ATP-LLaVA: Adaptive Token Pruning for Large Vision Language Models

Xubing Ye[1], Yukang Gan[2†], Yixiao Ge[2*], Xiao-Ping Zhang[1], Yansong Tang[1*]
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University, [2]ARC Lab, Tencent PCG
{yxb23@mails.,tang.yansong@sz.,xiaoping.zhang@sz.}tsinghua.edu.cn
{brucegan,yixiaoge}@tencent.com

Abstract

Large Vision Language Models (LVLMs) have achieved significant success across multi-modal tasks. However, the computational cost of processing long visual tokens can be prohibitively expensive on resource-limited devices. Previous methods have identified redundancy in visual tokens within the Large Language Model (LLM) decoder layers and have mitigated this by pruning tokens using a pre-defined or fixed ratio, thereby reducing computational overhead. Nonetheless, we observe that the impact of pruning ratio varies across different LLM layers and instances (image-prompt pairs). Therefore, it is essential to develop a layer-wise and instance-wise vision token pruning strategy to balance computational cost and model performance effectively. We propose ATP-LLaVA, a novel approach that adaptively determines instance-specific token pruning ratios for each LLM layer. Specifically, we introduce an Adaptive Token Pruning (ATP) module, which computes the importance score and pruning threshold based on input instance adaptively. The ATP module can be seamlessly integrated between any two LLM layers with negligible computational overhead. Additionally, we develop a Spatial Augmented Pruning (SAP) strategy that prunes visual tokens with both token redundancy and spatial modeling perspectives. Our approach reduces the average token count by 75% while maintaining performance, with only a minimal 1.9% degradation across seven widely used benchmarks. The project page can be accessed via the following link.

1. Introduction

The emergence of Large Vision Language Models (LVLMs) [2, 4, 12, 24, 28–30, 46, 55, 57] has significantly advanced visual understanding. These approaches leverage visual encoders to extract visual features, which are then processed jointly with text in Large Language Models (LLMs). Despite their impressive multi-modal understanding capabilities, the deployment of these models is often hindered by the substantial memory and computational costs when processing large number of visual tokens, especially in resource-constrained environments.

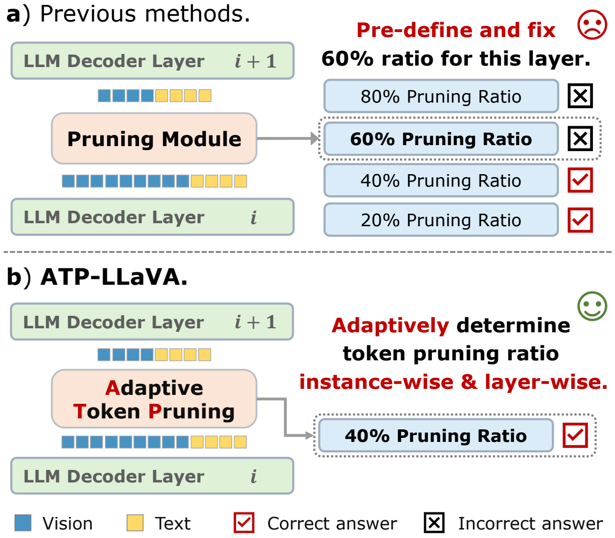

Figure 1. (a) Previous methods employ a fixed, pre-defined token pruning ratio. (b) Illustration of ATP-LLaVA, which dynamically selects the adaptive pruning ratio for each layer of the LLM decoder based on the instance-specific characteristics.

To address this issue, previous methods [8, 9, 43, 52] have focused on compressing visual tokens by pruning redundant ones, as visual information tends to be sparser compared to natural language information. While these methods can mitigate the loss of model's understanding capabilities caused by token pruning, they share a common limitation: requiring a fixed pruning ratio (*i.e.*, a fixed number of retained tokens or a non-learnable threshold) to be **pre-defined** for the model, as shown in Fig. 1 (a). Recently, some methods [7, 15] have explored training a single model that can handle varying visual token counts. However, it

Work was done when the author interned at ARC Lab, Tencent PCG. †Project lead. *Corresponding author.

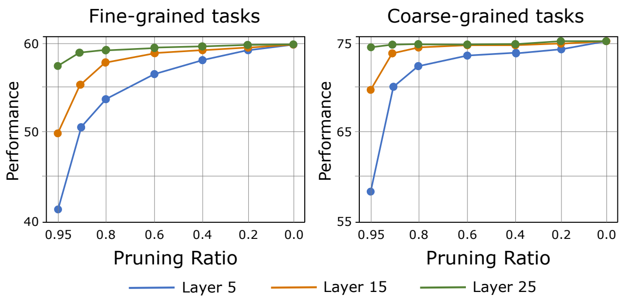

Figure 2. Comparison of vision token pruning at different LLM decoder layers and pruning ratios across fine-grained and coarse-grained tasks of the SEED-Image [23]. Fine-grained tasks include instance counting, spatial relation, etc. Coarse-grained tasks include scene understanding, etc.

remains necessary to manually specify a pruning ratio for each individual sample. Due to the varying complexity of different tasks and the differing levels of content within images, a pre-defined pruning ratio may result in either information loss or excessive information retention, consequently impacting the model's efficiency and effectiveness.

To further validate this, we conducted a preliminary experiment to examine the effects of pruning visual tokens at various layers of LVLMs and across different pruning ratios. As shown in Fig. 2, the impact of pruning ratio on performance is ***layer-wise***, with shallower layers being more sensitive to pruning and deeper layers exhibiting greater robustness. Moreover, we compared the performance of pruning across tasks of varying complexity and observed that the pruning ratio's impact is also ***instance-wise***. Fine-grained tasks, such as instance counting and spatial relation, demand detailed visual information, necessitating the retention of more tokens at each layer to prevent performance degradation. In contrast, coarse-grained tasks like scene understanding do not exhibit significant performance loss even at high pruning ratios. These observations indicate that the optimal pruning ratio, which achieves the best balance between performance and efficiency, varies for each instance and each layer. A pre-defined pruning ratio can lead to suboptimal model performance and efficiency.

