V-Pruner: A Fast and Globally-informed Token Pruning Framework for Vision Transformer

Guangzhen Yao[1, *, †], **Jiayun Zheng**[2, †], **Zezhou Wang**[3, †],
Wenxin Zhang[1], **Renda Han**[1], **Chuangxin Zhao**[1], **Zeyu Zhang**[1], **Runhao Liu**[1]

[1]School of Information Science and Technology, Northeast Normal University
[2]College of Engineering, University of Michigan
[3]School of Computing, Australian National University
guangzhenyao@163.com, zhengji@umich.edu, wangzezhou2002@outlook.com

Abstract

Vision Transformer (ViT) has become one of the cornerstones of the computer vision field, demonstrating exceptional performance. However, its inherent high computational complexity and inference latency still pose significant obstacles for deployment in resource-constrained environments. Token pruning, by removing less informative tokens, offers an effective strategy to reduce computational overhead. However, existing pruning methods largely rely on static or local token importance scores. This myopic approach fundamentally overlooks the sequential dependency of pruning decisions and fails to capture the interaction effects between pruning decisions across layers, often neglecting the global interactions between mask variables. To address this limitation, we propose **V-Pruner**, a fast and globally-informed token pruning framework for Vision Transformer. V-Pruner first leverages Fisher information to perform an initial assessment of token importance, providing a principled initial prior for pruning decisions. Building on this, V-Pruner introduces a Reinforcement Learning (RL) Proximal Policy Optimization (PPO) algorithm, refining token pruning into a global sequential decision process. The algorithm combines a composite reward signal that incorporates both model performance and computational cost to guide policy exploration, effectively evaluating the long-term impact of different pruning decision combinations on global model performance. Extensive experiments on ViT-L, DeiT-B, DeiT-S, and DeiT-T demonstrate that V-Pruner achieves a better balance between accuracy, GFLOPs, inference speed, and training time, surpassing existing mainstream ViT pruning algorithms in overall performance.

Code — https://github.com/Yaoguangzhen/V-Pruner

Introduction

Vision Language Models (VLMs) have gradually become the cornerstone of modern multimodal learning and demonstrate exceptional performance in numerous computer vision tasks (Laurençon et al. 2024; Lee et al. 2024; Yang et al. 2025; Xu et al. 2025; Wang et al. 2024; Chen et al. 2024). The key component driving this progress is the ViT (Dosovitskiy 2020), whose self-attention mechanism plays a crucial role in capturing long-range spatial relationships and building rich global visual representations. Despite the many advantages of the ViT architecture, it typically contains hundreds of millions to billions of parameters, leading to high computational costs during training and inference. This computational burden severely limits the practical deployment of ViTs in latency-sensitive or resource-constrained environments. Therefore, the development of effective compression techniques that can significantly reduce computation while maintaining accuracy has become an urgent need.

Among the various strategies for compressing ViT models (Ahmed et al. 2025; Ye et al. 2024; Yang et al. 2024; Rangwani et al. 2024; Choi and Kim 2025; Li et al. 2024), token pruning has emerged as a particularly promising technique (Yang et al. 2025; Liu et al. 2024; Zhang et al. 2024; Laurençon et al. 2024; Rahmanzadehgervi et al. 2024; Kim et al. 2024; Zhan et al. 2024). This technique removes spatial

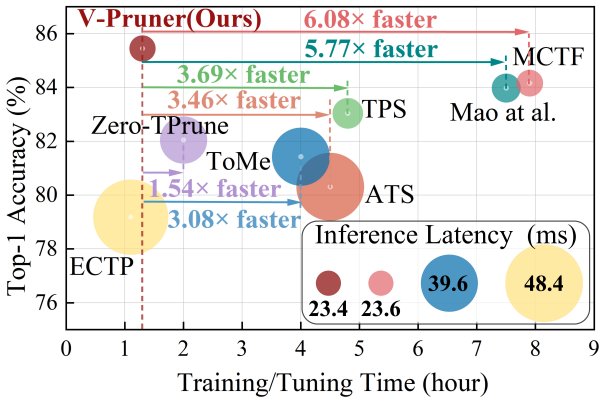

Figure 1: Under the 15 GFLOPs constraint for DeiT-B, V-Pruner not only outperforms existing mainstream ViT pruning methods in both accuracy and inference latency but also requires only 1.3 hours of training—approximately 1/6 of the time required by MCTF.

*Corresponding author: Guangzhen Yao
†These authors contributed equally.

tokens that contribute minimally to downstream predictions, directly reducing the number of elements processed by subsequent attention layers and MLP blocks, thereby significantly lowering FLOPs and memory usage.

Existing pruning algorithms can be broadly classified into two categories: retraining-based methods (Wei et al. 2023; Mao et al. 2025; Lee, Choi, and Kim 2024) and retraining-free methods (Bolya et al. 2023; Kim et al. 2022a; Yu et al. 2024; Wang, Dedhia, and Jha 2024). The former introduces learnable scoring modules (e.g., gating units) and relies on gradient-based optimization (fine-tuning) to learn pruning strategies, which, while yielding higher performance, also comes with high computational overhead. The latter, on the other hand, relies on carefully designed heuristic rules or static importance metrics (e.g., token similarity, class attention) to determine token significance in a one-shot manner, achieving extremely high time efficiency.

However, despite these methods having clear implementation paths, they share a common structural limitation in their decision-making mechanism: they overly depend on local significance scores. This strategy entirely overlooks the sequential dependency of pruning decisions, i.e., how the pruning masks chosen at one layer interact with deeper feature dependencies. This myopic decision-making process often leads to two globally suboptimal outcomes: either retaining locally important but globally redundant tokens, or prematurely discarding tokens that only reveal their importance after aggregation in deeper layers. This common flaw leads to poor global token configurations and fundamentally limits the performance ceiling of existing pruning methods.

