TPRL: Adaptive Visual Token Pruning in LVLMs via Language-Guided Reinforcement Learning

Sihan Cao[1], **Jianwei Zhang**[1], **Pengcheng Zheng**[1], **Jiaxin Yan**[1], **Caiyan Qin**[2], **Yalan Ye**[1], **Wei Dong**[3], **Peng Wang**[1], **Yang Yang**[1], **Chaoning Zhang**[1*]

[1]University of Electronic Science and Technology of China
[2]Harbin Institute of Technology, Shenzhen
[3]Xi'an University of Architecture and Technology

2023080903002@std.uestc.edu.cn, chaoningzhang1990@gmail.com

Abstract

Large Vision-Language Models (LVLMs) incur substantial inference costs due to the processing of a vast number of visual tokens. Existing methods typically rely on hand-engineered scoring rules that fail to capture the complex inter-dependencies of progressive visual token reduction, while directly deploying multi-step pruner during inference introduces unacceptable latency. To overcome this dilemma, we propose TPRL, a reinforcement learning framework that leverages sequential multi-step exploration during training to derive a highly adaptive, one-shot pruning policy for efficient inference. During the training phase, we formulate visual token reduction as a Markov Decision Process (MDP) with explicit state transitions, employing a self-supervised autoencoder to compress visual tokens for efficient policy learning. Initialized via learning from demonstrations, the policy is fine-tuned using Proximal Policy Optimization (PPO) to explicitly learn sequential token dependencies and jointly optimize end-task accuracy and computational efficiency. Crucially, at inference time, TPRL deploys this trained policy as a deterministic one-shot selector, amortizing the cost of sequential reasoning into a single forward pass. Our experimental results demonstrate that TPRL removes up to 66.7% of visual tokens and achieves up to a 65.3% reduction in FLOPs during inference while maintaining a near-lossless average accuracy drop of only 0.2% on LLaVA-1.5-7B. Extensive evaluations across diverse architectures further confirm the robust generalizability of our approach.

1 Introduction

Large Vision-Language Models (LVLMs) [4, 28, 33, 31, 59, 16, 27, 55, 11] have achieved remarkable success in multimodal understanding by integrating high-capacity visual encoders with powerful Large Language Models (LLMs) [46, 47, 7]. However, this superior performance comes at a high computational cost. As modern LVLMs increasingly adopt high-resolution inputs to capture fine-grained spatial details, the volume of visual tokens generated by the encoder grows significantly [52]. This influx of tokens leads to prohibitive inference latency and high memory consumption, posing a substantial challenge for deploying these models in real-time or resource-constrained applications.

To mitigate this computational bottleneck, extensive research efforts [22, 15, 2, 9, 57, 49, 56, 53, 24, 25, 8, 54, 43, 45, 44, 58] have been dedicated to visual token pruning and merging. The core motivation behind these methods is to exploit the inherent spatial redundancy of high-resolution visual

*Corresponding author.
†Code available at: https://github.com/MagicVicCoder/TPRL

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

inputs, where large background areas or repetitive textures can be compressed without sacrificing critical information. These existing approaches can be categorized into two distinct paradigms. The first paradigm relies on hand-engineered heuristic scoring rules [15, 2, 57, 56, 24]. These methods typically leverage structural priors or internal model activations such as attention weights or feature similarity between adjacent patches. By calculating these proxy metrics, heuristic approaches discard those deemed least informative tokens to the overall image representation. Conversely, a second line of research attempts to move beyond fixed rules by introducing learnable or dynamic token reduction mechanisms [53, 36, 52]. Rather than relying on static mathematical metrics, these approaches utilize auxiliary prediction networks, lightweight routing layers, or iterative decision modules. Their objective is to adaptively filter tokens in a data-driven manner, conditionally adjusting the retention of visual features based on the specific semantic context of the input image and the associated textual prompt.

While these paradigms have demonstrated promising results in reducing inference costs, they face critical limitations that hinder their optimal deployment. Heuristic-based methods are inherently task-agnostic and make decisions based on static snapshots. They entirely ignore how the progressive removal of certain tokens dynamically alters the semantic importance of the remaining sequence. On the other hand, learnable dynamic methods struggle to balance training effectiveness with inference efficiency. They often either fail to model the complex sequential dependencies of token pruning during optimization [53], or rely on heavy, iterative decision modules during inference [36]. This multi-step execution introduces substantial latency overhead, effectively negating the computational savings they aim to achieve. Consequently, resolving the tension between deep modeling of sequential token dependencies and single-pass inference efficiency remains a critical challenge.

To overcome these limitations, we propose TPRL, the first reinforcement learning framework that resolves the tension between deep dependency modeling and inference efficiency. Unlike previous works, TPRL formulates the token reduction discovery phase as a Markov Decision Process (MDP) [38] where the pruning policy is directly tied to end-task rewards. This sequential training paradigm allows the pruner to explore and capture the complex, step-to-step interdependencies between visual tokens. To make RL training efficient, we utilize a self-supervised autoencoder to compress visual tokens into a compact state representation and address exploration challenges by initializing the policy through Learning from Demonstrations (LfD) [19] followed by Proximal Policy Optimization (PPO). Crucially, to circumvent the latency bottlenecks of existing dynamic methods, the sequential reasoning learned during our RL phase is entirely internalized into the policy weights. This allows TPRL to be deployed as a highly adaptive yet deterministic one-shot selector at inference time, ensuring that retained tokens are semantically essential for the textual query without incurring any multi-step execution overhead.

Our main contributions are summarized as follows:

- We propose TPRL, the first reinforcement learning framework that introduces an amortized sequential reasoning paradigm to resolve the tension between dynamic token dependency modeling and inference efficiency.
- We design a tailored optimization pipeline to overcome the curse of dimensionality and exploration inefficiency inherent in high-dimensional multimodal token spaces. By combining a self-supervised autoencoder for extreme state compression with a LfD initialization, our approach enables stable and sample-efficient convergence of the policy network.
- Extensive experiments demonstrate that TPRL significantly enhances inference efficiency through adaptive token reduction while maintaining outstanding performance across a diverse range of benchmarks and models.

2 Related Work

2.1 Large Vision-Language Models

Large Vision-Language Models (LVLMs), such as LLaVA, Qwen-VL and InternVL [7, 46, 47, 39, 1, 51, 12, 11, 50, 55], have achieved outstanding performance by aligning visual features from a pre-trained Vision Transformer (ViT) with the embedding space of a Large Language Model (LLM) [4, 59, 13, 27, 33]. While these models excel in multimodal reasoning and instruction following, their inference efficiency is severely hampered by the massive number of visual tokens. As higher

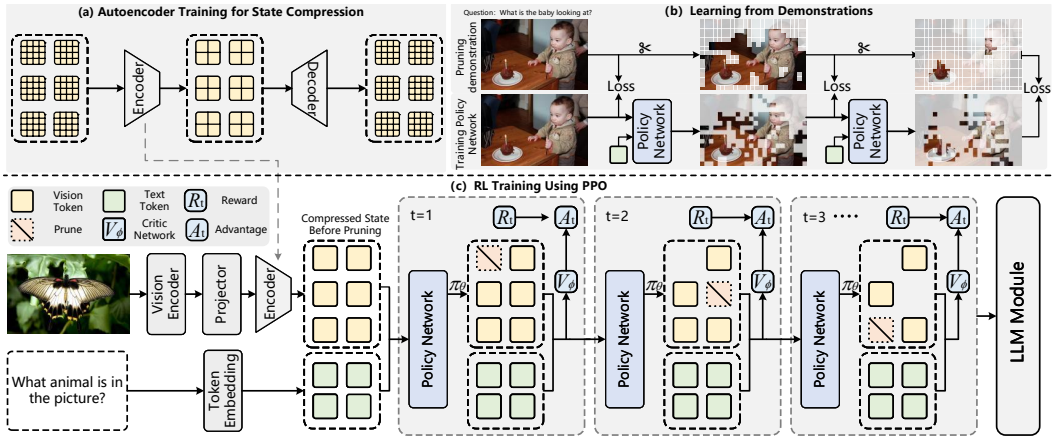


Figure 1: Overview of the TPRL framework consisting of three synergistic stages. **(a) Autoencoder training for state compression** learns a compact latent representation to reduce the feature dimensionality of each visual token. **(b) LfD** provides a foundational initialization for the pruning policy by imitating high-quality trajectories to ensure stable convergence. **(c) RL training using PPO** fine-tunes the policy through language-guided sequential optimization to achieve an optimal balance between task performance and computational efficiency.

resolutions become standard to capture fine-grained spatial information, the computational burden on the LLM backbone increases significantly. Our work addresses this challenge by introducing a token-level optimization framework that reduces the input density for the LLM without requiring architectural changes to the underlying models.

