LVLM, which is typically the penultimate layer in LLaVA-style models (Yang et al., 2025). Although such methods are simple and effective, attention scores can be affected by positional bias and are difficult to combine with efficient attention operators such as FlashAttention (Dao et al., 2022; Dao, 2023). Representation-based methods avoid direct dependence on attention maps and instead identify redundant tokens from visual features, such as feature similarity, approximation error, token duplication, or local variation (Zhang et al., 2025; Ma et al., 2026; Wen et al., 2025). For instance, ApET reconstructs the original visual tokens with a compact set of basis tokens and uses the approximation error to measure token informativeness (Ma et al., 2026). However, many existing criteria still depend on layer-specific features or local token relations, which may provide only a partial estimate of token importance. These methods usually evaluate token importance from static features or local relationships, rather than explicitly modeling how each token changes between adjacent vision-encoder layers. In contrast, EvoCut models token importance from multi-layer evolution behavior and measures whether a token persistently deviates from multiple shared group evolution directions.

3 Method

3.1 Preliminaries

An LVLM turns visual content into a token sequence that can be consumed by a language model (Huang et al., 2023). Given an image or video, the vision encoder first extracts patch-level representations, and a projector maps these representations into the same embedding space as text tokens. We denote the resulting visual token sequence as V. Together with a textual instruction T, these visual tokens form the multimodal context used by the LLM to generate an answer autoregressively:

$$y_t \sim p_\theta(y_t \mid V, T, y_{<t}), \tag{1}$$

where θ denotes the model parameters and $y_{<t}$ denotes previously generated tokens.

3.2 Layer-wise Token Evolution Analysis

To understand how visual tokens evolve inside the vision encoder, we analyze the layer-wise evolution direction of patch tokens between adjacent Transformer layers. Let $x_i^{(t)} \in \mathbb{R}^d$ denote the hidden state of the i-th patch token at the t-th vision-encoder layer. We define its layer-wise evolution direction from layer $t-1$ to layer t as

$$\vec{\Delta}_i^{(t)} = \frac{x_i^{(t)} - x_i^{(t-1)}}{\|x_i^{(t)} - x_i^{(t-1)}\|_2}. \tag{2}$$

Discovery of Group Evolution Directions. We use HDBSCAN (Campello et al., 2013) to cluster token evolution directions at each layer transition, since it can discover density-based direction groups without specifying the number of clusters. Each cluster is treated as a shared evolution direction followed by a group of tokens, and its group direction is defined as the normalized mean of all member directions. As shown in the right part of Figure 3, the number of clustered directions is small in shallow layers and increases in deeper layers on both POPE and TextVQA. This indicates that visual tokens do not evolve along a single global direction, but gradually split into multiple group evolution paths as the network deepens.

High-Deviation Tokens Correlate with Informative Regions. To study the properties of informative tokens during layer-wise evolution, we examine their deviations from group evolution directions. For each token, we compute its cosine deviation

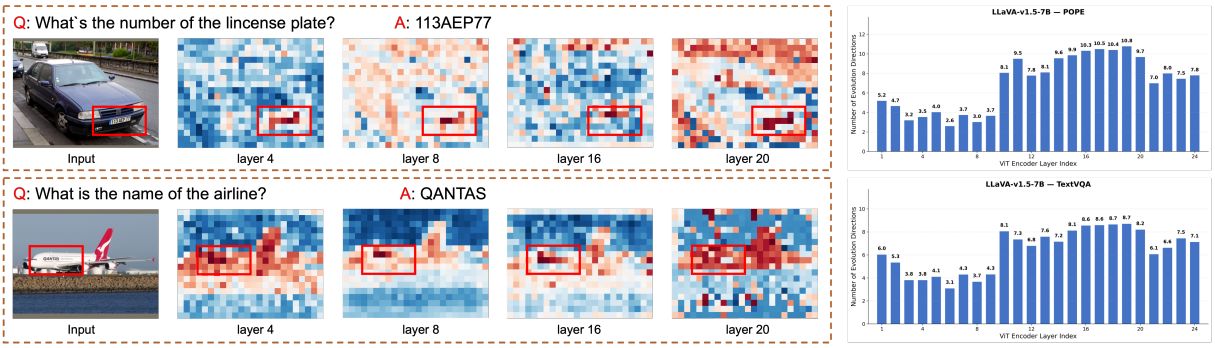

Figure 3: Visualization of multi-layer token evolution in LLaVA-1.5. **(Left)** Tokens with larger deviations from group evolution directions tend to correspond to visually informative regions. **(Right)** The number of clustered token evolution directions across vision-encoder layers on POPE and TextVQA, showing that token transitions form multiple group evolution directions. More results are provided in Appendix A.4.

from the nearest group direction at each layer transition. As shown in the left part of Figure 3, tokens corresponding to informative regions, such as the license plate and airplane, maintain large deviations across multiple layers. More generally, high-deviation tokens tend to appear on distinct foreground objects, text regions, and small targets, suggesting that directional deviation is associated with informative visual content.

Redundancy of Low-Deviation Tokens. In contrast, tokens that closely follow dominant group directions are mostly located in background regions, repetitive textures, or weakly informative edges. These tokens show highly predictable cross-layer behavior and carry less token-specific visual information for multimodal understanding. This complements the high-deviation cases and suggests that deviation from group directions can help distinguish informative tokens from less informative ones.

