

Scaling Vision Transformers Without Training: Prune-and-Merge Token Compression*

Vatsal

Department of Mathematics
Indian Institute of Technology Roorkee
vatsal@ma.iitr.ac.in

Soumil Malhotra

Department of Mathematics
Indian Institute of Technology Roorkee
soumil_m@ma.iitr.ac.in

Priyanshu Chahal

Department of Mathematics
Indian Institute of Technology Roorkee
priyanshu@ma.iitr.ac.in

Sanjeev Kumar

Department of Mathematics
Indian Institute of Technology Roorkee
sanjeev.kumar@ma.iitr.ac.in

Abstract

Vision Transformers achieve strong performance but incur high computational costs from quadratic token interactions. Training-free token reduction methods improve efficiency, but their robustness remains underexplored. We propose Prune-and-Merge ViT, a training-free framework for off-the-shelf models. It combines class-attention-based pruning with similarity-based merging to reduce redundancy while preserving semantic information. To evaluate robustness we introduce two metrics. Relative Accuracy Drop (RAD) measures relative accuracy degradation under transformations, and Prediction Flip Rate (PFR) quantifies prediction consistency. On pretrained ViT and DeiT, Prune-and-Merge reduces computation competitively while maintaining robustness and better accuracy than existing token reduction methods. Our analysis suggests that hybrid token reduction does not introduce hallucination-like behavior and preserves prediction stability for common image transformations.

1 Introduction

Vision Transformers (ViTs) forgo convolutional inductive biases and treat an image as a sequence of visual tokens [4]. Non-overlapping patches are projected into N tokens and passed, with a learnable class ([CLS]) token, through Transformer blocks. The token count is constant across layers, so cost scales quadratically, $O(N^2)$.

To address this issue, token pruning removes redundant tokens and merging aggregates similar ones to cut computation [2, 8]. Many such methods are training-free and apply directly to pretrained models. While such approaches improve efficiency, they may introduce instability under input transformations, potentially affecting prediction consistency. Aggressive reduction at lower keep rates can also degrade performance, as fewer tokens remain to capture essential visual information. This motivates a systematic analysis of prediction stability under aggressive reduction.

Existing approaches apply pruning or merging in isolation, which causes semantic loss or hallucination-like errors under input transformations. We address this with Prune-and-Merge ViT (Figure 1), a plug-and-play, training-free framework that combines class-attention-based pruning with similarity-driven merging for off-the-shelf ViT [4] and DeiT [13]. The first stage discards low-importance tokens via top- k selection on class-attention scores [8], which filters noise while

*Official Repository

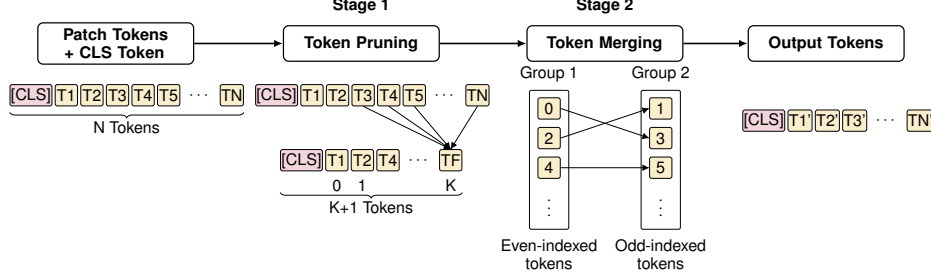


Figure 1: Overview of the Prune-and-Merge ViT architecture. The framework processes tokens through a dual-stage reduction at layers $l \in \{3, 6, 9\}$. Stage 1 (pruning) filters out background tokens based on class-attention scores k_p , while Stage 2 (merging) aggregates redundant salient tokens using bipartite matching k_m .

retaining salient features. Subsequently, a merging stage aggregates the remaining redundant tokens by bipartite matching on cosine similarities [2], preserving the semantic manifold in a compact bottleneck $\mathbf{T}' \in \mathbb{R}^{N' \times D}$, where $N' \ll N$. Unlike single-strategy methods, our framework controls the token budget finely: adjusting the pruning ratio and the merging threshold preserves semantic stability.