2.2 Visual Token Pruning for LVLMs

Visual token pruning seeks to mitigate the computational bottleneck of LVLMs by discarding redundant visual information, thereby reducing memory footprint and inference latency. Existing literature can generally be classified into three paradigms based on the architectural stage where the pruning operation occurs. First, *encoder-side* methods, such as ToMe [6], reduce token redundancy internally within the layers of the Vision Transformer. Second, *decoder-side* methods, including PACT [15], ATP-LLaVA [53], and FastV [9], dynamically drop tokens inside the language model backbone after cross-modal interactions have commenced. Finally, *interface-side* approaches, such as DivPrune [2], PruMerge+ [43], VisPruner [56] and ZOO-Prune [24], operate as an intermediate bridge, filtering tokens after visual encoding but prior to their ingestion by the LLM. Unlike existing heuristic approaches, TPRL is purposefully designed as an architecturally agnostic interface-side method that captures progressive semantic dependencies and maximizes computational savings by filtering redundant tokens before they enter the expensive LLM prefill and decoding stages.

3 Methodology

3.1 Overview

In this section, we present TPRL, a reinforcement learning framework for visual token pruning in LVLMs. The overall architecture of our method is shown in Figure 1. First, we formulate the adaptive pruning task as a MDP (Sec. 3.2) to enable sequential decision-making. This formulation defines a compact state representation, action space, and reward function. Second, we conduct *LfD* (Sec. 3.3). In this stage, the policy network is pre-trained with supervised learning on pruning trajectories generated by existing heuristic methods. Third, we detail the RL-based adaptive pruning framework (Sec. 3.4), including the architecture of the policy and value networks. Finally, we present the full training and inference protocol (Sec. 3.5), employing Proximal Policy Optimization (PPO) [42] for policy optimization.

3.2 Language-guided MDP Formulation

We formulate the adaptive token pruning task as MDP to enable sequential, language-guided decision-making. The MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$, $\mathcal{P}$, and $\mathcal{R}$ represent the state space, action space, state transition dynamics, and the multi-objective reward function, respectively, while $\gamma \in [0, 1]$ denotes the discount factor.

State Space. To enable efficient reinforcement learning, we learn a compact state representation by reducing the feature dimensionality of each visual token independently. We train a lightweight token-wise autoencoder. The encoder $\mathcal{E}$ maps a single visual token $\mathbf{h}_i \in \mathbb{R}^{d_v}$ to a low-dimensional code $\mathbf{z}_i \in \mathbb{R}^{d_l}$ with $d_l \ll d_v$. The decoder $\mathcal{D}$ reconstructs the original token from $\mathbf{z}_i$, and the autoencoder is trained with a per-token mean squared reconstruction objective:

$$\mathcal{L}_{\text{rec}} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{h}_i - \mathcal{D}(\mathbf{z}_i)\|_2^2. \tag{1}$$

This compression only reduces feature dimensionality and does not alter the number of visual tokens. After training, we freeze $\mathcal{E}$. At inference and during RL, we apply $\mathcal{E}$ once to obtain the compressed set $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_N] \in \mathbb{R}^{N \times d_l}$. The state at pruning step t is defined as $s_t = (Z_t, q)$, where $Z_t \in \mathbb{R}^{K_t \times d_l}$ contains codes for the K_t remaining tokens and $q \in \mathbb{R}^{d_q}$ denotes the text query embedding. The compressed codes serve only the pruning policy.

We introduce a binary mask vector $\mathbf{m} \in \{0, 1\}^N$ to track the retention of the original visual tokens $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]^\top \in \mathbb{R}^{N \times d_v}$. The mask is initialized to all ones. After the policy outputs actions at step t, any token with original index idx that is marked to be pruned has its corresponding mask entry set to zero: $\mathbf{m}[idx] = 0$. The original indices remain fixed even when earlier tokens are removed. During multi-step RL, the policy always operates on compressed codes, and intermediate masks are applied to the compressed representations for subsequent policy steps, while all mask updates refer to original indices. At inference, the policy produces a one-shot mask by thresholding the retention probabilities, and this mask is applied directly to the original uncompressed tokens. Formally, the retained visual tokens after pruning are given by $\tilde{\mathbf{H}} = \{\mathbf{h}_{idx} \mid \mathbf{m}[idx] = 1\}$, which are then forwarded to the downstream LLM. Compressed codes are used only by the pruning policy and are never sent to the LLM.

Action Space. At each step t, the pruner outputs a binary retention decision for each token in Z_t:

$$a_t = \{d_i \mid i \in \{1, \ldots, K_t\}, d_i \in \{0, 1\}\}, \tag{2}$$

where $d_i = 0$ denotes pruning token i, and $d_i = 1$ denotes retaining it. The state transition follows:

$$Z_{t+1} = \{\mathbf{z}_{t,i} \mid d_i = 1\}, \tag{3}$$

where $\mathbf{z}_{t,i}$ is the compressed feature of the i-th token at step t.

Reward Function. The reward at step t balances task performance and computational efficiency, computed over a mini-batch of B samples to reduce variance:

$$r_t = R_{\text{task}}(t) + R_{\text{eff}}(t), \tag{4}$$

$$R_{\text{task}}(t) = \alpha \cdot \frac{1}{B} \sum_{b=1}^{B} \left[\text{Score}(H_t^{(b)}, q^{(b)}) - \text{Score}(H_{t-1}^{(b)}, q^{(b)}) \right], \tag{5}$$

$$R_{\text{eff}}(t) = \beta \cdot \frac{1}{B} \sum_{b=1}^{B} \left(1 - \frac{K_t^{(b)}}{K_{t-1}^{(b)}} \right), \tag{6}$$

where α and β are weighting coefficients, and Score evaluates the downstream task performance given the retained tokens. Here $H_t^{(b)}$ denotes the set of original visual tokens that remain after applying the pruning mask at step t to the initial visual features $H^{(b)}$ (i.e., $H_t^{(b)} = \{\mathbf{h}_i^{(b)} \mid \mathbf{m}_t^{(b)}[i] = 1\}$).

3.3 Learning from Demonstrations

The policy network faces the challenge of inefficient exploration and may converge to local optima when trained from scratch. Inspired by [10, 19], we pre-train it using supervised learning on high-quality pruning demonstrations. These demonstrations are generated offline using existing heuristic pruning methods that provide reasonable, though imperfect, pruning trajectories.

Demonstration Generation. For each image–text pair in the training set we run a heuristic pruner to produce a short pruning trajectory. Each trajectory τ is recorded as a sequence of state–action pairs (s_t, a_t), where the state s_t includes the compressed visual representation Z_t and the textual query q, and a_t is the binary pruning action recommended by the heuristic at step t. Trajectories are generated by applying the reference pruner with progressively increased pruning rates, thereby producing stepwise demonstrations that remove more tokens at each step. Demonstration labels are assigned per token as

$$a_{t,i}^{\text{demo}} = \begin{cases} 1, & \text{if token } i \text{ is retained at step } t\ , \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Supervised Pre-training. The policy network π_θ (with parameters θ) is trained to imitate the demonstrated actions by minimizing the binary cross-entropy loss over all tokens and steps in the demonstration dataset $\mathcal{D}_{\text{demo}}$:

$$\begin{aligned} \mathcal{L}_{\text{BCE}}(\theta) = -\frac{1}{|\mathcal{D}_{\text{demo}}|} \sum_{\tau \in \mathcal{D}_{\text{demo}}} \Bigg[\sum_{t} \sum_{i=1}^{K_t} \Big(a_{t,i}^{\text{demo}} \log p_{t,i} \\ + (1 - a_{t,i}^{\text{demo}}) \log(1 - p_{t,i}) \Big) \Bigg] \end{aligned} \tag{8}$$

where $a_{t,i}^{\text{demo}}$ is the demonstration action for token i at step t, and $p_{t,i} = \pi_\theta(s_t)_i$ is the retention probability predicted by the policy network. This supervised pre-training phase equips the policy with a foundational understanding of token relevance, significantly improving sample efficiency and stability during the subsequent RL fine-tuning.

3.4 RL-based Adaptive Token Pruning

Our RL framework leverages the pre-trained policy network as a starting point and fine-tunes it using PPO to optimize the pruning strategy. The policy and value networks share an attention-based feature extractor [42, 20, 3] that operates on the compressed state s_t, effectively capturing the spatial and semantic interdependencies among visual tokens. This architecture reduces the number of parameters, improves training efficiency, and minimizes memory usage.