In this paper, we propose a framework called **A**daptive **T**oken **P**runing for large vision language models (**ATP-LLaVA**), as illustrated in Fig. 1 (b), which adaptively determines the pruning ratio. Specifically, we design an Adaptive Token Pruning (ATP) module that can be seamlessly integrated between any two LLM layers with negligible computation cost. The ATP module first employs a Spatial Augmented Pruning (SAP) policy to dynamically prune tokens for each instance. This policy adopts scores on two perspectives to evaluate the importance of each token. The first perspective, the redundant pruning score, computes token importance by leveraging both intra-modal and cross-modal correlations. The second perspective, the spatial pruning score, evaluates the spatial information within the token sets that are spatially uniform sampled at various granularity. Building upon these two scores, two learnable thresholds are introduced at each ATP module to dynamically select important tokens for each LLM layer and each instance. Tokens that are not selected will be discarded in subsequent layers. Lastly, to further enhance the training process, we propose an ATP-Loss function to strike a balance between token pruning efficiency and understanding capability. We summarize our contributions as follows:

- We reveal the importance of adaptively determining pruning ratios at the instance and LLM layer levels for effective visual token pruning, and propose ATP-LLaVA, a framework that dynamically reduces computational cost for large vision language models.
- To enable the model to learn the adaptive token pruning strategy, we introduce an Adaptive Token Pruning (ATP) module, along with an ATP-Loss function, thereby balancing pruning efficiency and model capabilities.
- To mitigate the loss in visual understanding performance caused by token pruning, we introduce a Spatial Augmented Pruning (SAP) approach , which preserves spatial modeling during the pruning process.
- ATP-LLaVA achieves a 75% average pruning ratio while maintaining 98.1% performance across seven widely used vision understanding benchmarks.

2. Related work

Large Vision-Language Models. The impressive progress of large language models (LLMs) [6, 10, 17, 19, 39, 42, 47, 48] has sparked interest in developing large vision language models (LVLMs) that can bridge the gap between visual and linguistic understanding. LVLMs have shown impressive capabilities in cross-modal understanding and visual language tasks through modality alignment and instruction tuning. Previous works [1–4, 11, 12, 24, 28, 29, 49, 50, 55, 57] have validated the efficacy of this training paradigm in visual understanding. The success of LVLMs has also been extended to the video domain [16, 21, 25, 27, 30–33, 36–38, 46, 53, 54]. Furthermore, research has shown that LVLMs can capture rich visual information for understanding and generation when provided with high-resolution images [4, 28]. However, the growing number of vision tokens occupy a substantial portion of the LLM's valuable context window and leading huge bottleneck for computational infrastructure. To address this, further innovation in token compression and pruning techniques is essential.

Vision Token Compression and Pruning. To compress vision information with less tokens, previous methods [11, 12, 24, 55, 57] have largely employed Q-Former [24], which maps images to fixed-length learnable queries. [27, 51] have applied simple pooling strategy to downsample visual features. [52] try to distill the LLMs' understanding paradigm of vision tokens into single VoCo token to reduce inference cost. [43] identifies redundant visual tokens through clustering analysis and prunes them. [9] reveals that vision tokens within LLM Transformer layers are also redundant, and pruning them internally incurs less penalty than pruning before inputting to the LLM. Although these methods can mitigate the loss caused by token compression, they are limited by relying on a pre-defined pruning rate. While [20, 22, 41] explore adaptive token pruning in Vision Transformers, this area remains relatively under-explored in the context of Large Vision Language Models. Recent methods [7, 15] offer flexible choices for the number of visual tokens, but they struggle to adaptively determine the optimal pruning ratio. ATP-LLaVA can adaptively determine the pruning ratio within any layer of the LLM, based on the specific instance characteristics.

3. Method

3.1. Overview

The ATP-LLaVA architecture, as shown in Fig. 3, centers around the **A**daptive **T**oken **P**runing (ATP) module that can be easily inserted between any two decoder layers in the LLM decoder backbone. The ATP module first computes importance scores for visual tokens based on the self-attention map from the previous layer, and then utilizes two lightweight prediction heads to learn pruning thresholds with minimal additional parameters. These two prediction heads generate learnable thresholds for redundant and spatial pruning of visual tokens, respectively, enabling layer-wise and instance-wise adaptive pruning.

In the following sections, we provide a brief overview of the LLM decoder in Sec. 3.2. We then present our Adaptive Token Pruning module in Sec. 3.3. Specifically, the ATP module consists of three key components: Spatial Augmented Pruning approach (Sec. 3.3.1), Learnable Thresholds (Sec. 3.3.2), and ATP-Loss (Sec. 3.3.3).

3.2. Preliminaries

The LLM decoder consists of multiple decoding layers (*i.e.*, 32 layers in LLaMA [47]). Each decoding layer typically uses self-attention with a causal mask. The input to each Transformer layer is a concatenation of vision and text tokens $H \in \mathbb{R}^{L\times D}$, where L is the total length of the tokens and D is the hidden dimension. For the causal attention mechanism (considering single-head attention as an example), the self-attention logits can be computed by

$$A_{\text{logits}} = QK^T/\sqrt{D}, \tag{1}$$

where $Q \in \mathbb{R}^{L\times D}$ and $K \in \mathbb{R}^{L\times D}$ are the query and key matrices, respectively. It yields the resulting logits matrix $A_{\text{logits}} \in \mathbb{R}^{L\times L}$. To accommodate the causal attention mechanism in the decoder, a lower triangular masking matrix $M \in \mathbb{R}^{L\times L}$ is element-wise added to the logits when computing final attention weights A, resulting in

$$A = \text{Softmax}(A_{\text{logits}} + M), \tag{2}$$

Following the self-attention, each layer is connected to a Feed Forward Network (FFN) layer, enabling the model to capture contextual relationships within each modality.