To evaluate robustness we introduce two metrics: Relative Accuracy Drop (RAD), which measures relative accuracy degradation under input transformations, and Prediction Flip Rate (PFR), which quantifies prediction consistency between original and transformed inputs. Experiments on pretrained ViT and DeiT show that Prune-and-Merge ViT competitively reduces computation while maintaining robustness and achieving strong accuracy. Our analysis further shows that hybrid token reduction preserves prediction stability under common image transformations.

Our contributions are summarized as follows:

- Prune-and-Merge ViT, a training-free hybrid token reduction framework.
- An analysis showing that hybrid pruning and merging preserve stability while reducing computation.
- RAD and PFR, which evaluate robustness and prediction stability under semantics-preserving transformations.

2 Related work

ViT [4] and DeiT [13] keep the sequence length fixed, so inference stays costly. DynamicViT [12] learns a pruning module, but it has to be trained. EViT [8] and Top-K need no training. They rank patches by class-attention and discard the rest, while V-Pruner [15] ranks globally rather than locally, and Zhan et al. [17] carry the paradigm to state-space models.

ToMe [2] fuses similar tokens by bipartite matching instead, which shortens the sequence physically rather than masking it. The same mechanism accelerates Stable Diffusion [1], and Haurum et al. [6] find that ToMe works best when at least half the token budget is kept. Spatial-preserving hybrids [5, 9] enhance locality and VisionTrim [16] favors throughput, but neither line offers robustness guarantees, and hybrids of the two families remain underexplored.

We address that gap in the plug-and-play spirit of ToCom [7] and its vision-language extensions [18], where transfer to a narrow subset such as ImageNet-100 needs careful alignment [11]; see Section A.

3 Method

3.1 Framework overview

We propose Prune-and-Merge ViT (see Figure 1), a dual-stage, training-free framework. It treats token reduction as sequential filtering and compression. Given N input tokens $\mathbf{x}$, we apply a hierarchical

reduction at layers $l \in \{3, 6, 9\}$ [6]. This schedule preserves high-resolution spatial information in early layers before progressively reducing the sequence length in deeper blocks.

The composite keep rate $K = k_p \cdot k_m$ governs reduction intensity at each stage. Here, $k_p \in (0, 1]$ is the pruning keep rate, the fraction of tokens retained according to semantic importance; pruning thus acts as a semantic filter. Conversely, $k_m \in (0, 1]$ is the merging keep rate, the fraction remaining after bipartite matching of the pruned subset. This multiplicative form allows fine-grained control over the total token budget while ensuring the reduction scales proportionally with the remaining sequence length.

3.2 Stage 1: class-attention pruning

Stage 1 retains semantically important tokens using the [CLS] token’s class-attention, following EViT [8]. Multi-head self-attention over input tokens $\mathbf{x} \in \mathbb{R}^{B \times N \times C}$ produces an attention map $\mathbf{A} \in \mathbb{R}^{B \times H \times N \times N}$. We measure importance by [CLS] attention to the $P = N - n_s$ patch tokens,

$$\mathbf{A}_{\text{cls}} = \mathbf{A}[:, :, 0, n_s:] \in \mathbb{R}^{B \times H \times P}, \quad \bar{\mathbf{a}} = \frac{1}{H} \sum_{h=1}^H \mathbf{A}_{\text{cls}}^{(h)} \in \mathbb{R}^{B \times P}, \quad (1)$$

where n_s counts the special tokens. The top- k attentive tokens are selected using k_p ,

$$k = \max(1, \lceil k_p \cdot P \rceil), \quad \mathcal{I}_{\text{att}} = \text{TopK}(\bar{\mathbf{a}}, k). \quad (2)$$

To prevent total loss of background context, we aggregate the remaining inattentive tokens $\mathcal{I}_{\text{inatt}}$ into one representative token $\mathbf{x}_{\text{fused}}$ by attention-weighted averaging,

$$\mathbf{x}_{\text{fused}} = \sum_{i \in \mathcal{I}_{\text{inatt}}} \bar{a}_i \mathbf{x}_i. \quad (3)$$

The resulting sequence $\mathbf{x}_{\text{stage1}} = [\mathbf{x}_{\mathcal{I}_{\text{att}}}, \mathbf{x}_{\text{fused}}] \in \mathbb{R}^{B \times (k+1) \times C}$ concentrates computation on salient regions while preserving global context.