To address this challenge, we propose **V-Pruner**, a fast and globally-informed token pruning framework for Vision Transformer. This framework innovatively redefines token pruning as a global sequential decision problem and adopts the PPO algorithm from RL. By exploring the end-to-end impact of different pruning decisions on model performance, V-Pruner achieves **global awareness**, effectively **avoiding the trap of local optima**. Specifically, V-Pruner first uses Fisher information to establish an initial estimate of token significance across layers, providing a principled, gradient-based perspective on token sensitivity. Building on this prior information, V-Pruner further deploys the PPO algorithm, transforming token pruning into a sequential decision process. During this process, the PPO agent explores based on a composite reward signal that combines model performance and computational cost, effectively evaluating the long-term impact of different pruning decision combinations on the global performance of the model. As shown in Figure 1, leveraging this **prior-guided** strategy, V-Pruner maintains high accuracy while achieving significant training efficiency. Compared with existing mainstream pruning methods, it attains a **6.08×** training speedup, demonstrating its fast characteristic.

The main contributions of this study are as follows:

- We propose V-Pruner, an RL-based adaptive token pruning method for ViTs. To the best of our knowledge, this represents the first application of RL to ViT token pruning, establishing a new paradigm for ViT architecture compression.
- V-Pruner formulates the pruning process as a sequential decision-making task. By optimizing a composite reward signal that balances model performance and computational cost via the PPO algorithm, it effectively explores and learns the long-term impact of various pruning decisions on global model performance.
- Extensive experiments on ViT-L, DeiT-B, DeiT-S, and DeiT-T demonstrate that V-Pruner achieves a superior trade-off among accuracy, GFLOPs, inference speed, and training time. Its overall performance outperforms existing mainstream pruning methods, fully validating its effectiveness and superiority.

Related Work

Reinforcement Learning Algorithms

Reinforcement Learning (RL) (Sutton, Barto et al. 1998) is a trial-and-error learning paradigm in which an agent continuously interacts with the environment and makes sequential decisions according to a policy to maximize long-term cumulative rewards. Typical RL algorithms can be broadly categorized into several types. Value-based methods, such as Q-learning (Watkins and Dayan 1992) and SARSA, estimate state-action value functions to guide policy improvement. Policy gradient methods, such as REINFORCE (Sutton, Barto et al. 1998), estimate policy gradients by sampling complete trajectories to update parameters. Actor-Critic methods, including A2C (Mnih et al. 2016) and DDPG (Lillicrap et al. 2015), combine policy optimization with value function estimation to achieve both efficiency and stability. Trust region-based methods, such as TRPO (Schulman et al. 2015) and PPO (Schulman et al. 2017), constrain the magnitude of policy updates to effectively enhance training stability and convergence.

Reinforcement Learning-based Token Pruning

In recent years, RL methods have been increasingly applied to model pruning and acceleration tasks (Yu, Mazaheri, and Jannesari 2022; Wu et al. 2020; Gangopadhyay, Dasgupta, and Dey 2023; Alwani, Wang, and Madhavan 2022; Graesser et al. 2022; Liu et al. 2019), demonstrating outstanding performance across multiple datasets. In CNN pruning, AMC (He et al. 2018) used RL to automatically search for compression strategies, significantly improving compression efficiency and reducing manual intervention; DRL (Chen, Chen, and Pan 2020) achieved efficient dynamic pruning by combining runtime and static channel importance; Wang et al. (Wang and Li 2022) introduced Monte Carlo Tree Search to enhance decision foresight, overcoming the short-sighted nature of traditional channel pruning. In Transformer pruning, G-Pruner (Yao et al. 2024) leveraged RL to improve the global adaptability of pruning policies, and LTP (Kim et al. 2022b) dynamically removed redundant tokens to reduce computational overhead. Building on these advancements, we propose V-Pruner, which introduces RL into ViT pruning. Our work investigates the applicability of RL methods for token pruning, providing new insights and a practical foundation for related research.

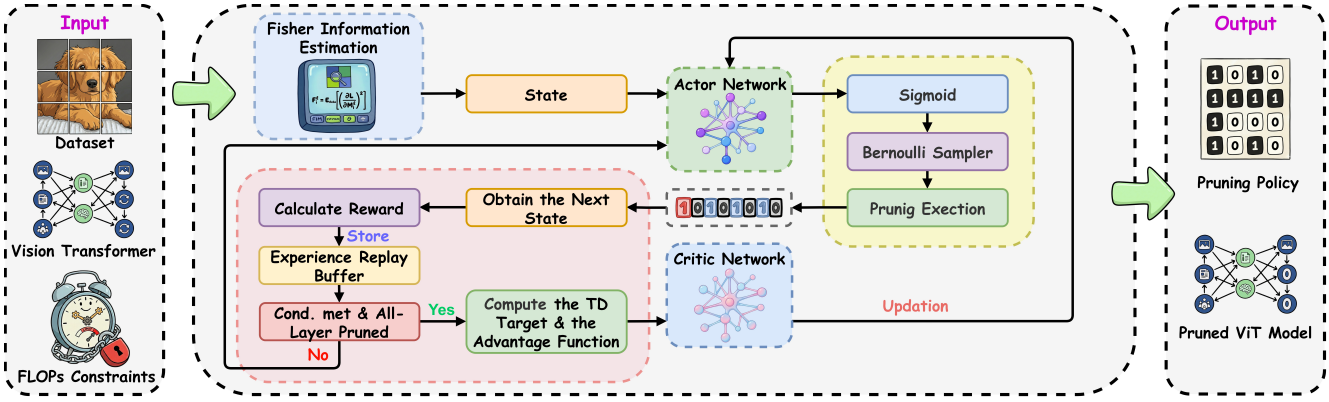

Figure 2: Overview of the V-Pruner framework. Adopting a prior-guided strategy, V-Pruner leverages Fisher information to estimate token importance as an informative prior, guiding the PPO agent to learn the globally optimal pruning mask.

