THE PRICE OF A MERGE IS PAID IN DEPTH: A TIGHT, LABEL-FREE BOUND ON TOKEN MERGING IN VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Token merging accelerates Vision Transformers by replacing pairs of similar tokens with their average, and no existing method states what a merge costs the layer that receives it. We give that cost. Because the next layer aggregates tokens in proportion to the attention they receive, the perturbation a merge induces downstream is bounded by an *influence-weighted* error that one forward pass computes with no labels; the salience-proportional average minimizes the bound, and uniform averaging inflates it by $(s_i+s_j)^2/4s_is_j$. We measure the bound on a pretrained ViT-B/16. It holds on every one of 52,190 merged pairs and is tight to 1.6%; uniform averaging realizes $2.27\times$ the downstream perturbation of the proportional average; and the price is paid in depth, with a ratio of 1.00 in the shallow blocks, 3.1 in the deep ones, and a per-pair cost that rises by more than an order of magnitude from block 2 to block 11. Two further measurements say what the price buys. Halving the perturbation moves ImageNet accuracy by 0.06 pp, so the network absorbs it; forcing merges into the deep blocks costs five to eleven times the marginal rate of a schedule that avoids them. Accuracy is set by where and how much a method merges, not by how it averages. We turn the bound into a schedule, ADAMERGE, which spends a merge budget where the bound says merging is cheap and is calibrated on 256 unlabeled images: its margin over TOME grows with compression on four backbones, from $+0.01$ to $+1.67$ pp on ViT-B/16 and $+0.53$ to $+4.64$ pp on ViT-L/16, and it survives 30 epochs of fine-tuning. The bound applies to any merging method and costs one column sum of a matrix the method already forms.

1 INTRODUCTION

Token merging makes Vision Transformers faster by discarding information, and no method states how much. TOME (Bolya et al., 2023) pairs similar tokens across a bipartite split, replaces each pair by its mean, and removes a fixed number of tokens per layer; its successors change which tokens are paired (Kim et al., 2024; Heo et al., 2024; Tran et al., 2024) and keep the rest. PiToMe motivates its score by spectrum preservation of the token graph (Tran et al., 2024); what no method provides is an account of the merge itself: what the layer that receives a merged token loses, which of the two tokens it should have kept more of, and where in the network the loss is large. The methods are tuned against accuracy, which is the right target and a poor instrument, because a merge that costs the next layer a great deal may cost the classifier nothing, and a fixed schedule that is harmless at one depth may be ruinous at another.

This paper supplies the account and then measures it. The object is the aggregate the next layer forms, $\mathbf{Y} = \hat{\mathbf{A}}\mathbf{X}$ for a row-stochastic affinity $\hat{\mathbf{A}}$: what each token reads from the sequence. For a merged pair (i, j) replaced by $\tilde{x}$, we prove that the change in this aggregate is bounded by $s_i\|\tilde{x} - x_i\|^2 + s_j\|\tilde{x} - x_j\|^2$, where s_i, the column sum of $\hat{\mathbf{A}}$, is the *influence* of token i on the layer. The bound is minimized by the salience-proportional average $\tilde{x}^\star = (s_ix_i + s_jx_j)/(s_i + s_j)$; the uniform average inflates it by $(s_i + s_j)^2/4s_is_j$, a factor set by the pair's influence asymmetry alone; and an influence that is wrong in ordering is worse than none, because the excess is quadratic in the weight error. The identities are exact and need no noise model. In other words, the bound is a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: **The price of a merge, measured.** ViT-B/16, 256 ImageNet images, all twelve blocks, the 52,190 pairs an influence-weighted matching selects at $r_{\max}$=17 (Section 5). (a) The cumulative token-equality penalty $P(r)$, the excess bound of uniform over proportional averaging, is non-decreasing in the merge count at every block (log scale; blocks 1, 5, 9, 11). (b) Its per-pair increments are front-loaded in matching order. (c) The realized downstream perturbation of the uniform average relative to the proportional one, by block: 1.00–1.02 at blocks 0, 2, 3 and 2.7–3.1 at blocks 7, 8, 11, 2.27× overall. The bound holds on 100% of pairs and the realized perturbation is 98.4–98.7% of it.

per-merge price that any method can compute from one forward pass, with no labels, at the cost of one column sum of a matrix it already forms.

Measured on a pretrained ViT-B/16 (Figure 1, Section 5), the bound holds on every one of 52,190 merged pairs and is tight to 1.6%; uniform averaging realizes 2.27× the perturbation of the proportional average; and the factor is 1.00–1.02 at blocks 0, 2, 3 and 2.7–3.1 at blocks 7, 8, 11, while the per-pair cost of the cheapest merges rises from 21 to 388 feature units between blocks 2 and 11: the price of a merge is paid in depth. Two further measurements say what it buys. Replacing the proportional average by the uniform one, everything else fixed, doubles the perturbation and moves ImageNet accuracy by 0.06 pp; forcing merges into the deep blocks a calibrated schedule leaves alone costs 1.8 pp per GFLOP saved, five to eleven times the schedule's own marginal rate. The network absorbs the perturbation the average creates and not the one depth creates, so accuracy is set by where and how much a method merges, and only weakly by how it averages, which is the opposite of where the literature has spent its design effort.

The bound also implies a schedule, and we build it to show that the account is actionable. AdaMerge keeps ToMe's bipartite matching, combines pairs by $\tilde{x}^\star$, and sets the number of merges at each layer from the layer's mean influence-weighted similarity, the bound's own estimate of how cheaply the layer can be compressed, standardized against two moments calibrated once on 256 unlabeled images; no gradients, labels, or parameters are involved. On ImageNet-1k at matched FLOPs its margin over ToMe is non-negative at every tier on ViT-S/16, ViT-B/16, DeiT-B/16, and ViT-L/16 and rises monotonically on each ($+0.01 \to +1.67$ pp on ViT-B/16, $+0.53 \to +4.64$ pp on ViT-L/16), persists after fine-tuning, and is reproduced by PiToMe, an independently designed importance-aware method ($+0.93 \to +5.56$ pp on ViT-L/16); a proxy that orders tokens differently loses 8.59 pp, as the bound predicts. The schedule is 5–9% slower than ToMe at matched FLOPs on ViT-S/B and 7% faster on ViT-L, PiToMe is 0.4–1.3 pp more accurate on ViT-L, and on LLaVA-1.5-7B it transfers without regression but does not widen.

In summary: we prove a label-free, one-forward-pass bound on what a merge costs the next layer and the average that minimizes it (Section 3); we measure it and find it tight, its price paid in depth, and its accuracy consequences carried by depth rather than by the average (Section 5); and we turn it into a calibrated schedule whose margin over ToMe widens with compression on four backbones (Sections 4–7).

2 Related Work

Token reduction decides three things and bounds none of them. Which tokens to drop: learned scorers (Rao et al., 2021; Yin et al., 2022; Pan et al., 2021; Tang et al., 2022; Goyal et al., 2020; Kim et al., 2022), [CLS]-attention scores (Liang et al., 2022b; Fayyaz et al., 2022; Xu et al., 2022;

Kong et al., 2022), diversity terms (Long et al., 2023), and the finding that plain top-k is hard to beat (Haurum et al., 2023). How to combine the rest: ToMe averages matched pairs (Bolya et al., 2023), as do its diffusion and learned-rate variants (Bolya & Hoffman, 2023; Chen et al., 2023); ToFu switches between pruning and merging by depth (Kim et al., 2024), DSM delays merging (Heo et al., 2024), PiToMe protects informative tokens at partition time (Tran et al., 2024), and pooling and clustering methods learn the aggregation (Ryoo et al., 2021; Marin et al., 2023; Xu et al., 2024; Zeng et al., 2022); Hourglass and Token Adapter reduce tokens reversibly for dense prediction (Liang et al., 2022a; Su et al., 2024), and FastV drops visual tokens inside a vision-language model (Chen et al., 2024). How many: almost always a fixed r per layer, with the exceptions in adaptive computation below. Every one of these choices is validated against end accuracy, and PiToMe's partition is motivated by a spectral argument on the token graph; none comes with a per-merge statement of what the merge does to the layer that receives it, so none can say which decision its accuracy comes from. Sections 3–5 supply the statement and the answer.

