DCP: Dual-Cue Pruning for Efficient Large Vision-Language Models

Lei Jiang[1] **Zixun Zhang**[2] **Yuting Zeng**[1,3] **Chunzhao Xie**[1]
Tongxuan Liu[1,3*] **Zhen Li**[2] **Lechao Cheng**[4*] **Xiaohua Xu**[1*]
[1]University of Science and Technology of China
[2]The Chinese University of Hong Kong, Shenzhen
[3]JD.COM [4]Hefei University of Technology
{jianglei0510, yuting_zeng, xiechunzhao, tongxuan.ltx*}@mail.ustc.edu.cn
{zixunzhang@link.,lizhen@}cuhk.edu.cn
chenglc@hfut.edu.cn* xiaohuaxu@ustc.edu.cn*

Abstract

Large Vision-Language Models (LVLMs) achieve remarkable performance in multimodal tasks but suffer from high computational costs due to the large number of visual tokens. Existing pruning methods either apply after visual tokens enter the LLM or perform pre-pruning based solely on visual attention. Both fail to balance efficiency and semantic alignment, as post-pruning incurs redundant computation, while visual-only pre-pruning overlooks multimodal relevance. To address this limitation, we propose Dual-Cue Pruning (DCP), a novel cross-modal pruning framework that jointly considers textual semantics and visual self-attention. DCP consists of a text-aware computation module, which employs a gradient-weighted attention mechanism to enhance text-visual alignment, and an image-aware computation module, which utilizes deep-layer self-attention distributions to retain essential structural information. By integrating both cues, DCP adaptively selects the most informative visual tokens, achieving efficient inference acceleration while maintaining strong task performance. Experimental results show that DCP can retain only 25% of the visual tokens, with a minimal performance degradation of 0.063% on LLaVA-1.5-13B, demonstrating its effectiveness in balancing efficiency and accuracy.

1 Introduction

In recent years, Large Vision Language Models (LVLMs) (Li et al., 2023a; Zhu et al., 2023; Team et al., 2023; Wang et al., 2024) have exhibited remarkable capabilities in diverse multimodal scenarios, propelling advancements in intricate tasks such as image and language comprehension. These models typically involve a substantial number of visual tokens, ranging from hundreds to thousands (Cai et al., 2024). The large quantity of visual tokens significantly amplifies the training and inference costs of LVLMs (Chen et al., 2024a).

Previous methods aimed at reducing the computational overhead caused by visual tokens can be broadly classified according to the pruning stage within the vision-to-language pipeline. The first category applies pruning after the visual embeddings have been passed into the LLM. These methods identify important visual tokens by analyzing attention weights from LLM text tokens to visual tokens during inference (Chen et al., 2024a; Xing et al., 2024; Ye et al., 2024). However, post-input pruning does not reduce the number of tokens fed into the LLM, resulting in limited computational savings during the prefilling stage. The second category performs pruning before the visual tokens are input into the LLM (Bolya et al., 2023; Shang et al., 2024; Yang et al., 2024; Li et al., 2024b; Jiang et al., 2025). For example, TOME (Bolya et al., 2023) accelerates ViTs by merging similar image features via feature-space clustering. PruMerge (Shang et al., 2024) and VisionZip (Yang et al., 2024) leverage image attention to select core tokens and merge redundant tokens to mitigate information loss. G-Prune (Jiang et al., 2025) constructs a similarity graph among visual tokens to identify key tokens. However, these pre-input pruning methods focus solely on visual-centric features and lack the textual guidance needed to preserve text-relevant visual content.

This limitation becomes evident when analyzing the proportion of visual and textual tokens across several multimodal datasets, as shown in Figure 1. The diagram reveals that visual tokens overwhelmingly dominate most datasets, with proportions reaching as high as 96% (GQA (Hudson and Manning, 2019)) and 97% (TextVQA (Singh et al., 2019)), leaving relatively few tokens for textual information. This imbalance emphasizes the need for pruning strategies that not only reduce the number of visual tokens but also ensure that

*Corresponding authors: Tongxuan Liu, Lechao Cheng, Xiaohua Xu.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 21203–21216
November 4-9, 2025 ©2025 Association for Computational Linguistics

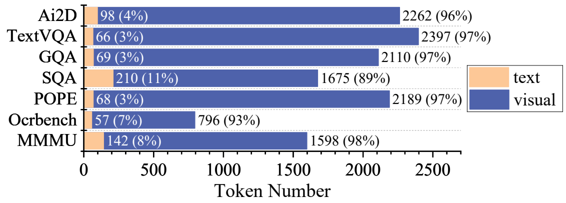

Figure 1: The proportion of visual tokens and textual tokens in seven different datasets.

text-relevant visual information is preserved.

To address these limitations, we propose the **Dual-Cue Pruning (DCP)** method, which enhances pruning efficiency in LVLMs while ensuring that essential semantic and structural information is preserved. Our method consists of two key modules: the *text-aware computation module* and the *image-aware computation module*, followed by a balanced pruning strategy. The text-aware module guides token selection by emphasizing the relevance of visual tokens to the input text, using a gradient-weighted attention mechanism to capture text-visual interactions. The image-aware module focuses on preserving the structural relationships among visual tokens, leveraging deep-layer self-attention distributions to maintain the integrity of visual details. Finally, the pruning strategy combines both text-aware and image-aware scores to select the most informative tokens, ensuring efficient pruning without sacrificing the model's performance. Our contributions can be summarized as follows:

- We introduce a training-free cross-modal pruning framework for LVLMs, which prunes visual tokens *before* inputting them into the LLM.
- We design an efficient token selection strategy by incorporating text-aware score and image-aware score, ensuring robust performance preservation.
- We establish a comprehensive evaluation, demonstrating that DCP achieves 2.26x speedup in Time to First Token (TTFT) on LLaVA-1.5-13B with an average performance degradation of only 0.063%.

2 Related Work

Large Vision-Language Models (LVLMs). The advancement of Large Language Models (LLMs) (Achiam et al., 2023; Touvron et al., 2023) has spurred progress in Large Vision-Language Models (LVLMs) (Yin et al., 2023), extending LLMs' reasoning and understanding to the visual domain by converting visual data into token sequences. A cross-modal projector facilitates this integration by bridging the visual encoder and LLMs (Bai et al., 2023; Liu et al., 2024a) which is achieved through a lightweight Q-Former (Li et al., 2023a) or simpler projection networks like linear layers (Zhu et al., 2023) or MLPs (Liu et al., 2024a).