Shared Attention Module. The module processes the concatenated sequence of compressed visual tokens and the projected query:

$$X = [Z_t; \tilde{q}] \in \mathbb{R}^{(K_t+1)\times d}, \tag{9}$$

where $\tilde{q}$ is the linearly projected query embedding. Multi-head self-attention with residual connections models token relationships:

$$\text{Attention}(X) = \text{softmax}\left(\frac{XW_Q(XW_K)^\top}{\sqrt{d_k}}\right) XW_V, \tag{10}$$

$$F = \text{LayerNorm}(X + \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O), \tag{11}$$

producing enriched features $F = [f_1, \ldots, f_{K_t}, f_q]$.

Policy Network. For each visual token i, the policy network computes a retention probability:

$$g_i = \text{LayerNorm}(f_i + W_2 \cdot \text{GELU}(W_1 f_i + b_1) + b_2), \tag{12}$$

$$p_{t,i} = \sigma(w^\top g_i + b). \tag{13}$$

The action for each visual token is sampled from a Bernoulli distribution whose success probability is discounted by a step-wise factor during RL:

$$a_{t,i} \sim \text{Bernoulli}\big(\lambda_{\text{disc}}^{t} p_{t,i}\big), \tag{14}$$

$$P(a_{t,i}=1) = \lambda_{\text{disc}}^{t} p_{t,i}, \quad P(a_{t,i}=0) = 1 - \lambda_{\text{disc}}^{t} p_{t,i}, \tag{15}$$

where $\lambda_{\text{disc}} \in (0,1)$ is a step-wise discount factor that decays the retention probability with step index t, encouraging more aggressive pruning in later steps and accelerating training. We sample actions during RL for exploration and gradient estimation.

At inference we use deterministic one-shot pruning:

$$a_{t,i} = \mathbf{1}\{p_{t,i} > \tau\}. \tag{16}$$

Value Network. The value network estimates the state value by aggregating token features:

$$\bar{f} = \frac{1}{K_t} \sum_{i=1}^{K_t} f_i, \tag{17}$$

$$V_\phi(s_t) = u^\top \text{LayerNorm}(\bar{f} + U_2 \cdot \text{GELU}(U_1 \bar{f} + d_1) + d_2) + b_v. \tag{18}$$

3.5 Training and Inference

Training Strategy. Our training proceeds in three phases: (1) Autoencoder training for state compression via $\mathcal{L}_{\text{rec}}$; (2) Supervised policy pre-training via $\mathcal{L}_{\text{BCE}}$ using demonstrations; (3) RL fine-tuning using PPO. During RL, we collect trajectories over $T_{\max}$ pruning steps per sample. The advantage $\hat{A}_t$ is estimated using Generalized Advantage Estimation (GAE) [41]:

$$\delta_t = r_t + \gamma V_{\phi_{\text{old}}}(s_{t+1}) - V_{\phi_{\text{old}}}(s_t) \tag{19}$$

$$\hat{A}_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+l}. \tag{20}$$

The PPO objective combines clipped policy loss, value loss, and entropy regularization:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min\Big(r_t(\theta)\hat{A}_t,\ \text{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big)\hat{A}_t \Big) \right], \tag{21}$$

$$L^{\text{VF}}(\phi) = \mathbb{E}_t \left[\big(V_\phi(s_t) - V_t^{\text{target}}\big)^2 \right], \tag{22}$$

$$L^{\text{S}}(\theta) = \mathbb{E}_t \left[\mathcal{H}(\pi_\theta(\cdot|s_t)) \right]. \tag{23}$$

The total optimization objective is then defined as:

$$L(\theta,\phi) = -L^{\text{CLIP}}(\theta) + c_1 L^{\text{VF}}(\phi) - c_2 L^{\text{S}}(\theta), \tag{24}$$

where $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ is the probability ratio between the new and old policies, $\mathcal{H}$ denotes the entropy of the policy distribution, and c_1 and c_2 are weighting coefficients that balance the value regression loss and entropy regularization, respectively.

Inference Protocol. During deployment, TPRL utilizes a deterministic single-pass pruning protocol. The trained policy evaluates all compressed tokens simultaneously to output retention probabilities $\{p_i\}$ and prunes any token where $p_i \leq \tau$. Although trained via a sequential MDP to capture complex interdependencies, this simultaneous execution minimizes reasoning latency. This design eliminates autoregressive overhead while effectively leveraging the pruning logic acquired through the sequential optimization phase. An adjustable threshold τ explicitly dictates the balance between accuracy and efficiency. Finally, the remaining tokens form a shortened sequence for the downstream LVLM.

Table 1: Performance comparison on multiple benchmarks with the LLaVA-1.5-7B backbone. All baselines use a frozen LLM by default. F and L denote the frozen LLM and LoRA-tuned LLM settings, respectively. **Avg Acc. (%)** means the average of normalized relative percentage accuracy across all benchmarks. Bold indicates the best result in each column.

Method	GQA	MMB	MME	POPE	SEED	SQA^I	VQA^{v2}	VizWiz	Avg Acc. (%)
*Full-token Model (576 Tokens) (**100%**)*									
LLaVA-1.5-7B	62.0	64.6	1553	85.9	60.3	70.4	78.9	50.0	100
Retain ≈192 Tokens (↓ 66.7%)									
ToMe (ICLR 2023)	54.3	60.5	1485	72.4	53.1	65.2	67.9	50.2	91.1 (↓8.9%)
FastV (ECCV 2024)	52.7	61.2	1312	64.7	50.9	65.4	67.2	50.8	88.0 (↓12.0%)
SparseVLM (ICML 2025)	57.6	62.5	1473	83.6	53.0	67.2	75.6	50.8	94.6 (↓5.4%)
VisPruner (ICCV 2025)	58.1	62.9	1475	84.9	**59.0**	69.2	76.5	51.2	97.6 (↓2.4%)
ZOO-Prune (CVPR 2026)	60.0	62.9	1502	**87.2**	58.3	**69.4**	77.4	51.0	98.2 (↓1.8%)
TPRL*$_F$ (Ours)	60.8	65.1	1534	85.2	58.6	68.8	78.1	51.2	99.4 (↓0.6%)
TPRL*$_L$ (Ours)	**61.1**	**66.1**	**1554**	85.7	58.9	69.0	**78.5**	**51.3**	**99.8** (↓0.2%)
Retain ≈128 Tokens (↓ 77.8%)									
ToMe (ICLR 2023)	52.4	53.3	1343	62.8	50.9	59.6	63.0	50.5	84.6 (↓15.4%)
FastV (ECCV 2024)	49.6	56.1	1195	59.2	48.5	59.7	61.8	51.3	82.4 (↓17.6%)
SparseVLM (ICML 2025)	56.0	60.0	1360	80.5	50.4	65.1	73.2	51.4	92.0 (↓8.0%)
PruMerge+ (ICCV 2025)	57.8	61.3	1420	81.5	57.7	67.6	74.7	52.4	95.7 (↓4.3%)
VisPruner (ICCV 2025)	57.9	62.7	1461	84.5	**58.7**	68.5	75.8	**52.7**	97.4 (↓2.6%)
ZOO-Prune (CVPR 2026)	59.5	61.8	1447	**87.1**	57.9	**68.9**	**76.5**	51.2	96.9 (↓3.1%)
TPRL*$_F$ (Ours)	59.6	64.9	1470	83.0	58.1	68.7	76.0	52.3	97.8 (↓2.2%)
TPRL*$_L$ (Ours)	**60.2**	**65.4**	**1472**	83.3	58.4	68.7	76.1	52.5	**98.2** (↓1.8%)
Retain ≈64 Tokens (↓ 88.9%)									
ToMe (ICLR 2023)	48.6	43.4	993	52.5	50.3	50.2	57.3	50.2	74.8 (↓25.2%)
FastV (ECCV 2024)	46.1	47.9	1020	48.0	50.9	51.1	55.0	50.8	74.8 (↓25.2%)
VisPruner (ICCV 2025)	55.4	61.3	1370	80.4	55.8	**69.3**	72.7	**53.3**	94.6 (↓5.4%)
ZOO-Prune (CVPR 2026)	**58.5**	61.2	1425	**85.8**	55.6	68.2	**75.0**	51.0	95.4 (↓4.6%)
TPRL*$_F$ (Ours)	55.8	63.2	1440	81.4	57.4	67.1	74.5	52.3	95.6 (↓4.4%)
TPRL*$_L$ (Ours)	55.9	**63.5**	**1443**	81.6	**57.6**	67.3	74.7	52.3	**95.8** (↓4.2%)

4 Experiments

4.1 Experimental Setup

4.1.1 Implementation Details

For LLaVA-1.5 [31], we adopt 7B and 13B language models paired with the CLIP-ViT-L/14 vision encoder. For Qwen-VL-2.5 [4], we utilize 3B and 7B models equipped with NaViT [14]. We also evaluate our method on LLaVA-NeXT-7B [32] and InternVL3-8B [11]. The experimental results for Qwen-VL-2.5-7B, LLaVA-NeXT-7B, and InternVL3-8B are provided in Appendix C.