Influence, saliency, and weighted estimation. That some units matter more than others is the oldest idea in network compression (LeCun et al., 1990; Molchanov et al., 2017; Michel et al., 2019). For tokens, attention flow traces contributions to an output (Abnar & Zuidema, 2020; Chefer et al., 2021), DINO shows attention concentrating on objects (Caron et al., 2021), registers absorb background mass (Darcet et al., 2024), and SBAM uses the affinity column sum to choose masks during pre-training (Choi et al., 2024). Weighting observations by reliability rather than uniformly is the principle of generalized least squares (Aitken, 1935). We give the column sum a role with a proof: it is the first-order influence of a token on the next layer's aggregate, and the merge error it weights is minimized by the salience-proportional average.

What ViTs tolerate. ViT representations are more uniform across depth than convolutional ones (Raghu et al., 2021), self-attention acts as a low-pass filter that flattens the loss landscape (Park & Kim, 2022), and a few tokens can carry global information (Darcet et al., 2024). Section 5 gives a quantitative instance of this tolerance and of its limit in depth. Token reduction composes with kernel-level attention (Dao et al., 2022) and better-trained backbones (Touvron et al., 2021; Steiner et al., 2022; Zhai et al., 2022); we evaluate supervised checkpoints of four sizes and the CLIP tower of LLaVA-1.5 (Radford et al., 2021; Liu et al., 2024), leaving self-supervised ones (He et al., 2022; Oquab et al., 2024) to future work.

How many tokens to keep. Early-exit, spatially adaptive, and input-adaptive networks set their budget with trained gates (Huang et al., 2018; Elbayad et al., 2020; Figurnov et al., 2017; Wang et al., 2021; Meng et al., 2022; Han et al., 2022); AdaMerge sets the merge count per layer from the bound evaluated at that layer, with no learned parameter (Section 4).

3 What a Merge Costs the Next Layer

This section states what a merge does to the aggregate the next layer forms, which average minimizes it, and what a wrong influence costs; every statement is exact (proofs in Appendix A).

Objective. The quantity a merge should preserve is what the next layer sees. We take the aggregate $\mathbf{Y} = \hat{\mathbf{A}}\mathbf{X}$, whose row $y_k = \sum_m \hat{A}_{km} x_m$ is what token k reads from the sequence, as a first-order surrogate for it; the argument needs only that the rows of $\hat{\mathbf{A}}$ are stochastic, and this is the matrix the merge step already forms.[1] Merging a matched pair (i, j) into $\tilde{x}$ replaces x_i and x_j by $\tilde{x}$ while $\hat{A}_{ki}$ and $\hat{A}_{kj}$ are, to first order, unchanged.

Lemma 1 (Influence bound). *Let $\Delta\mathbf{Y}$ be the change in $\mathbf{Y}$ when x_i and x_j are both replaced by $\tilde{x}$. Then*

$$\|\Delta\mathbf{Y}\|_F^2 \;\le\; J_{ij}(\tilde{x}) \;:=\; s_i\|\tilde{x} - x_i\|^2 + s_j\|\tilde{x} - x_j\|^2. \tag{1}$$

In other words, an error on token i is felt downstream in proportion to how much the sequence attends to i.

[1] A real block uses $\mathrm{softmax}(QK^\top)V$ with learned projections; the same lemma holds for any row-stochastic weight matrix.

Proposition 1 (Salience-proportional averaging is optimal). *J_{ij} is minimized over $\tilde{x} \in \mathbb{R}^d$ at $\tilde{x}^\star = (s_i x_i + s_j x_j)/(s_i + s_j)$, with $J_{ij}(\tilde{x}^\star) = \frac{s_i s_j}{s_i + s_j}\|x_i - x_j\|^2$. Uniform averaging $\bar{x} = (x_i + x_j)/2$ incurs*

$$J_{ij}(\bar{x}) - J_{ij}(\tilde{x}^\star) \;=\; \frac{(s_i - s_j)^2}{4(s_i + s_j)}\,\|x_i - x_j\|^2 \;\geq\; 0, \qquad \frac{J_{ij}(\bar{x})}{J_{ij}(\tilde{x}^\star)} = \frac{(s_i + s_j)^2}{4 s_i s_j}, \tag{2}$$

with equality if and only if $s_i = s_j$.

A token with four times its partner's influence makes uniform averaging inflate the bound by $25/16$; the ratio depends on the asymmetry alone. We call the left side of Eq. 2 the *token-equality penalty*; $\tilde{x}^\star$ is the aggregation rule of AdaMerge (Eq. 7), and ToMe is the case $s_i \equiv s_j$.

Corollary 1 (A wrong influence can be worse than none). *If the merge uses weight $\hat{w}$ instead of $w^\star$, then $J_{ij}(\hat{w}) - J_{ij}(w^\star) = (s_i + s_j)(\hat{w} - w^\star)^2\|x_i - x_j\|^2$. Hence the merge is worse than uniform averaging exactly when $|\hat{w} - w^\star| > |\frac{1}{2} - w^\star|$.*

The excess is quadratic in the weight error, so an influence estimate wrong in ordering is penalized more than ignoring influence (Section 7: [CLS]-attention and feature-norm proxies lose 8.59 and 3.20 pp, both below ToMe).

Proposition 2 (The penalty does not shrink with compression). *Let pairs enter the merge set in the order produced by matching, $(i_1, j_1), (i_2, j_2), \ldots$, and let $p_k = \frac{(s_{i_k} - s_{j_k})^2}{4(s_{i_k} + s_{j_k})}\|x_{i_k} - x_{j_k}\|^2$ be the k-th pair's penalty. The cumulative penalty $P(r) = \sum_{k \leq r} p_k$ is non-decreasing in r for any matching. If, in addition, (A1) pairs are admitted in non-increasing order of cosine similarity and features are ℓ_2-normalized, so that $\|x_{i_k} - x_{j_k}\|^2 = 2(1 - \cos_k)$ is non-decreasing in k, and (A2) the asymmetry factor $(s_{i_k} - s_{j_k})^2/(s_{i_k} + s_{j_k})$ is non-decreasing in k, then p_k is non-decreasing and $P(r)$ is convex in r.*

(A1) holds for ToMe's matching by construction. (A2) is an assumption about which pairs a larger budget admits, and we do not prove it; the monotone statement is what the paper uses, and Section 5 reports what we measured about the convex one.

Proposition 1 also says how cheap a merge is: $J^\star_{ij}$ is small when an influential token has a partner it resembles. Section 4.4 turns the layer average of this quantity into a budget, which is why the budget cannot be computed from unweighted cosine similarity: it would then estimate the cost of the wrong objective (Table 13).