Despite their effectiveness, existing LVLMs face challenges caused by poor visual token representations (Tong et al., 2024) and visual hallucinations (Huang et al., 2024). Recent work has focused on enhancing visual perception by increasing the resolution of input images. For instance, LLaVA-NeXT (Liu et al., 2024c) and InternVL 1.5 (Chen et al., 2024b) introduce Anyres practice, processing multiple sub-images and the original image's thumbnail independently and then concatenating to project before being input into LLMs, leading to significant improvements in performance for tasks requiring text recognition or reducing hallucinated outputs However, while enhancing the understanding of high-resolution images, this approach also introduces a greater number of visual tokens.

Token Reduction in LVLMs. Visual tokens in LVLMs often outnumber text tokens and exhibit high spatial redundancy, limiting inference efficiency due to autoregressive generation and token redundancy. To address token redundancy, existing methods fall into training-based compression and training-free pruning. Training-based approaches, such as Q-Former (Li et al., 2023a), Resampler (Bai et al., 2023) and Abstractor (Cha et al., 2024) select relevant tokens using learnable queries or convolutional aggregation. LLaVA-Mini (Zhang et al., 2025) introduces modality pre-fusion, thereby facilitating the extreme compression of visual tokens. These methods are effective but suffer from limited generalizability, as they require extensive retraining for each LLM or dataset.

In contrast, training-free methods focus on reducing tokens by merging similar tokens (Bolya et al., 2023; Jiang et al., 2025) or selecting important tokens based on attention scores. FastV (Chen et al., 2024a) prunes low-attention tokens in the LLM backbone, while some others, such as PruMerge (Shang et al., 2024) and VisionZip (Yang et al., 2024) prune tokens with low attention extracted from the CLIP encoder and merge tokens via k-nearest neighbor clustering. G-Prune (Jiang et al.,

2025) proposes a graph-based method that treats tokens as nodes to identify key tokens. However, these approaches often focus on internal visual token attention and overlook text-image correlations, resulting in suboptimal selection. Rather than relying solely on internal image information, PDrop (Xing et al., 2024) drops part of the image tokens based on the attention between all the image tokens and the last token of the instruction. Recent work (Wen et al., 2025) shows that the attention between text and visual tokens in LLMs may not always reflect the actual relevance, limiting the effectiveness of text-guided approaches. Unlike previous work that prunes tokens based on the similarity between textual and image token features (Chen et al., 2025), our approach achieves modality alignment in the visual encoder through attention mechanisms and gradient-based information, enabling more adaptive and informative pruning.

3 Method

To enhance the pruning process for LVLMs, we propose Dual-Cue Pruning (DCP), which incorporates text-aware and image-aware mechanisms to retain semantically significant visual tokens, as illustrated in Figure 2. Based on the analysis and observations presented in Sec. 3.1, our DCP method achieves significant computational savings while maintaining model performance. A detailed technical description of the method is provided in Sec. 3.2.

3.1 Preliminary and Analysis

Image text correlation. Contrastive Language-Image Pretraining (CLIP) (Radford et al., 2021) is a cross-modal model trained on large-scale image-text pairs. It consists of a Vision Encoder and a Text Encoder, both Transformer-based, which project images and text into a shared embedding space through contrastive learning. Given an image I_v and text I_t, CLIP encodes them into feature embeddings $f_v = V(I_v)$ and $f_t = T(I_t)$, and aligns them via cosine similarity:

$$S = \frac{f_v \cdot f_t}{\|f_v\| \, \|f_t\|}. \tag{1}$$

While CLIP effectively captures global image-text alignment, understanding the contribution of individual visual tokens to the final similarity score is critical for interpretability and pruning strategies. To address this, gradient-based visualization techniques such as Grad-CAM (Selvaraju et al., 2017) are widely used for feature attribution analysis. Following this principle, we compute the gradient of the similarity score with respect to the vision encoder's attention matrix:

$$\nabla \mathbf{A}^{(i)} = \frac{\partial S}{\partial A^{(i)}}, \tag{2}$$

where $\nabla \mathbf{A}^{(i)}$ represents the sensitivity of the attention matrix at the i-th layer to the similarity score. To highlight image-text correlations, we compute the element-wise product:

$$\mathbf{M}^{(i)} = \nabla \mathbf{A}^{(i)} \odot \mathbf{A}^{(i)}, \tag{3}$$

where positive values indicate visual regions that strongly align with the text description, while negative values correspond to tokens that contribute less to the cross-modal representation.

By leveraging this mechanism, we derive text-aware saliency scores for visual tokens, facilitating the identification of features that exhibit strong semantic alignment with the input text. This systematic quantification of multimodal interactions provides a foundation for text-guided visual token pruning.

Imbalance Attention in Vision Encoder. Inspired by He et al. (2023), we quantify and visualize the attention maps from selected layers (Layer 1 to 23) in the CLIP model, as shown in Figure 3. We observe that while the shallow layers exhibit relatively balanced attention distribution, the deep layers present a phenomenon known as mode collapse, where over 80% of the attention is concentrated on less than 25% of the tokens. This imbalance in attention suggests that only a few visual tokens with high attention scores contain critical visual information. Based on the phenomenon of Imbalance Attention in Vision Encoders, we propose an image-aware saliency score that quantifies visual token importance through multi-pooling feature representations derived from deep attention maps.