3.3 Stage 2: similarity-based token merging

The Stage 1 sequence of length $N_p = k + 1$ undergoes bipartite merging inspired by ToMe [2]. We partition its tokens into two disjoint sets $\mathcal{A}$ and $\mathcal{B}$ of roughly equal size, then match them by cosine similarity between the key features $\mathbf{k}$. Every token in $\mathcal{A}$ draws exactly one edge to its most similar counterpart in $\mathcal{B}$. The keep rate k_m governs intensity through the number of merges r ,

$$r = \lfloor (1 - k_m) \cdot N_p \rfloor. \quad (4)$$

By retaining the r most similar edges, we ensure that only the most redundant features are fused.

Token size tracking and proportional attention. As tokens merge, we track a token size vector $\mathbf{s}$, where s_i counts the original patches a token represents [2]. We merge tokens i and j by a weighted average,

$$\mathbf{x}_{\text{merged}} = \frac{s_i \mathbf{x}_i + s_j \mathbf{x}_j}{s_i + s_j}. \quad (5)$$

To keep the correct softmax distribution in later layers, we apply proportional attention [2],

$$\mathbf{A} = \text{softmax} \left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}} + \log \mathbf{s} \right). \quad (6)$$

The $\log \mathbf{s}$ term in Eq. (6) ensures that a merged token exerts the same influence as if its patches were still present as separate, identical copies. This dual-stage consolidation keeps the model expressive and accurate even under high compression.

4 Experiments

4.1 Implementation details

We test ViT and DeiT at Tiny, Small and Base scales with official pretrained weights, on the ImageNet-100 validation set (an ImageNet-1K [3] subset) under semantics-preserving transformations. Prune-and-Merge blocks sit at layers $l \in \{3, 6, 9\}$. In Stage 1 we average [CLS]-token attention over all

H heads to score patch importance [8]. Stage 2 and all ToMe-based baselines use proportional attention [2]. We compare against the raw backbone, Random Drop, Top-K, EViT [8] and ToMe [2]. All are scored at composite keep rates $K \in \{0.25, 0.5, 0.7, 0.9\}$ by top-1 accuracy, GFLOPs, latency (ms/img) and throughput (imgs/sec), under identical hardware and software (Section D).

4.2 Results and discussion

Accuracy-efficiency trade-offs. While our framework maintains a computational profile competitive with EViT and ToMe, it exhibits higher top-1 accuracy across all architectural scales (Tables 1 and 3). This performance gap is most pronounced in aggressive compression regimes (e.g., $K = 0.25$), representing a marginal but consistent improvement over standalone techniques. This suggests that the interplay between pruning and merging preserves the representational power of the backbones even under significant reduction.

Controlled aggression. A primary contribution of this work is the introduction of ‘Controlled Aggression’ within the reduction pipeline. Unlike fixed-strategy baselines, our framework lets the two stages be balanced dynamically. At low keep rates, prioritizing aggressive merging over pruning yields higher representational stability (Table 2). By shifting the reduction burden from Stage 1 (noise removal) to Stage 2 (information fusion) as K decreases, we mitigate the ‘cliff-edge’ accuracy degradation typically observed in pure-pruning baselines. Critical semantic tokens are fused rather than discarded, which maintains a more coherent global context.

4.3 Stability and robustness analysis

We use two metrics to distinguish raw classification accuracy from the stability of the model’s decision-making process. The Relative Accuracy Drop (RAD),

$$\text{RAD} = \frac{\text{Acc}(Y, T(x))}{\text{Acc}(Y, x)} - \frac{\text{Acc}(X, T(x))}{\text{Acc}(X, x)}, \quad (7)$$

measures the difference in top-1 accuracy ratios between frameworks X and Y under a semantics-preserving transformation T of input x . The Prediction Flip Rate (PFR),

$$\text{PFR} = \frac{1}{N} \sum_{i=1}^N \mathbb{I}[f(x_i) \neq f(T(x_i))], \quad (8)$$

averages the indicator function $\mathbb{I}$ over N samples. Both quantify hallucination H , a lack of faithfulness between a model’s representation and its prediction. We measure it as the flip-rate deviation $H = \Delta\text{PFR} = |\text{PFR}_{\text{raw}} - \text{PFR}_{\text{hybrid}}|$, where PFR_{raw} is the backbone flip rate without token reduction. A ΔPFR closer to zero indicates higher robustness against hallucinations (Section E).