4 The Schedule the Bound Implies: AdaMerge

To show that the bound is actionable we build the schedule it implies. AdaMerge is a drop-in block before every Transformer layer (Figure 8, Algorithms 1–5 in Appendix K): at layer l it receives the patch tokens $\mathbf{X}_l \in \mathbb{R}^{N_l \times d}$ (the [CLS] token is set aside and reattached), decides *how many* pairs to merge, r_l, and *how* to merge them, and returns $\mathbf{X}_{l+1} \in \mathbb{R}^{(N_l - r_l) \times d}$. Both decisions come from Section 3; nothing is learned.

4.1 Influence and the Pair Cost

We write $\hat{\mathbf{A}} = \mathrm{softmax}(\mathbf{X}\mathbf{X}^\top)$ for the row-normalized affinity and define the influence of token i as the mass the sequence places on it,

$$\mathbf{s} = \hat{\mathbf{A}}^\top \mathbf{1}, \qquad s_i = \sum_{k=1}^{N} \hat{A}_{ki}, \tag{3}$$

which is the feature-affinity centrality of Choi et al. (2024). Within a layer we use the per-image affine rescaling $\hat{s}_i = (s_i - \min_k s_k)/(\max_k s_k - \min_k s_k) \in [0, 1]$ so that $\tilde{\mathbf{S}}$ below is comparable across depths (constant rows give $\hat{s} \equiv \frac{1}{2}$); the rescaling preserves the ordering of influences, and Appendix E.3 measures both variants.

By Proposition 1, the least downstream perturbation that merging a pair (i, j) can achieve is

$$J^\star_{ij} \;=\; \frac{s_i s_j}{s_i + s_j}\,\|x_i - x_j\|^2 \;=\; \tfrac{1}{2}\,\mathrm{H}(s_i, s_j)\,\|x_i - x_j\|^2, \tag{4}$$

half the harmonic mean of the two influences times the pair's squared distance. The harmonic mean is dominated by the smaller influence, so a pair joining an influential token to an unimportant near-duplicate is cheap to merge: the merged token will be almost entirely the influential one. AdaMerge therefore does not protect influential tokens from matching; it ensures that, once matched, they dominate the result (Section 4.3).

4.2 Matching: What the Influence-Weighted Similarity Measures

As in ToMe, tokens are split by index into $\mathcal{A} = \{1, \ldots, N/2\}$ and $\mathcal{B} = \{N/2{+}1, \ldots, N\}$, each source $i \in \mathcal{A}$ gets one target in $\mathcal{B}$, and the r_l best-scoring sources are merged into their targets. Where ToMe scores a pair by cosine similarity, AdaMerge scores it by

$$\tilde{S}_{ij} = \hat{s}_i\,\cos(x_i, x_j), \qquad \pi(i) = \arg\max_{j \in \mathcal{B}} \tilde{S}_{ij}, \qquad \mathcal{M}_l = \text{Top-}r_l\big\{\tilde{S}_{i\pi(i)}\big\}_{i \in \mathcal{A}}. \tag{5}$$

The score has an exact reading in terms of Eq. 4 on the unit-normalized features the matching uses (the aggregation acts on the raw features). There $\|x_i - x_j\|^2 = 2(1 - \cos(x_i, x_j))$, and because the harmonic mean is at most the smaller argument, $J^\star_{ij} \le U_{ij} := 2\hat{s}_i\,(1 - \cos(x_i, x_j))$. Hence

$$\tilde{S}_{ij} \;=\; \hat{s}_i - \tfrac{1}{2}\,U_{ij}, \qquad U_{ij} \;\ge\; J^\star_{ij}. \tag{6}$$

In other words, the score is the influence a merge retains minus half an upper bound on what it costs; for a fixed source $\pi(i)$ minimizes the cost bound as in ToMe, and across sources the top-r_l rule prefers pairs that retain much influence per unit of cost.

4.3 Aggregation

Each selected pair is replaced by the minimizer of its pair cost,

$$\tilde{x}_{\,i\pi(i)} = \frac{\hat{s}_i\,x_i + \hat{s}_{\pi(i)}\,x_{\pi(i)}}{\hat{s}_i + \hat{s}_{\pi(i)}}, \tag{7}$$

which is Proposition 1, with ToMe the case $\hat{s}_i \equiv \hat{s}_{\pi(i)}$. Influence is recomputed at every layer from the current tokens (Eq. 3), so a merged token carries whatever influence the reduced sequence assigns it at the next layer (Algorithm 4).

4.4 Budget: How Many Pairs a Layer Can Afford

The row maxima of $\tilde{\mathbf{S}}$ are what the matching would score if every source were merged; their mean,

$$\bar{S}_l = \frac{1}{|\mathcal{A}|}\sum_{i \in \mathcal{A}} \tilde{S}_{i\pi(i)} = \underbrace{\frac{1}{|\mathcal{A}|}\sum_{i \in \mathcal{A}} \hat{s}_i}_{\text{mean influence}} \;-\; \tfrac{1}{2}\;\underbrace{\frac{1}{|\mathcal{A}|}\sum_{i \in \mathcal{A}} U_{i\pi(i)}}_{\text{mean best-case cost bound}}, \tag{8}$$

is the influence the layer could keep per token if it merged everything, net of half the cost bound of doing so, the *redundancy statistic*. Its distribution depends on depth and on upstream compression (Table 18), so we standardize it against per-layer moments $\theta_l = (\mu_l, \sigma_l)$ and gate the z-score,

$$z_l = \frac{\bar{S}_l - \mu_l}{\sigma_l}, \qquad r_l = \Big\lfloor r_{\max}\,\sigma\big(\alpha\,z_l/T\big)\Big\rfloor, \tag{9}$$

with $\alpha = 3$ and $T = 1$ throughout (Table 17 varies T). A layer more redundant than its calibrated norm spends more of $r_{\max}$, a less redundant one spends less, and the gate is closed for $z_l \ll 0$. In our ImageNet evaluations $\bar{S}_l$ is averaged over the evaluation batch of 64 images, so the schedule there is a calibrated depth allocation that reacts to the batch; at batch size 1 it reacts to the image (Figure 2a), and we do not attribute the accuracy results to per-image reaction. This is the logic of halting and early-exit rules (Figurnov et al., 2017; Huang et al., 2018; Elbayad et al., 2020) with the merge objective as the score and two moments as the threshold.

Calibration as a fixed point. The moments $\theta = (\theta_1, \ldots, \theta_L)$ are not free: the distribution of $\bar{S}_l$ depends on how much was merged before layer l, hence on $\theta_{<l}$. Let $\mathcal{T}(\theta)$ be the per-layer mean and standard deviation of $\bar{S}_l$ over an unlabeled calibration set $\mathcal{D}_c$ when Eq. 9 runs with moments θ. The consistent statistics are the fixed point

$$\theta^\star = \mathcal{T}(\theta^\star), \qquad \theta^{(k+1)} = \mathcal{T}\big(\theta^{(k)}\big),\ \theta^{(0)} = (0.5,\, 0.1)^{\otimes L}, \tag{10}$$

and we approximate it by two applications of $\mathcal{T}$, i.e., two forward passes over $\mathcal{D}_c$ with no gradients and no labels. Empirically $\mathcal{T}$ contracts by more than 90% in the first two iterations from three different initializations and then oscillates within $\|\Delta\theta\|_2 < 0.005$ (Figure 3, Appendix F); $|\mathcal{D}_c| = 256$ images already reproduce the accuracy of a 12,811-image calibration to within 0.04 pp (Appendix F.3). The prior $\theta^{(0)}$ is far from $\theta^\star$ on purpose: without calibration the true scale of $\bar{S}_l$ (0.02–0.30, Table 18) lies outside it, the gate stays closed, and AdaMerge reduces to the identity (Appendix F.4).