Deviation Noise in Deeper Layers. We also observe that single-layer deviation becomes less reliable in deeper layers. As self-attention repeatedly aggregates global context, token representations become more mixed, and background tokens may show large deviations due to their association with foreground regions. This makes isolated layer-wise scores unstable and motivates accumulating deviation signals across multiple layer transitions.

Overall, these observations suggest that token importance is better characterized by persistent deviation from group evolution directions than by a single-layer snapshot. This motivates EvoCut to score visual tokens using multi-layer evolution deviation.

3.3 Evolution-Aware Token Compression

Building on the above observations, we propose EvoCut, a training-free visual token compression method that estimates token importance from multi-layer evolution in the vision encoder. Instead of relying on a single-layer representation snapshot, EvoCut tracks how each visual token evolves across adjacent layers and identifies tokens whose evolution directions deviate from shared group evolution directions. As illustrated in Figure 4, EvoCut consists of three steps: group evolution modeling, evolution-deviation scoring, and multi-layer score accumulation.

Group Evolution Modeling. Let $X^{(\ell)} = \{x_1^{(\ell)}, \ldots, x_N^{(\ell)}\}$ denote the vision-encoder hidden states at layer ℓ. For each adjacent layer pair, we compute the token evolution direction $\vec{\Delta}_i^{(\ell)}$ using the normalized layer-wise difference in Equation 2.

Since tokens may evolve along multiple shared directions, we cluster all token evolution directions for the transition from layer $\ell - 1$ to layer ℓ with K-means. For simplicity and efficient inference, EvoCut uses a fixed number of clusters for all layer transitions. For each cluster, its center is computed as the mean of the assigned token evolution directions and then normalized to form a group evolution direction. EvoCut uses the M cluster centers as group evolution directions:

$$\mathcal{U}^{(\ell)} = \{\vec{u}_1^{(\ell)}, \vec{u}_2^{(\ell)}, \ldots, \vec{u}_M^{(\ell)}\}. \tag{3}$$

These directions capture the dominant evolution directions shared by visual tokens at the current layer transition.

Evolution-Deviation Scoring. For each token, EvoCut measures its deviation from the closest

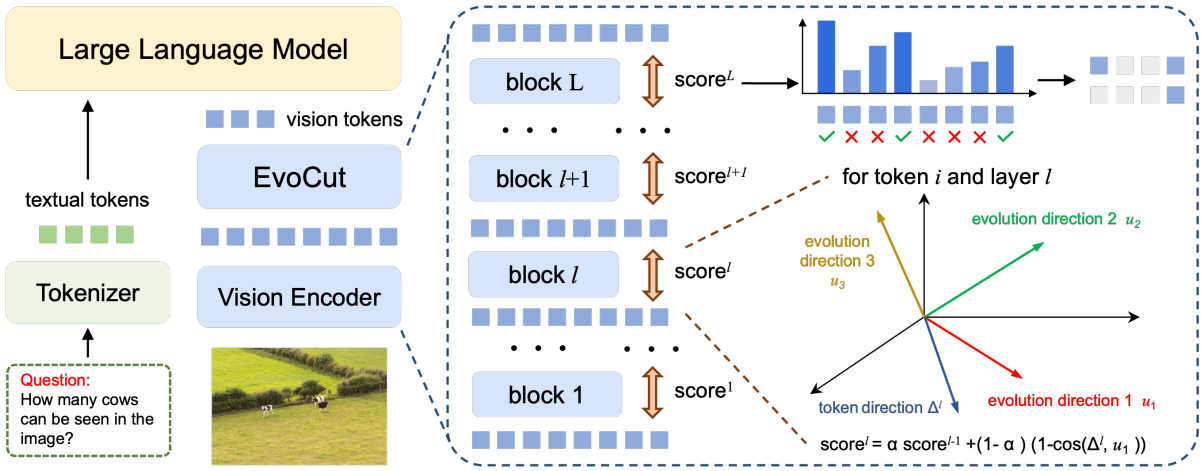

Figure 4: Overview of EvoCut. EvoCut estimates visual token importance from multi-layer token evolution, measures deviation from multiple group evolution directions, aggregates scores across layers, and retains top-ranked visual tokens before the multimodal projector and LLM prefilling.

group evolution direction. We compute the best cosine alignment between $\vec{\Delta}_i^{(\ell)}$ and the direction set $\mathcal{U}^{(\ell)}$:

$$a_i^{(\ell)} = \max_m \cos(\vec{\Delta}_i^{(\ell)}, \vec{u}_m^{(\ell)}). \tag{4}$$

The layer-wise importance score is then defined as the directional deviation:

$$r_i^{(\ell)} = 1 - a_i^{(\ell)}. \tag{5}$$

A larger score indicates stronger deviation from shared group evolution directions, suggesting that the token is more likely to contain distinctive visual information.

Multi-Layer Score Accumulation. To reduce the instability of single-layer estimates, EvoCut accumulates token scores across layer transitions with an exponential moving average:

$$s_i^{(\ell)} = \alpha s_i^{(\ell-1)} + (1-\alpha) r_i^{(\ell)}, \tag{6}$$

where α controls the contribution of historical evolution information. After the final vision-encoder layer, EvoCut keeps the top-K patch tokens according to s_i and sends the compressed visual sequence to the multimodal projector. The complete pseudocode is provided in Appendix A.1.