Robustness to hallucinations. We benchmark ΔPFR (Table 4) for ViT-Tiny and ViT-Base over Rotation, Color Jitter and Horizontal Flip. ViT-Base is significantly more stable than ViT-Tiny at $K = 0.25$. This suggests that its greater parameter density provides more stable ‘anchor’ tokens, allowing both stages to operate without disrupting the core decision boundary (Section E).

Manifold integrity via RAD. $\text{RAD} \approx 0$ suggests no more hallucination than the baseline, whereas a positive value indicates a higher rate. Against the raw ViT, RAD stays near zero or slightly positive (Table 5), so the hybrid model preserves semantic faithfulness to the pretrained backbone. Negative values against other reduction frameworks indicate lower hallucination rates. The ToMe-versus-hybrid comparison is undefined at $K = 0.25$ (a dash in Table 5), because standalone merging cannot exceed 50% reduction while keeping a complete bipartite graph. Our framework, in contrast, keeps RAD stable across the entire spectrum of K . This validates the utility of Stage 1 pruning in stabilizing the token set and removing noise before fusion occurs in Stage 2.

5 Conclusion

We introduced Prune-and-Merge ViT, a dual-stage token reduction framework designed to optimize the trade-off between inference efficiency and representational faithfulness. Under ‘Controlled

Aggression’ it transitions from top- k pruning in the early stages to bipartite matching for information fusion in deeper layers. On ImageNet-100 it significantly outperforms standalone pruning and merging baselines, achieving a $3\times$ speedup on ViT-Base at aggressive keep rates while maintaining superior accuracy (Tables 1 and 7). Furthermore, RAD and PFR confirm that the hybrid approach mitigates hallucinations and maintains a more robust attention manifold than fixed-strategy alternatives. These results validate that a hardware-aware, hybrid reduction paradigm is essential for scaling ViTs to resource-constrained environments without sacrificing reliability.

References

- [1] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion, 2023.
- [2] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster, 2023.
- [3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.
- [5] Omar Erak, Omar Alhussein, Hatem Abou-Zeid, Mehdi Bennis, and Sami Muhaidat. Adaptive token merging for efficient transformer semantic communication at the edge, 2025.
- [6] Joakim Bruslund Haurum, Sergio Escalera, Graham W. Taylor, and Thomas B. Moeslund. Which tokens to use? investigating token reduction in vision transformers, 2023.
- [7] Shibo Jie, Yehui Tang, Jianyuan Guo, Zhi-Hong Deng, Kai Han, and Yunhe Wang. Token compensator: Altering inference cost of vision transformer without re-tuning, 2024.
- [8] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations, 2022.
- [9] Junzhu Mao, Yang Shen, Jinyang Guo, Yazhou Yao, and Xiansheng Hua. Efficient token compression for vision transformer with spatial information preserved, 2025.
- [10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* 32, pages 8024–8035, 2019.
- [11] Liu Peng and Yaochu Jin. Robustness of probabilistic models to low-quality data: A multi-perspective analysis, 2025.
- [12] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification, 2021.
- [13] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2021.
- [14] Ross Wightman. PyTorch image models, 2019. <https://github.com/rwightman/pytorch-image-models>.
- [15] Guangzhen Yao, Jiayun Zheng, Zezhou Wang, Wenxin Zhang, Renda Han, Chuangxin Zhao, Zeyu Zhang, and Runhao Liu. V-Pruner: A fast and globally-informed token pruning framework for vision transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(40): 34396–34404, 2026.

- [16] Hanxun Yu, Wentong Li, Xuan Qu, Song Wang, Junbo Chen, and Jianke Zhu. VisionTrim: Unified vision token compression for training-free MLLM acceleration, 2026.
- [17] Zheng Zhan, Zhenglun Kong, Yifan Gong, Yushu Wu, Zichong Meng, Hangyu Zheng, Xuan Shen, Stratis Ioannidis, Wei Niu, Pu Zhao, and Yanzhi Wang. Exploring token pruning in vision state space models, 2024.
- [18] Jusheng Zhang, Xiaoyang Guo, Kaitong Cai, Qinhan Lv, Yijia Fan, Wenhao Chai, Jian Wang, and Keze Wang. HybridToken-VLM: Hybrid token compression for vision-language models, 2025.