3.3. Adaptive Token Pruning

Given the output of the i-th Transformer decoder layer, denoted as H_i. It consists of two components: the hidden states of visual tokens, represented as $V_i \in \mathbb{R}^{L_v\times D}$, and the hidden states of text tokens, represented as $T_i \in \mathbb{R}^{L_t\times D}$ (we ignore the system prompt input for simplicity). The ATP module removes redundant tokens from V, resulting in unpruned visual tokens, denoted as $V_i^p \in \mathbb{R}^{L_v^p\times D}$, where $L_v^p < L_v$. The maintained visual tokens are then concatenated with the original text tokens T and fed into the following LLM layer. Visual tokens pruned at the current layer are irretrievable in subsequent layers.

3.3.1. Spatial Augmented Pruning

Token pruning in large vision language models disrupts spatial modeling, which is crucial for vision understanding. To mitigate this, we introduce the **S**patial **A**ugmented **P**runing (SAP) approach. SAP consists of two stages: (1) redundant pruning, which assigns importance scores to visual tokens and prunes them based on a threshold, and (2) spatial pruning, which uniformly samples tokens in the spatial dimension. The pruning masks from both stages are then combined to form the final pruning strategy.

Redundant Pruning Score. To perform redundant token pruning, we define the importance score for each token. Intuitively, we consider both the vision token's self-modality importance and its importance to the text modality. For the current layer's visual tokens $V_i = \{v_1, \dots, v_{L_v}\}$, a visual token v_n is deemed important if it receives significant attention from other visual tokens in the same layer. Specifically, as shown in the *self-attn map* of Fig. 3 (left), this importance score can be obtained from the attention logits A^i_{logits} of the i-th Transformer layer. The self-modality importance score S_n^{self} for token v_n is defined as:

$$S_n^{self} = \frac{1}{L_v}\sum_{m=1}^{L_v} A^i_{\text{logits}}(v_n, v_m) \in \mathbb{R}^{L_v}, \tag{3}$$

Similarly, visual tokens that receive greater attention from all text tokens should be assigned a higher importance score in the text modality. As shown in the *text-vision map* of Fig. 3 (left), this importance score can be obtained from the attention map A^i of the i-th Transformer layer. Given

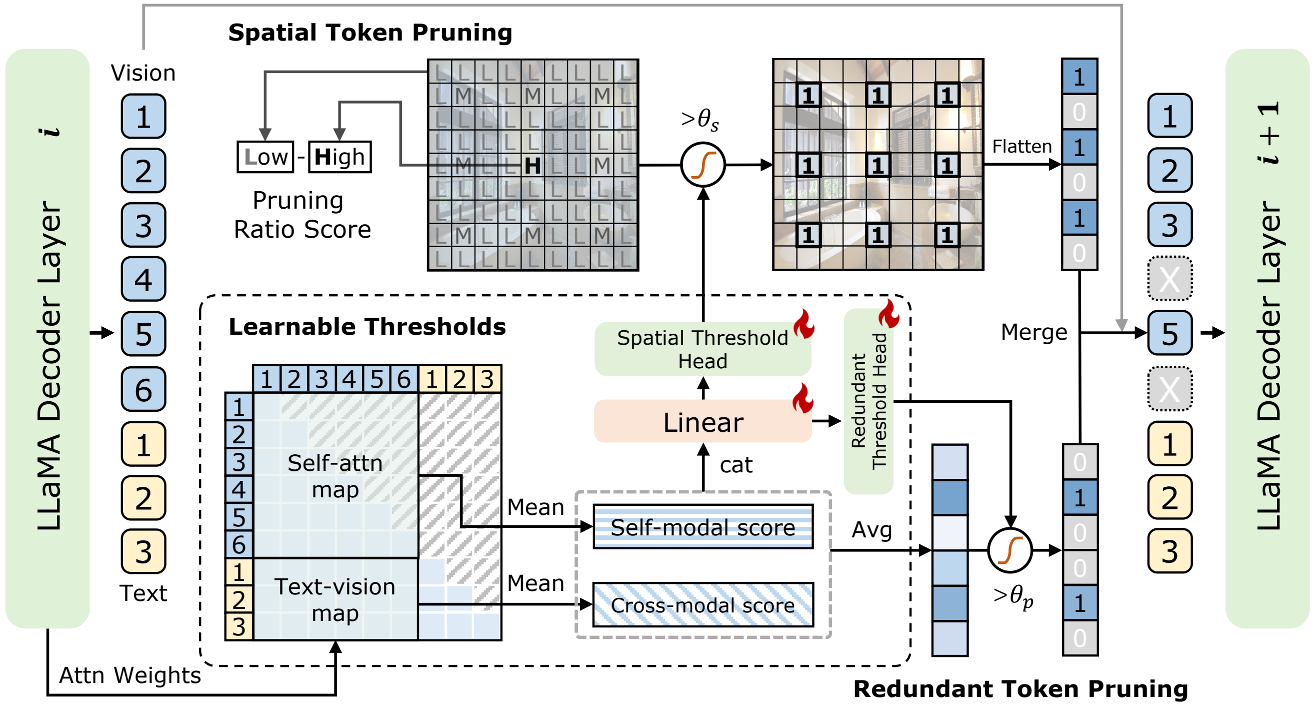

Figure 3. Illustration of the **A**daptive **T**oken **P**runing (ATP) module. The ATP module can be flexibly inserted between any two LLaMA decoder layers. It adaptively predicts pruning thresholds for current layer and instance. Redundant or text-irrelevant visual tokens are pruned at this stage, and they will be ignored by other tokens in subsequent LLaMA decoder layers.

the current layer's text tokens $T_i = \{t_1, \ldots, t_{L_t}\}$, we define the cross-modality importance score S_n^{cross} for token v_n as:

$$S_n^{cross} = \frac{1}{L_t} \sum_{m=1}^{L_t} A^i(v_n, t_m) \in \mathbb{R}^{L_v}, \tag{4}$$

where the probability $A^i(v_n, t_m)$ represents the normalized value that token t_m focuses on v_n. We calculate the final score $S_n^{redundant}$ by taking the average of S_n^{self} and S_n^{cross}.