Methodology

Framework Overview

As shown in Figure 2, V-Pruner takes a pre-trained ViT model, a target dataset, and FLOPs/latency constraints as input, and outputs a compressed ViT model ready for deployment. Unlike methods that rely solely on RL to search for pruning policies from scratch, V-Pruner adopts a **prior-guided** strategy. We first compute a reliable importance score vector for each token using Fisher information. Then, the RL agent treats this score vector as an informative prior (i.e., part of the state) to learn a globally optimal pruning mask. The overall execution process is detailed in Algorithm 1.

Token Importance Evaluation

We recognize that the combinatorial space of token masks is extremely large. Directly relying on RL to explore this high-dimensional discrete space from scratch incurs prohibitive computational costs. To alleviate this issue, we introduce a token importance initialization stage based on Fisher information, which significantly reduces the subsequent RL search complexity while maintaining effectiveness.

Specifically, we use Fisher information to compute gradient-related statistics, quantifying the sensitivity of each token (or its corresponding mask variable) to the overall model loss. This process provides a rich and efficient initialization point for subsequent RL, making policy optimization more stable and accelerating convergence.

In ViT, an input image x is divided into N non-overlapping patches of size $P \times P$. Each patch is flattened and linearly projected into a D-dimensional feature space, forming the initial token sequence:

$$z_0 = [x_{\text{class}}, e_1, e_2, \ldots, e_N] \in \mathbb{R}^{(N+1)\times D} \tag{1}$$

where x_{class} is a learnable classification token. After passing through multiple Transformer encoder layers, the updated sequence at layer l is represented as:

$$z_l = [x^l_{\text{class}}, e^l_1, e^l_2, \ldots, e^l_N] \tag{2}$$

To evaluate the importance of each token during initialization, we introduce a set of continuous mask variables for layer l:

$$M^l = [M^l_1, M^l_2, \ldots, M^l_N], \quad M^l_i \in [0,1] \tag{3}$$

where each M^l_i is associated with the corresponding non-class token e^l_i (e.g., via the element-wise product $M^l_i \cdot e^l_i$ for gating). Note that these continuous masks are only used for gradient computation and importance estimation; the final pruning decisions in the RL stage remain binary.

We characterize the influence of each token on the model loss L using Fisher information. Specifically, the importance of the i-th token in layer l is defined as the diagonal entry of the Fisher information matrix, which can be approximated by the expected squared gradient of the loss with respect to the mask variable:

$$F^l_i = \mathbb{E}_{\text{data}}\left[\left(\frac{\partial L}{\partial M^l_i}\right)^2\right] \tag{4}$$

where the gradient $\frac{\partial L}{\partial M^l_i}$ is computed via standard backpropagation and reflects the change in total loss when token e^l_i is slightly masked or removed.

After computing this expectation over a representative dataset, we obtain the importance vector:

$$F^l = [F^l_1, F^l_2, \ldots, F^l_N] \tag{5}$$

which serves as prior knowledge input to the RL agent, guiding it in exploring a globally optimal pruning policy. This initialization mechanism enables V-Pruner to maintain global search capability while significantly reducing the RL exploration space and accelerating policy convergence.

Defining Key Elements

In the RL framework for token pruning, the Actor-Critic model requires a precise definition of key elements to ensure effective learning and decision-making. The main elements are as follows:

• **State s_t**: At each time step t (corresponding to the l-th layer of ViT), the state s_t contains all the input information required for the agent's decision, defined as:

$$s^l_t = \{E^l, F^l, B^l\} \tag{6}$$

where $E^l = \{e_1^l, e_2^l, \ldots, e_N^l\}$ is the sequence of non-class tokens in layer l; $F^l = \{F_1^l, F_2^l, \ldots, F_N^l\}$ is the token importance vector precomputed during the initialization stage; and B^l is a scalar representing either the pruning budget or the layer index, providing the agent with global context.

• **Action a_t**: The action a_t is the decision generated by the policy π_θ after observing state s_t, i.e., the binary pruning mask for the current layer:

$$a_t^l = M^l = \{M_1^l, M_2^l, \ldots, M_N^l\}, \quad M_i^l \in \{0, 1\} \quad (7)$$

where $M_i^l = 1$ indicates that the i-th token is retained, and $M_i^l = 0$ indicates that the token is pruned. Unlike the continuous mask variables in the initialization stage, discrete decisions are used here to achieve final pruning.

• **Policy π_θ**: The policy function π_θ, parameterized by θ, defines a probability distribution over actions a_t given the state s_t:

$$\pi_\theta(a_t \mid s_t) = P(M^l \mid s_t^l; \theta) \quad (8)$$

PPO constrains the magnitude of policy updates to improve training stability and optimizes θ to maximize the expected cumulative reward, thereby balancing model performance and computational cost.

Reward Function Design

We model token pruning as an episodic task with sparse rewards. A complete episode contains L time steps ($t = 1, \ldots, L$), corresponding to the L layers of the ViT. In reinforcement learning, the reward function R_t quantifies the quality of the agent's action a_t at time step t. To guide the pruning policy more effectively, we explicitly distinguish between *immediate rewards* for intermediate steps and *terminal rewards* at the end of an episode.

Immediate Reward R_t ($t < L$). At each intermediate time step $t < L$ (i.e., non-final layers), the agent receives only an immediate reward R_t to provide fast local feedback. This reward consists of two components:

• **Exploration Reward R_{explore}**: To prevent the pruning policy from prematurely converging to suboptimal mask configurations, we introduce an intrinsic exploration reward based on the policy entropy H:

$$R_{\text{explore}} = w_e \cdot \frac{\mathcal{H}(\pi_\theta(\cdot \mid s_t))}{\mathcal{H}_{\max}} \quad (9)$$

where $\mathcal{H}_{\max}$ is the maximum entropy of the current action space for normalization. This term encourages the policy to maintain moderate uncertainty in state s_t and explore more potential mask configurations.