3.2 Dual-Cue Pruning

To improve pruning efficiency while preserving essential semantic and structural information, we propose **Dual-Cue Pruning (DCP)**, which consists of two key components: *text-aware computation module* and *image-aware computation module*, followed by a balanced pruning strategy. *Text-aware computation module* aims to enhance cross-modal alignment by extracting core textual information and guiding token selection based on text-visual

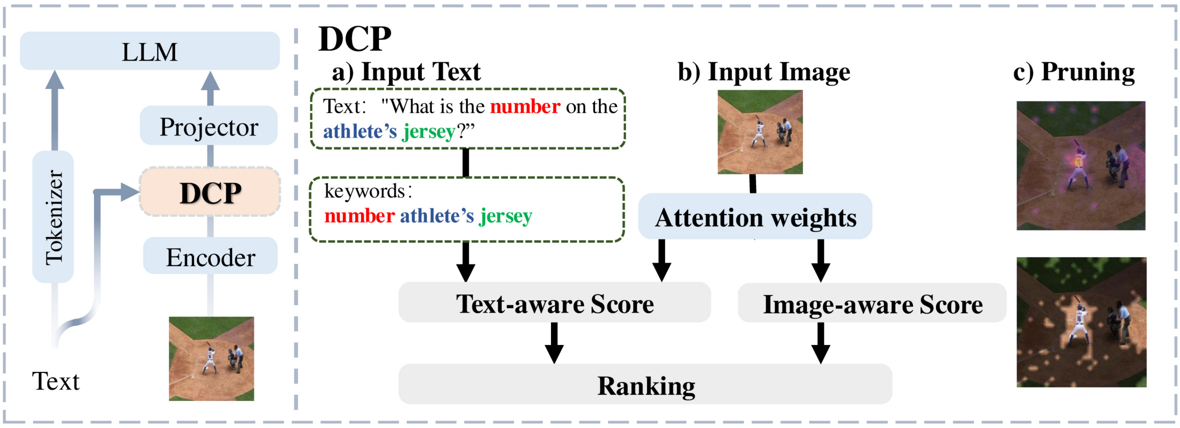

Figure 2: The framework of Dual-Cue Pruning for LVLMs. (a) The text-aware module extracts keywords and computes token relevance using gradient-weighted attention. (b) The image-aware module captures structural relationships via deep-layer self-attention. (c) Both scores are combined to rank and prune visual tokens before feeding into the LLM.

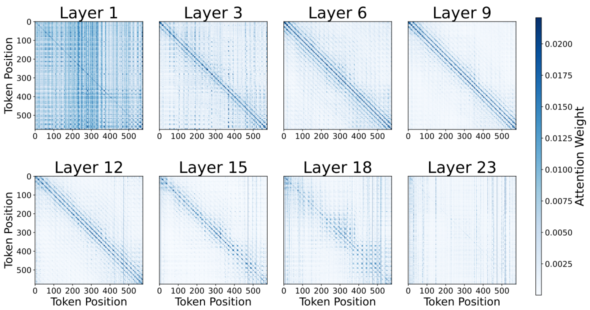

Figure 3: The attention map of CLIP in different layers.

interactions. First, a lightweight NLP model is used to extract key problem-related words from the input text by removing system prompts, response instructions, and options. Then, a gradient-weighted attention mechanism is introduced to emphasize visual tokens that strongly respond to textual content, ensuring effective cross-modal guidance. *Image-aware computation module* focuses on preserving structural information by leveraging deep-layer self-attention distributions. The self-attention matrix from a selected deep layer is extracted, excluding the influence of the `CLS` token, and used to compute the mean attention score for visual tokens. This captures inherent structural relationships, complementing the text-aware module to prevent the removal of critical visual details. Finally, the *pruning strategy* balances textual relevance and structural importance by ranking visual tokens based on both text-aware and image-aware scores. A token budget is allocated by selecting the most informative tokens from each ranking, ensuring a refined subset that maintains both semantic fidelity and computational efficiency.

3.2.1 Text-aware Computation Module

Existing pruning methods for LVLMs lack sufficient modeling of textual information, which may result in the inadvertent removal of visual tokens closely related to textual content during the pruning process. To address this issue, we introduce a gradient-weighted attention mechanism to enhance cross-modal interaction.

We first apply regular expressions to remove system prompts, response instructions, and answer options, retaining only the core problem description. Next, we perform syntactic parsing using spaCy (`en_core_web_sm`), a lightweight 12 MB NLP model (Explosion, 2023), to obtain part-of-speech tags and dependency structures. Based on these results, we extract noun phrases and discard non-semantic stopwords, yielding up to N problem-related keywords. This results in a compact yet semantically informative textual representation, aligned with CLIP's 77-token input constraint. More details can be found in appendix.

Then we compute the cosine similarity between the image embedding and text embedding. Since this similarity function is differentiable, we can trace its gradient to capture the response of visual tokens to textual information. For selected layers of the visual encoder (e.g., the last two layers), we compute the gradient of the attention matrix $A^{(i)}$ with respect to the scoring function S, denoted as $\nabla A^{(i)}$. We then define the gradient-enhanced attention mapping:

$$\mathrm{TRM}^{(i)} = \mathrm{ReLU}\big(A^{(i)} \odot \nabla A^{(i)}\big), \quad (4)$$

where element-wise multiplication highlights the visual tokens that contribute significantly to the final decision.

To incorporate multi-layer information, we adopt a cumulative update strategy via batch matrix multiplication:

$$R \leftarrow R + \text{bmm}\big(\text{TRM}^{(i)}, R\big), \tag{5}$$

where R is initialized as an identity matrix. The final text-aware importance score r_{text} is obtained by normalizing R. Ablation studies indicate that fusing information from the last two layers leads to better preservation of text-related visual details.

3.2.2 Image-Aware Computation Module

To achieve more precise pruning while preserving core visual information, we design an image-aware computation module. As discussed in Sec. 3.1, this module leverages the imbalance of attention distributions in deeper layers by first obtaining the multi-head attention (MHA) output from the vision encoder. We compute the mean attention across all heads and subsequently perform an additional mean operation along the token dimension to derive the importance of each token:

$$r_{\text{att}} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{H}\sum_{h=1}^{H} A_{h,j}^{L}\right), \tag{6}$$

where H is the number of attention heads, N denotes the number of non-CLS tokens, and $A_{h,j}^{L}$ represents the attention score of the j-th token in the L-th layer for the h-th head. Through ablation studies, we find that extracting the attention matrix from the 18th layer best captures the structural relationships and relative importance of visual tokens, effectively compensating for potentially missing visual information in text-guided pruning.

3.2.3 Dual-Cue Balanced Pruning Strategy

Given an input text-image pair, DCP first extracts keywords from the text and computes text-visual relevance scores. Simultaneously, it leverages deep-layer self-attention distributions to estimate visual token importance. After obtaining the text relevance score r_{text} and the self-attention score r_{att}, a balanced pruning strategy is adopted to retain the most informative tokens. For a predetermined token budget K, the procedure is as follows: 1) Independently sort the visual tokens based on r_{text} and r_{att}. 2) Select the top $\frac{K}{2}$ tokens from the r_{text} ranking. 3) From the remaining tokens, select the top $\frac{K}{2}$ tokens according to r_{att}. 4) Merge the selected token indices and reorder them based on their original sequence to preserve the input order for downstream processing. This dual-cue pruning strategy effectively combines textual guidance with inherent visual structure, ensuring robust and efficient inference acceleration. The overall procedure is summarized in Algorithm 1.