A Extended related work

This appendix expands the condensed discussion of prior work given in the main text, covering Vision Transformer backbones, learned and training-free token pruning, token merging, and plug-and-play compression in turn.

A.1 Vision Transformer foundations and distillation

The Vision Transformer (ViT) architecture [4] has been further optimized through specialized training strategies. Notably, DeiT [13] introduced a distillation token that interacts with patch embeddings via self-attention to inherit the inductive biases of convolutional teacher models. That work demonstrated that hard distillation can significantly outperform soft distillation, and that the interplay between the [CLS] and distillation tokens provides a more robust classifier than either token in isolation. While effective, these models still maintain a fixed sequence length, leading to high inference costs.

A.2 Dynamic and learned token pruning

To reduce computation, several methods focus on discarding redundant tokens. DynamicViT [12] introduced a learned prediction module that hierarchically prunes tokens throughout the network. A key finding in learned pruning is that masking tokens at the input layer significantly harms performance; sparsification is therefore typically applied between transformer blocks. While learned modules offer high accuracy, they require additional training and backpropagation. In contrast, EViT [8] and Top-K strategies provide training-free alternatives by ranking token importance based on class-attention scores, identifying and removing background patches that contribute minimally to the final classification. Recent advances such as V-Pruner [15] refine this idea by employing globally informed importance metrics, ensuring that locally important but globally redundant features are identified accurately. Extensions to state-space models [17] further refine this paradigm, yet hybrids remain underexplored.

A.3 Token merging and compression

A more recent paradigm shifts from discarding information to aggregating it. Token merging (ToMe) [2] uses bipartite soft matching to fuse similar tokens based on the dot product of their key vectors. Unlike pruning, which often relies on masking that does not always translate into hardware speedups, merging physically reduces the number of tokens passed to the MLP block, leading to tangible throughput gains. Recent extensions have explored unequal bipartite graphs to achieve higher compression ratios ($> 50\%$) and have applied these techniques to Stable Diffusion in order to accelerate generative tasks [1]. Furthermore, studies such as “Which Tokens to Use?” [6] suggest that ToMe is most effective when at least 50% of the token budget is kept, often outperforming baseline DeiT models by leveraging proportional attention to account for the varying weights of merged tokens. Spatial-preserving hybrids such as that of Mao et al. [9] and adaptive merging [5] enhance locality but lack robustness guarantees, which our framework addresses. In addition, hardware-centric frameworks such as VisionTrim [16] have begun optimizing token reduction for specific edge-device latencies, though they often prioritize throughput over prediction stability.

A.4 Plug-and-play compression and transferability

Recent research has addressed the gap between models trained at different compression degrees. ToCom [7] identifies that transferring a model between different compression levels is often inefficient. To mitigate this, it introduces a pretrained plug-in module that models the performance gap between source and target compression degrees, enabling more efficient inference-time scaling without retraining. Extensions to vision-language models [18] further validate this paradigm. Our work builds on this plug-and-play philosophy by combining pruning and merging into a single, training-free framework that offers more granular control than these monolithic strategies. Finally, as noted by Peng and Jin [11], the transferability of compressed weights to smaller, specialized subsets such as ImageNet-100 requires careful alignment in order to maintain the original semantic robustness of the model.

B Result tables

This section reports the full accuracy and complexity results referenced in Section 4.2. A dash marks a configuration that standalone merging cannot reach, since bipartite matching cannot exceed a 50% reduction rate while keeping a complete bipartite graph over the tokens.

Table 1: Top-1 Accuracy comparison for DeiT and ViT models under different keep rates.

Keep Rates	DeiT-Tiny Baseline: 88.42				DeiT-Small Baseline: 92.62				DeiT-Base Baseline: 93.64			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
Random Drop	87.94	84.02	75.22	33.18	92.06	90.52	85.00	50.34	92.68	90.62	82.14	25.86
EViT	88.44	86.42	80.34	52.54	92.46	91.80	89.02	70.16	93.76	93.06	89.00	51.32
ToMe	88.44	86.70	82.92	–	92.42	91.54	89.96	–	93.26	92.00	85.30	–
Top-K	88.48	86.28	80.14	52.02	92.46	91.60	89.16	68.78	93.74	93.18	88.86	53.04
Hybrid (Ours)	88.06	86.96	84.00	67.62	92.30	91.86	90.60	79.58	93.56	92.58	89.72	47.88