4 Experiments

4.1 Experimental Setup

Models and Baselines. To validate the effectiveness and generality of the proposed method, we evaluate it on representative LVLM backbones covering both image and video understanding: LLaVA-1.5-7B, LLaVA-NeXT-7B, Qwen-2.5-VL-7B, and Video-LLaVA-7B (Liu et al., 2023a, 2024; Bai et al., 2025; Lin et al., 2023). These models cover fixed-length, high-resolution, dynamic-resolution, and temporal visual-token settings. We compare against representative training-free visual token reduction methods covering attention-based and representation-based compression, including FastV, SparseVLM, PDrop, V^{2}Drop, VisionZip, and ApET (Chen et al., 2024a; Zhang et al., 2025; Xing et al., 2025; Chen et al., 2026; Yang et al., 2025; Ma et al., 2026).

Implementation Details. All methods are evaluated under the standard inference configurations of their corresponding LVLMs without additional training or fine-tuning. Our method can be integrated into all evaluated backbones: it accumulates token-importance scores from intermediate vision-encoder layers and applies token selection to the final vision-encoder output before the multimodal projector. For EvoCut, we start computing token importance from the middle of the vision encoder, i.e., layer 12 for LLaVA-1.5, LLaVA-NeXT, and Video-LLaVA, and layer 16 for Qwen-2.5-VL. The number of K-means clusters is set to $M = 8$, and the EMA decay factor is set to $\alpha = 0.95$ for all experiments. All experiments are conducted on NVIDIA A800-80G GPUs.

4.2 Main Results

Image understanding tasks. We evaluate EvoCut on LLaVA-1.5-7B, LLaVA-NeXT-7B, and Qwen-2.5-VL-7B during inference without addi-

Table 1: Performance comparison on image understanding benchmarks with LLaVA-1.5-7B under different visual token budgets. Avg. reports the normalized average performance relative to the vanilla upper bound.

Method	GQA	MMB	MMB^{CN}	MME	POPE	SQA	VQA^{V2}	VQA^{Text}	SEED	LLaVA-B	Avg.
Upper Bound, 576 Tokens **(100%)**											
Vanilla	61.9	64.7	58.1	1862	85.9	69.5	78.5	58.2	59.3	66.9	100.0%
Retain 192 Tokens (↓ 66.7%)											
FastV	52.7	61.2	57.0	1612	64.8	67.3	67.1	52.5	57.1	49.4	88.2%
SparseVLM	57.6	62.5	53.7	1721	83.6	69.1	75.6	56.1	55.8	66.1	95.7%
PDrop	57.3	63.2	56.8	1766	82.3	69.0	75.1	56.1	54.7	65.8	96.1%
VisionZip	59.3	63.0	57.3	1778	85.2	68.7	76.6	57.3	56.4	**67.7**	97.8%
V^2Drop	58.5	63.7	56.6	1796	85.1	**69.3**	74.9	55.6	56.4	66.5	97.1%
ApET	**60.2**	63.4	**57.9**	**1808**	86.3	68.5	76.2	54.4	56.8	66.1	97.6%
EvoCut	**60.2**	**64.2**	57.6	1794	**86.5**	68.5	**76.7**	**57.6**	**57.3**	66.4	**98.4%**
Retain 128 Tokens (↓ 77.8%)											
FastV	49.6	56.1	56.4	1490	59.6	60.2	61.8	50.6	**55.9**	52.0	83.8%
SparseVLM	56.0	60.0	51.1	1696	80.5	67.1	73.8	54.9	53.4	62.7	92.5%
PDrop	57.1	61.1	56.6	1644	82.3	68.4	72.9	54.8	53.3	61.9	93.6%
VisionZip	57.6	62.0	56.2	1759	83.2	**68.9**	75.6	56.8	54.9	64.8	95.9%
V^2Drop	56.3	61.8	54.5	1712	80.9	68.8	72.1	53.8	53.8	62.9	93.4%
ApET	58.9	**62.3**	56.4	**1801**	**86.1**	68.7	75.1	53.9	54.7	64.2	96.1%
EvoCut	**59.2**	**62.3**	**56.7**	1790	85.7	68.7	**76.2**	**57.1**	55.2	**65.1**	**97.0%**
Retain 64 Tokens (↓ 88.9%)											
FastV	46.1	48.0	52.7	1256	48.0	51.1	55.0	47.8	51.9	46.1	74.5%
SparseVLM	52.7	56.2	46.1	1505	75.1	62.2	68.2	51.8	51.1	57.5	85.7%
PDrop	47.5	58.8	50.5	1092	55.9	**69.2**	69.2	45.9	40.0	59.2	80.1%
VisionZip	55.1	60.1	55.3	1687	77.0	69.0	72.4	55.5	52.2	62.9	92.6%
V^2Drop	50.5	55.2	49.7	1470	75.1	68.9	71.2	51.8	51.4	62.4	87.8%
ApET	**56.9**	61.2	54.4	**1714**	**84.4**	68.9	72.5	53.0	52.4	**63.0**	93.6%
EvoCut	56.6	**61.8**	**56.0**	1692	83.9	68.8	**73.4**	**55.7**	**53.2**	**63.0**	**94.4%**