The token-wise autoencoder for state compression consists of a two-layer MLP encoder and a symmetric decoder, where the encoder reduces the feature dimensionality of each visual token to 1/16 of its original size by mapping 4096-dimensional embeddings through a hidden layer of 1024 units with LayerNorm and GELU activation to a 256-dimensional latent space. The decoder mirrors this structure to reconstruct the original features, after which the encoder is frozen for all subsequent stages. To generate demonstrations for pre-training, we utilize Vispruner [56] to construct 50K pruning trajectories from the training data. For each image–text pair, we generate a short trajectory by applying two pruning rates sequentially from no mask to a 25% pruning mask and finally to a 50% pruning mask. These short demonstration trajectories are intentionally used to accelerate policy convergence by providing a reasonable initialization for visual token relevance. The subsequent RL stage then refines this initialization and learns longer pruning sequences beyond the demonstrated steps to optimize for final task rewards. Further details regarding RL hyperparameters, training configurations, and inference threshold settings are provided in Appendix A.

Table 2: Performance comparison with LLaVA-1.5-13B.

Method	Token	GQA	MMB	MME	POPE	SQA^I	Avg.
LLaVA-1.5-13B	576	63.3	68.6	1522	86.0	72.9	100%
FastV	192	54.2	64.8	1290	64.1	69.9	86.4%
	128	48.5	61.2	1152	58.2	68.0	80.5%
Ours	192*	62.8	68.95	1500	85.5	71.8	99.2%
	128*	61.2	67.5	1465	83.5	70.8	97.1%

Table 3: Performance comparison with Qwen-VL-2.5-3B.

Method	Token	GQA	SQA^I	MME	Avg.
Qwen-VL-2.5-3B	256	61.5	70.2	1450	100%
FastV	144	54.7	64.6	1305	90.3%
	128	51.5	61.1	1262	85.9%
Ours	144*	60.4	68.2	1399	97.3%
	128*	60.1	67.7	1393	96.7%

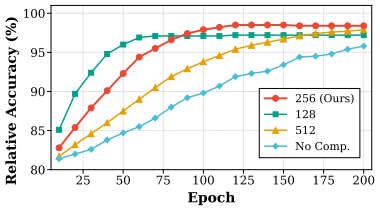


Figure 2: Impact of state compression dimension d_l on accuracy.

Table 6: Efficiency analysis including cache storage memory, latency and the FLOPs.

Method	Avg. Token	Accuracy	Storage Memory (MB) ↓	Latency (ms) ↓	FLOPs (T) ↓
Upper Bound	576	100%	302.4	57.8	9.6
FastV	192	88.0%	113.4 ↓62.5%	34.9 ↓39.6%	3.7 ↓61.1%
SparseVLM	192	94.6%	100.9 ↓66.6%	36.5 ↓36.9%	3.3 ↓65.3%
Ours	192*	99.8%	104.8 ↓65.3%	35.4 ↓38.8%	3.3 ↓65.3%
	128*	98.2%	71.2 ↓76.5%	31.3 ↓45.8%	2.3 ↓76.1%
	64*	95.8%	37.6 ↓87.6%	27.3 ↓52.8%	1.2 ↓87.0%

4.1.2 Datasets and Metrics

We conduct experiments on eight benchmarks: GQA [23], MMBench [34], MME [17], POPE [29], SEED-Bench [26], ScienceQA-Image [37], VQA-v2 [18], and VizWiz [5]. These benchmarks cover a wide range of capabilities including visual question answering, reasoning, perception, and hallucination evaluation. For training the token-wise autoencoder, generating pruning demonstrations, and supervised pre-training of the policy, we use the training splits of MME and VQA-v2. We report absolute task performance on each benchmark as well as the relative percentage with respect to the corresponding upper bound model.

4.2 Experimental Results

We denote the retained token counts for TPRL with (*) in the following experiments to indicate that these are average approximations resulting from our adaptive token pruning.

Main Results. We first evaluate TPRL extensively on the LLaVA-1.5-7B backbone, adjusting the threshold τ to align the average visual token retention to 192, 128, and 64 (yielding pruning rates of 66.7%, 77.8%, and 88.9%). As summarized in Table 1, TPRL achieves near-lossless compression across all eight benchmarks. Notably, TPRL even exceeds the unpruned Upper Bound on benchmarks like MMBench, MME and VizWiz. This demonstrates that the RL pruner effectively filters out redundant visual noise that otherwise interferes with the LLM's reasoning process. Furthermore, to validate the scalability of our approach, we apply TPRL to LLaVA-1.5-13B and Qwen-VL-2.5-3B. As shown in Table 2 and Table 3, TPRL consistently delivers exceptional efficiency-accuracy trade-offs on these models as well. These robust findings across diverse model scales confirm the task-awareness and generalizability of the TPRL framework.

Efficiency Analysis. We provide a comprehensive evaluation of TPRL by analyzing its cache storage memory, latency, and FLOPs based on the total inference complexity $\mathcal{F}_{\text{total}} = \mathcal{F}_{\text{vision}} + \mathcal{F}_{\text{pruner}} + \mathcal{F}_{\text{prefill}} + \mathcal{F}_{\text{decode}}$, where $\mathcal{F}_{\text{vision}}$ and $\mathcal{F}_{\text{pruner}}$ represent the computational costs of the vision encoder and our pruning module, respectively. The LLM backbone workload comprises the prefill phase $\mathcal{F}_{\text{prefill}} \approx 2P \cdot (N_{v'} + N_t)$ for parallel input processing and the decoding phase $\mathcal{F}_{\text{decode}}$ for autoregressive generation. During the decoding stage, which accounts for a substantial portion of the inference time, reducing $N_{v'}$ directly minimizes the size of the KV cache and decreases the computational cost of the attention mechanism for every newly generated token. As summarized in Table 6, the reduction in visual tokens yields substantial efficiency gains across all execution metrics. While heuristic methods like FastV and SparseVLM achieve negligible $\mathcal{F}_{\text{pruner}}$ through simple attention-based dropping, TPRL establishes a superior accuracy-efficiency trade-off.

4.3 Ablation Studies and Analysis

Table 7: Ablation of sequential decision modeling under matched token budgets with a frozen LLM.

Method	GQA	MMB	MME	POPE	SEED	SQA^I	VQA^{v2}	VizWiz	Avg Rel. (%)
Retain ≈192 Tokens (↓ 66.7%)									
VisPruner, teacher	58.1	62.9	1475	84.9	**59.0**	**69.2**	76.5	**51.2**	97.6
One-shot Distillation	57.0	60.7	1392	81.5	56.1	66.2	73.2	50.0	94.4
One-step RL, $T_{\max} = 1$	59.0	62.2	1437	83.6	58.2	67.9	75.4	50.4	96.3
Full TPRL, $T_{\max} = 3$	**60.8**	**65.1**	**1534**	**85.2**	58.6	68.8	**78.1**	**51.2**	**99.4**
Retain ≈128 Tokens (↓ 77.8%)									
VisPruner, teacher	57.9	62.7	1461	**84.5**	**58.7**	68.5	75.8	**52.7**	97.4
One-shot Distillation	55.5	59.5	1374	79.4	55.5	64.8	70.5	49.7	93.3
One-step RL, $T_{\max} = 1$	58.8	63.0	1407	81.9	57.8	67.3	74.9	51.5	96.0
Full TPRL, $T_{\max} = 3$	**59.6**	**64.9**	**1470**	83.0	58.1	**68.7**	**76.0**	52.3	**97.8**
Retain ≈64 Tokens (↓ 88.9%)									
VisPruner, teacher	55.4	61.3	1370	80.4	55.8	**69.3**	72.7	**53.3**	94.6
One-shot Distillation	52.0	56.5	1305	77.2	52.9	61.5	66.1	49.8	90.2
One-step RL, $T_{\max} = 1$	54.8	60.8	1391	80.2	55.1	66.5	71.4	51.0	93.0
Full TPRL, $T_{\max} = 3$	**55.8**	**63.2**	**1440**	**81.4**	**57.4**	67.1	**74.5**	52.3	**95.6**

We conduct our ablation studies and analyses based on LLaVA-1.5-7B with 192 visual tokens, using LLM fine-tuning as the default setting. Comprehensive ablation studies are presented in Appendix B.