• **Layer-wise Budget Penalty P_l**: We assign a separate computation budget τ_l for each layer. A penalty is applied if the relative computation of layer l exceeds the threshold:

$$P_l = -w_{p_l} \cdot \max\left(0, \frac{\text{MACs}_p^l}{\text{MACs}_b^l} - \tau_l\right) \quad (10)$$

where MACs_p^l and MACs_b^l denote the MACs of the pruned and baseline models at layer l, respectively, and τ_l is the layer-level budget threshold.

Thus, for intermediate steps ($t < L$), the reward function is:

$$R_t = R_{\text{explore}} + P_l \quad (11)$$

Algorithm 1: Workflow for Globally-Informed Token Pruning

Input: Pre-trained ViT, Dataset, FLOPs/Latency Constraint, Experience buffer $\mathcal{D} \leftarrow \emptyset$, K epochs
Output: Pruned ViT Model
Initialization: Actor π_θ and Critic V_ω parameters, $\mathcal{D} \leftarrow \emptyset$ Compute Fisher importance F^l ▷ Using Eq. (5)
for *each training iteration* **do**
  `// Experience Collection Phase`
  for $j = 1...N$ *episodes* **do**
    `// Run one full episode (L layers)`
    for *each layer* $l = 1...L$ **do**
      Observe state $s_t^l = \{E^l, F^l, B^l\}$
      Generate action probabilities: $\mathbf{p} \leftarrow \pi_\theta(\cdot \mid s_t^l)$
      Sample pruning mask: $a_t^l \sim \text{Bernoulli}(\mathbf{p})$
      Apply mask a_t^l and get next state s_{t+1}^{l+1}
      if $l < L$ **then**
        Compute intermediate reward R_l
      end
      else
        // Terminal step $l = L$
        Compute terminal reward R_L
      end
      Store exp: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t^l, a_t^l, R_t, s_{t+1}^{l+1})\}$
    end
  end
  `// Policy Optimization Phase`
  for K *epochs* **do**
    Sample mini-batch $B \sim \mathcal{D}$
    for *each* $(s_t^l, a_t^l, R_t, s_{t+1}^{l+1}) \in B$ **do**
      Compute TD target y_l: ▷ Using Eq. (22)
      Compute advan $A(s_t, a_t)$: ▷ Using Eq. (24)
    end
    Update ω and θ ▷ Using Eq. (26) and (27)
  end
end
return Trained pruning policy π_{θ^*}

Terminal Reward R_t ($t = L$). Only at the end of an episode ($t = L$), when the agent has generated complete pruning masks $\{M^1, \ldots, M^L\}$ for all L layers, can we perform a full forward pass to evaluate the overall performance of the compressed model. The terminal reward consists of the following global terms along with the immediate reward of the current layer:

• **Accuracy Preservation Reward R_{acc}**: Encourages the pruned model to achieve accuracy close to or better than the baseline:

$$R_{\text{acc}} = w_a \cdot \frac{\text{Acc}_p}{\text{Acc}_b} \quad (12)$$

• **Information Retention Reward R_{fisher}**: Encourages the pruned model to retain more total Fisher information:

$$R_{\text{fisher}} = w_f \cdot \frac{F_p}{F_b} \quad (13)$$

where F_b is the total information of the baseline model, and F_p is the sum of precomputed Fisher scores of the retained

tokens in the pruned model:

$$F_p = \sum_{l=1}^{L} \sum_{i=1}^{N} M_i^l \cdot F_i^l \tag{14}$$

• **Global Efficiency Reward R_{macs}**: Rewards the reduction of overall computation:

$$R_{\text{macs}} = w_m \cdot \left(1 - \frac{\text{MACs}_p^{\text{total}}}{\text{MACs}_b^{\text{total}}}\right) \tag{15}$$

• **Global Budget Constraint P_g**: Applies a penalty if the total relative computation exceeds the global budget threshold τ_g:

$$P_g = -w_{p_g} \cdot \max\left(0, \frac{\text{MACs}_p^{\text{total}}}{\text{MACs}_b^{\text{total}}} - \tau_g\right) \tag{16}$$

Complete Reward Function R_t. In summary, the total reward received by the agent at $t = L$ is the sum of the terminal reward and the immediate reward for the current layer. The complete reward function R_t is defined as:

$$R_t = \begin{cases} R_{\text{explore}} + P_l & t < L \\ R_{\text{acc}} + R_{\text{fisher}} + R_{\text{macs}} + P_g + R_{\text{explore}} + P_L & t = L \end{cases} \tag{17}$$

All w_* are weight coefficients that balance different optimization objectives. In our experiments, we set the initial weights and thresholds as: the initial weights and thresholds are set as follows: $w_f = 1.2$, $w_a = 1.0$, $w_m = 0.7$, $w_e = 0.3$, $w_{p_l} = 10.0$, $\tau_l = 0.8$, $w_{p_g} = 5.0$, and $\tau_g = 0.75$.

Actor Network and Pruning Execution

The Actor network serves as the parameterized implementation of the pruning policy $\pi_\theta(a_t \mid s_t)$. Its goal is to generate pruning decisions that maximize long-term cumulative rewards, thereby balancing model performance and computational efficiency. At time step t, the network computes an independent retention probability for each non-class token e_i^t in the sequence based on the current state s_t. Specifically, the network first generates an unnormalized logit $f_\theta(s_t)^i$ for each token, which is then mapped to a retention probability via the Sigmoid activation function σ:

$$\pi_\theta(a_t^i = 1 \mid s_t) = \sigma(f_\theta(s_t)^i) \tag{18}$$

where $a_t^i = 1$ indicates that the i-th token is retained.

Based on these probabilities, we perform Bernoulli sampling to generate a binary decision for each token. Collectively, these decisions form a binary pruning mask $M^l \in \{0,1\}^N$, where N is the number of non-class tokens:

$$M_i^l \sim \text{Bernoulli}(\pi_\theta(a_t^i = 1 \mid s_t)) \tag{19}$$

with $M_i^l = 1$ indicating retention and $M_i^l = 0$ indicating pruning.