Keep Rates	ViT-Tiny Baseline: 89.22				ViT-Small Baseline: 93.18				ViT-Base Baseline: 95.52			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
Random Drop	63.42	57.04	42.22	13.20	88.62	83.82	67.48	19.16	92.08	90.04	80.66	33.90
EViT	89.08	86.92	79.72	44.06	93.06	91.28	82.80	35.48	95.54	95.04	92.48	61.00
ToMe	88.76	86.86	83.06	–	92.76	91.68	86.58	–	95.30	94.64	91.42	–
Top-K	89.08	86.78	79.22	42.90	93.14	91.22	81.82	34.44	95.58	95.10	92.46	58.72
Hybrid (Ours)	88.90	87.62	84.14	60.12	92.88	91.94	88.14	55.58	95.36	95.20	93.62	71.16

Table 2: Balanced vs pruning-heavy vs merging-heavy hybrid strategies (Accuracy %)

Keep Rates	DeiT-Tiny				DeiT-Small				DeiT-Base			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
Balanced	88.06	86.96	84.00	67.62	92.30	91.86	90.60	79.58	93.56	92.58	89.72	47.88
Pruning-heavy	88.28	86.60	82.66	68.00	92.26	91.96	90.40	79.76	93.44	92.96	90.24	52.44
Merging-heavy	88.00	86.88	84.38	74.98	92.28	92.06	90.26	84.84	93.46	92.70	88.70	68.58

Keep Rates	ViT-Tiny				ViT-Small				ViT-Base			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
Balanced	88.90	87.62	84.14	60.12	92.88	91.94	88.14	55.58	95.36	95.20	93.62	71.16
Pruning-heavy	89.30	87.52	82.56	60.06	92.86	92.02	87.58	55.98	95.48	95.20	93.62	74.08
Merging-heavy	89.10	87.74	84.36	71.42	92.84	92.24	88.50	71.16	95.28	95.16	93.12	83.56

Table 3: GFLOPs comparison for DeiT and ViT models under different keep rates.

Keep Rates	DeiT-Tiny Baseline: 1.079				DeiT-Small Baseline: 4.250				DeiT-Base Baseline: 16.867			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
EViT	0.989	0.752	0.576	0.432	3.817	2.892	2.203	1.634	14.985	11.336	8.608	6.349
ToMe	0.983	0.746	0.571	–	3.787	2.863	2.178	–	14.865	11.216	8.507	–
Top-K	0.981	0.744	0.570	0.428	3.786	2.862	2.177	1.618	14.863	11.215	8.506	6.287
Hybrid (Ours)	0.992	0.753	0.577	0.429	3.823	2.893	2.204	1.624	15.005	11.337	8.608	6.309

Keep Rates	ViT-Tiny Baseline: 1.079				ViT-Small Baseline: 4.250				ViT-Base Baseline: 16.867			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
EViT	0.989	0.752	0.576	0.432	3.817	2.892	2.203	1.634	14.985	11.336	8.608	6.349
ToMe	0.983	0.746	0.571	–	3.787	2.863	2.178	–	14.865	11.216	8.507	–
Top-K	0.981	0.744	0.570	0.428	3.786	2.862	2.177	1.618	14.863	11.215	8.506	6.287
Hybrid (Ours)	0.992	0.753	0.577	0.429	3.823	2.893	2.204	1.624	15.005	11.337	8.608	6.309

Table 4: Average Δ PFR across augmentations.

Model	0.25	0.50	0.70	0.90
ViT-Base	0.3611	0.0422	0.0099	0.0022
ViT-Tiny	0.4105	0.0943	0.0272	0.0059

Table 5: Relative Accuracy Drop (RAD).

Model	Comparison	0.25	0.50	0.70	0.90
Tiny	Raw	0.073	0.023	0.003	−0.004
	ToMe	–	−0.003	−0.003	−0.003
	EViT	−0.030	−0.002	0.000	−0.003
Base	Raw	0.045	0.009	0.001	−0.001
	ToMe	–	0.002	−0.001	−0.001
	EViT	−0.026	0.000	0.001	−0.002

C Extended hardware performance metrics

While the main text focuses on accuracy and GFLOPs, we present here the comprehensive latency and throughput benchmarks, conducted on a T4 GPU using a Google Colab environment.