Having obtained the token scores, a learnable pruning threshold is introduced to selectively retain tokens. A detailed discussion on this is presented in Sec. 3.3.2.

Spatial Pruning Score. Uniform spatial sampling of visual tokens was proposed by LLaVA-PruMerge [43] to preserve spatial visual information. However, we observe that the impact of sampling ratio varies across different input instances and LLM layers. We found that excessive sampling of spatial tokens significantly increases token count, with the sampling ratio's impact varying across different input instances and LLM layers. To address this, we design a dynamic uniform spatial sampling strategy.

As illustrated in Fig. 3 (top), we define a spatial pruning score $S^{spatial} \in \mathbb{R}^{L_v}$ within the range of (0, 1]. The sampling rate R^s is defined as the ratio of the uniform sampled tokens to the visual tokens L_v. Under this sampling rate, we acquire a set of sampled visual tokens V_s. The score of the sampled visual tokens is defined as:

$$S_n^{spatial} = 1 - R^s \cdot \lambda_{sample}, \quad v_n \in V_s, \tag{5}$$

where λ_{sample} is a scaling coefficient. Tokens sampled at higher rates are given higher scores. We introduce a learnable threshold for spatial pruning, allowing us to dynamically adjust the spatial pruning rate instance-wise and layer-wise, despite the fixed spatial pruning score.

Positional Embeddings for Token Pruning. Merely applying uniform spatial sampling is insufficient to enhance spatial modeling, as the sampled tokens will be flattened into a sequence and fed into the LLM. To address this, we employ 2D rotary embedding [45, 49], which enhances the spatial information. Furthermore, previous methods typically reorganize the retained tokens' position embeddings into a contiguous sequence (*e.g.*, $0, 1, 2, \ldots$), which may disrupt spatial features under 2D positional encoding. In contrast, we preserve the original position embeddings of the retained tokens after pruning.

3.3.2. Adaptive Pruning with Learnable Thresholds

To adapt to the dynamic changes in pruning thresholds across different layers and instances, we introduce a MLP with dual prediction heads to learn instance-specific thresholds. The determination of pruning thresholds should be highly correlated with the instance itself, and thus we leverage the self-modality and cross-modality scores computed in Sec. 3.3.1 as inputs to the prediction module.

Specifically, given the visual token scores S_n^{self} and S_n^{cross}, we compute the redundant pruning threshold θ_r and spatial pruning threshold θ_s as follows:

$$z = \text{Linear}(cat(S_{V_i}^{self}, S_{V_i}^{cross})), \tag{6}$$

$$\theta_r = \sigma(\text{Linear}_r(z)) \in \mathbb{R}^1, \tag{7}$$

$$\theta_s = \sigma(\text{Linear}_s(z)) \in \mathbb{R}^1, \tag{8}$$

However, the hard masks generated by threshold prevent gradient backpropagation, rendering the threshold prediction module untrainable. Inspired by [5], we convert the hard masks to differentiable soft masks:

$$Mask_i^r = \sigma((S_{V_i}^{redundant} - \theta_r) \cdot T) \in \mathbb{R}^{L_v}, \tag{9}$$

$$Mask_i^s = \sigma((S_{V_i}^{spatial} - \theta_s) \cdot T) \in \mathbb{R}^{L_v}, \tag{10}$$

$$Mask_i = max(Mask_i^r, Mask_i^s) \in \mathbb{R}^{L_v}, \tag{11}$$

where T is a temperature coefficient that, when sufficiently large, renders the sigmoid function a differentiable mask.

Besides, directly discarding unselected visual tokens using the mask is non-differentiable for the learnable threshold module, hindering end-to-end learning of token pruning strategies. And pruning varying numbers of tokens per instance within a batch complicates parallel training. To address these challenges, we followed [41] and utilize masks during the attention *Softmax* operation, effectively eliminating the influence of pruned tokens on others and ensuring a differentiable process. Specifically, we multiply the exponential results by the pruning mask after computing the exponentials and before summing them up. We only apply the softmax mask during the training phase. In the inference phase, the pruned tokens are directly discarded and do not participate in any subsequent layer computations.

3.3.3. Budget-Constrained Training

Our goal is to encourage the model to learn an optimal pruning strategy that balances computational cost and performance loss. To achieve this, we design an ATP loss function that trains the model under limited pruning constraints. Specifically, we introduce a penalty term that discourages the model from retaining excessive tokens. The penalty term should be designed to increase with both the number of remaining tokens and the layer depth, accounting for the diminishing returns of deep pruning on computational cost.

Given the index set $I = \{i_0, \ldots, i_n\}$ of LLM layers where ATP module is introduced, the differentiable pruning masks are denoted as $Masks = \{Mask_{i_k} \mid i_k \in I\}$. To facilitate backpropagation, we compute the sum of $Mask_{i_k}$ to obtain the number of remaining tokens after pruning at layer i_k. The penalty term can be expressed as:

$$\mathcal{L}_{\text{ATP}} = \sum_{i_k}^{I} \frac{Sum(Mask_{i_k})}{576} * i_k, \tag{12}$$

To constrain the average token count with a target value, we compute the average token count $\overline{N}$ across all layers within a batch and minimize the difference between $\overline{N}$ and the target token value N_{target} using:

$$\mathcal{L}_{\text{target}} = || \, \overline{N} - N_{\text{target}} \, ||, \tag{13}$$

We adopt a supervised fine-tuning (SFT) setting for visual language models as our training paradigm. The full training objective for ATP module is:

$$\mathcal{L} = \mathcal{L}_{\text{ntp}} + \mathcal{L}_{\text{ATP}} * \lambda_{\text{ATP}} + \mathcal{L}_{\text{target}} * \lambda_{\text{target}}, \tag{14}$$

where λ_{ATP} and λ_{target} are scaling coefficients that control the impact of computational budget constraints on training.