4.5 Cost

Per layer, AdaMerge adds the $N \times N$ affinity, its softmax, a column sum, and a scalar decision; against a block costing about $12Nd^2 + 2N^2d$, the relative overhead is

$$\rho(N, d) \;\approx\; \frac{N^2 d}{12Nd^2 + 2N^2 d} \;=\; \frac{N}{12d + 2N}, \tag{11}$$

3.9%, 2.0%, 1.5% at $d = 384, 768, 1024$ for N=196. It shrinks with width while the savings, $\propto (2Nr_l - r_l^2)d$, grow with it, which predicts a wall-clock crossover between small and large backbones (Section 6.3, Appendix H).

5 The Bound, Measured

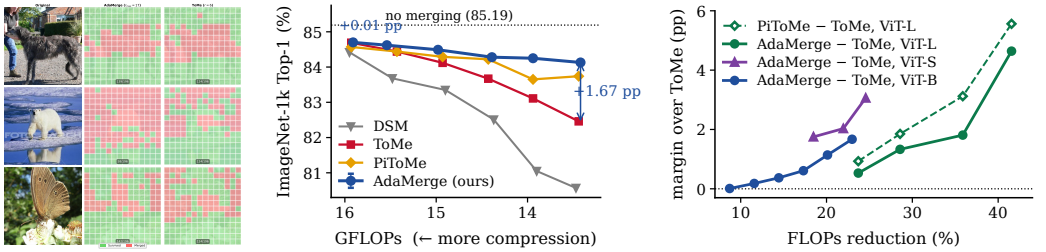
Everything in Section 3 concerns one forward pass, so we checked it on one: 256 ImageNet-1k training images, all twelve blocks of ViT-B/16, every pair the matching of Eq. 5 selects at $r_{\max}$=17 (52,190 pairs), recording the realized perturbation $\|\Delta\mathbf{Y}\|_F^2$ and the bound J_{ij} for the uniform average and for the proportional average with raw s and with rescaled $\hat{s}$ (Figure 1, Tables 15–16, Appendix E.3).

The bound holds and is tight. Eq. 1 holds on every pair for every rule, and the realized perturbation is 98.7% of the bound for the uniform average and 98.4% for the proportional one (92% at the loosest block); the slack is small because few rows of $\hat{\mathbf{A}}$ attend to both members of a pair. With $\hat{\mathbf{A}}$ held fixed this tests the inequality and its slack; the accuracy measurements below test the first-order approximation behind it.

The price is paid in depth. Uniform averaging realizes 207.9 units of downstream perturbation per pair against 91.4 for the proportional average, a factor of 2.27. The factor is 1.00–1.02 at blocks 0, 2, and 3 and 2.7–3.1 at blocks 7, 8, and 11, and the reason is visible in the influence itself: in the shallow blocks $\hat{\mathbf{A}}$ is close to the identity, so every column sum is 1.00 ± 0.08 and the average cannot matter, whereas by block 11 the column sums range from 0 to 116 as a few tokens absorb most of the attention. The median pair has a ratio of 1.000; the mean is carried by the asymmetric pairs, and those sit deep. The minimum cost itself, $\bar{J}^\star$, rises from 21 at block 2 to 388 at block 11 (Table 16). In other words, the same merge is nearly free early and expensive late; a fixed schedule keeps merging in the depths as r grows, and a schedule that reads the bound stops there first (Section 6.4).

Monotone, and front-loaded in our order. The increments p_k are positive on 99.8% of pairs, so $P(r)$ is non-decreasing as Proposition 2 states (Figure 1a); in the order $\hat{s}_i$ cos produces they decrease with k in every block (Figure 1b), because the score admits the most asymmetric pairs first. That is a property of our order, not ToMe's, and we do not claim convexity for ToMe's merge sets.

The network absorbs the average and not the depth. Holding matching and budget fixed at $r_{\max}$=17 and changing only the merge weights, the proportional average with $\hat{s}$, with raw s, and the uniform average reach 84.13, 84.07, and 84.07% Top-1 on a second machine on which the default reproduces at 84.13% (Table 1: 84.28%; raw vs. default not significant, McNemar p=0.52), while



(a) Where it merges. (b) ViT-B/16 at matched FLOPs. (c) Margin over ToMe, three backbones.

Figure 2: **The schedule in practice.** (a) Token survival maps (green kept, red merged) for AdaMerge ($r_{\max}$=17, left) and ToMe (r=6, right) at batch size 1: ToMe keeps 124/196 tokens on every image, AdaMerge keeps 58/196 on the uniform polar-bear scene and 139–142/196 on the cluttered ones. (b) Top-1 against FLOPs on ViT-B/16 (AdaMerge: mean±std over 5 runs); the margin over ToMe rises from $+0.01$ to $+1.67$ pp with non-decreasing increments $0.17, 0.19, 0.24, 0.53, 0.53$ pp. (c) The same margin against FLOPs reduction on ViT-B/16, ViT-S/16 (three tiers matched with PiToMe), and ViT-L/16, where it reaches $+4.64$ pp; PiToMe, which protects influential tokens at partition time, widens over ToMe the same way ($+0.93 \rightarrow +5.56$ pp, dashed).

realizing 91.7, 91.4, and 207.9 units of perturbation. Halving the perturbation the average creates is worth 0.06 pp at this tier on this backbone: the classifier absorbs it. The perturbation depth creates is different. At $r_{\max}$=23 the calibrated schedule leaves blocks $l \geq 9$ untouched; forcing a floor of $r_l \geq 1$ or 2 there lowers accuracy from 84.15 to 84.07 and 84.04% for FLOPs of 13.47 and 13.44 G, i.e., 1.8 pp per GFLOP saved (1.2–2.5 within the three-run spread), against 0.23 pp per GFLOP along the schedule's own curve between $r_{\max}$=20 and 23: five to eleven times the marginal rate (Table 22). This pair of numbers is the paper's central finding: accuracy is set by where and how much a method merges, and only weakly by how it averages.

6 Experiments

The schedule is a consequence of the bound, and this section tests whether the consequence holds: the margin over uniform averaging with a fixed schedule should be non-negative at every budget (P1), grow with the budget (P2), and collapse when the influence is replaced by a proxy that orders tokens differently (P3). The protocol is ImageNet-1k validation (Deng et al., 2009) at 224^2 with the `augreg2` ViT-B/16 checkpoint (Steiner et al., 2022) (85.19% Top-1, 17.45 GFLOPs), official implementations of ToMe, PiToMe, and DSM, matched FLOPs, evaluation batches of 64 (over which the budget statistic is averaged), and AdaMerge as mean±std over five runs; calibration uses a 1% training subset unless stated (Appendix L).

6.1 Main Result: The Margin Is Non-Negative and Widens (P1, P2)

Table 1 and Figure 2 compare the four methods at six FLOPs tiers from 8.8% to 23.1% reduction. AdaMerge has the highest accuracy at every tier, and the margin over ToMe (Table 4) rises from $+0.01$ to $+1.67$ pp with increments $0.17, 0.19, 0.24, 0.53, 0.53$ pp, none of them negative; the $+0.01$ at the mildest tier is parity within noise. Two features of the table matter for the mechanism. First, PiToMe, which partitions tokens by an importance score, sits between ToMe and AdaMerge at four of six tiers and ties ToMe at a fifth: protecting influential tokens at partition time helps, and the influence-informed budget adds to it. Second, DSM, whose schedule is fixed by depth, loses 4.62 pp at the aggressive tier where AdaMerge loses 1.06. Full sweeps are in Tables 3–7 (Appendix B).