Finally, the generated pruning mask $M^l = [M_1^l, M_2^l, \ldots, M_N^l]$ is applied to the token sequence z_l of the current layer, producing the pruned sequence z_l':

$$z_l' = [x_{\text{class}}^l, M_1^l e_1^l, M_2^l e_2^l, \ldots, M_N^l e_N^l] \tag{20}$$

To preserve task-relevant global information, the classification token x_{class}^l is always retained and is not subject to pruning.

Critic Network Optimization

The Critic network estimates the state-value function $V_\omega(s_t)$, representing the expected cumulative reward from state s_t under the current policy π_θ. By evaluating the long-term value of a state, the Critic provides a stable learning signal for updating the Actor's policy and guides it toward actions with higher expected returns.

We train the Critic network by minimizing the temporal difference (TD) error, with the loss function defined as:

$$L(\omega) = \mathbb{E}_{s_t \sim D}\left[(y_t - V_\omega(s_t))^2\right] \tag{21}$$

where the target value y_t is the sum of the immediate reward R_t and the discounted value of the next state $V_\omega(s_{t+1})$:

$$y_t = R_t + \gamma V_\omega(s_{t+1}) \tag{22}$$

with D representing the state distribution sampled from the experience replay buffer, and γ the discount factor.

In practice, we approximate the loss function using mini-batch gradient descent over a batch of N samples:

$$L(\omega) \approx \frac{1}{N} \sum_{i=1}^{N} \left(y_i - V_\omega(s_i)\right)^2 \tag{23}$$

where y_i is the target value corresponding to state s_i, representing the expected cumulative reward from that state.

Pruning Algorithm Update

During training, we first compute the advantage function $A(s_t, a_t)$, which represents the relative desirability of selecting a specific action in the current state. The advantage function is defined as:

$$A(s_t, a_t) = y_t - V_\omega(s_t) \tag{24}$$

where $y_t = R_t + \gamma V_\omega(s_{t+1})$ is the target value.

Based on this advantage estimate, the PPO algorithm optimizes the Actor network by maximizing the following clipped objective function:

$$J^{\text{CLIP}}(\theta) = \mathbb{E}_{(s_t, a_t) \sim \pi_{\text{old}}}\Big[\min\left(r_t(\theta)A(s_t, a_t),\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A(s_t, a_t)\right)\Big] \tag{25}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\text{old}}(a_t|s_t)}$ is the probability ratio between the new policy and the old policy used to collect the current batch of data. The clip function constrains this ratio within $[1-\epsilon, 1+\epsilon]$ to prevent overly large policy updates and ensure training stability.

We update the Actor network parameters θ using gradient ascent:

$$\theta \leftarrow \theta + \alpha^A \nabla_\theta J^{\text{CLIP}}(\theta) \tag{26}$$

where α^A is the learning rate of the Actor network.

For the Critic network parameters ω, we minimize the value function loss using gradient descent:

$$\omega \leftarrow \omega - \alpha^C \nabla_\omega L(\omega) \tag{27}$$

where α^C is the learning rate of the Critic network.

Method	DeiT-T (2GFLOPS) Top-1 Acc.	Params	Throughput	Inf. Latency	Speedup	Method	DeiT-S (5GFLOPS) Top-1 Acc.	Params	Throughput	Inf. Latency	Speedup
ECTP	66.23%	5.7 M	2489 img/s	4.3ms	1.14×	ECTP	73.94%	22.1M	1176 img/s	8.5ms	1.14×
ATS	67.92%	5.7 M	2650 img/s	4.1ms	1.20×	ATS	74.56%	22.1M	1382 img/s	7.8ms	1.24×
ToMe	68.34%	5.7 M	2741 img/s	3.8ms	1.29×	ToMe	75.90%	22.1M	1552 img/s	6.9ms	1.41×
Zero-TPrune	69.57%	5.7 M	2455 img/s	4.0ms	1.23×	Zero-TPrune	76.28%	22.1M	1415 img/s	7.5ms	1.29×
TPS	70.46%	5.7 M	2710 img/s	3.7ms	1.32×	TPS	77.28%	22.1M	1428 img/s	6.8ms	1.43×
Mao et al.	70.95%	5.7 M	2730 img/s	3.8ms	1.29×	Mao et al.	78.63%	22.1M	1462 img/s	7.1ms	1.37×
MCTF	71.55%	5.7 M	2777 img/s	3.6ms	1.36×	MCTF	79.05%	22.1M	1558 img/s	6.5ms	1.49×
V-Pruner	**71.90%**	**5.7 M**	**2845 img/s**	**3.5ms**	**1.40×**	**V-Pruner**	**79.93%**	**22.1M**	**1847 img/s**	**5.5ms**	**1.76×**
Method	**DeiT-B (15GFLOPS) Top-1 Acc.**	**Params**	**Throughput**	**Inf. Latency**	**Speedup**	**Method**	**ViT-L (35GFLOPS) Top-1 Acc.**	**Params**	**Throughput**	**Inf. Latency**	**Speedup**
ECTP	79.18%	86.6M	526 img/s	48.4ms	1.15×	ECTP	82.47%	304M	180 img/s	92.1ms	1.14×
ATS	80.31%	86.6M	573 img/s	44.6ms	1.25×	ATS	83.55%	304M	201 img/s	84.7ms	1.24×
ToMe	81.43%	86.6M	635 img/s	39.6ms	1.40×	ToMe	84.08%	304M	226 img/s	75.0ms	1.40×
Zero-TPrune	82.05%	86.6M	745 img/s	34.1ms	1.63×	Zero-TPrune	84.82%	304M	210 img/s	80.8ms	1.30×
TPS	83.04%	86.6M	802 img/s	25.6ms	2.17×	TPS	86.73%	304M	235 img/s	71.4ms	1.47×
Mao et al.	83.98%	86.6M	816 img/s	24.8ms	2.24×	Mao et al.	87.75%	304M	232 img/s	73.9ms	1.42×
MCTF	84.16%	86.6M	838 img/s	23.6ms	2.35×	MCTF	87.93%	304M	248 img/s	68.8ms	1.53×
V-Pruner	**85.45%**	**86.6M**	**857 img/s**	**23.4ms**	**2.37×**	**V-Pruner**	**89.27%**	**304M**	**252 img/s**	**67.7ms**	**1.55×**

Table 1: Quantitative comparison with state-of-the-art pruning methods on DeiT and ViT models.