Algorithm 1 Dual-Cue Pruning (DCP)

Require: Input text T, input image I, token budget K
Ensure: Selected visual tokens $\mathcal{V}_K$
Text-Aware Computation
Extract keywords from T using NLP model
Compute text-visual similarity:
  $S = \cos(\text{Embed}(I), \text{Embed}(T))$
Compute gradient-weighted attention:
  $\text{TRM}^{(i)} = \text{ReLU}(A^{(i)} \odot \nabla A^{(i)})$
Fuse multi-layer attention:
  $R \leftarrow R + \text{bmm}(\text{TRM}^{(i)}, R)$
Compute text-importance score:
  $r_{\text{text}} = \text{Normalize}(R)$
Image-Aware Computation
Extract deep-layer self-attention matrix A^L
Remove `CLS` token influence
Compute mean attention score:
  $r_{\text{att}} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{H}\sum_{h=1}^{H} A_{h,j}^{L}\right)$
Dual-Cue Balanced Pruning Strategy
Rank visual tokens based on r_{text} and r_{att}
Select top $\frac{K}{2}$ tokens from r_{text} ranking
Select remaining $\frac{K}{2}$ tokens from r_{att} ranking
Merge and reorder selected tokens to preserve input order
return $\mathcal{V}_K$

4 Experiments

4.1 Experimental Setup

Datasets. We utilize 7 widely used multimodal datasets to evaluate the performance, including POPE (Li et al., 2023b), MMMU (Yue et al., 2024), ScienceQA (SQA) (Lu et al., 2022), Ai2D (Kembhavi et al., 2016), GQA (Hudson and Manning, 2019), TextVQA (VQA^{T}) (Singh et al., 2019) and OCRBench (OCR) (Liu et al., 2023). More details of datasets can be found in appendix.

Implementation Details. All experiments are conducted on the LMMS-Eval platform (Zhang et al., 2024), a unified and reproducible benchmark covering 50+ multimodal datasets and 10+ LVLMs. We evaluate on five representative models: `LLaVA-1.5-7B`, `LLaVA-1.5-13B` (Liu et al., 2024b), `LLaVA-NeXT-7B`, `LLaVA-NeXT-13B` (Liu et al., 2024c) and `OneVision-Qwen2-7B` (Li et al., 2024a). Our DCP framework supports variable visual token retention ratios, with experiments conducted under multiple settings (e.g., 25%, 50%, 75%) to evaluate scalability and robustness. Inference efficiency is measured using Time to First Token (TTFT) and Time Per Output Token (TPOT).

We compare against two categories of training-free pruning baselines, categorized by pruning stage: post-input pruning (FastV, PDrop) and pre-input pruning (TOME, PruMerge, G-Prune). All methods are configured according to official or widely adopted settings. Specifically, FastV prunes at layer 3; PDrop uses layers 8/16/24 with retention ratios p_1 = [0.75, 0.375, 0.1875], p_2 = [0.5, 0.25, 0.125], and p_3 = [0.25, 0.125, 0.0625]; other baselines use their default settings. All evaluations are re-run under the same platform to ensure consistency.

4.2 Main Results

Comparison with state-of-the-art methods. Table 1 presents the accuracy and inference efficiency of various pruning methods on LLaVA-1.5–7B across three different token retention ratios. Our DCP consistently outperforms prior methods across multiple evaluation datasets. At the 75% retention ratio, DCP achieves the highest average accuracy of 55.26%, outperforming the second best FastV by 0.03%, even higher than the original model without pruning. Notably, DCP achieves the highest efficiency, with speedup ratios of 1.20× (TTFT) and 1.29× (TPOT) over the original model. Similarly, our DCP ranks first in overall average accuracy with 54.91% and achieves the best performance in four subsets at the 50% pruning level, while retaining strong efficiency with a speedup of 1.52x and 1.29x, respectively. Even under the most aggressive 25% pruning ratio, DCP leads with an average score of 54.51%, surpassing the second best FastV and TOME by 1.87%. These results align with our motivation. Unlike post-input pruning methods that rely on cross-modal attention but do not reduce LLM input size, and pre-input methods that prune early but ignore text relevance, DCP combines both strengths—performing early token reduction while leveraging textual and visual signals. This leads to more relevant token selection and a better trade-off between accuracy and efficiency.

DCP on different LVLMs. To evaluate the generalizability of our method, we apply DCP across various LVLMs, including `LLaVA-1.5-13B`, `LLaVA-NeXT-7B`, `LLaVA-NeXT-13B`, as reported in Figure 4 and Table 2. Figure 4 shows that as token retention increases, model accuracy generally improves. However, different datasets stabilize at

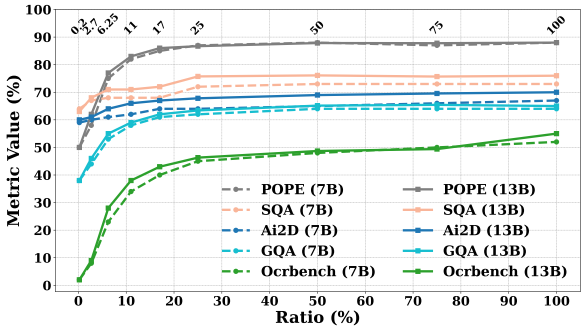

Figure 4: Performance metrics across visual token retention ratios for the `LLaVA-NeXT-7B` and `LLaVA-NeXT-13B` models on five datasets.

varying retention ratios. For instance, SQA, Ai2D, and POPE maintain good accuracy even with low retention, while OCRBench requires higher retention to achieve significant accuracy. Interestingly, Ai2D and SQA remain robust even at extreme pruning levels, suggesting that key information is concentrated in a small number of tokens, making them less sensitive to pruning.

As shown in Table 2, DCP can effectively prune a significant number of redundant tokens without sacrificing much accuracy, enabling efficient inference. For instance, at 25% token retention, the `LLaVA-NeXT-13B` model maintains an accuracy of 67.94%, while the `LLaVA-1.5-13B` and `LLaVA-NeXT-7B` models achieve similar improvements across datasets, demonstrating that DCP's pruning approach is effective across different architectures. The inference speedup is also notable, with `LLaVA-NeXT-13B` achieving up to a 2.52x speedup, further proving the scalability and consistency of DCP across different model configurations.