6.2 The Widening Is a Property of the Penalty, Not of ViT-B (P2)

If the widening came from the penalty rather than from one checkpoint, it should appear on other backbones with no re-tuning, and for any method that spends a fixed budget in the deep blocks. With the same calibration and no other change, the margin over ToMe is monotone on all four backbones

Table 1: FLOPs-matched comparison on ImageNet-1k with ViT-B/16 at six tiers. AdaMerge is mean±std over 5 runs; **bold** is the best accuracy per tier. d is DSM's `delay_layer`. Speedup is measured throughput relative to the uncompressed model (A6000, batch 64).

Tier	Method	Top-1 Acc	Δ Acc	FLOPs (G)	Speedup
∼15.9G	AdaMerge ($r_{\max}$=9)	**84.70**$_{\pm 0.01}$%	**−0.49%**	15.91	1.01×
	ToMe (r=3)	84.69%	−0.50%	15.93	1.10×
	PiToMe (r=3)	84.57%	−0.62%	15.93	1.04×
	DSM (r=12, d=6)	84.42%	−0.77%	15.95	1.06×
∼15.5G	AdaMerge ($r_{\max}$=11)	**84.62**$_{\pm 0.00}$%	**−0.57%**	15.54	1.05×
	ToMe (r=4)	84.44%	−0.75%	15.43	1.10×
	PiToMe (r=4)	84.44%	−0.75%	15.43	1.04×
	DSM (r=18, d=6)	83.68%	−1.51%	15.48	1.09×
∼14.9G	AdaMerge ($r_{\max}$=14)	**84.49**$_{\pm 0.03}$%	**−0.70%**	14.98	1.09×
	ToMe (r=5)	84.12%	−1.07%	14.93	1.15×
	PiToMe (r=5)	84.29%	−0.90%	14.93	1.08×
	DSM (r=16, d=5)	83.35%	−1.84%	14.90	1.12×
∼14.4G	AdaMerge ($r_{\max}$=17)	**84.28**$_{\pm 0.02}$%	**−0.91%**	14.39	1.13×
	ToMe (r=6)	83.67%	−1.52%	14.43	1.19×
	PiToMe (r=6)	84.22%	−0.97%	14.43	1.12×
	DSM (r=14, d=4)	82.51%	−2.68%	14.37	1.21×
∼13.9G	AdaMerge ($r_{\max}$=20)	**84.25**$_{\pm 0.03}$%	**−0.94%**	13.94	1.17×
	ToMe (r=7)	83.11%	−2.08%	13.94	1.25×
	PiToMe (r=7)	83.65%	−1.54%	13.94	1.17×
	DSM (r=17, d=4)	81.05%	−4.14%	13.90	1.26×
∼13.4G	AdaMerge ($r_{\max}$=23)	**84.13**$_{\pm 0.02}$%	**−1.06%**	13.42	1.22×
	ToMe (r=8)	82.46%	−2.73%	13.44	1.31×
	PiToMe (r=8)	83.74%	−1.45%	13.44	1.23×
	DSM (r=20, d=4)	80.57%	−4.62%	13.47	1.28×

Table 2: Four backbones, one calibration recipe. Margin of AdaMerge over ToMe (pp) at matched FLOPs from the mildest to the most aggressive tier, and the three-way comparison with PiToMe. Full per-tier tables and throughput: Appendix C.

Backbone	Params	Tiers	Margin over ToMe	PiToMe at matched FLOPs
ViT-S/16	22M	5	+0.86 → +3.06	AdaMerge ahead by +0.27 to +0.56 at all three matched tiers
ViT-B/16	86M	6	+0.01 → +1.67	AdaMerge ahead at all six tiers (Table 1)
DeiT-B/16	86M	4	+0.25 → +1.08	— (wall-clock parity with ToMe)
ViT-L/16	304M	4	+0.53 → +4.64	PiToMe +0.93 → +5.56 over ToMe; 0.40–1.31 above AdaMerge, which is fastest at 3 of 4 tiers

(Table 2). PiToMe reproduces the pattern independently, and the table also records the one ordering that runs against AdaMerge: on the supervised ViT-L/16 checkpoint PiToMe is 0.40–1.31 pp more accurate at matched FLOPs while AdaMerge is faster (Table 10). Token Adapter (Su et al., 2024) trails AdaMerge by 0.71 and 1.27 pp (Table 11).

6.3 Throughput

On the FLOPs axis AdaMerge is the upper envelope at every tier (Figure 2b); on measured throughput it is not, offering higher accuracy at 5–9% lower throughput than ToMe at each tier of Table 1. Eq. 11 says why and when this reverses: the overhead scales with $N^2 d$ and the block with Nd^2, and across the other backbones the measured gap moves as predicted, from 6–9% slower on ViT-S through parity on DeiT-B to 7% faster on ViT-L, where AdaMerge is faster *and* more accurate than ToMe at every operating point (Table 24, Appendix H.2).

6.4 Where the Budget Goes

The schedule was given no per-layer prior, so where it spends the budget is a measurement: over the 50,000 validation images, at $r_{\max} \leq 11$ the last block merges most ($\mathbb{E}[r_{11}] = 5.18$), at $r_{\max} \in \{14, 17\}$ blocks 0–1 lead and block 11 contracts ($3.12 \rightarrow 0.84$), and at $r_{\max} \geq 20$ blocks $l \geq 10$ ($l \geq 9$ at 23) merge nothing (Table 21, Figure 4). The budget leaves the blocks where Table 16 says a merge costs most; per-image maps are in Appendix J (Figures 5–7).

7 Analysis

Neither half works alone (leave-one-out). Table 13 (Appendix E.1) removes each decision in turn at the $\sim$15.0G tier. Influence-weighted matching and aggregation under ToMe's fixed budget loses 0.15 pp to ToMe; the adaptive budget under ToMe's uniform similarity loses 0.48 pp; both together gain 0.33 pp. With uniform similarity the statistic of Eq. 8 collapses to the mean best cosine, which measures proximity rather than the cost of Eq. 4, so the gate opens on layers that are similar but not cheap and merges 68.8 tokens where the calibrated method merges 47.2. Merging fewer tokens is not the explanation: at $r_{\max}$=24 full AdaMerge merges 70.7 tokens and still leads the budget-only variant by 0.51 pp. The two decisions are coupled because they read the same objective (Appendix E.1).

A wrong influence is worse than none (P3, counterfactual). Corollary 1 predicts that replacing the influence by a proxy that orders tokens differently should cost more than uniform averaging. Swapping only the influence signal at $r_{\max}$=17, the [CLS]-attention score of EViT (Liang et al., 2022b) drops accuracy from 84.14 to 75.55% and the feature ℓ_2 norm to 80.94%, both below ToMe's 82.46% at the matched 13.4G tier. FLOPs differ at fixed $r_{\max}$ because the budget reacts to the signal (ℓ_2 merges 109.9 tokens, ours 51.2), so the FLOPs-matched comparison to ToMe is the controlled one (Table 14).

Fine-grained classes, calibration, fine-tuning. At 13.4G the 118 dog breeds lose 1.80 pp under AdaMerge and 4.51 pp under ToMe (Table 23); 256 calibration images match the full 1% within 0.04 pp (Table 19, Appendix F); after 30 epochs of fine-tuning AdaMerge leads ToMe by +0.42, +0.57, +0.72 pp, again widening (Table 12).