C.1 Inference latency

Table 6 details the average per-image inference latency for both DeiT and ViT architectures. We observe that the dual-stage overhead of our hybrid approach is negligible, consistently maintaining parity with, or outperforming, standalone pruning and merging methods. Notably, in the ViT-Base configuration at aggressive keep rates ($K = 0.25$), our hybrid method achieves the lowest latency of 4.222 ms/img, outperforming EViT (5.061 ms/img) and Top-K (4.417 ms/img). Similarly, for ViT-Tiny at $K = 0.9$, our approach achieves lowest latency of 0.971 ms/img among the evaluated methods. These results demonstrate that the combination of pruning and merging stages does not introduce a computational bottleneck, even when scaling to larger backbone architectures.

Table 6: Latency comparison (ms/img)

Keep Rates	DeiT-Tiny Baseline: 1.629				DeiT-Small Baseline: 3.092				DeiT-Base Baseline: 12.772			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
EViT	1.241	0.984	0.751	0.747	3.256	2.390	1.763	1.253	11.559	8.771	6.728	5.058
ToMe	1.042	0.888	0.900	–	2.952	2.217	1.621	–	10.437	7.958	6.092	–
Top-K	1.156	0.903	0.729	0.633	2.892	2.138	1.569	1.311	10.107	7.660	5.845	4.421
Hybrid (Ours)	1.110	1.116	1.249	1.107	2.930	2.165	1.628	1.193	10.189	7.626	5.885	4.532

Keep Rates	ViT-Tiny Baseline: 1.322				ViT-Small Baseline: 3.440				ViT-Base Baseline: 12.721			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
EViT	1.096	1.036	0.753	0.660	3.260	2.362	1.744	1.254	11.568	8.815	6.698	5.061
ToMe	1.059	1.031	0.777	–	2.745	2.203	1.647	–	10.301	7.361	6.022	–
Top-K	1.208	0.901	0.751	0.719	2.876	2.136	1.575	1.150	10.114	7.646	5.825	4.417
Hybrid (Ours)	0.971	0.932	1.305	0.697	2.822	2.093	1.560	1.227	9.805	7.340	5.631	4.222

C.2 System throughput

Table 7 presents a comparative analysis of system throughput (imgs/sec) across varied keep rates for DeiT and ViT architectures. Our hybrid framework demonstrates superior scaling characteristics, particularly as model complexity increases from Tiny to Base variants. In the ViT-Base regime, our method achieves the highest throughput across almost all keep rates. Most notably, at the most aggressive compression level ($K = 0.25$), the hybrid approach reaches 236.85 imgs/sec, representing a $3.01\times$ speedup over the ViT-Base baseline (78.61 imgs/sec) and significantly outperforming EViT (197.60 imgs/sec). Even at higher keep rates such as $K = 0.9$, our method maintains a lead with 101.98 imgs/sec, validating that the dual-stage reduction logic provides an immediate efficiency boost without requiring a heavy computational overhead. For smaller scales like ViT-Tiny, our method achieves a peak throughput of 1073.18 imgs/sec at $K = 0.7$, further illustrating its versatility for high-speed edge deployment.

D Supplemental implementation details

This section expands upon the implementation details provided in Section 4.1. We provide specific hyperparameter configurations and environmental constraints to ensure full reproducibility.

Model families. We evaluate our framework across two major model families: the standard Vision Transformer (ViT) and the Data-efficient Image Transformer (DeiT), covering Tiny, Small and Base scales (16×16 patch size, 224×224 resolution). All inference is performed in full precision (FP32) to ensure a high-fidelity evaluation of the token manifolds, and the Prune-and-Merge blocks are integrated at layers $l \in \{3, 6, 9\}$, specifically positioned post-attention and pre-MLP.