3.4. Efficiency Analysis

At inference time, tokens with values below the threshold are directly discarded, which reduces the computational overhead of the model during inference. We only consider the computation of multi-head attention and feed-forward network module of LLM in the FLOPs estimation. The theoretical FLOPs of each LLM layer with unpruned tokens can be calculated as:

$$\text{FLOPs}^L = 4Ld^2 + 2L^2d + 2Ldm, \tag{15}$$

where L represents the total number of input tokens to the first layer of the LLM without pruning, d denotes the hidden state size, and m is the intermediate size of FFN. Example as the ATP module is inserted after the i_k-th layer, $i_k \in I$. Let $L_{i_k}^p$ represent the preserved token length after pruning, i_N denote the final layer index in the LLM decoder. We compute the FLOPs reduction across the entire model as:

$$\sum_{i_k}^{\{I, i_N\}} (i_{k+1} - i_k) * (\text{FLOPs}^L - \text{FLOPs}^{L_{i_k}^p}). \tag{16}$$

Additionally, our ATP-LLaVA introduces a slight additional computational overhead for the ATP module, which is further elaborated in the supplementary material.

4. Experiments

4.1. Implementation Details

Regarding the training strategy and data, ATP-LLaVA follows a standard vision instruction tuning stage. LLaVA-1.5 [28] is chosen as the base model for ATP-LLaVA. Specifically, we employ the pre-trained CLIP-ViT-L [40] as our visual encoder followed by a linear projector to align text and vision modality. We directly utilize pretrained projectors from LLaVA-1.5, which were trained on the filtered CC3M dataset [44] with the fixed language model and vision encoder. For pre-trained large language models, we utilize Vicuna-7B-1.5 [10]. For training data, we use 665k

Method	Token	**GQA**	**MMB**	**MME**	**POPE**	**SEED**	**SQA^I**	**VQA^{v2}**	**Avg.**
Upper Bound Model									
LLaVA-1.5 [28]	576	62.0 100%	64.3 100%	1510.7 100%	85.8 100%	58.6 100%	71.6 100%	78.5 100%	- 100%
Methods with Pre-defined Pruning Ratio									
PruMerge+ [43]	144	- -	64.9 100.9%	1462.4 96.8%	84.0 97.9%	- -	68.3 95.4%	**76.8** 97.8%	- 97.8%
FastV [9]	192	52.7 83.8%	61.2 95.2%	1312.4 86.9%	64.8 75.5%	50.8 86.7%	65.4 91.3%	67.1 85.5%	- 86.4%
	128	49.6 80.0%	56.1 87.2%	1187.9 78.6%	59.6 69.5%	48.1 82.1%	59.7 83.4%	61.8 78.7%	- 79.9%
SparseVLM [56]	192	57.6 92.9%	62.5 97.2%	1382.8 91.5%	83.6 97.4%	53.0 90.4%	67.2 93.9%	75.6 96.3%	- 94.2%
	128	56.0 90.3%	60.0 93.3%	1296.7 85.8%	80.5 96.3%	50.2 85.7%	65.5 91.5%	73.8 94.0%	- 91.0%
Methods with Adaptive Pruning Ratio									
ATP-LLaVA	144*	**59.5** 96.0%	**66.0** 102.6%	**1473.9** 97.6%	**84.2** 98.1%	**57.3** 97.8%	**69.1** 96.5%	76.4 97.3%	- **98.1%**
	88*	56.8 91.6%	64.7 100.6%	1401.5 92.8%	82.6 96.3%	55.7 95.1%	67.2 93.9%	73.3 91.8%	- **94.6%**

Table 1. Comparison with previous approaches on vision token pruning within LLM decoder layers using common visual understanding benchmarks. (Token) indicates the average token count for all layers in language model. Token count with (*) means the retained token count is not pre-defined and adaptively determined by learnable ATP module. We calculate the average token count during inference across all benchmarks. The percentage represents the compression retention rates to Upper Bound model.

visual instruction following data [28] to tune our model. We train ATP-LLaVA using lr of 2e-5 for LLM and 1e-4 for the ATP module for 1 epoch. All other hyperparameters and settings are identical to those used in LLaVA-1.5. As for the scaling coefficients, λ_{sample} is set as 3, λ_{target} is set as 0.2, and λ_{ATP} is set within the range of 0.01 to 0.1. Varying the value of λ_{ATP} during model training has an impact on the final average FLOPs of the resulting model.

4.2. Datasets

We conduct experiments on several common visual understanding benchmarks for vision token pruning in this work. Specifically, we report results on GQA [18], MMB (MMBench) [34], MME [13], POPE [26], SEED-Bench [23], SQA^I (Image-based setting in ScienceQA) [35] and VQA^{v2} (VQA V2) [14]. We can assess the impact of visual information loss during the pruning process by comparing the model's performance on these visual understanding benchmarks before and after pruning. We follow the evaluation details outlined in [29] to assess the model's performance on these visual understanding benchmarks.