Transfer to a vision-language model is a check, not a claim. On LLaVA-1.5-7B (Liu et al., 2024) (CLIP ViT-L/14 tower (Radford et al., 2021)), AdaMerge leads ToMe and PiToMe on POPE (Li et al., 2023) and GQA (Hudson & Manning, 2019) at the moderate tier within sampling error and does not widen; at the aggressive tier both fixed-schedule methods score *above* the uncompressed model on POPE (+0.86, +1.07 pp) through a shift toward "yes" answers, while AdaMerge's answer distribution stays at baseline (Table 25).

8 Limitations

The bound is first-order: it holds the affinity fixed across a merge, so we validate its consequences by accuracy rather than by the bound alone. Our study covers image classification at 224^2 and one vision-language model; video, dense prediction, and self-supervised backbones are outside its scope, as for the methods we compare against. The schedule needs one calibration pass per backbone and budget on a few hundred unlabeled images, and its wall-clock advantage depends on model width (Eq. 11).

9 Conclusion

A merge has a price, computable from one forward pass without labels, and it is paid in depth. We proved the bound, showed it tight on 52,190 pairs, and found that the network absorbs the part the average creates (0.06 pp for a 2× reduction) but not the part depth creates (5–11× the marginal rate). A schedule that reads the bound widens its margin over ToMe on four backbones; the bound audits any merge before a label is seen.

Reproducibility Statement

The supplementary material contains the anonymized code, the precomputed `stats.json`, and commands reproducing every number in Table 1; checkpoints, preprocessing, seeds, and hardware are in Appendix L and proofs in Appendix A.

AI Use Statement

Large language models were used for editing (grammar, wording, and LaTeX formatting) and to check the algebra of Appendix A against the statements in Section 3. They were not used to design the method, run or analyze experiments, generate results, or write the related-work survey; all references were verified by the authors against the original venues.

References

Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Alexander C. Aitken. On least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 55:42–48, 1935.

Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations (ICLR)*, 2023.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *International Conference on Computer Vision (ICCV)*, 2021.

Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

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 (ECCV)*, 2024.

Mengzhao Chen, Wenqi Shao, Peng Xu, Mingbao Lin, Kaipeng Zhang, Fei Chao, Rongrong Ji, Yu Qiao, and Ping Luo. DiffRate: Differentiable compression rate for efficient vision transformers. In *International Conference on Computer Vision (ICCV)*, 2023.

Hyesong Choi, Hyejin Park, Kwang Moo Yi, Sungmin Cha, and Dongbo Min. Salience-based adaptive masking: Revisiting token dynamics for enhanced pre-training. In *European Conference on Computer Vision (ECCV)*, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International Conference on Learning Representations (ICLR)*, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. In *International Conference on Learning Representations (ICLR)*, 2020.

Mohsen Fayyaz, Soroush Abbasi Koohpayegani, Farnoush Rezaei Jafari, Sunando Sengupta, Hamid Reza Vaezi Joze, Eric Sommerlade, Hamed Pirsiavash, and Jürgen Gall. Adaptive token sampling for efficient vision transformers. In *European Conference on Computer Vision (ECCV)*, 2022.

Michael Figurnov, Maxwell D. Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh M. Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. PoWER-BERT: Accelerating BERT inference via progressive word-vector elimination. In *International Conference on Machine Learning (ICML)*, 2020.

Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2022.

Joakim Bruslund Haurum, Sergio Escalera, Graham W. Taylor, and Thomas B. Moeslund. Which tokens to use? Investigating token reduction in vision transformers. In *International Conference on Computer Vision Workshops (ICCVW)*, pp. 773–783, 2023.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Jung Hwan Heo, Seyedarmin Azizi, Arash Fayyazi, and Massoud Pedram. Training-free acceleration of ViTs with delayed spatial merging. In *ICML Workshop on Efficient Systems for Foundation Models (ES-FoMo)*, 2024.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q. Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations (ICLR)*, 2018.

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

Minchul Kim, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. Token fusion: Bridging the gap between token pruning and token merging. In *Winter Conference on Applications of Computer Vision (WACV)*, 2024.

Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. Learned token pruning for transformers. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2022.

Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, Minghai Qin, and Yanzhi Wang. SPViT: Enabling faster vision transformers via latency-aware soft token pruning. In *European Conference on Computer Vision (ECCV)*, 2022.

Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems (NIPS)*, 1990.

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

Weicong Liang, Yuhui Yuan, Henghui Ding, Xiao Luo, Weihong Lin, Ding Jia, Zheng Zhang, Chao Zhang, and Han Hu. Expediting large-scale vision transformer for dense prediction without fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022a.

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. In *International Conference on Learning Representations (ICLR)*, 2022b.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Sifan Long, Zhen Zhao, Jimin Pi, Shengsheng Wang, and Jingdong Wang. Beyond attentive tokens: Incorporating token importance and diversity for efficient vision transformers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Dmitrii Marin, Jen-Hao Rick Chang, Anurag Ranjan, Anish Prabhu, Mohammad Rastegari, and Oncel Tuzel. Token pooling in vision transformers for image classification. In *Winter Conference on Applications of Computer Vision (WACV)*, 2023.

Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. AdaViT: Adaptive vision transformers for efficient image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *International Conference on Learning Representations (ICLR)*, 2017.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2024.

Bowen Pan, Rameswar Panda, Yifan Jiang, Zhangyang Wang, Rogerio Feris, and Aude Oliva. IA-RED^2: Interpretability-aware redundancy reduction for vision transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Namuk Park and Songkuk Kim. How do vision transformers work? In *International Conference on Learning Representations (ICLR)*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Michael S. Ryoo, Anurag Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. TokenLearner: Adaptive space-time tokenization for videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your ViT? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research (TMLR)*, 2022.

Diwei Su, Cheng Fei, and Jianxu Luo. Removing rows and columns of tokens in vision transformer enables faster dense prediction without retraining. In *European Conference on Computer Vision (ECCV)*, 2024.

Yehui Tang, Kai Han, Yunhe Wang, Chang Xu, Jianyuan Guo, Chao Xu, and Dacheng Tao. Patch slimming for efficient vision transformers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning (ICML)*, 2021.

Hoai-Chau Tran, Duy M. H. Nguyen, Duy M. Nguyen, TrungTin Nguyen, Ngan Le, Pengtao Xie, Daniel Sonntag, James Zou, Binh T. Nguyen, and Mathias Niepert. Accelerating transformers with spectrum-preserving token merging. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yulin Wang, Rui Huang, Shiji Song, Zeyi Huang, and Gao Huang. Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Xuwei Xu, Sen Wang, Yudong Chen, Yanping Zheng, Zhewei Wei, and Jiajun Liu. GTP-ViT: Efficient vision transformers via graph-based token propagation. In *Winter Conference on Applications of Computer Vision (WACV)*, 2024.

Yifan Xu, Zhijie Zhang, Mengdan Zhang, Kekai Sheng, Ke Li, Weiming Dong, Liqing Zhang, Changsheng Xu, and Xing Sun. Evo-ViT: Slow-fast token evolution for dynamic vision transformer. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

Hongxu Yin, Arash Vahdat, Jose Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-ViT: Adaptive tokens for efficient vision transformer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Wang Zeng, Sheng Jin, Wentao Liu, Chen Qian, Ping Luo, Wanli Ouyang, and Xiaogang Wang. Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

APPENDIX OVERVIEW

- **A Proofs** of Lemma 1, Proposition 1, Corollary 1, and the identities of Eqs. 4–6.
- **B Full sweeps** on ViT-B/16 for ADAMERGE, TOME, PiToMe, and DSM.
- **C Other backbones** (ViT-S, DeiT-B, ViT-L), Token Adapter, and **D fine-tuning**.
- **E Ablations**: coupling, proxy swap, direct measurement of the bound (tightness, depth profile, aggregation-only accuracy), temperature.
- **F Calibration**: fixed point, per-layer statistics, initialization, data efficiency, no-calibration control.
- **G Where the budget goes**, the forced-activation control, and class-stratified accuracy.
- **H Wall-clock** across scale and the overhead model.
- **I LLaVA-1.5**, **J qualitative maps**, **K algorithms**, **L reproducibility**.