Table 7: Throughput comparison (imgs/sec)

Keep Rates	DeiT-Tiny Baseline: 613.92				DeiT-Small Baseline: 323.46				DeiT-Base Baseline: 78.30			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
EViT	805.73	1016.41	1330.76	1338.06	307.11	418.34	567.28	797.83	86.51	114.02	148.62	197.69
ToMe	960.08	1126.47	1110.56	–	338.74	451.12	616.72	–	95.81	125.66	164.14	–
Top-K	864.74	1107.43	1371.64	1579.34	345.74	467.62	637.31	762.66	98.94	130.56	171.08	226.19
Hybrid (Ours)	917.30	895.66	800.48	903.08	341.34	461.98	614.11	838.28	98.14	131.13	169.92	220.63

Keep Rates	ViT-Tiny Baseline: 756.22				ViT-Small Baseline: 290.71				ViT-Base Baseline: 78.61			
	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25	0.9	0.7	0.5	0.25
EViT	912.27	965.17	1328.21	1514.27	306.75	423.41	573.48	797.14	86.45	113.44	149.30	197.60
ToMe	944.52	969.48	1287.70	–	364.32	454.01	607.31	–	97.07	135.85	166.05	–
Top-K	827.67	1109.70	1331.42	1390.24	347.75	468.19	634.78	869.25	98.87	130.79	171.68	226.38
Hybrid (Ours)	1029.53	1073.18	766.50	1434.45	354.34	477.73	640.87	815.07	101.98	136.25	177.57	236.85

Baselines. We benchmark our method against the original, uncompressed backbone operating on the full sequence length; Random Drop, a naive baseline where tokens are discarded uniformly at random, representing a lower bound for pruning; Top-K pruning, a basic strategy that retains high-attention tokens but discards the remainder without context fusion; and EViT [8] and ToMe [2], the state of the art in semantic pruning and similarity-based merging, respectively. For a fair comparison, all baselines are evaluated under identical hardware and software constraints.

Hyperparameter configuration. All experiments were conducted using the PyTorch framework [10] and the timm library [14] on a Google Colab T4 GPU environment. For our hybrid Prune-and-Merge blocks, the similarity metric used for token merging in Stage 2 was the cosine similarity computed on the token features (embeddings). Proportional attention is applied in Stage 2, and for all ToMe-based baselines, to maintain mathematical consistency during self-attention.

Hardware environment. All benchmarks, namely the latency figures in Table 6 and the throughput figures in Table 7, were measured on a single NVIDIA T4 GPU (16 GB VRAM).

ImageNet-100 class mapping. The 100 classes were selected as a subset of the ImageNet-1K label set. The backbone was initialized with ImageNet-1K pre-trained weights and the linear projection weights corresponding to the selected 100 classes were retained to obtain the 100-class output space.

E Additional results

In this section, we provide detailed quantitative results supporting our analysis. We report (i) the Prediction Flip Rate (PFR, Table 8) and (ii) the absolute change in PFR (Δ PFR, Table 9) across different augmentations and keep rates. RAD serves as an accuracy trade-off metric, measuring the difference in top-1 accuracy ratios between frameworks when subjected to transformations. While faithfulness is often tested by removing important tokens, we evaluate the inverse: a robust model should maintain its prediction if an input is slightly transformed such that its semantic information remains unchanged. We calculate Δ PFR for ViT-Tiny and ViT-Base to capture trends across increasing model scales.

E.1 Prediction flip rate

E.2 Absolute change in the prediction flip rate

Table 8: Prediction Flip Rate (PFR) for raw and hybrid models across augmentations and keep rates.

Model	Augmentation	Raw	0.25	0.50	0.70	0.90
ViT-Base	Rotation	0.0472	0.4528	0.0938	0.0552	0.0458
	Color Jitter	0.0128	0.3552	0.0524	0.0250	0.0162
	Horizontal Flip	0.0112	0.3464	0.0516	0.0206	0.0130
ViT-Tiny	Rotation	0.1856	0.6086	0.2902	0.2162	0.1898
	Color Jitter	0.0416	0.4516	0.1394	0.0698	0.0498
	Horizontal Flip	0.0356	0.4342	0.1160	0.0584	0.0408

Table 9: Absolute change in Prediction Flip Rate (Δ PFR) between raw and hybrid models.

Model	Augmentation	0.25	0.50	0.70	0.90
ViT-Base	Rotation	0.4056	0.0466	0.0080	0.0014
	Color Jitter	0.3424	0.0396	0.0122	0.0034
	Horizontal Flip	0.3352	0.0404	0.0094	0.0018
ViT-Tiny	Rotation	0.4230	0.1046	0.0306	0.0042
	Color Jitter	0.4100	0.0978	0.0282	0.0082
	Horizontal Flip	0.3986	0.0804	0.0228	0.0052