4.3 Ablation Studies

Results on SigLIP as encoder. DCP is not only applicable to models utilizing CLIP as the encoder but also demonstrates strong performance when implemented with SigLIP as the encoder. In the computation of DCP, we continue to leverage the accumulated gradient information from the last two layers along with attention weight information to enhance the extraction of critical features. Experimental results on Table 3 indicate that the OneVision-Qwen2-7B model, which employs SigLIP as the encoder and Qwen2 as the LLM, achieves consistently strong performance across multiple datasets. Notably, even when the visual input is reduced to 25%, the model maintains robust performance, par-

Ratio	Method	Accuracy Performance								Inference Efficiency			
		Ai2D	GQA	MMMU	SQA	POPE	VQA^T	OCR	Avg.	TTFT (ms	S_p)	TPOT (ms/tok.	S_p)
100	Vanilla	55.50	61.97	35.30	69.51	86.98	46.00	31.20	55.21	74	-	27.71	-
	TOME(ICLR'23)	54.31	59.36	35.78	68.87	**86.98**	41.47	29.20	53.71	91	0.81x	32.41	0.86x
	PruMerge(2024.03)	53.21	60.41	36.33	68.47	85.60	40.66	29.60	53.47	91	0.81x	26.56	1.02x
	Fastv(ECCV'24)	55.34	61.61	36.11	**69.51**	86.69	**46.08**	**31.30**	<u>55.23</u>	69	1.07x	27.40	1.01x
75	G-Prune(AAAI'25)	54.70	58.63	34.89	68.22	86.32	40.96	29.30	53.29	<u>66</u>	<u>1.12x</u>	<u>25.18</u>	<u>1.10x</u>
	PDrop(CVPR'25)	<u>55.38</u>	<u>61.64</u>	**36.78**	<u>69.11</u>	<u>86.87</u>	45.65	30.90	55.19	184	0.40x	25.91	1.07x
	DCP	**55.86**	**61.65**	<u>36.67</u>	68.82	86.86	<u>45.89</u>	<u>31.10</u>	**55.26**	**62**	**1.20x**	**21.45**	**1.29x**
	TOME(ICLR'23)	54.33	59.61	36.11	68.71	<u>87.23</u>	40.33	28.20	53.50	89	0.83x	26.82	1.03x
	PruMerge(2024.03)	54.24	56.82	<u>36.56</u>	**69.36**	79.63	39.45	27.80	51.98	86	0.44x	26.33	1.05x
	Fastv(ECCV'24)	**55.08**	<u>60.33</u>	35.89	68.67	85.20	<u>45.51</u>	<u>30.60</u>	54.47	59	1.26x	27.16	1.02x
50	G-Prune(AAAI'25)	<u>54.95</u>	57.29	36.00	**69.36**	83.78	40.89	29.30	53.08	<u>52</u>	<u>1.42x</u>	25.30	1.10x
	PDrop(CVPR'25)	54.53	60.16	**36.78**	<u>69.31</u>	86.18	45.24	29.80	<u>54.57</u>	154	0.48x	<u>24.02</u>	<u>1.15x</u>
	DCP	54.83	**60.85**	35.89	68.52	**87.26**	**45.61**	**31.40**	**54.91**	**49**	**1.52x**	**21.41**	**1.29x**
	TOME(ICLR'23)	<u>54.08</u>	<u>58.67</u>	<u>36.33</u>	68.12	**87.24**	38.04	26.00	<u>52.64</u>	79	0.94x	22.93	1.21x
	PruMerge(2024.03)	53.85	53.48	36.00	<u>69.56</u>	75.40	38.14	26.70	50.45	74	1.00x	<u>21.73</u>	<u>1.28x</u>
	Fastv(ECCV'24)	53.95	57.47	35.44	68.86	81.21	42.56	<u>29.00</u>	<u>52.64</u>	51	1.44x	27.28	1.02x
25	G-Prune(AAAI'25)	**54.40**	54.04	35.22	**69.71**	79.38	40.85	28.20	51.69	<u>44</u>	<u>1.68x</u>	25.58	1.08x
	PDrop(CVPR'25)	53.30	57.13	35.11	69.36	82.40	<u>44.23</u>	22.70	52.03	135	0.55x	22.11	1.25x
	DCP	**54.40**	**58.92**	**36.56**	68.72	<u>87.14</u>	**44.82**	**31.00**	**54.51**	**38**	**1.97x**	**20.89**	**1.33x**

Table 1: Accuracy and inference efficiency of different methods using `LLaVA-1.5-7B`. Inference efficiency is measured on the POPE dataset. **Bold** indicates the best, <u>underlined</u> the second-best result. *Avg.* is the average value, *ms* denotes milliseconds, S_p represents the speedup ratio and *ms/tok.* indicates milliseconds per token.

Model	Ratio	Accuracy Performance								Inference Efficiency			
		Ai2D	GQA	MMMU	SQA	POPE	VQA^T	OCR	Avg.	TTFT (ms	S_p)	TPOT (ms/tok.	S_p)
	100	59.49	63.25	34.80	72.88	87.09	48.73	33.70	57.13	138	-	33.41	-
LLaVA-1.5-13B	75	58.65	61.11	**36.11**	**72.98**	**87.91**	48.24	33.30	56.90	110	1.25x	32.56	1.03x
	50	57.67	60.94	**35.89**	**74.07**	**87.82**	47.95	33.60	56.85	88	1.57x	32.04	1.04x
	25	57.29	59.43	**37.33**	**73.43**	**87.18**	47.03	**33.80**	56.50	61	**2.26x**	31.5	**1.06x**
	100	66.58	64.24	35.10	70.15	87.61	64.90	52.20	62.97	88	-	23.70	-
LLaVA-NeXT-7B	75	65.06	**64.44**	**37.11**	70.00	**87.80**	63.39	49.30	62.44	81	1.09x	23.55	1.01x
	50	65.16	64.06	**37.44**	68.82	**88.00**	62.34	48.30	62.02	60	1.47x	23.05	1.03x
	25	64.38	62.70	**37.11**	67.97	87.57	60.38	45.30	60.77	50	**1.78x**	21.97	**1.08x**
	100	70.30	65.37	35.90	73.57	87.56	67.10	55.10	64.99	198	-	38.30	-
LLaVA-NeXT-13B	75	70.01	65.30	**37.56**	73.23	**87.92**	65.30	49.90	64.17	153	1.29x	36.06	1.06x
	50	69.43	65.05	**37.11**	**74.17**	**87.98**	63.34	49.60	63.81	116	1.70x	33.35	1.15x
	25	67.94	64.14	**36.22**	73.08	**87.89**	62.78	47.10	62.74	79	**2.52x**	30.94	**1.24x**