tional training, covering fixed-resolution, high-resolution, and dynamic-resolution visual encoding settings. For LLaVA-1.5-7B, Table 1 reports results on a broad set of image understanding benchmarks, including visual question answering, perception, hallucination evaluation, science reasoning, text-rich understanding, and multimodal reasoning tasks (Goyal et al., 2017; Hudson and Manning, 2019; Singh et al., 2019; Lu et al., 2022; Fu et al., 2023; Li et al., 2023c; Liu et al., 2023b). EvoCut achieves the best normalized average performance under all token budgets, retaining 98.4%, 97.0%, and 94.4% of the vanilla performance with 192, 128, and 64 tokens, respectively. Under the most aggressive 64-token setting, EvoCut still outperforms the strongest baseline by 0.8 percentage points in average retention, with clear gains on MMB, MMB^{CN}, VQA^{V2}, TextVQA, and SEED.

Tables 2 and 3 further show consistent gains on LLaVA-NeXT-7B and Qwen-2.5-VL-7B. On LLaVA-NeXT-7B, EvoCut obtains the best average retention under all token budgets, reaching 98.0%, 94.9%, and 91.4% when retaining 640, 320, and 160 tokens. The advantage becomes larger as the token budget decreases, suggesting that multi-layer evolution information is particularly useful under aggressive compression. On Qwen-2.5-VL-7B, EvoCut improves the best competing average retention from 95.5% to 96.2% at the 20% token budget and from 90.5% to 91.7% at the 10% token budget. These results indicate that multi-layer evolution deviation provides a robust token-importance criterion across different LVLM visual encoding settings.

Video understanding tasks. We next apply EvoCut to Video-LLaVA-7B to evaluate whether the proposed criterion generalizes to video understanding, where each input contains substantially more visual tokens across multiple frames. We evaluate on TGIF-QA, MSVD-QA, and MSRVTT-QA (Jang et al., 2017; Chen and Dolan, 2011; Xu et al., 2016). Detailed dataset descriptions are provided

Table 2: Main results on LLaVA-NeXT-7B. Avg. reports the normalized average performance relative to the vanilla upper bound.

Method	GQA	MMB	MMB^{CN}	MME	POPE	VQA^{V2}	VQA^{Text}	Avg.
Upper Bound, 2880 Tokens **(100%)**								
Vanilla	64.2	67.4	60.6	1851	86.5	81.8	61.3	100%
Retain 640 Tokens (↓ 77.8%)								
PDrop	60.6	65.5	58.5	1781	83.7	78.3	57.4	95.8%
VisionZip	61.3	66.2	57.8	1787	85.9	79.1	**60.2**	97.1%
ApET	63.0	65.3	**59.3**	**1815**	**87.2**	**79.2**	57.9	97.6%
EvoCut	**63.2**	**66.3**	59.2	1804	86.3	**79.2**	59.7	**98.0%**
Retain 320 Tokens (↓ 88.9%)								
PDrop	56.4	63.4	56.2	1663	77.6	73.5	54.4	90.4%
VisionZip	59.3	63.1	55.3	1702	82.1	76.2	**58.9**	93.3%
ApET	**61.0**	63.5	56.6	**1783**	85.6	75.8	54.4	94.2%
EvoCut	60.9	**63.7**	**56.9**	1743	**85.8**	**76.4**	57.9	**94.9%**
Retain 160 Tokens (↓ 94.4%)								
PDrop	54.9	61.8	**54.9**	1513	72.3	70.2	52.7	86.4%
VisionZip	55.5	60.1	52.7	1628	74.8	71.4	56.2	88.0%
ApET	58.4	60.8	52.3	**1680**	**82.6**	72.7	53.8	90.1%
EvoCut	**59.4**	**62.0**	53.6	1653	82.2	**73.0**	**57.3**	**91.4%**

Table 3: Performance on Qwen-2.5-VL-7B for image understanding. The original number of visual tokens is dynamic, ranging from 256 to 2048.

Method	GQA	POPE	SQA	MME	MMB	Avg.
Upper Bound, 256–2048 Tokens **(100%)**						
Vanilla	60.5	86.2	76.7	2327	83.3	100%
Retain 20% Tokens on Average (↓ 80%)						
PDrop	55.1	78.4	70.9	2117	77.3	91.6%
VisionZip	56.8	82.4	76.3	2134	**79.3**	95.2%
ApET	57.0	**83.6**	75.5	2211	77.4	95.5%
EvoCut	**57.1**	83.4	**76.7**	**2227**	78.5	**96.2%**
Retain 10% Tokens on Average (↓ 90%)						
PDrop	52.0	74.8	69.7	1886	73.6	86.6%
VisionZip	52.4	78.9	74.1	2003	75.6	90.3%
ApET	**53.4**	**79.3**	74.0	2030	73.8	90.5%
EvoCut	52.8	79.2	**74.5**	**2124**	**75.8**	**91.7%**

Table 4: Performance on video understanding tasks. The original number of visual tokens per video in Video-LLaVA is 2048, while all compression methods retain only 256 tokens.