4.3. Results

Main Results. We report results of our ATP-LLaVA on various common visual understanding benchmarks to presents the vision token pruning performance. Besides, to rigorously quantify the performance loss of ATP-LLaVA during token pruning, we also report the compression retention rates to the Upper Bound model (*i.e.*, LLaVA-1.5 [28] in this paper). We compare our method with previous token pruning methods. For fair comparisons, we constrain ATP-LLaVA with a limited budget and average token count. In particular, we adjust the scaling coefficients and the target token number in Eq. (13) and Eq. (14). The token count is an average value across all decoder layers (*i.e.*, 32 layers in LLaMA [47]) on a uniform sampled set across all reported benchmarks during inference. As shown in Tab. 1, it can be observed that our method preserves the original vision understanding capability to a large extent. Notably, we achieved an average compression retention rate of 98.1% and 94.6% across seven widely used benchmarks, when pruning from 576 to 144 and 88 tokens, respectively. Especially on MMBench and SEED-Bench, our method

Pruning Strategy	Avg. Tokens	Pruning Ratio	Pruning Layer Indexes	Avg. Retained Tokens	**MMB**	**GQA**	**VQA^{v2}**	**SEED**
Upper Bound	576	-	-	-	64.3	62.0	78.5	58.6
Pre-defined Ratio	144	$3/4$	[1]	[130]	59.6	56.2	71.6	53.1
	144	$2/3$	[4, 14, 24]	[162, 54, 18]	62.7	58.0	74.6	55.7
	144	$1/2$	[4, 14, 24]	[136, 68, 34]	63.2	58.2	73.9	55.3
ATP-LLaVA	144*	-	[4, 14, 24]	[126*, 88*, 20*]	**66.0**	**59.5**	**76.4**	**57.3**
	88*		[1, 13, 25]	[98*, 79*, 16*]	64.7	56.8	73.3	55.7

Table 2. Comparison with pre-defined pruning strategy under same traing setting using common benchmarks. (Pruning Layer Indexes) specify the LLM layer indexes at which vision tokens are pruned, starting from 0 and occurring prior to input.

LM	Token	**MMB**	**GQA**	**VQA^{v2}**	**SEED**
Frozen	144*	63.1	57.9	75.2	54.9
Trainable	144*	66.0	59.5	76.4	57.3

Table 3. Ablation study on training strategy for ATP-LLaVA using common visual understanding benchmarks. (*Frozen*) means only the ATP module is trainable while freezing the language model.

RP	SP	PP	**MMB**	**GQA**	**VQA^{v2}**	**SEED**
✓	✓	✓	**66.0**	**59.5**	**76.4**	**57.3**
✓			65.3	58.3	75.1	56.5
	✓		64.3	57.1	74.2	55.1
	✓	✓	65.2	58.6	74.9	56.3

Table 4. Ablation study on SAP approach for ATP-LLaVA using common visual understanding benchmarks. (RP) indicates using redundant pruning strategy, (SP) indicates using spatial pruning strategy, and (PP) indicates using pruning positional embedding.

S^{self}	S^{cross}	**MMB**	**GQA**	**VQA^{v2}**	**SEED**
✓	✓	66.0	**59.5**	**76.4**	**57.3**
✓		**66.1**	58.7	75.8	56.9
	✓	65.4	59.2	76.1	56.3

Table 5. Ablation study on redundant pruning score for ATP-LLaVA using common visual understanding benchmarks. (S^{self}) indicates the self-modality importance score in Eq. (3), and (S^{cross}) indicates the cross-modality importance score in Eq. (4).

achieves and even surpasses the performance of the Upper Bound model. This demonstrates that ATP-LLaVA can substantially enhance inference efficiency with only a negligible performance degradation.

Pruning Strategy. To evaluate the adaptability of Adaptive Token Pruning (ATP) module, we conduct experiments using pre-defined pruning ratio strategy. Specifically, we replace the ATP module with a pre-defined pruning ratio strategy that prunes visual tokens at a fixed ratio across all layers, while maintaining the other modules. The pruned tokens will be directly discarded, eliminating the need of differentiable design introduced in Sec. 3.3.2. The pre-defined ratio models are trained under the same setting and dataset as ATP-LLaVA. We report the performance of the model on common visual understanding benchmarks at various pruning rates ($3/4$, $2/3$, and $1/2$) and compare it to the ATP strategy with the same average number of pruned tokens. As shown in Tab. 2, ATP-LLaVA consistently outperforms models trained with fixed pruning rates across all four benchmarks. Especially, ATP-LLaVA attains noticeable improvements compared with the best fixed pruning ratio model, with large margins of 2.8, 1.6 and 1.6 absolute points on MMBench, VQA^{v2} and SEED. These results illustrate that tailoring adaptive pruning strategies to specific instances and layers can help mitigate performance degradation when pruning tokens within a constrained budget.

Training Strategy. We further investigate the influence of training strategies on the performance of ATP-LLaVA. In our main experiments, we adopt the training setting of visual instruction tuning, where both the ATP module and language model are trainable. To explore more efficient training paradigms, we conduct additional experiments by freezing the language model and solely training the lightweight ATP module. We omit pruning positional embedding from these comparative experiments, as it is incompatible with the frozen language model training paradigm. As evident in Tab. 3, ATP-LLaVA is capable of learning adaptive token pruning strategies and mitigating the degradation in capability caused by token pruning under the training paradigm with a frozen language model. Concurrently, fine-tuning the language model enables the LLM to better adapt to visual understanding with a limited number of tokens. More training details can be found in the supplementary materials.

Token Pruning Technique. We conduct several ablations to evaluate the effectiveness of the key components in ATP-LLaVA. Firstly, we remove the redundant pruning strategy (RP), spatial pruning strategy (SP) and pruning positional embedding (PP), respectively. The average token count in these experiments is adjusted to be identical, *i.e.*, 144.

Method	Avg. Token	Accuracy	Storage Memory (MB)	Δ	CUDA Time (ms) ↓	Δ	FLOPs (T) ↓	Δ
Upper Bound	576	100%	302.4	-	432.7	-	9.6	-
FastV [9]	144	81.7%	75.6	75%	259.1	40.1%	2.0	79.2%
ATP-LLaVA	144	**98.1%**	75.6	75%	266.4	38.4%	2.1	78.1%
	88	**94.6%**	46.2	84.7%	226.8	47.6%	1.5	84.4%

Table 6. Efficiency analysis of ATP-LLaVA including cache storage memory, CUDA times and the FLOPs. Δ denotes the reduction ratio.