Table 2: Accuracy performance and inference efficiency under different LVLMs. Inference efficiency is measured on the POPE dataset. Results better compared to no pruning are **bold**. *Avg.* is the average value, *ms* denotes milliseconds, S_p represents the speedup ratio and *ms/tok.* indicates milliseconds per token.

ticularly on POPE and TextVQA, where the impact of data reduction is minimal.

Effectiveness of dual-cue importance indicators. DCP is composed of two key importance indicators: text-aware and image-aware significant score. Specifically, DCP uses the penultimate two layers for text-related accumulation and selects Layer 18 as the image-aware attention matrix. The *random* baseline represents a setting where tokens are pruned randomly. In this experiment, we retain 25% of the tokens and evaluate performance on `LLaVA-1.5-7B`. The results on Table 4 demonstrate that removing either text-aware or image-aware components leads to a noticeable performance drop compared to the full DCP method. Specifically, the absence of text-aware features results in a decrease in average performance from 63.63% to 63.04%, while removing image-aware features lowers the score to 63.13%. This highlights the complementary nature of both components. Additionally, the random pruning baseline performs significantly worse, resulting in a particularly poor performance

Model	Ratio	Ai2D	GQA	POPE	VQA^T
OneVision-Qwen2-7B	100%	81.35	62.22	89.13	76.03
	75%	80.70	**62.47**	**89.36**	76.01
	50%	79.89	**62.38**	**89.73**	75.98
	25%	79.44	60.87	89.11	75.99

Table 3: DCP on `OneVision-Qwen2-7B` models, whose encoder is SigLIP.

Method	VQA^T	GQA	POPE	AVG
random	30.89	58.39	84.14	57.81
w/o text-aware	44.44	58.44	86.24	63.04
w/o image-aware	44.27	58.34	86.79	63.13
DCP	**44.82**	**58.92**	**87.14**	**63.63**

Table 4: Ablation study on `LLaVA-1.5-7B` with 25% token retention. The table compares different ablation settings of the proposed DCP method. "random" refers to a baseline where tokens are pruned randomly. "w/o text-aware" removes the text-awareness component, and "w/o image-aware" eliminates the image-awareness component. The "DCP" row represents our full method, which integrates both text-aware and image-aware importance indicators. The final column presenting the average performance across all datasets.

on TextVQA (30.89%), indicating that visual token pruning guided by cross-modal dual-cue is crucial for maintaining high performance and key information of image.

Ablation study on text-aware component. We analyze the effect of selecting different layers for computing gradient-based importance in the text-aware component. Layers 6, 12, and 18 use gradients from a single layer, whereas Layer 23 accumulates gradients from Layers 23 and 24, following LLaVA's practice of using the penultimate layer for image feature extraction. The results on Table 5 show that early layers perform worse, with Layer 12 yielding the lowest average score of 59.51. Layer 6 improves slightly to 61.48, while Layer 18 achieves 61.79. The best performance is obtained with Layer 23, reaching an average score of 63.13, demonstrating that integrating gradients from deeper layers enhances text-aware token selection.

Ablation study on image-aware component. To investigate the effect of using attention maps from different layers in the image-aware component, we evaluate the performance when selecting attention maps from a single layer. The results on

Layer	VQA^T	GQA	POPE	AVG
6	38.76	59.16	86.52	61.48
12	34.67	58.28	85.57	59.51
18	41.44	58.19	85.74	61.79
23	**44.27**	**58.34**	**86.79**	**63.13**

Table 5: Ablation study on text-aware component using gradients from different layers. Layers 6, 12, and 18 use gradients from a single layer, while Layer 23 accumulates gradients from the last two layers (23-24).

Layer	VQA^T	GQA	POPE	AVG
6	30.67	56.65	84.37	57.23
12	44.16	58.58	86.51	63.08
18	**44.44**	**58.44**	**86.24**	**63.04**
23	44.23	57.94	85.07	62.41
24	39.29	57.91	82.99	60.06

Table 6: Ablation study on the image-aware component using attention maps from different layers.

Table 6 show that Layer 6 performs the worst, with an average score of 57.23%, indicating that early-layer attention does not effectively capture meaningful image features. Performance improves significantly when using Layer 12 (63.08%) and Layer 18 (63.04), suggesting that mid-to-deep layers contain more informative spatial relationships. However, Layer 23 (62.41%) and Layer 24 (60.06%) show performance degradation, especially in POPE and TextVQA, implying that attention from the final layers may be less reliable for capturing fine-grained image token importance. These findings suggest that selecting attention maps from mid-to-deep layers is more beneficial for enhancing the image-aware component.

5 Conclusion

In this paper, we propose Dual-Cue Pruning (DCP), a novel pruning framework that enhances the efficiency of Large Vision-Language Models (LVLMs) by jointly leveraging textual and visual cues. Unlike existing methods that focus solely on visual features, DCP integrates a text-aware computation module, which enhances cross-modal alignment using a gradient-weighted attention mechanism, and an image-aware computation module, which extracts deep-layer self-attention distributions to retain structural visual information. By balancing these two cues, DCP effectively prunes visual tokens while maintaining model fidelity. Extensive experiments demonstrate that DCP achieves substantial speedup while preserving model accuracy, outperforming existing pruning approaches.

Limitations

Despite its effectiveness, DCP has several limitations. First, it relies on a pre-trained NLP model for keyword extraction, which may introduce inaccuracies when handling complex or ambiguous prompts. Second, while DCP demonstrates strong performance across standard multimodal benchmarks, the degree of efficiency gain may vary with different model architectures and prompt styles. Future work could investigate adaptive pruning strategies tailored to specific vision-language tasks or dynamic prompt characteristics to further improve robustness and generalization.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) with Grant No.62172383 and No.62231015, Anhui Provincial Key R&D Program with Grant No.S202103a05020098, Research Launch Project of University of Science and Technology of China (USTC) with Grant No.KY0110000049.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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. *Preprint*, arXiv:2308.12966.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2023. Token merging: Your vit but faster. In *ICLR*.