Method	TGIF		MSVD		MSRVTT		Avg.	
	Acc	Score	Acc	Score	Acc	Score	Acc	Score
Video-LLaVA	46.9	3.34	69.8	3.91	57.1	3.49	100%	100%
FastV	44.2 94.2%	3.29 98.5%	60.3 86.4%	3.72 95.1%	40.6 77.1%	3.18 91.1%	83.9%	94.9%
SparseVLM	45.9 **98.9%**	3.32 99.4%	68.6 98.3%	3.90 99.7%	32.9 57.6%	3.02 86.5%	84.6%	95.2%
PDrop	40.3 85.9%	3.21 96.1%	61.5 88.1%	3.74 95.7%	41.8 73.2%	3.19 91.4%	82.4%	94.4%
VisionZip	44.3 94.5%	3.29 98.5%	65.2 93.4%	3.83 98.0%	54.5 95.4%	3.43 98.3%	94.4%	98.2%
EvoCut	**46.4** **98.9%**	**3.34** **100%**	**69.2** **99.1%**	**3.93** **100.5%**	**55.8** **97.7%**	**3.46** **99.1%**	**98.6%**	**99.9%**

in Appendix A.2.

As shown in Table 4, EvoCut retains 98.6% of the average accuracy and 99.9% of the average score while reducing the number of visual tokens per video from 2048 to 256. Compared with the strongest baseline, EvoCut improves average accuracy retention from 94.4% to 98.6% and average score retention from 98.2% to 99.9%. This shows that evolution-aware scoring remains effective when important visual evidence is distributed across both spatial and temporal dimensions.

Efficiency Analysis. We also compare total inference time, prefilling time, and FLOPs to evaluate practical efficiency gains. As shown in Tables 5 and 6, EvoCut consistently reduces total inference time and prefilling latency under matched token budgets. On LLaVA-1.5-7B, EvoCut reduces the total evaluation time from 17:05 to 11:53 and the prefilling time from 102ms to 68.4ms, achieving 1.44× and 1.49× speedups, respectively. On LLaVA-NeXT-7B, EvoCut further achieves 2.90× total-time and 2.98× prefilling speedups. Although VisionZip obtains marginally lower TFLOPs, EvoCut achieves the lowest wall-clock latency on both backbones, indicating limited scoring overhead while effectively reducing the cost of subsequent multimodal inference. A detailed FLOPs analysis is presented in Appendix A.3.

Qualitative Analysis. Qualitative examples in Appendix A.5 show that EvoCut tends to retain tokens on foreground objects, text regions, and small targets, while filtering out background or repetitive-texture regions. This supports cross-layer evolution deviation as a useful signal for identifying informative visual tokens.

4.3 Ablation Studies

Ablation studies examine three key design choices in EvoCut: the number of group evolution directions, the starting layer for score accumulation, and the EMA decay factor. All ablations are conducted on LLaVA-1.5-7B with 64 retained visual tokens unless otherwise specified.

Effect of the number of evolution directions. To examine how the granularity of group evolution modeling affects token scoring, this ablation varies the number of group evolution directions M. As shown in Table 7, increasing M from 2 to 8 improves the normalized average performance from 91.8% to 95.0%, while further increasing it to 16 reduces the average to 93.5%. Accordingly, $M = 8$

Table 5: Efficiency analysis on LLaVA-1.5-7B using one NVIDIA A800 GPU on POPE.

Methods	Token	Total Time↓	Δ ↑	Prefilling Time↓	Δ ↑	TFLOPs
LLaVA-1.5-7B	576	17:05	1.00×	102ms	1.00×	8.82
+ FastV	64	15:32	1.10×	87.3ms	1.17×	2.26
+ SparseVLM	64	15:57	1.07×	90.1ms	1.13×	2.31
+ PDrop	64	12:56	1.32×	72.5ms	1.41×	2.16
+ VisionZip	64	12:10	1.40×	69.2ms	1.47×	**2.03**
+ V^2Drop	64	12:30	1.37×	71.0ms	1.44×	2.12
+ ApET	64	12:02	1.42×	68.9ms	1.48×	2.09
+ EvoCut	64	**11:53**	**1.44**×	**68.4ms**	**1.49**×	2.04

Table 6: Efficiency analysis on LLaVA-NeXT-7B using one NVIDIA A800 GPU on POPE.

Methods	Token	Total Time↓	Δ ↑	Prefilling Time↓	Δ ↑	TFLOPs
LLaVA-NeXT-7B	2880	35:16	1.00×	206ms	1.00×	31.03
+ FastV	160	28:27	1.24×	117ms	1.76×	7.35
+ SparseVLM	160	30:26	1.16×	136ms	1.51×	7.62
+ PDrop	160	13:45	2.56×	79.5ms	2.59×	6.78
+ VisionZip	160	12:32	2.81×	69.9ms	2.95×	**4.72**
+ V^2Drop	160	12:51	2.74×	73.8ms	2.79×	5.22
+ ApET	160	12:24	2.84×	69.7ms	2.96×	4.79
+ EvoCut	160	**12:10**	**2.90**×	**69.1ms**	**2.98**×	4.74

is used as the default cluster number. This result supports our observation that visual tokens follow multiple evolution directions, while using too many directions may over-fragment shared directions and introduce noisy estimates.

Table 7: Ablation on the number of evolution directions. All variants retain 64 visual tokens on LLaVA-1.5-7B, and $M = 8$ is used as the default setting.