A PROOFS AND IDENTITIES

Weighted-mean inequality. For $\alpha, \beta \geq 0$ and $u, v \in \mathbb{R}^d$, $\|\alpha u+\beta v\|^2 \leq (\alpha+\beta)(\alpha\|u\|^2+\beta\|v\|^2)$. *Proof.* Expand the left side: $\alpha^2\|u\|^2 + \beta^2\|v\|^2 + 2\alpha\beta\, u^\top v$. The right side is $\alpha^2\|u\|^2 + \beta^2\|v\|^2 + \alpha\beta(\|u\|^2 + \|v\|^2)$. The difference is $\alpha\beta\,\|u - v\|^2 \geq 0$. □

Lemma 1. With $\alpha = \hat{A}_{ki}$, $\beta = \hat{A}_{kj}$, $u = \tilde{x} - x_i$, $v = \tilde{x} - x_j$, the inequality gives $\|\Delta y_k\|^2 \leq (\hat{A}_{ki} + \hat{A}_{kj})(\hat{A}_{ki}\|u\|^2 + \hat{A}_{kj}\|v\|^2)$, and $\hat{A}_{ki} + \hat{A}_{kj} \leq \sum_m \hat{A}_{km} = 1$. Summing over k and using $\sum_k \hat{A}_{ki} = s_i$ gives Eq. 1. The bound is tight when, for every k, $\hat{A}_{ki} + \hat{A}_{kj} = 1$ and $u = v$, i.e., when the two tokens are identical and the row attends to nothing else. □

Proposition 1, in the weight parametrization. For $\tilde{x} = wx_i + (1-w)x_j$ and $D = \|x_i - x_j\|^2$, $\tilde{x} - x_i = -(1-w)(x_i - x_j)$ and $\tilde{x} - x_j = w(x_i - x_j)$, so $J_{ij}(w) = \big(s_i(1-w)^2 + s_j w^2\big)D$, a convex parabola with minimizer $w^\star = s_i/(s_i + s_j)$ and minimum $J_{ij}(w^\star) = \frac{s_i s_j}{s_i+s_j} D$. At $w = \frac{1}{2}$, $J_{ij} = \frac{s_i+s_j}{4} D$. The difference is $\frac{(s_i+s_j)^2 - 4s_i s_j}{4(s_i+s_j)} D = \frac{(s_i-s_j)^2}{4(s_i+s_j)} D$. Minimizing over all $\tilde{x} \in \mathbb{R}^d$ rather than over the segment gives the same point because J_{ij} is a weighted sum of two squared distances and its unconstrained minimizer is the weighted centroid. □

Corollary 1. $J_{ij}(w) - J_{ij}(w^\star) = (s_i + s_j)(w - w^\star)^2 D$ follows by completing the square in the parabola above; the comparison with uniform averaging is the case $w = \frac{1}{2}$. □

Harmonic-mean form (Eq. 4). $\frac{s_i s_j}{s_i+s_j} = \frac{1}{2} \cdot \frac{2 s_i s_j}{s_i+s_j} = \frac{1}{2}\mathrm{H}(s_i, s_j)$, and $\mathrm{H}(s_i, s_j) \leq \min(s_i, s_j) \leq s_i$.

Surrogate identity (Eq. 6). For unit-norm x_i, x_j, $\|x_i - x_j\|^2 = 2 - 2\cos(x_i, x_j)$. Then $U_{ij} = \hat{s}_i\|x_i - x_j\|^2 = 2\hat{s}_i(1-\cos(x_i, x_j)) \geq J^\star_{ij}$ by the previous paragraph, and $\hat{s}_i - \frac{1}{2}U_{ij} = \hat{s}_i - \hat{s}_i(1-\cos) = \hat{s}_i \cos = \tilde{S}_{ij}$.

Proposition 2. $P(r) = \sum_{k \leq r} p_k$ with $p_k \geq 0$ is non-decreasing for any sequence of pairs; this uses only the sign of Eq. 2 applied pair by pair and no assumption on how pairs interact. Under (A1)–(A2), p_k is the product of $\frac{1}{4}(s_{i_k} - s_{j_k})^2/(s_{i_k} + s_{j_k})$ and $2(1 - \cos_k)$, both non-negative and non-decreasing in k, hence non-decreasing; a partial-sum sequence with non-decreasing increments is convex. □

B FULL SWEEPS ON VIT-B/16

ADAMERGE across $r_{\max}$ (Table 3). Accuracy decays sub-linearly in the compression: doubling the FLOPs reduction from 7.7% to 16.3% roughly doubles the loss ($-0.43 \to -0.87$ pp), and a further increase to 23.1% adds only 0.19 pp. Configurations $r_{\max} \in \{12, 13, 14\}$ form a plateau at ∼84.5%, and the merged-token count grows smoothly with $r_{\max}$ (24.7 to 68.2), so $r_{\max}$ acts as a continuous knob rather than a discrete schedule.

Baselines (Tables 5–7). TOME's loss accelerates with r (-0.75 at r=4, -2.73 at r=8), the signature of a penalty whose increments grow (Proposition 2). PiToMe degrades more slowly than TOME, and its r=7 vs. r=8 non-monotonicity (83.65 vs. 83.74%) reproduces with the official code; it stems from the energy-threshold partition shifting discontinuously between r values. DSM's loss grows as its delay layer is moved earlier, reaching -4.62 pp at d=4, r=20.

Table 3: AdaMerge sweep across $r_{\max}$ (ViT-B/16; mean $\pm$ std over 5 runs).

Setting	Top-1 Acc	Δ Acc	FLOPs (G)	FLOPs $\downarrow$	Speedup	Merged Tokens
Baseline	85.19%	–	17.45	–	1.00×	0
$r_{\max} = 8$	$84.76_{\pm 0.01}\%$	-0.43%	16.10	7.7%	1.01×	24.7
$r_{\max} = 9$	$84.70_{\pm 0.01}\%$	-0.49%	15.91	8.8%	1.01×	27.8
$r_{\max} = 10$	$84.66_{\pm 0.02}\%$	-0.53%	15.73	9.9%	1.03×	30.8
$r_{\max} = 11$	$84.62_{\pm 0.00}\%$	-0.57%	15.54	10.9%	1.05×	33.9
$r_{\max} = 12$	$84.49_{\pm 0.02}\%$	-0.70%	15.36	12.0%	1.06×	36.7
$r_{\max} = 13$	$84.53_{\pm 0.01}\%$	-0.66%	15.20	12.9%	1.07×	39.2
$r_{\max} = 14$	$84.49_{\pm 0.03}\%$	-0.70%	14.98	14.1%	1.09×	42.6
$r_{\max} = 15$	$84.34_{\pm 0.04}\%$	-0.85%	14.78	15.3%	1.10×	45.6
$r_{\max} = 16$	$84.32_{\pm 0.03}\%$	-0.87%	14.61	16.3%	1.12×	48.1
$r_{\max} = 17$	$84.28_{\pm 0.02}\%$	-0.91%	14.39	17.5%	1.13×	51.7
$r_{\max} = 18$	$84.25_{\pm 0.01}\%$	-0.94%	14.27	18.2%	1.14×	53.4
$r_{\max} = 19$	$84.22_{\pm 0.03}\%$	-0.97%	14.08	19.3%	1.15×	56.3
$r_{\max} = 20$	$84.25_{\pm 0.03}\%$	-0.94%	13.94	20.1%	1.17×	58.1
$r_{\max} = 21$	$84.18_{\pm 0.03}\%$	-1.01%	13.72	21.4%	1.19×	61.6
$r_{\max} = 22$	$84.16_{\pm 0.04}\%$	-1.03%	13.55	22.3%	1.20×	63.8
$r_{\max} = 23$	$84.13_{\pm 0.02}\%$	-1.06%	13.42	23.1%	1.22×	65.3
$r_{\max} = 24$	$84.11_{\pm 0.01}\%$	-1.08%	13.22	24.2%	1.23×	68.2