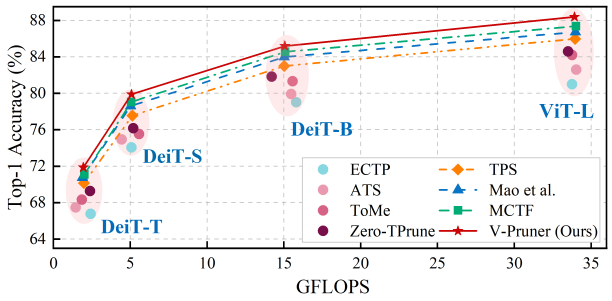

Figure 3: Performance comparison of V-Pruner and various existing pruning algorithms on ViT-L, DeiT-B, DeiT-S, and DeiT-T models based on Top-1 accuracy and GFLOPs.

Experiments

Experimental Setup

Pretrained models and Evaluation Metrics. To comprehensively evaluate the performance of different token pruning methods, we select a series of ViT variants with varying scales as benchmark models, including ViT-L (Dosovitskiy 2020), DeiT-B (Touvron et al. 2021), DeiT-S, and DeiT-T. The image classification experiments are conducted on the standard ImageNet-1K dataset (Deng et al. 2009). All experiments are performed on a single NVIDIA A100 GPU. In the evaluation, we compare different methods across multiple metrics, including Top-1 accuracy, GFLOPs, image throughput, inference latency, and training time.

Comparison with State-of-the-art Methods

Competitors. We benchmark V-Pruner against representative baselines, categorized into training-free methods (ATS (Kim et al. 2022a), ToMe (Bolya et al. 2023), RCTP (Yu et al. 2024), Zero-TPrune (Wang, Dedhia, and Jha 2024)) and retraining-based methods (TPS (Wei et al. 2023), Mao et al. (Mao et al. 2025), MCTF (Lee, Choi, and Kim 2024)). All baselines are re-implemented under a unified setup to ensure fair comparison.

Accuracy and GFLOPs. As shown in Figure 3, V-Pruner consistently achieves the highest Top-1 accuracy across all budgets. Under the same GFLOPs constraint, V-Pruner outperforms training-free methods (ATS, ToMe, RCTP, Zero-TPrune) by **2–6%** and retraining-based methods (TPS, Mao et al., MCTF) by ∼1%. These results confirm that our RL framework can more accurately assess token importance and make more effective dynamic pruning decisions.

Inference Latency. As shown in Table 1, V-Pruner demonstrates excellent inference efficiency under different GFLOPs constraints. For example, under the 15 GFLOPs constraint for the DeiT-B model, V-Pruner successfully compresses it into a highly efficient lightweight model, achieving an inference latency of only **23.4 ms (2.37× speedup)** while maintaining a high Top-1 accuracy of **85.45%**. Its overall performance surpasses all compared methods. In particular, compared with training-free approaches such as ToMe (39.6 ms) and ECTP (48.4 ms), V-Pruner reduces the latency by 40.9% and 51.7%, respectively. These results fully demonstrate that V-Pruner effectively preserves critical information while significantly improving computational efficiency, achieving an optimal balance between model accuracy and inference speed.

Training Time. As shown in Figure 1, under the 15 GFLOPs constraint for DeiT-B, V-Pruner requires only **1.3 hours** for training, approximately **1/6** of the time needed by MCTF and Mao et al. Although slightly longer than the training-free ECTP (1.1 hours), V-Pruner significantly outperforms it in terms of performance. This demonstrates that the prior-guided mechanism effectively reduces the complexity of RL-based policy search. While introducing reinforcement learning for policy exploration incurs some additional overhead, the total training time remains only a matter of tens of minutes. Compared to end-to-end fine-tuning, which typically takes several days, this time cost is almost negligible. Crucially, this limited time investment yields substantial performance gains, fully reflecting V-Pruner's excellent balance between computational efficiency and model accuracy.

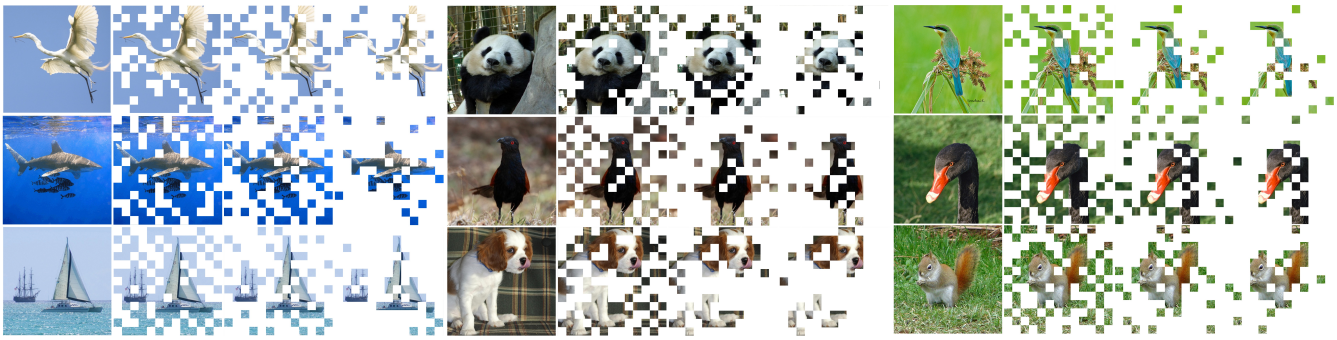

Figure 4: Visualization examples of the token pruning process of V-Pruner on ImageNet.