Variant	GQA	MMB	MME	POPE	SQA	VQA^{Text}	Avg.
$M = 2$	55.1	56.7	1640	82.6	67.7	53.9	91.8%
$M = 4$	54.6	58.7	1676	82.2	68.2	55.1	92.9%
$M = 8$	**56.6**	**61.8**	**1692**	**83.9**	**68.8**	55.7	**95.0%**
$M = 16$	55.6	59.4	1662	83.0	67.8	**55.9**	93.5%

Effect of the starting layer. To study how much cross-layer history is needed for stable token scoring, this ablation varies the layer from which EvoCut starts accumulating token-importance scores. Table 8 shows that starting from layer 12 achieves the best MME and POPE scores and ties for the best TextVQA result, while keeping prefilling latency close to the fastest setting. Accordingly, the middle-layer setting is used as the default. This suggests that starting too early may include noisy low-level variations, whereas starting too late loses useful cross-layer evolution history.

Table 8: Ablation on the starting layer for computing token importance. All variants retain 64 visual tokens on LLaVA-1.5-7B.

Variant	MME	POPE	SQA	VQA^{Text}	Prefilling Time↓
Start at layer 1	1691	83.2	**69.2**	55.6	69.5ms
Start at layer 8	1684	82.6	68.4	**55.7**	68.9ms
Start at layer 12	**1692**	**83.9**	68.8	**55.7**	68.4ms
Start at layer 20	1636	82.5	68.2	55.2	**67.7ms**

Effect of EMA decay factor. To assess the effect of historical-score weighting, this ablation studies the EMA decay factor α for multi-layer score accumulation. As shown in Table 9, $\alpha = 0.95$ achieves the best normalized average performance of 95.0% and obtains the best results on most reported benchmarks. Accordingly, $\alpha = 0.95$ is adopted as the default EMA decay factor. This indicates that retaining sufficient history is important for stable token scoring, but overly slow updates may suppress informative changes from later layers.

Table 9: Ablation on the EMA decay factor α for multi-layer evolution scores. All variants retain 64 visual tokens on LLaVA-1.5-7B.

Variant	GQA	MMB	MME	POPE	SQA	VQA^{Text}	Avg.
$\alpha = 0.80$	54.2	58.7	1596	77.8	67.3	53.5	90.6%
$\alpha = 0.90$	55.2	60.4	1643	77.0	68.7	54.8	92.2%
$\alpha = 0.95$	56.6	**61.8**	**1692**	83.9	**68.8**	**55.7**	**95.0%**
$\alpha = 0.97$	**56.9**	61.4	1657	**84.2**	67.9	55.0	94.3%

5 Conclusion

In this paper, we observe that visual tokens follow multiple group evolution directions across vision-encoder layers, and that tokens persistently deviating from these group directions tend to carry important information. Based on this insight, we propose EvoCut, a training-free visual token compression method. EvoCut first clusters token evolution directions, then scores tokens by their deviation from these group directions, and accumulates these scores across layers. Experiments on image and video understanding across multiple LVLM backbones demonstrate that EvoCut consistently outperforms existing training-free compression methods under various token budgets.

Limitations

Although EvoCut achieves strong performance on image and video understanding tasks, it still has several limitations. First, EvoCut requires access to the internal hidden states of the vision encoder to compute token evolution directions and perform clustering. This makes it incompatible with closed-

source or API-only LVLMs where the vision encoder is not exposed, limiting its applicability in scenarios where only black-box model access is available. Second, our experiments focus on representative image and video understanding benchmarks; future work can further evaluate EvoCut on longer videos, higher-resolution inputs, more diverse multimodal reasoning tasks, and a broader range of vision-encoder backbones.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, and 22 others. 2022. Flamingo: A visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*.

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*.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*.

Ricardo J. G. B. Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172. Springer.

David L. Chen and William B. Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Annual Meeting of the Association for Computational Linguistics*.

Junjie Chen, Xuyang Liu, Zichen Wen, Yiyu Wang, Siteng Huang, and Honggang Chen. 2026. Variation-aware vision token dropping for faster large vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024a. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. *arXiv preprint arXiv:2403.06764*.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. 2024b. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *arXiv preprint arXiv:2312.14238*.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven C. H. Hoi. 2023. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Advances in Neural Information Processing Systems*.

Tri Dao. 2023. FlashAttention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Yingqi Fan, Anhao Zhao, Jinlan Fu, Junlong Tong, Hui Su, Yijie Pan, Wei Zhang, and Xiaoyu Shen. 2025. VisiPruner: Decoding discontinuous cross-modal dynamics for efficient multimodal LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 18885–18902.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. 2023. MME: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Barun Patra, Qiang Liu, Kriti Aggarwal, Zewen Chi, Johan Bjorck, Vishrav Chaudhary, Subhojit Som, Xia Song, and

Furu Wei. 2023. Language is not all you need: Aligning perception with language models. In *Advances in Neural Information Processing Systems*.

Drew A. Hudson and Christopher D. Manning. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. 2017. TGIF-QA: Toward spatio-temporal reasoning in visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Lei Jiang, Zixun Zhang, Yuting Zeng, Chunzhao Xie, Tongxuan Liu, Zhen Li, Lechao Cheng, and Xiaohua Xu. 2025. DCP: Dual-cue pruning for efficient large vision-language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21191–21204.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. 2023a. SEED-Bench: Benchmarking multimodal large language models. *arXiv preprint arXiv:2307.16125*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. 2023b. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning*.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023c. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. 2023. Video-LLaVA: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023a. Visual instruction tuning. In *Advances in Neural Information Processing Systems*.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. 2023b. MMBench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems*.