Impact of State Compression Dimension. We evaluate the impact of the compressed feature dimension d_l (Figure 2). Reducing the original 4096-dimensional features to 256 maintains convergence speeds and accuracies identical to the uncompressed baseline, confirming that critical high-level semantics are preserved in this compact latent space. Conversely, further reduction creates a severe information bottleneck that consistently degrades performance. Therefore, TPRL adopts this optimal $16\times$ compression ratio to retain highly informative state representations while keeping the pruning overhead computationally negligible.

Table 4: Ablation study on policy initialization, refinement and LLM Fine-tuning

Method	VQA^{v2}	MMB	MME	POPE
Vispruner (Frozen LLM)	97.0%	97.4%	95.0%	98.8%
Vispruner (Fine-tuned LLM)	97.5%	97.8%	95.6%	99.1%
RL scratch	96.3%	97.5%	95.7%	97.9%
LfD only	94.2%	94.7%	93.9%	94.8%
TPRL (Frozen LLM)	98.8%	99.5%	98.5%	99.2%
TPRL (Fine-tuned LLM)	99.6%	102.3%	99.4%	99.7%

Table 5: Ablation study on shared attention architecture.

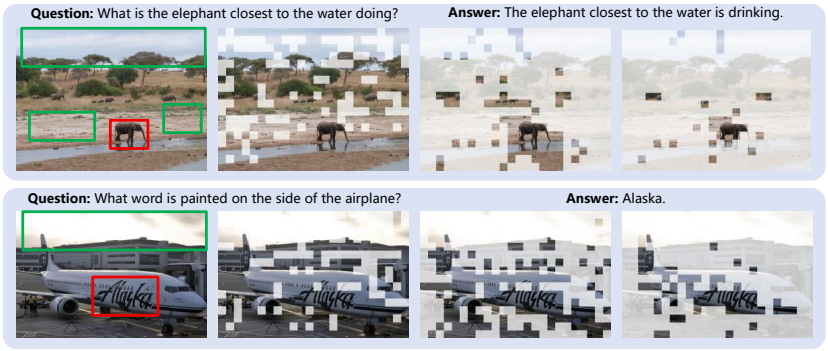
Architecture	VQA^{v2}	MMB	MME	POPE
MLP	87.4%	89.6%	87.9%	89.3%
MLP(shared)+MLP	87.6%	89.7%	87.5%	89.4%
Attn.(shared)+MLP	99.6%	102.3%	99.4%	99.7%

Contribution of Sequential Decision Modeling. To validate the importance of sequential decision modeling in TPRL, we compare our full framework with two simplified variants: one-shot distillation and one-step RL. As shown in Table 7, directly learning the final pruning mask without sequential optimization leads to consistent performance degradation across different token budgets, indicating that static mask prediction is insufficient to capture complex token dependencies. Although one-step RL improves over direct distillation by incorporating task-oriented rewards, it remains inferior to Full TPRL, demonstrating the necessity of modeling progressive pruning decisions through sequential MDP optimization.

Effect of Policy Initialization and Refinement. Table 4 shows that training from scratch faces convergence issues in vast exploration spaces, while Vispruner lacks text-guided adaptability. TPRL overcomes these via LfD, which bypasses initial exploration bottlenecks, followed by PPO training for downstream rewards.

Impact of LLM Fine-tuning. Table 4 shows that fine-tuning significantly improves performance by adapting internal representations to sparse visual inputs. This optimization enables the LLM to better interpret critical tokens and mitigate information loss from pruning. Consequently, fine-tuning ensures robust multimodal reasoning despite substantial token reduction.

Impact of Shared Attention Architecture. We evaluate our shared attention design against a global MLP baseline (Table 5). The MLP yields inferior pruning precision because it fails to capture complex relational redundancies between tokens. Conversely, our shared self-attention module enables holistic interaction across the entire visual field before feeding features into the policy and value networks.



(a) Sequential Pruning Trajectories

(b) Retention Probability Distribution

Figure 3: Qualitative visualization of TPRL. **(a) Sequential Pruning Trajectories:** The policy progressively filters out irrelevant backgrounds to isolate task-critical visual features based on the textual query. Red boxes highlight regions aligned with the required semantic information, whereas green boxes represent redundant background areas. **(b) Retention Probability Distribution:** The one-shot inference probability map. Red regions indicate higher retention probabilities, while blue regions indicate lower ones.

RL Optimization Method Comparison. We compare PPO, MAPO [30], and GRPO [48] to assess optimization robustness. As shown in Figure 4, all methods yield comparable accuracy with PPO showing a marginal stability advantage. This algorithmic insensitivity confirms that TPRL's efficacy originates primarily from our MDP formulation rather than the specific policy gradient technique.

Visualization Results. Figure 3(a) visualizes sequential pruning trajectories via mask superimposition. The four columns show the original image followed by three progressive pruning steps, where masked areas indicate pruned tokens. During training, the policy network successfully learns to prioritize pruning uniform background panels first, subsequently filtering out other redundant regions that are irrelevant to the semantic context. Figure 3(b) illustrates the one-shot retention probability distribution. This heat map confirms the policy has learned to assign high probabilities to task-critical regions. These results demonstrate that the sequential reasoning acquired during MDP training successfully translates into an accurate, text-guided one-shot token selector. Additional visualization are provided in Appendix D.

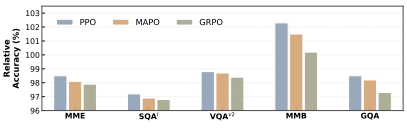


Figure 4: Comparison of RL methods.

5 Conclusion

In this paper, we present TPRL, a reinforcement learning framework designed to mitigate the prohibitive inference costs of processing vast visual tokens in LVLMs. By formulating token reduction as a language-guided MDP, our approach bridges the gap between dynamic semantic modeling and single-pass execution efficiency. Our optimization pipeline leverages a state-compressing autoencoder to overcome high-dimensional exploration bottlenecks, followed by LfD and PPO to capture complex token interdependencies. Extensive evaluations across diverse benchmarks and architectures confirm TPRL establishes a strong accuracy-efficiency trade-off, providing a principled compression solution for efficiently deploying robust LVLMs.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Natural Science Foundation of China (General Program) under Grant No. 62576267, and in part by the Major Science and Technology Innovation Projects of Xianyang under Grant Nos. L2025-ZDKJ-ZDGG-ZYRGZN-002 and L2024-ZDKJ-ZDGG-GY-0010.

References

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

[2] Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9392–9401, 2025.

[3] Maryam Ansarifard, Mohit K Sharma, Kishor C Joshi, and George Exarchakos. Transformer actor-critic for efficient freshness-aware resource allocation. *arXiv preprint arXiv:2602.22774*, 2026.

[4] 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*, 2(1):1, 2023.

[5] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, et al. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 333–342, 2010.

[6] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] Feng Chen, Yefei He, Lequan Lin, Jing Liu, Chenhui Gou, Bohan Zhuang, and Qi Wu. Sparsity forcing: reinforcing token sparsity of mllms. In *International Conference on Learning Representations*, volume 2026, pages 155748–155762, 2026.

[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. In *European Conference on Computer Vision*, pages 19–35. Springer, 2024.

[10] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[11] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024.

[12] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[13] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

[14] Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim M Alabdulmohsin, et al. Patch n'pack: Navit, a vision transformer for any aspect ratio and resolution. *Advances in Neural Information Processing Systems*, 36:2252–2274, 2023.

[15] Mohamed Dhouib, Davide Buscaldi, Sonia Vanier, and Aymen Shabou. Pact: Pruning and clustering-based token reduction for faster visual language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 14582–14592, 2025.

[16] 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.

[17] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023.

[18] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[19] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[20] Lior Hirsch and Gilad Katz. Multi-objective pruning of dense neural networks using deep reinforcement learning. *Information Sciences*, 610:381–400, 2022.

[21] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[22] Kai Huang, Hao Zou, Ye Xi, BoChen Wang, Zhen Xie, and Liang Yu. Ivtp: Instruction-guided visual token pruning for large vision-language models. In *European conference on computer vision*, pages 214–230. Springer, 2024.

[23] Drew A Hudson and Christopher D Manning. 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, 2019.

[24] Youngeun Kim, Youjia Zhang, Huiling Liu, Aecheon Jung, Sunwoo Lee, and Sungeun Hong. Training-free token pruning via zeroth-order gradient estimation in vision-language models. *arXiv preprint arXiv:2509.24837*, 2025.

[25] Ao Li, Yuxiang Duan, Jinghui Zhang, Congbo Ma, Yutong Xie, Gustavo Carneiro, Mohammad Yaqub, and Hu Wang. Transprune: Token transition pruning for efficient large vision-language model, 2025.