Backbone	GFLOPs	Top-1 Accuracy (%) of Different Policies			
		Random	REINFORCE	A2C	PPO (Ours)
DeiT-T	2.0	65.83	67.92	70.68	**71.90**
	2.5	68.02	70.25	73.12	**74.11**
DeiT-S	3.0	69.05	71.28	73.89	**75.36**
	5.0	73.24	75.46	78.25	**79.93**
DeiT-B	10.0	75.84	78.15	81.12	**82.75**
	15.0	78.95	81.28	84.25	**85.45**
ViT-L	30.0	80.04	82.35	85.35	**87.95**
	35.0	82.16	84.45	87.45	**89.27**

Table 2: Ablation study of different optimization policies.

Ablation Studies

We validate PPO against Random Pruning, REINFORCE, and A2C. As shown in Table 2, PPO exhibits clear advantages across all evaluation metrics. In contrast, REINFORCE suffers from high-variance gradient estimates and fails to achieve stable and efficient convergence, while A2C is susceptible to performance fluctuations caused by destructive updates during training. For example, on ViT-L (35.0 GFLOPs), PPO achieves a Top-1 accuracy of **89.27%**, significantly surpassing A2C's 87.45% and REINFORCE's 84.45%, highlighting the stability of PPO in such sequential decision-making tasks.

Visualization of the Pruning Process

Figure 4 visualizes the pruning progression on DeiT-B, displaying the remaining tokens after Blocks 3, 6, and 9. Taking the egret as an example, V-Pruner exhibits a clear coarse-to-fine refinement: shallow layers (Block 3) rapidly remove redundant background regions, mid-level layers (Block 6) concentrate on the main object, and deeper layers (Block 9) retain only the most discriminative parts (e.g., beak, claws, wings). This progression demonstrates that V-Pruner effectively selects informative tokens, achieving an excellent balance between efficiency and accuracy.

Downstream Experiments

To verify the generality of V-Pruner, we systematically evaluate its performance on multiple downstream classification and segmentation benchmark tasks using DeiT-B as the backbone. As shown in Table 3 and Table 4, the results further demonstrate the efficiency and applicability of V-Pruner.

Dataset	DeiT-B	+ToMe	+Zero-TPrune	+MCTF	+V-Pruner
CIFAR-100	99.5	97.2	97.9	98.3	**98.8**
Flowers	98.7	96.3	96.1	96.8	**97.2**
Pets	91.4	88.3	89.2	88.5	**89.8**
DTD	75.5	72.0	72.3	72.8	**73.1**
Indoor67	78.4	75.2	75.2	76.0	**76.4**
CUB200	79.7	76.3	76.0	76.4	**77.1**
Aircrafts	80.5	78.1	78.2	77.8	**78.9**
Cars	92.5	88.5	88.7	89.4	**89.6**

Table 3: Zero-shot accuracy (%) on downstream image classification tasks.

Model	Pascal VOC mIoU	COCO mIoU	GFLOPs
DeiT-B	79.5	59.9	17.6
+ECTP	77.2	57.8	10.9
+ATS	77.8	58.2	10.3
+ToMe	78.0	58.4	10.2
+Zero-TPrune	78.2	58.6	10.1
+TPS	78.4	58.8	10.2
+Mao et al.	77.8	58.2	10.7
+MCTF	78.1	58.7	10.5
+V-Pruner (Ours)	**78.6**	**59.0**	**10.1**

Table 4: Object segmentation performance (mIoU %) on Pascal VOC and COCO datasets using the DeiT-B backbone.

Conclusion

This paper proposes V-Pruner, a fast and globally-informed token pruning framework for Vision Transformer, aiming to address the limitations of existing methods that overly rely on local scores while ignoring inter-layer decision interactions. The core innovation of V-Pruner lies in reformulating pruning as a sequential decision-making process: it first uses Fisher information for initialization and then introduces the PPO algorithm, guided by a composite reward signal that combines model performance and computational cost. This enables V-Pruner to explore and learn the long-term impact of different pruning decisions on global model performance. Extensive experiments on ViT-L, DeiT-B, DeiT-S, and DeiT-T demonstrate that V-Pruner achieves a superior balance among accuracy, GFLOPs, inference latency, and training time, consistently outperforming all mainstream pruning methods and validating its effectiveness and superiority.

References

Ahmed, S.; Al Arafat, A.; Najafi, D.; Mahmood, A.; Rizve, M. N.; Al Nahian, M.; Zhou, R.; Angizi, S.; and Rakin, A. S. 2025. DeepCompress-ViT: Rethinking Model Compression to Enhance Efficiency of Vision Transformers at the Edge. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 30147–30156.

Alwani, M.; Wang, Y.; and Madhavan, V. 2022. Decore: Deep compression with reinforcement learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12349–12359.

Bolya, D.; Fu, C.-Y.; Dai, X.; Zhang, P.; Feichtenhofer, C.; and Hoffman, J. 2023. Token Merging: Your ViT But Faster. In *ICLR*.

Chen, B.; Xu, Z.; Kirmani, S.; Ichter, B.; Sadigh, D.; Guibas, L.; and Xia, F. 2024. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14455–14465.

Chen, J.; Chen, S.; and Pan, S. J. 2020. Storage efficient and dynamic flexible runtime channel pruning via deep reinforcement learning. *Advances in neural information processing systems*, 33: 14747–14758.

Choi, D.; and Kim, H. 2025. GradQ-ViT: Robust and Efficient Gradient Quantization for Vision Transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 16019–16027.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Dosovitskiy, A. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Gangopadhyay, B.; Dasgupta, P.; and Dey, S. 2023. Safety aware neural pruning for deep reinforcement learning (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 16212–16213.

Graesser, L.; Evci, U.; Elsen, E.; and Castro, P. S. 2022. The state of sparse training in deep reinforcement learning. In *International Conference on Machine Learning*, 7766–7792. PMLR.