Qiankun Ma, Ziyao Zhang, Haofei Wang, Jie Chen, Zhen Song, and Hairong Zheng. 2026. ApET: Approximation-error guided token compression for efficient VLMs. *arXiv preprint arXiv:2602.19870*.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

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. In *International Conference on Machine Learning*.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. 2025. LLaVA-PruMerge: Adaptive token reduction for efficient large multimodal models. In *IEEE/CVF International Conference on Computer Vision*, pages 22857–22867.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Roziere, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Zichen Wen, Yifeng Gao, Shaobo Wang, Junyuan Zhang, Qintong Zhang, Weijia Li, Conghui He, and Linfeng Zhang. 2025. Stop looking for "important tokens" in multimodal language models: Duplication matters more. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9961–9980.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and Dahua Lin. 2025. PyramidDrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. MSR-VTT: A large video description dataset for bridging video and language. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. 2025. VisionZip: Longer is better but not necessary in vision

language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Hang Zhang, Xin Li, and Lidong Bing. 2023a. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*.

Hang Zhang, Xin Li, and Lidong Bing. 2023b. VideoChat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. 2025. SparseVLM: Visual token sparsification for efficient vision-language model inference. In *International Conference on Machine Learning*.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

A Additional Method Details

A.1 Pseudocode of EvoCut

Algorithm 1 summarizes the complete inference-time procedure of EvoCut. Given the hidden states from the vision encoder, EvoCut starts from a selected layer ℓ_0, computes normalized token evolution directions between adjacent layers, estimates M group evolution directions, and accumulates token-level deviation scores across layers. After the final vision-encoder layer, the top-K patch tokens are retained and sent to the multimodal projector.

Algorithm 1 EvoCut visual token compression

Require: Vision-encoder hidden states $\{X^{(\ell)}\}_{\ell=\ell_0-1}^{L}$, number of retained tokens K, group direction number M, EMA decay factor α

Ensure: Compressed visual token sequence $\hat{X}^{(L)}$

Initialize token scores $s_i^{(\ell_0-1)} \leftarrow 0$ for all patch tokens $i = 1, \ldots, N$
for $\ell = \ell_0$ **to** L **do**
  for $i = 1$ **to** N **do**
    $\vec{\Delta}_i^{(\ell)} \leftarrow \dfrac{x_i^{(\ell)} - x_i^{(\ell-1)}}{\|x_i^{(\ell)} - x_i^{(\ell-1)}\|_2}$
  end for
  Cluster $\{\vec{\Delta}_i^{(\ell)}\}_{i=1}^{N}$ into M groups $\{C_m^{(\ell)}\}_{m=1}^{M}$ with K-means
  for $m = 1$ **to** M **do**
    $\vec{u}_m^{(\ell)} \leftarrow \mathrm{Norm}\left(\frac{1}{|C_m^{(\ell)}|} \sum_{i \in C_m^{(\ell)}} \vec{\Delta}_i^{(\ell)}\right)$
  end for
  for $i = 1$ **to** N **do**
    $a_i^{(\ell)} \leftarrow \max_m \cos(\vec{\Delta}_i^{(\ell)}, \vec{u}_m^{(\ell)})$
    $r_i^{(\ell)} \leftarrow 1 - a_i^{(\ell)}$
    $s_i^{(\ell)} \leftarrow \alpha s_i^{(\ell-1)} + (1-\alpha) r_i^{(\ell)}$
  end for
end for
$S \leftarrow$ indices of the top-K tokens according to $s_i^{(L)}$
$\hat{X}^{(L)} \leftarrow \{x_i^{(L)} \mid i \in S\}$
return $\hat{X}^{(L)}$

A.2 Dataset Descriptions

We evaluate EvoCut on a diverse set of image and video understanding benchmarks to examine whether the proposed token-compression criterion preserves different types of multimodal information. The datasets used in our experiments are summarized below.

Image understanding benchmarks.

VQAv2. VQAv2 evaluates general visual question answering over natural images and requires models to answer diverse questions about objects, attributes, actions, and scenes (Goyal et al., 2017).

GQA. GQA focuses on compositional visual reasoning and tests whether models can answer questions that require object recognition, relation understanding, and multi-step reasoning (Hudson and Manning, 2019).

TextVQA. TextVQA measures text-rich image understanding, where models must recognize and reason over textual content appearing in images (Singh et al., 2019).

ScienceQA. ScienceQA is a multimodal science question answering benchmark that evaluates knowledge-intensive reasoning over image-question pairs (Lu et al., 2022).

MME. MME provides a comprehensive evaluation of LVLM perception and cognition abilities, covering tasks such as object existence, count, position, color, and reasoning (Fu et al., 2023).

POPE. POPE evaluates object hallucination in LVLMs through binary questions and includes random, popular, and adversarial sampling settings (Li et al., 2023c).

MMBench and MMBench-CN. MMBench evaluates multimodal understanding with multiple-choice questions, and MMBench-CN provides a Chinese counterpart for testing cross-lingual multimodal capability (Liu et al., 2023b).

SEED-Bench. SEED-Bench is a multi-dimensional benchmark for evaluating generative multimodal comprehension across images and videos (Li et al., 2023a).