[26] 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.

[27] 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 *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[28] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *European Conference on Computer Vision*, pages 323–340. Springer, 2024.

[29] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 292–305, 2023.

[30] Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc Le, and Ni Lao. Memory augmented policy optimization for program synthesis with generalization. *arXiv preprint arXiv:1807.02322*, 2018.

[31] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024.

[32] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llavanext: Improved reasoning, ocr, and world knowledge, 2024.

[33] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[34] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer, 2024.

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[36] Chenglong Lu, Shen Liang, Xuewei Wang, and Wei Wang. Reinforcement learning-based token pruning in vision transformers: A markov game approach, 2025.

[37] Pan Lu, Swaroop Mishra, Tanglin 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. *Advances in neural information processing systems*, 35:2507–2521, 2022.

[38] Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

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

[40] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: international conference for high performance computing, networking, storage and analysis*, pages 1–16. IEEE, 2020.

[41] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[42] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[43] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22857–22867, 2025.

[44] Xiaole Tang, Xiaoyi He, Jiayi Xu, Xiang Gu, and Jian Sun. Learning continuous wasserstein barycenter space for generalized all-in-one image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

[45] Xiaole Tang, Jiayi Xu, Xiang Gu, Yan Yang, and Jian Sun. Binding multiple modalities via multimodal wasserstein barycenter. *arXiv preprint arXiv:2609.33800*, 2026.

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

[47] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[48] Milan Vojnovic and Se-Young Yun. What is the alignment objective of grpo? *arXiv preprint arXiv:2502.18548*, 2025.

[49] Jianchao Wang, Qingfeng Li, Pengcheng Zheng, Xiaorong Pu, and Yazhou Ren. Chronoselect: Robust learning with noisy labels via dynamics temporal memory. In *Proceedings of the 7th ACM International Conference on Multimedia in Asia*, pages 1–7, 2025.

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

[51] BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.

[52] Senqiao Yang, Junyi Li, Xin Lai, Jinming Wu, Wei Li, Zejun MA, Bei Yu, Hengshuang Zhao, and Jiaya Jia. Visionthink: Smart and efficient vision language model via reinforcement learning. *Advances in Neural Information Processing Systems*, 38:95187–95227, 2026.

[53] Xubing Ye, Yukang Gan, Yixiao Ge, Xiao-Ping Zhang, and Yansong Tang. Atp-llava: Adaptive token pruning for large vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24972–24982, 2025.

[54] Jianwei Zhang, Chaoning Zhang, Sihan Cao, Wang Liu, Pengcheng Zheng, Jiaxin Huang, Caiyan Qin, Yalan Ye, Wei Dong, and Yang Yang. Rcp: Representation consistency pruner for mitigating distribution shift in large vision-language models. *arXiv preprint arXiv:2604.04972*, 2026.

[55] Pan Zhang, Xiaoyi Dong, Bin Wang, Yuhang Cao, Chao Xu, Linke Ouyang, Zhiyuan Zhao, Haodong Duan, Songyang Zhang, Shuangrui Ding, et al. Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition. *arXiv preprint arXiv:2309.15112*, 2023.

[56] Qizhe Zhang, Aosong Cheng, Ming Lu, Renrui Zhang, Zhiyong Zhuo, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. Beyond text-visual attention: Exploiting visual cues for effective token pruning in vlms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20857–20867, 2025.

[57] 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.

[58] Pengcheng Zheng, Chaoning Zhang, Jiaxin Yan, Sihan Cao, Jianwei Zhang, Xudong Wang, Jiaquan Zhang, Jewon Lee, Tae-Ho Kim, Yang Yang, and Heng Tao Shen. Mor-mllm: Mixture of recursions for efficient multimodal large language models. *IEEE Transactions on Multimedia*, pages 1–13, 2026.

[59] 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.

A Implementation Details and Training Configurations

In this section, we provide comprehensive details regarding the hardware setup, optimization strategies, and hyperparameter configurations that we use to train the TPRL framework.

Hardware and Pre-training. All experiments are conducted on a computing server equipped with $8 \times$ NVIDIA GeForce RTX 4090 (48GB) GPUs. The implementation is based on the PyTorch framework and utilizes DeepSpeed ZeRO-2 [40] for efficient memory management and distributed training optimization, which partitions optimizer states and gradients to overcome the memory bottlenecks associated with fine-tuning Large Vision-Language Models (LVLMs). The token-wise autoencoder is trained for 10 epochs using the AdamW optimizer [35] with a learning rate of 1×10^{-4},

a weight decay of 0.01, and a batch size of 256. During the supervised pre-training phase, the Policy Network is trained to imitate the 50K trajectories generated by Vispruner [56]. We employ the AdamW optimizer with a learning rate of 5×10^{-5} and a batch size of 128. This phase is trained for 5 epochs to ensure a stable initialization before RL fine-tuning.

RL Fine-tuning (PPO). After supervised pre-training, we fine-tune the policy with PPO. For the Proximal Policy Optimization (PPO) stage, the Policy Network (Actor) and Value Network (Critic) share the attention-based feature extractor but utilize separate Multi-Layer Perceptron (MLP) heads. We use distinct learning rates of 1×10^{-5} for the Actor and 3×10^{-5} for the Critic to ensure stable value estimation. The RL training is conducted for 200 epochs. Each PPO update utilizes a rollout batch size of 64 and a mini-batch size of 16, with an optimization clip ratio (ϵ) set to 0.2. During RL fine-tuning, we collect trajectories with up to $T_{\max} = 3$ pruning steps per sample, allowing the policy to explore and learn longer sequential pruning strategies than those present in the demonstrations. PPO training uses a discount factor of $\gamma = 0.99$, a GAE parameter $\lambda = 0.95$, and reward weights $\alpha = 1.0$ (task reward) and $\beta = 0.1$ (efficiency reward). The PPO objective uses a value loss coefficient $c_1 = 0.5$ and an entropy regularization coefficient $c_2 = 0.01$. The stepwise discount factor for action sampling is set to $\lambda_{\text{disc}} = 0.5$. To adapt internal representations to sparse visual inputs , we fine-tune the LLM backbone using Low-Rank Adaptation (LoRA) [21] with a rank $r = 128$ and scaling factor $\alpha = 256$, utilizing the AdamW optimizer at a learning rate of 2×10^{-5} and employing DeepSpeed ZeRO-2 for efficient memory management.

Threshold Selection and Adaptive Token Retention. Our thresholding strategy enables sample-adaptive computation by dynamically adjusting the retained visual tokens according to the information complexity of each image-query pair. Specifically, simple or redundant inputs may retain fewer tokens, while information-dense samples can preserve more visual tokens. For each target token budget, the retention threshold is determined on a separate held-out calibration set containing 500 examples, which is disjoint from all evaluation benchmarks. We first perform a coarse grid search followed by binary search to identify the threshold that matches the desired average token budget. The selected thresholds are fixed during evaluation and are not further adjusted based on evaluation samples, labels, task scores, or retained-token statistics. Under this setting, the actual average retained-token counts over the full 1,000-example evaluation set are 193.6, 129.3, and 63.3 for target budgets of 192, 128, and 64 tokens, respectively.

Inference Setup. At inference time, we perform one-shot pruning: the policy outputs per-token retention probabilities, and we prune tokens with $p_i \leq \tau$. Specifically, we modulate the average pruning rate by adjusting the threshold τ. For instance, in our experiments on LLaVA-1.5-7B (averaged across benchmarks), setting the threshold $\tau = 0.55$ yields about 192 retained visual tokens, $\tau = 0.74$ yields about 128 tokens, and $\tau = 0.87$ yields about 64 tokens.

B Hyperparameter Ablations

To validate the stability of our RL training process, we conduct ablation studies on key hyperparameters governing the PPO objective and advantage estimation. All evaluations are performed on the LLaVA-1.5-7B backbone using the VQA-v2 benchmark, reporting relative accuracy with respect to the Upper Bound. Unless otherwise specified, the average number of retained visual tokens is fixed at approximately 192 for consistency across studies. Note that these ablations are conducted without any fine-tuning of the LLM backbone.

B.1 Loss Scaling Coefficients (c_1 and c_2)

The PPO objective requires balancing the policy gradient loss, the value function regression loss (c_1), and the entropy regularization term (c_2). As illustrated in Figure 5, $c_1 = 0.5$ provides the most stable advantage estimations. A higher c_1 over-prioritizes value network fitting, which can impede policy refinement. Figure 6 demonstrates that $c_2 = 0.01$ provides sufficient exploration to avoid early local optima without inducing instability in later training stages.