He, Y.; Lin, J.; Liu, Z.; Wang, H.; Li, L.-J.; and Han, S. 2018. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European conference on computer vision (ECCV)*, 784–800.

Kim, M.; Gao, S.; Hsu, Y.-C.; Shen, Y.; and Jin, H. 2024. Token fusion: Bridging the gap between token pruning and token merging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 1383–1392.

Kim, S.; Shen, S.; Thorsley, D.; Gholami, A.; Kwon, W.; Hassoun, J.; and Keutzer, K. 2022a. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 784–794.

Kim, S.; Shen, S.; Thorsley, D.; Gholami, A.; Kwon, W.; Hassoun, J.; and Keutzer, K. 2022b. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, 784–794.

Laurençon, H.; Tronchon, L.; Cord, M.; and Sanh, V. 2024. What matters when building vision-language models? *Advances in Neural Information Processing Systems*, 37: 87874–87907.

Lee, S.; Choi, J.; and Kim, H. J. 2024. Multi-criteria token fusion with one-step-ahead attention for efficient vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15741–15750.

Lee, T.; Tu, H.; Wong, C. H.; Zheng, W.; Zhou, Y.; Mai, Y.; Roberts, J. S.; Yasunaga, M.; and Yao, N. I. P. S. 2024. Vhelm: A holistic evaluation of vision language models. 37: 140632–140666.

Li, Y.; Xu, S.; Lin, M.; Cao, X.; Liu, C.; Sun, X.; and Zhang, B. 2024. Bi-vit: Pushing the limit of vision transformer quantization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 3243–3251.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Liu, H.; Li, C.; Li, Y.; and Lee, Y. J. 2024. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 26296–26306.

Liu, Z.; Mu, H.; Zhang, X.; Guo, Z.; Yang, X.; Cheng, K.-T.; and Sun, J. 2019. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF international conference on computer vision*, 3296–3305.

Mao, J.; Shen, Y.; Guo, J.; Yao, Y.; Hua, X.; and Shen, H. 2025. Prune and merge: Efficient token compression for vision transformer with spatial information preserved. *IEEE Transactions on Multimedia*.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 1928–1937. PmLR.

Rahmanzadehgervi, P.; Bolton, L.; Taesiri, M. R.; and Nguyen, A. T. 2024. Vision language models are blind. In *Proceedings of the Asian Conference on Computer Vision*, 18–34.

Rangwani, H.; Mondal, P.; Mishra, M.; Asokan, A. R.; and Babu, R. V. 2024. Deit-lt: Distillation strikes back for vision transformer training on long-tailed datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 23396–23406.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *International conference on machine learning*, 1889–1897. PMLR.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sutton, R. S.; Barto, A. G.; et al. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; and Jegou, H. 2021. Training Data-Efficient Image Transformers & Distillation through Attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *PMLR*, 10347–10357.

Wang, H.; Dedhia, B.; and Jha, N. K. 2024. Zero-TPrune: Zero-shot token pruning through leveraging of the attention graph in pre-trained transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16070–16079.

Wang, J.; Ming, Y.; Shi, Z.; Vineet, V.; Wang, X.; Li, S.; and Joshi, N. 2024. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. *Advances in Neural Information Processing Systems*, 37: 75392–75421.

Wang, Z.; and Li, C. 2022. Channel pruning via lookahead search guided reinforcement learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2029–2040.

Watkins, C. J.; and Dayan, P. 1992. Q-learning. *Machine learning*, 8: 279–292.

Wei, S.; Ye, T.; Zhang, S.; Tang, Y.; and Liang, J. 2023. Joint token pruning and squeezing towards more aggressive compression of vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2092–2101.

Wu, C.; Cui, Y.; Ji, C.; Kuo, T.-W.; and Xue, C. J. 2020. Pruning deep reinforcement learning for dual user experience and storage lifetime improvement on mobile devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(11): 3993–4005.

Xu, G.; Jin, P.; Wu, Z.; Li, H.; Song, Y.; Sun, L.; and Yuan, L. 2025. Llava-cot: Let vision language models reason step-by-step. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2087–2098.

Yang, S.; Chen, Y.; Tian, Z.; Wang, C.; Li, J.; Yu, B.; and Jia, J. 2025. Visionzip: Longer is better but not necessary in vision language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 19792–19802.

Yang, Z.; Li, Z.; Zeng, A.; Li, Z.; Yuan, C.; and Li, Y. 2024. Vitkd: Feature-based knowledge distillation for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1379–1388.

Yao, G.; Wang, Y.; Xu, H.; Zhang, L.; and MiaoQI, M. 2024. Global-pruner: A stable and efficient pruner for retraining-free pruning of encoder-based language models. In *Proceedings of the 28th Conference on Computational Natural Language Learning*, 46–55.

Ye, H.; Yu, C.; Ye, P.; Xia, R.; Tang, Y.; Lu, J.; Chen, T.; and Zhang, B. 2024. Once for both: Single stage of importance and sparsity search for vision transformer compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5578–5588.

Yu, S.; Mazaheri, A.; and Jannesari, A. 2022. Topology-aware network pruning using multi-stage graph embedding and reinforcement learning. In *International conference on machine learning*, 25656–25667. PMLR.

Yu, Y.-C.; Weng, M.-C.; Lin, M.-G.; and Wu, A.-Y. A. 2024. Retraining-free Constraint-aware Token Pruning for Vision Transformer on Edge Devices. In *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, 1–5. IEEE.

Zhan, Z.; Kong, Z.; Gong, Y.; Wu, Y.; Meng, Z.; Zheng, H.; Shen, X.; Ioannidis, S.; Niu, W.; Zhao, P.; et al. 2024. Exploring token pruning in vision state space models. *Advances in Neural Information Processing Systems*, 37: 50952–50971.

Zhang, J.; Huang, J.; Jin, S.; and Lu, S. 2024. Vision-language models for vision tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 46(8): 5625–5644.