LLaVA-Bench. LLaVA-Bench contains open-ended image-instruction examples and is used to assess instruction-following multimodal response quality (Liu et al., 2023a).

Video understanding benchmarks.

TGIF-QA. TGIF-QA evaluates video question answering over GIF-style videos and emphasizes temporal events, actions, and repeated motion patterns (Jang et al., 2017).

MSVD-QA. MSVD-QA is built on the MSVD video dataset and tests question answering over short web videos with diverse daily activities (Chen and Dolan, 2011).

MSRVTT-QA. MSRVTT-QA evaluates open-ended video question answering on a large collection of web videos covering broad topics and scenes (Xu et al., 2016).

A.3 FLOPs Analysis

EvoCut reduces the computational cost mainly by shortening the visual sequence before the multimodal projector and the LLM prefilling stage. Since compression is performed at the output of the vision encoder, EvoCut does not reduce the FLOPs of the vision encoder itself. Instead, it reduces the FLOPs of all subsequent modules whose cost depends on the number of visual tokens.

Let N denote the original number of visual tokens and K denote the number of retained visual tokens after compression. Let T be the number of text tokens, d be the hidden size of the LLM, d_{ff} be the feed-forward dimension, and L_{llm} be the number of LLM layers. For a decoder-only Transformer layer, the prefilling FLOPs can be approximated, up to constant factors, as

$$\mathcal{F}_{\text{layer}}(S) \approx 4Sd^2 + 2S^2d + 2Sdd_{\text{ff}}, \quad (7)$$

where $S = T+N$ before compression and $S = T+K$ after compression. The three terms correspond to attention projections, self-attention computation, and feed-forward layers, respectively. Therefore, the LLM-side FLOPs reduction is approximately

$$\Delta\mathcal{F}_{\text{llm}} \approx L_{\text{llm}}\big[\mathcal{F}_{\text{layer}}(T+N) - \mathcal{F}_{\text{layer}}(T+K)\big]. \quad (8)$$

This reduction contains both linear terms in the sequence length and the quadratic self-attention term. Thus, the benefit becomes more significant for high-resolution images and videos, where N is large.

The multimodal projector also benefits from token reduction. If the projector maps each visual token from dimension d_v to the LLM hidden size d, its FLOPs decrease from approximately $2Nd_vd$ to $2Kd_vd$. Although this saving is smaller than the LLM-side reduction, it further lowers the prefilling cost.

EvoCut introduces only a lightweight scoring overhead. For L_e evaluated layer transitions, token-direction computation costs $O(L_eNd_v)$. K-means clustering and nearest-direction scoring cost $O(L_eINMd_v)$ and $O(L_eNMd_v)$, respectively, where I is the number of K-means iterations and M is the number of group evolution directions. Since we use a small default value $M = 8$ and do not perform any additional vision-encoder forward pass, this overhead is small compared with the FLOPs saved in the LLM prefilling stage. For LLaVA-1.5-7B, we estimate this overhead under the default setting with $N = 576$, $d_v = 1024$, $M = 8$, $L_e = 13$, and $I = 2$ K-means iterations. The resulting EvoCut scoring cost is approximately 0.40 GFLOPs, i.e., 0.00040 TFLOPs. Compared with the 2.04 TFLOPs of LLaVA-1.5-7B after compression to 64 visual tokens, this overhead accounts for only about 0.020% of the total inference FLOPs.

The measured TFLOPs in Tables 5 and 6 are consistent with this analysis. On LLaVA-1.5-7B, reducing visual tokens from 576 to 64 lowers TFLOPs from 8.82 to 2.04, corresponding to a 76.9% reduction. On LLaVA-NeXT-7B, reducing visual tokens from 2880 to 160 lowers TFLOPs from 31.03 to 4.74, corresponding to an 84.7% reduction. These results show that visual token compression substantially reduces the dominant prefilling computation, while the remaining cost mainly comes from the unchanged vision encoder and lightweight EvoCut scoring.

A.4 Evolution Deviation Visualization

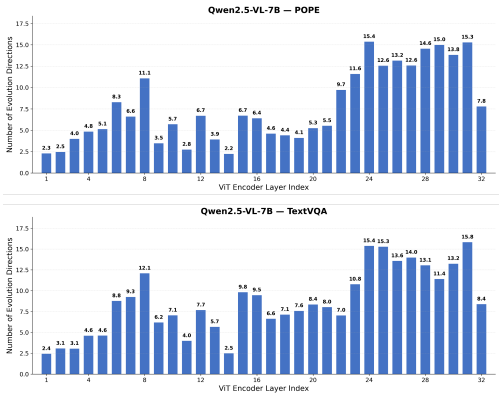

Figure 5: Supplementary visualization for the layer-wise token evolution analysis in Section 3.2. The figure further illustrates how token evolution directions deviate from group evolution directions across multiple vision-encoder layers, complementing the findings in Figure 3.

A.5 Qualitative Visualization

Figure 6 provides qualitative examples of token retention after EvoCut compression. Retained tokens are concentrated on visually informative regions, while filtered tokens mostly lie in background and repetitive-texture areas.

Figure 6: Qualitative visualization of token retention after EvoCut compression. Retained tokens (highlighted) are concentrated on visually informative regions, while filtered tokens mostly lie in background and repetitive-texture areas.