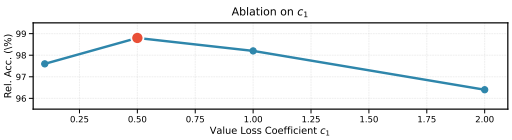


Figure 5: Impact of Value Loss Coefficient c_1 on task performance. Evaluated with $c_2 = 0.01$.

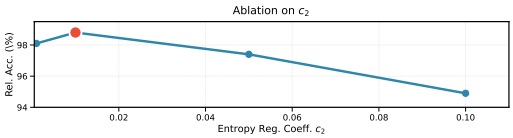


Figure 6: Impact of Entropy Coefficient c_2 on task performance. Evaluated with $c_1 = 0.5$.

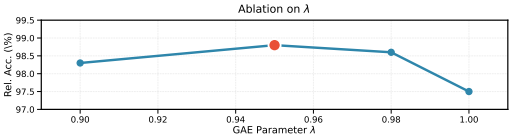


Figure 7: Ablation on GAE Parameter λ for advantage estimation balance.

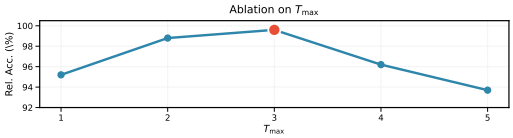


Figure 8: Impact of maximum pruning steps $T_{\max}$.

B.2 Generalized Advantage Estimation (λ)

The GAE parameter λ dictates the trade-off between bias and variance in advantage estimation. As shown in Figure 7, $\lambda = 0.95$ yields the best empirical performance, effectively balancing immediate rewards with long-term value estimations during the multi-step pruning process.

B.3 Reward Coefficient Analysis

Our reward function jointly optimizes task accuracy (R_{task}) and computational efficiency (R_{eff}). We investigate this balance by varying the efficiency reward coefficient β while fixing the task reward coefficient $\alpha = 1.0$ and the target retention ratio $\tau = 0.55$. Figure 9 visualizes the resulting trade-off between retained tokens and accuracy. Low values of β (e.g., 0.01) prioritize accuracy but fail to achieve meaningful acceleration. Conversely, excessive β (e.g., 0.5) forces aggressive pruning that discards critical semantic features, leading to a significant performance drop. Our choice of $\beta = 0.1$ establishes a superior Pareto frontier, maintaining near-lossless reasoning with high efficiency.

B.4 Maximum Pruning Steps Analysis

We evaluate the impact of the maximum pruning steps $T_{\max}$ by varying it from 1 to 5. As illustrated in Figure 8, performance shows a clear upward trend as $T_{\max}$ increases from 1 to 3, confirming that a multi-step MDP formulation enables the pruner to capture more precise token dependencies. Notably, a performance decline is observed when $T_{\max}$ exceeds 3. Excessive pruning rounds exponentially expand the exploration space and complicate credit assignment, thereby hindering stable policy convergence. Consequently, we fix $T_{\max} = 3$ to achieve an optimal balance between sequential reasoning depth and optimization stability.

B.5 Impact of the Retention Decay Factor (λ_{disc})

To evaluate the effectiveness of the explicit exploration mechanism in our multi-step MDP formulation, we analyze the impact of the stepwise retention probability decay factor λ_{disc} on VQA-v2. As shown in Figure 10, removing the decay factor results in overly conservative pruning, preventing the policy from exploring compressed states and retaining redundant tokens. Conversely, an excessively aggressive decay rate causes premature token removal, leading to significant performance degradation due to information loss. We find that $\lambda_{\text{disc}} = 0.5$ provides the best trade-off between accuracy and efficiency. Larger values gradually increase token retention while reducing accuracy, indicating that insufficient exploration limits the policy's ability to capture sequential token dependencies. Consequently, an accurately calibrated decay factor is indispensable for ensuring optimal sequential reasoning and achieving an ideal balance between task accuracy and computational efficiency.

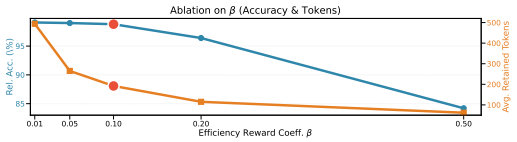


Figure 9: Trade-off analysis between task performance and token retention under varying efficiency reward coefficients β. The solid blue line denotes the relative accuracy and the orange line tracks the average number of retained tokens.

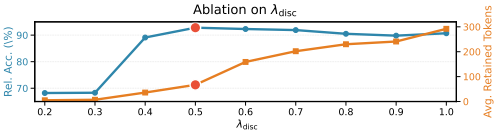


Figure 10: Ablation study on the stepwise retention probability decay factor λ_{disc}. The solid blue line denotes the relative accuracy and the orange line tracks the average number of retained tokens.

Table 8: Performance comparison on InternVL3-8B.

Method	MMB	MME	Avg Acc. (%)
Full-token Model (1280 Tokens)			
InternVL3-8B	83.9	2394	100
Retain 256 Tokens			
FastV	82.6	2257	96.4 (↓3.6%)
DivPrune	80.8	2249	95.1 (↓4.9%)
TPRL* (Ours)	**83.5**	**2334**	**98.5** (↓1.5%)
Retain 128 Tokens			
FastV	81.3	2250	95.4 (↓4.6%)
DivPrune	78.4	2166	92.0 (↓8.0%)
TPRL* (Ours)	**82.1**	**2282**	**96.6** (↓3.4%)

Table 9: Performance comparison on LLaVA-NeXT-7B.

Method	GQA	MMB	MME	POPE	Avg Acc. (%)
Full-token Model (2880 Tokens)					
LLaVA-NeXT-7B	62.2	64.0	1525	85.8	100
Retain 640 Tokens					
FastV	58.9	63.1	1413	79.5	94.7 (↓5.3%)
SparseVLM	61.2	65.9	1456	85.3	99.1 (↓0.9%)
TPRL* (Ours)	**62.4**	65.1	**1495**	**86.2**	**100.1** (↑0.1%)
Retain 320 Tokens					
FastV	49.8	53.4	1099	49.5	73.3 (↓26.7%)
SparseVLM	57.9	63.1	1386	76.9	93.0 (↓7.0%)
TPRL* (Ours)	**60.2**	**63.9**	**1426**	**84.6**	**97.2** (↓2.8%)

C Further Quantitative Results and Analysis

As indicated in the main text, we provide additional experimental results for LLaVA-NeXT-7B, Qwen-VL-2.5-7B, and InternVL3-8B in this section to further validate the architectural flexibility and robustness of our framework. We compare TPRL against recent competitive baselines including FastV, SparseVLM, and DivPrune. Table 9 presents the performance comparison on the LLaVA-NeXT-7B backbone where TPRL consistently establishes a superior balance between task accuracy and computational cost. Most notably, when retaining a moderate portion of tokens, our token selector achieves a relative average accuracy exceeding the dense upper bound. This remarkable result indicates that intelligently discarding redundant visual patches can effectively filter out distracting visual noise and consequently enhance the fundamental reasoning capabilities of the multimodal backbone. Furthermore, the advantages of our trained pruner become increasingly prominent under extreme compression scenarios. When the visual token budget is drastically restricted, heuristic methods like FastV suffer a severe performance collapse. In stark contrast, TPRL maintains highly robust performance by accurately preserving the most informative semantic regions. Similarly, the detailed comparisons on Qwen2.5-VL-7B and InternVL3-8B in Table 10 and Table 8 consistently demonstrate resilient performance across different model families even under aggressive token reduction rates. While baseline methods may occasionally show competitive metrics at moderate compression levels, they lack the text-guided semantic awareness required for stable extreme pruning. The RL-optimized policy network successfully internalizes complex cross-modal alignments, ensuring that the LVLM retains critical visual evidence regardless of the underlying architectural design.

Table 10: Performance comparison on Qwen-VL-2.5-7B.

Method	GQA	MMB	MME	POPE	Avg Acc. (%)
Full-token Model (1296 Tokens)					
Qwen2.5-VL-7B	60.8	84.1	2310	86.3	100
Retain 256 Tokens (↓ 80.2%)					
FastV (ECCV 2024)	54.7	79.6	2238	77.6	92.9(↓7.1%)
DivPrune (CVPR 2025)	60.0	79.6	2173	83.4	96.0(↓4.0%)
TPRL* (Ours)	**60.1**	**80.3**	**2325**	**84.1**	**98.1**(↓1.9%)
Retain 128 Tokens (↓ 90.1%)					
FastV	47.4	72.9	2008	69.5	83.0(↓17.0%)
DivPrune	55.5	76.0	2054	79.0	90.5(↓9.5%)
TPRL* (Ours)	**56.3**	**77.8**	**2107**	**80.5**	**92.4**(↓7.6%)

D More Visualization Examples

To further dissect the learning dynamics and the inferential efficacy of our framework, we provide comprehensive quantitative and qualitative visualizations in this section. Specifically, we first analyze the evolution of token retention throughout the reinforcement learning optimization process. We then present additional qualitative examples of our one-shot pruning execution to highlight the model's semantic sensitivity. Finally, we explore the dynamics of the inference threshold, demonstrating how our policy adaptively modulates the token retention budget based on the inherent complexity and semantic density of the visual input under a fixed threshold.