Mu Cai, Haotian Liu, Siva Karthik Mustikovela, Gregory P Meyer, Yuning Chai, Dennis Park, and Yong Jae Lee. 2024. Vip-llava: Making large multimodal models understand arbitrary visual prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12914–12923.

Junbum Cha, Wooyoung Kang, Jonghwan Mun, and Byungseok Roh. 2024. Honeybee: Locality-enhanced projector for multimodal llm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13817–13827.

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. In *European Conference on Computer Vision*, pages 19–35. Springer.

Yi Chen, Jian Xu, Xu-Yao Zhang, Wen-Zhuo Liu, Yang-Yang Liu, and Cheng-Lin Liu. 2025. Recoverable compression: A multimodal vision token recovery mechanism guided by text information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 2293–2301.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, and 1 others. 2024b. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 67(12):220101.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and 1 others. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Explosion. 2023. en_core_web_sm. https://huggingface.co/spacy/en_core_web_sm. Version 3.7.1.

Jingxuan He, Lechao Cheng, Chaowei Fang, Dingwen Zhang, Zhangye Wang, and Wei Chen. 2023. Mitigating undisciplined over-smoothing in transformer for weakly supervised semantic segmentation. *arXiv preprint arXiv:2305.03112*.

Wen Huang, Hongbin Liu, Minxin Guo, and Neil Zhenqiang Gong. 2024. Visual hallucinations of multi-modal large language models. *arXiv preprint arXiv:2402.14683*.

Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709.

Yutao Jiang, Qiong Wu, Wenhao Lin, Wei Yu, and Yiyi Zhou. 2025. What kind of visual tokens do we need? training-free visual token pruning for multi-modal large language models from the perspective of graph. *arXiv preprint arXiv:2501.02268*.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. 2016. A diagram is worth a dozen images. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 235–251. Springer.

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

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jianke Zhu, and Lei Zhang. 2024b. Token-packer: Efficient visual projector for multimodal llm. *arXiv preprint arXiv:2407.02392*.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306.

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

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024c. Llava-next: Improved reasoning, ocr, and world knowledge.

Yuliang Liu, Zhang Li, Biao Yang, Chunyuan Li, Xucheng Yin, Cheng-lin Liu, Lianwen Jin, and Xiang Bai. 2023. Ocrbench: On the hidden mystery of ocr in large multimodal models. *arXiv preprint arXiv:2305.07895*.

Pan Lu, Swaroop Mishra, Tanglin 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. *Advances in Neural Information Processing Systems*, 35:2507–2521.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626.

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*.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. 2024. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9568–9578.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, and 1 others. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Zichen Wen, Yifeng Gao, Weijia Li, Conghui He, and Linfeng Zhang. 2025. Token pruning in multimodal large language models: Are we solving the right problem? *arXiv preprint arXiv:2502.11501*.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and 1 others. 2024. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. *arXiv preprint arXiv:2410.17247*.

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*.

Weihao Ye, Qiong Wu, Wenhao Lin, and Yiyi Zhou. 2024. Fit and prune: Fast and training-free visual token pruning for multi-modal large language models. *arXiv preprint arXiv:2409.10197*.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2023. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, and 1 others. 2024.

Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and 1 others. 2024. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*.

Shaolei Zhang, Qingkai Fang, Zhe Yang, and Yang Feng. 2025. Llava-mini: Efficient image and video large multimodal models with one vision token. *arXiv preprint arXiv:2501.03895*.

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 Datasets

We utilize 7 widely used multimodal datasets to evaluate the performance of our method, covering diverse vision-language reasoning scenarios:

- **POPE** (Li et al., 2023b): Designed to assess object hallucination in multimodal models. It evaluates whether the model can accurately determine the existence of objects mentioned in the text within the image.
- **MMMU** (Yue et al., 2024): A comprehensive benchmark spanning multiple university-level subjects (e.g., biology, physics, math), requiring high-level reasoning across image and text modalities.
- **ScienceQA** (Lu et al., 2022): Focused on scientific question answering, this dataset includes diagrams, textual descriptions, and multiple-choice questions across various science topics.
- **Ai2D** (Kembhavi et al., 2016): Targets the interpretation of complex scientific and educational diagrams. It tests a model's ability to perform diagram-based reasoning and question answering.
- **GQA** (Hudson and Manning, 2019): A large-scale visual question answering benchmark emphasizing compositional reasoning and spatial relationships grounded in real-world images.
- **TextVQA** (Singh et al., 2019): Involves questions related to text present in images. It evaluates the model's capability to localize, read, and reason about textual content embedded in complex visual scenes.
- **OCRBench** (Liu et al., 2023): Focuses on optical character recognition in natural images, measuring the model's ability to extract and understand text from images with varied layout, quality, and language content.

These datasets jointly test diverse capabilities of LVLMs, including object grounding, scientific reasoning, diagram interpretation, text recognition, and cross-modal understanding. All evaluations are conducted using standardized metrics and protocols provided by LMMS-Eval (Zhang et al., 2024).

B Vision-Language Inference Pipeline and Latency Analysis

B.1 Large Vision-Language Models (LVLMs)

LVLMs are aimed at generating textual responses based on input images and instructions (Yin et al., 2023). A typical LVLM consists of three key modules: a vision encoder, an advanced LLM, and a projector, which serves as a bridge for modality alignment. First, the vision encoder transforms the input image into visual embeddings $\mathbf{E_v}$, often utilizing the ViT architecture (Dosovitskiy et al., 2020). Next, the projector converts these visual embeddings into visual tokens $\mathbf{T_v}$ by mapping them into the text space, making them understandable to the LLM. Given the generated visual tokens $\mathbf{T_v}$ and instructions' textual tokens $\mathbf{T_t}$, the LLM then produces the L-length output response $\mathbf{Y}$ in an auto-regressive manner based on the following probability distribution:

$$P(\mathbf{Y}|\mathbf{T_t}, \mathbf{T_v}) = \prod_{i=1}^{L} P(\mathbf{Y}_i|\mathbf{T_t}, \mathbf{T_v}, \mathbf{Y}_{<i}). \quad (7)$$

As shown in the formula, the inference efficiency and memory requirements of LVLMs heavily depend on the length of the input tokens that the LLM needs process, which consist of both textual and visual tokens. In fact, due to the auto-regressive nature of LLM decoding, the computational complexity of the LLM is proportional to the square of the input token length. This indicates that reducing the input tokens is crucial for improving the inference efficiency of LVLMs.