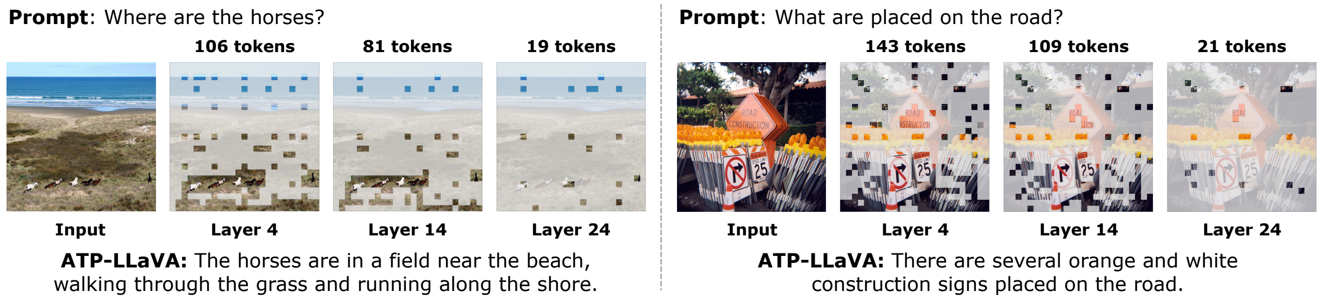
Prompt: Where are the horses?
106 tokens
81 tokens
19 tokens
Input
Layer 4
Layer 14
Layer 24
ATP-LLaVA: The horses are in a field near the beach, walking through the grass and running along the shore.
Prompt: What are placed on the road?
143 tokens
109 tokens
21 tokens
Input
Layer 4
Layer 14
Layer 24
ATP-LLaVA: There are several orange and white construction signs placed on the road.

Figure 4. Visualized vision token pruning results of ATP-LLaVA. White tokens represents the pruned tokens. The uniform sampled tokens are pruned by spatial pruning threshold, while the sparse tokens are pruned by redundant pruning threshold. Zoom in to have a better view.

Tab. 3 shows that removing redundant pruning and spatial pruning leads to an average drop of 1.05 and 1.0 absolute points across four common benchmarks, respectively. Furthermore, in comparison to employing spatial sampling pruning in isolation, the synergistic application of spatial pruning and pruning positional encoding yields more pronounced performance improvements. Specifically, the integration of pruning positional encoding results in an improvement of 0.9 absolute points when spatial pruning strategy is applied. These results demonstrate the benefit of exploiting the pruning technique from both token redundancy and spatial modeling perspective.

Pruning Importance Score. In this ablation study, we separately utilize either the self-modality importance score or the cross-modality importance score for redundant token pruning. As shown in Tab. 5, relying solely on self-score and cross-score results in average absolute performance drops of 0.43 and 0.55 points, respectively, across four common benchmarks. These results illustrate that both intra-modal and cross-modal importance scores are equally significant in quantifying the redundancy of visual tokens.

Efficiency Analysis. We evaluate the inference efficiency of ATP-LLaVA through a comparative analysis with the Upper Bound model and FastV [9]. Our results, presented in Tab. 6, demonstrate that ATP-LLaVA achieves significant reductions in KV cache storage memory (75%), CUDA time (38.4%), and FLOPs (78.1%) compared to the Upper Bound model, while maintaining 98.1% performance. In comparison to FastV, the introduction of threshold prediction linear layers in ATP module incurs a minor computational overhead (1.1% in FLOPs, 1.7% in CUDA time), but yields a substantial performance gain (16.4% accuracy improvement). We conclude that the performance benefits outweigh the negligible increase in computational cost. More discussions are provided in the supplementary materials.

Visualization Results. In Fig. 4, we visualize the instance-specific pruning results using the optimal ATP-LLaVA model from Tab. 1, which has an average token count of 144. The uniformly distributed tokens in the figure result from spatial token pruning, whereas the sparse tokens arise from redundancy token pruning. The example on the left illustrates an image of lower complexity, where the model prunes extensively in the shallow and middle layers (4-th and 14-th layers), while selectively retaining prompt-relevant tokens (e.g., “horses”). In contrast, the example on the right presents an image of higher complexity, where the model retains a larger number of tokens in the shallow and middle layers, particularly those related to the object’s location (*e.g.*, “on the road”). In the 24-th layer, token preservation is minimal, as visual information has been largely consolidated into text tokens through preceding layers. These results demonstrate that ATP-LLaVA adaptively prunes tokens across layers based on instance-specific visual complexity and text-vision relevance. We provide additional visualizations in the supplementary materials.

5. Conclusion

In this paper, we propose ATP-LLaVA, which adaptively prunes vision tokens for large vision language models on layer and instance level. By introducing an Adaptive Token

Pruning module, our method can calculate the importance score for vision tokens and determine the pruning threshold dynamically. Based on Spatial Augmented Pruning strategy, our method can prune redundant vision tokens and maintain spatial modeling, while minimizing information loss. In summary, our approach offers a promising solution for flexible vision token pruning of LVLMs, making them more scalable in resource constrained-environments.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, and et al. Flamingo: a visual language model for few-shot learning, 2022. 2

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

[3] Sule Bai, Yong Liu, Yifei Han, Haoji Zhang, and Yansong Tang. Self-calibrated clip for training-free open-vocabulary segmentation. *arXiv preprint arXiv:2411.15869*, 2024.