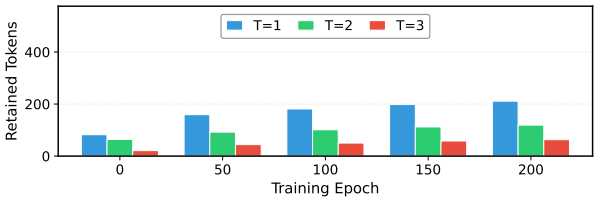


Figure 11: Evolution of the retained token count during RL training. The grouped bars denote tokens surviving after progressive pruning steps ($T = 1, 2, 3$).

Evolution of Token Retention During Training. We track the average number of retained visual tokens across different training epochs to understand the policy network's behavior during the PPO optimization. As illustrated in Figure 11, the token count is recorded after each progressive pruning step ($T = 1, 2,$ and 3) within our MDP formulation. Interestingly, as training progresses, the policy learns to retain a progressively larger number of tokens rather than mindlessly discarding them. This trend clearly demonstrates that the decision network discovers the necessity of preserving sufficient visual context to maximize the downstream text-generation reward. By actively avoiding the over-pruning trap, TPRL successfully prevents the catastrophic loss of critical semantic information that typically plagues naive heuristic methods.

Qualitative Visualization of Inference. Figure 12 presents additional qualitative examples of our trained TPRL pruner operating as an adaptive, one-shot token selector during inference. For each sample, we display three distinct components. First, the original picture highlights the ground-truth critical regions with red bounding boxes. Second, the one-shot pruning result visualizes the discarded tokens by masking them with semi-transparent white patches. Third, the retention probability distribution heatmap uses warmer red colors to indicate higher retention probabilities and cooler blue colors to represent lower probabilities. For these qualitative results, we set the threshold $\tau = 0.5$, which yields approximately 50 retained visual tokens on average per sample. The visualizations confirm that the trained pruner accurately localizes and retains the specific visual elements dictated by the corresponding textual queries. Furthermore, as explicitly demonstrated in the middle and bottom examples of the right column (which involve reading text on a piece of paper and text on a device), our token selector exhibits a profound sensitivity to fine-grained textual information within the image. The network intelligently assigns high retention probabilities to these high-frequency text patches, ensuring that the multimodal LLM retains the essential features required to accurately answer OCR-related queries.

Inference Threshold Dynamics. To address how token retention varies across different images for a fixed threshold τ, we analyze the pruning policy's adaptability to varying levels of semantic density. As shown in Figure 13, a fixed τ does not result in a rigid, uniform token count. Instead, the model dynamically adjusts the retention rate based on the complexity of the visual scene and the textual query. For simple contexts with focused subjects (Figure 13a), such as identifying a single object or attribute, the model aggressively prunes redundant background information, retaining a minimal number of tokens. Conversely, for complex and dense contexts that demand holistic understanding or detailed spatial descriptions (Figure 13b), the policy retains a substantially larger portion of tokens to preserve essential fine-grained details. This confirms that TPRL intelligently allocates the token budget based on scene complexity rather than applying a static dropping rate.

E Limitations and Future Works

While TPRL achieves significant efficiency gains and robust performance, certain limitations offer opportunities for future research. The current token pruner utilizes an attention-based policy network that introduces a computational overhead during inference, which, although offset by the overall reduction in total FLOPs, remains a constraint for deployment on resource-constrained edge devices.

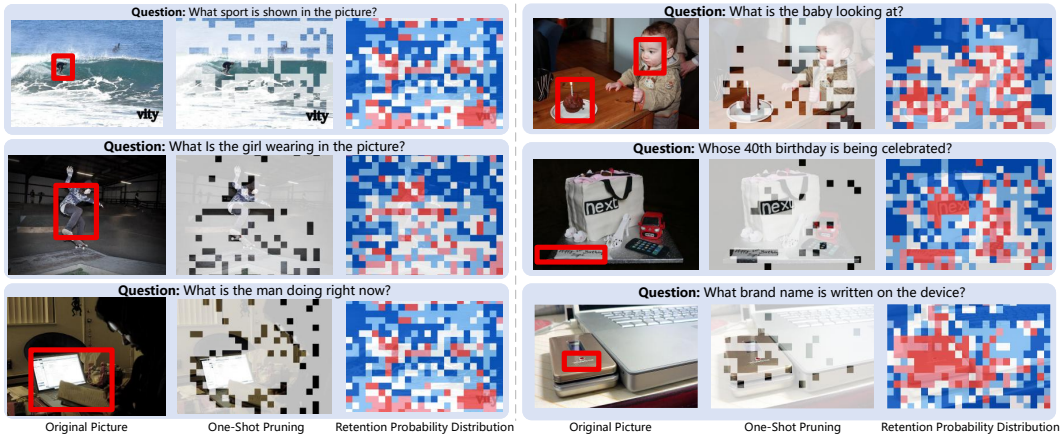
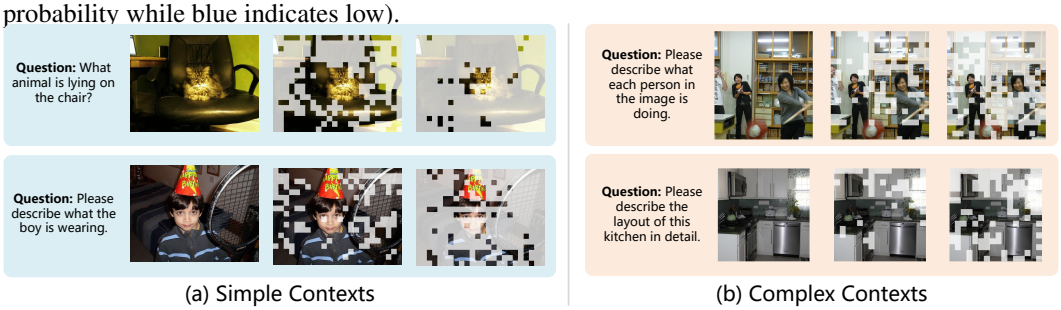


Figure 12: Additional qualitative visualizations of the TPRL pruning process during inference. For each sample, we provide the original picture (with key regions in red boxes), the one-shot pruned image (discarded tokens masked in white), and the retention probability heatmap (red indicates high probability while blue indicates low).



(a) Simple Contexts (b) Complex Contexts

Figure 13: Qualitative comparison of adaptive token retention at different thresholds (original, $\tau = 0.2$, and $\tau = 0.5$). (a) **Simple Contexts**: The policy aggressively prunes backgrounds for focused queries. (b) **Complex Contexts**: The policy adaptively retains a higher density of tokens to capture fine-grained details required for complex descriptions.

Future efforts will prioritize developing more lightweight and hardware-aware architectures to further minimize this decision-making footprint. Additionally, while this study focuses on static image understanding, extending the multi-step MDP formulation to video understanding and long-form reasoning presents a logical progression. We aim to adapt our reinforcement learning strategy to capture complex temporal dynamics and exploit temporal redundancies to achieve higher compression ratios without sacrificing semantic continuity.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state our claims, contributions, and scope in the Abstract and Section 1 (Introduction).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our proposed method in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This paper focuses on empirical results and does not include theoretical proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All necessary information required to reproduce the main experimental results is detailed in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The experiments use publicly available models and benchmarks, and the link to the full code will be provided upon the publication of this paper.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed experimental settings, including hyperparameters, data splits, and baselines, are provided in Appendix A and section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper reports deterministic benchmark scores under standard evaluation protocols but does not include error bars or statistical significance tests. This is consistent with common LVLM evaluation practice, where full benchmark evaluation is computationally expensive.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details regarding the compute resources (e.g., GPU types and training configurations) are explicitly stated in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in this paper strictly conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper primarily focuses on improving the inference efficiency of large vision-language models and does not explicitly discuss broader societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our research does not introduce or release models or datasets that pose a high risk for misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite all utilized existing public datasets and codebases, and their license information is detailed.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not introduce or release any new datasets or novel assets requiring specific documentation.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This research involves no crowdsourcing or experiments with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Not applicable as this research does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core methodology and research do not utilize LLMs (any use was strictly for writing or formatting).

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.