B.2 LLM Inference Pipeline

The inference process of the LLM consists of two computationally distinct stages:

B.2.1 Prefill Stage

The **prefill stage** processes the entire input sequence $\mathbf{X}$ in one forward pass through the transformer layers. At each layer l, self-attention is applied to the entire sequence:

$$\mathbf{Z}^{(l)} = \mathrm{MHSA}^{(l)}(\mathbf{X}^{(l-1)}) + \mathbf{X}^{(l-1)}$$

The multi-head self-attention involves computing attention weights:

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)\mathbf{V}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{(N_t+N_v)\times d_k}$ are projected query, key, and value matrices.

The overall complexity is:

$$\mathcal{O}\left((N_t + N_v)^2 \cdot d\right)$$

This quadratic scaling means that large N_v (as is common in LVLMs) leads to significant latency during this stage.

B.2.2 Decoding Stage

Once the KV caches are built during prefill, decoding proceeds token by token. At step i, the model generates the i-th output token $\hat{y}_i$ given the previous tokens and the cached states:

$$\hat{y}_i = f_{\text{llm}}(\hat{y}_{<i}, \text{cache})$$

Each decoding step only attends to the newly generated token and previously cached keys and values, with attention complexity:

$$\mathcal{O}(N_{\text{ctx}} \cdot d)$$

where N_{ctx} is the context length (fixed during inference). This stage is relatively lightweight compared to prefill.

B.3 Latency Metrics

To quantify the latency performance of pruning methods, we use two common metrics:

- **Time to First Token (TTFT)**: Measures the wall-clock time from input submission to the generation of the first output token. It corresponds to the entire prefill stage:

$$\text{TTFT} \approx \text{Latency}_{\text{prefill}} \propto (N_t + N_v)^2$$

- **Time Per Output Token (TPOT)**: Measures the average latency of decoding each token after the first:

$$\text{TPOT} = \frac{\text{Total Decoding Time}}{\text{Number of Output Tokens}} \propto N_{\text{ctx}} \cdot d$$

B.4 Efficiency Implication of Pruning

Post-input pruning (e.g., FastV, PDrop) operates after visual tokens are passed into the LLM. These methods may reduce computation during decoding, but have minimal impact on TTFT since the prefill complexity remains unchanged.

Pre-input pruning aims to reduce inference latency by removing visual tokens before they are passed into the LLM, thereby directly decreasing N_v and the computational cost of the prefill stage. Existing pre-input methods such as TOME (Bolya et al., 2023), PruMerge (Shang et al., 2024), and G-Prune (Jiang et al., 2025) typically rely on visual-only heuristics—e.g., token similarity, attention within the image encoder, or clustering—without considering the accompanying text prompt. As a result, these approaches may discard visually redundant tokens that are actually semantically important in the current context, leading to suboptimal performance on language-grounded tasks.

Unlike post-input pruning methods that rely on cross-modal attention but do not reduce LLM input size, and pre-input methods that prune early but ignore text relevance, DCP combines both strengths—performing early token reduction while leveraging textual and visual signals. This leads to more relevant token selection and a better trade-off between accuracy and efficiency.

Theoretical vs. Actual Speedup. Although pruning methods aim to reduce inference latency by discarding tokens, there exists a clear gap between theoretical token reduction and actual speedup in practice. This discrepancy is often due to substantial computational overhead introduced by post-processing (e.g., masked attention, index tracking) or inefficient integration with transformer architectures. As shown in Table 1, these methods may reduce the number of tokens but introduce non-trivial computation, leading to no actual speedup or even performance degradation, such as TOME and PruMerge. In contrast, DCP achieves a TPOT speedup of 1.33×, the highest among all compared methods at 25% ratio, with a total latency of 20.89 ms/tok. This highlights DCP's capability to achieve true computational reduction rather than superficial token sparsity, thanks to its pre-pruning strategy with minimal additional overhead. This advantage generalizes across architectures. As shown in Table 2, DCP achieves consistent real-world speedups on all tested LVLMs. In summary, DCP not only delivers superior accuracy under aggressive compression but also achieves practical inference acceleration. It addresses the limitations of prior methods by minimizing redundant computations and aligning token pruning with the actual execution flow, thereby bridging the gap between theoretical and realized gains.

C Implementation Details

C.1 Hardware Setup

All experiments are conducted on a single NVIDIA A100 GPU with 40GB memory. Our approach is lightweight and inference-efficient: it also runs smoothly on consumer-grade GPUs such as NVIDIA RTX 4090 (24GB). No distributed inference or model parallelism is required.

C.2 Text Preprocessing with spaCy

To extract problem-relevant keywords from complex input prompts, we perform multi-stage text preprocessing prior to CLIP encoding. This helps generate a compact and semantically rich representation that fits within CLIP's 77-token constraint.

The steps are as follows:

1. **Prompt Filtering:** We apply regular expressions to remove non-semantic content such as system prompts (e.g., `"Use the data..."`), response instructions (e.g., `"Answer the question..."`), and multiple-choice answer options (e.g., `"A. ..."` to `"D. ..."`).

2. **Syntactic Parsing:** We use the spaCy English parser (`en_core_web_sm`) (Explosion, 2023), a lightweight 12MB NLP model, to perform part-of-speech tagging and dependency parsing.

3. **Keyword Extraction:** From the parsed text, we identify noun phrases (`doc.noun_chunks`) and filter out common stopwords using spaCy's built-in stopword list. For each chunk, we retain meaningful tokens and reconstruct lowercased key phrases:

   keywords = { `" ".join(`
   `w.lower() for w in chunk`
   `if w not in STOP_WORDS)` }

   Up to $N = 10$ high-confidence keywords are retained to form a concise query aligned with the visual content.

4. **Output Formatting:** The extracted phrases are concatenated using commas to produce a compact query string:

   `"girl, car, cat..."`

   This compressed representation captures the semantic core of the question while ensuring compatibility with the CLIP text encoder's length limit.