[4] Rohan Bavishi, Erich Elsen, Curtis Hawthorne, Maxwell Nye, Augustus Odena, Arushi Somani, and Sağnak Taşırlar. Introducing our multimodal models, 2023. 1, 2

[5] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. 5

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and et al. Language models are few-shot learners, 2020. 2

[7] Mu Cai, Jianwei Yang, Jianfeng Gao, and Yong Jae Lee. Matryoshka multimodal models. *arXiv preprint arXiv:2405.17430*, 2024. 1, 3

[8] Junbum Cha, Wooyoung Kang, Jonghwan Mun, and Byungseok Roh. Honeybee: Locality-enhanced projector for multimodal llm, 2024. 1

[9] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models, 2024. 1, 3, 6, 8

[10] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023. 2, 5

[11] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023. 2

[12] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, 2022. 1, 2

[13] 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. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023. 6

[14] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6

[15] Wenbo Hu, Zi-Yi Dou, Liunian Harold Li, Amita Kamath, Nanyun Peng, and Kai-Wei Chang. Mqt-llava: Matryoshka query transformer for large vision-language models, 2024. 1, 3

[16] Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. Vtimellm: Empower llm to grasp video moments, 2023. 2

[17] Wei Huang, Xudong Ma, Haotong Qin, Xingyu Zheng, Chengtao Lv, Hong Chen, Jie Luo, Xiaojuan Qi, Xianglong Liu, and Michele Magno. How good are low-bit quantized llama3 models? an empirical study, 2024. 2

[18] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

[19] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and et al. Mistral 7b, 2023. 2

[20] Cao Jianjian, Ye Peng, Li Shengze, Yu Chong, Tang Yansong, Lu Jiwen, and Chen Tao. Madtp: Multimodal alignment-guided dynamic token pruning for accelerating vision-language transformer. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 3

[21] Peng Jin, Ryuichi Takanobu, Wancai Zhang, Xiaochun Cao, and Li Yuan. Chat-univi: Unified visual representation empowers large language models with image and video understanding, 2024. 2

[22] Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 784–794, 2022. 3

[23] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023. 2, 6

[24] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023. 1, 2

[25] Kunchang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*, 2023. 2

[26] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. 6

[27] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. *arXiv preprint arXiv:2311.17043*, 2023. 2

[28] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023. 1, 2, 5, 6

[29] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 2, 6

[30] Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World model on million-length video and language with ringattention. *arXiv preprint*, 2024. 1, 2

[31] Ruyang Liu, Chen Li, Yixiao Ge, Ying Shan, Thomas H Li, and Ge Li. One for all: Video conversation is feasible without video instruction tuning. *arXiv preprint arXiv:2309.15785*, 2023.

[32] Yong Liu, Ran Yu, Jiahao Wang, Xinyuan Zhao, Yitong Wang, Yansong Tang, and Yujiu Yang. Global spectral filter memory network for video object segmentation. In *European Conference on Computer Vision*, pages 648–665. Springer, 2022.

[33] Yong Liu, Ran Yu, Fei Yin, Xinyuan Zhao, Wei Zhao, Weihao Xia, and Yujiu Yang. Learning quality-aware dynamic memory for video object segmentation. In *European Conference on Computer Vision*, pages 468–486. Springer, 2022. 2

[34] 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. Mmbench: Is your multi-modal model an all-around player? *arXiv:2307.06281*, 2023. 6

[35] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 6

[36] Ruipu Luo, Ziwang Zhao, Min Yang, Junwei Dong, Da Li, Pengcheng Lu, Tao Wang, Linmei Hu, Minghui Qiu, and Zhongyu Wei. Valley: Video assistant with large language model enhanced ability, 2023. 2

[37] Zhuoyan Luo, Yicheng Xiao, Yong Liu, Shuyan Li, Yitong Wang, Yansong Tang, Xiu Li, and Yujiu Yang. Soc: Semantic-assisted object cluster for referring video object segmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

[38] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv:2306.05424*, 2023. 2

[39] OpenAI. Gpt-4 technical report. *arXiv:2303.08774*, 2023. 2

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 5

[41] Yongming Rao, Zuyan Liu, Wenliang Zhao, Jie Zhou, and Jiwen Lu. Dynamic spatial sparsification for efficient vision transformers and convolutional neural networks. *arXiv preprint arXiv:2207.01580*, 2022. 3, 5

[42] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, and et al. Bloom: A 176b-parameter open-access multilingual language model, 2023. 2

[43] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models, 2024. 1, 3, 4, 6

[44] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*, 2018. 5

[45] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. 4

[46] Gemini Team, Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry, Lepikhin, Timothy Lillicrap, Jean baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew Dai, Katie Millican, Ethan Dyer, Mia Glaese, and et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024. 1, 2

[47] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. 2, 3, 6

[48] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, and et al. Llama 2: Open foundation and fine-tuned chat models, 2023. 2

[49] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 2, 4

[50] Zhao Yang, Jiaqi Wang, Xubing Ye, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip HS Torr. Language-aware vision transformer for referring segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 2

[51] Linli Yao, Lei Li, Shuhuai Ren, Lean Wang, Yuanxin Liu, Xu Sun, and Lu Hou. Deco: Decoupling token compression from semantic abstraction in multimodal large language models, 2024. 2

[52] Xubing Ye, Yukang Gan, Xiaoke Huang, Yixiao Ge, Ying Shan, and Yansong Tang. VoCo-LLaMA: Towards Vision Compression with Large Language Models, 2024. 1, 3

[53] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023. 2

[54] Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, Jifeng Dai, and Xiaojie Jin. Flash-vstream: Memory-based real-time understanding for long video streams, 2024. 2

[55] Pan Zhang, Xiaoyi Dong, Bin Wang, Yuhang Cao, Chao Xu, Linke Ouyang, Zhiyuan Zhao, Shuangrui Ding, Songyang Zhang, Haodong Duan, Wenwei Zhang, Hang Yan, Xinyue Zhang, Wei Li, Jingwen Li, Kai Chen, Conghui He, Xingcheng Zhang, Yu Qiao, Dahua Lin, and Jiaqi Wang. Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition. *arXiv preprint arXiv:2309.15112*, 2023. 1, 2

[56] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024. 6

[57] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023. 1, 2