Table 4: Margin of AdaMerge over ToMe (pp) at the six matched-FLOPs tiers of Table 1, and its increments. The margin is non-negative at every tier (P1) and its increments are non-negative (P2).

Tier	∼15.9G	∼15.5G	∼14.9G	∼14.4G	∼13.9G	∼13.4G
Margin	+0.01	+0.18	+0.37	+0.61	+1.14	+1.67
Increment	–	0.17	0.19	0.24	0.53	0.53

Table 5: ToMe sweep across r (ViT-B/16; official implementation, reproduction within ± 0.05 pp of the published numbers).

Setting	Top-1 Acc	Δ Acc	FLOPs (G)	FLOPs $\downarrow$	Speedup	Merged Tokens
Baseline	85.19%	–	17.45	–	1.00×	0
ToMe $r = 3$	84.69%	-0.50%	15.93	8.7%	1.10×	36
ToMe $r = 4$	84.44%	-0.75%	15.43	11.6%	1.10×	48
ToMe $r = 5$	84.12%	-1.07%	14.93	14.4%	1.15×	60
ToMe $r = 6$	83.67%	-1.52%	14.43	17.3%	1.19×	72
ToMe $r = 7$	83.11%	-2.08%	13.94	20.1%	1.25×	84
ToMe $r = 8$	82.46%	-2.73%	13.44	23.0%	1.31×	96

Table 6: PiToMe sweep across r (ViT-B/16). Official implementation (https://github.com/hchautran/PiToMe) with default energy threshold on the `augreg2` checkpoint; Tran et al. (2024) report classification on ViT-MAE-H, so published numbers are not directly comparable, and our run is cross-validated against the official code under identical conditions.

Setting	Top-1 Acc	Δ Acc	FLOPs (G)	FLOPs $\downarrow$	Speedup	Merged Tokens
Baseline	85.19%	–	17.45	–	1.00×	0
PiToMe $r = 3$	84.57%	-0.62%	15.93	8.7%	1.00×	36
PiToMe $r = 4$	84.44%	-0.75%	15.43	11.6%	1.04×	48
PiToMe $r = 5$	84.29%	-0.90%	14.93	14.4%	1.08×	60
PiToMe $r = 6$	84.22%	-0.97%	14.43	17.3%	1.12×	72
PiToMe $r = 7$	83.65%	-1.54%	13.94	20.1%	1.17×	84
PiToMe $r = 8$	83.74%	-1.45%	13.44	23.0%	1.23×	96

Table 7: DSM sweep across `delay_layer` and r (ViT-B/16).

`delay_layer`	r	Top-1 Acc	Δ Acc	FLOPs (G)	FLOPs $\downarrow$	Speedup
6	12	84.42%	−0.77%	15.95	8.6%	1.06×
	13	84.10%	−1.09%	15.85	9.2%	1.08×
	14	83.98%	−1.21%	15.78	9.6%	1.09×
	15	83.85%	−1.34%	15.70	10.0%	1.10×
	16	83.70%	−1.49%	15.63	10.4%	1.10×
	17	83.52%	−1.67%	15.55	10.9%	1.11×
	18	83.68%	−1.51%	15.48	11.3%	1.09×
5	12	84.07%	−1.12%	15.40	11.8%	1.10×
	13	83.61%	−1.58%	15.24	12.7%	1.12×
	14	83.43%	−1.76%	15.13	13.3%	1.15×
	15	83.13%	−2.06%	15.01	13.9%	1.16×
	16	83.35%	−1.84%	14.90	14.6%	1.12×
	17	82.64%	−2.55%	14.79	15.2%	1.18×
	18	82.81%	−2.38%	14.68	15.9%	1.17×
4	12	83.52%	−1.67%	14.75	15.4%	1.15×
	13	82.83%	−2.36%	14.53	16.7%	1.19×
	14	82.51%	−2.68%	14.37	17.6%	1.21×
	15	82.20%	−2.99%	14.22	18.5%	1.23×
	16	82.49%	−2.70%	14.11	19.1%	1.21×
	17	81.05%	−4.14%	13.90	20.3%	1.26×
	18	81.53%	−3.66%	13.75	21.2%	1.26×
	20	80.57%	−4.62%	13.47	22.8%	1.28×

C OTHER BACKBONES

All three additional backbones use the same two-pass calibration (Section 4.4) and no other change. On DeiT-B/16, TOME $r \in \{4, 5, 6, 8\}$ is matched with ADAMERGE $r_{\max} \in \{8, 9, 11, 14\}$ (ADAMERGE at 0.06–0.25 G lower FLOPs in each pair), giving margins $+0.25$, $+0.33$, $+0.53$, $+1.08$ pp. Table 8 summarizes the margin over TOME; Tables 9 and 10 give the three-way comparisons on ViT-S/16 and ViT-L/16.

Table 8: Margin (ADAMERGE − TOME, pp) at matched FLOPs from the mildest to the most aggressive tier.

Backbone	Margin over TOME (mild → aggressive)	Trend
ViT-S/16 (5 tiers)	$+0.86 \rightarrow +3.06$	monotone
DeiT-B/16 (4 tiers)	$+0.25 \rightarrow +1.08$	monotone
ViT-L/16 (4 tiers)	$+0.53 \rightarrow +4.64$ (81.10 vs. 76.46% at 1.77×/1.72×)	monotone
ViT-B/16 (6 tiers, Table 4)	$+0.01 \rightarrow +1.67$	monotone

Table 9: ViT-S/16 (baseline 81.40%): TOME, PiToMe, and ADAMERGE at matched FLOPs.

Tier	r	TOME	PiToMe	ADAMERGE	Δ(ADAMERGE − PiToMe)
∼3.75G	6	78.57%	79.81% (3.73G)	$80.32_{\pm 0.01}$% (3.77G)	+0.51
∼3.59G	7	77.69%	79.45% (3.60G)	$79.72_{\pm 0.02}$% (3.58G)	+0.27
3.47G (exact)	8	76.56%	79.06% (3.47G)	$79.62_{\pm 0.03}$% (3.47G)	+0.56

Table 10: ViT-L/16 (baseline 85.90%, 62.4 img/s): Top-1 / measured throughput (img/s, RTX 3090, protocol of Appendix H). †Fixed-r PiToMe validated against the official implementation on ViT-B/16 within 0.03 pp.

Tier	TOME	PiToMe†	ADAMERGE
∼47G	84.00 / 79.3	84.93 / 79.4	$84.53_{\pm 0.04}$ / 82.4
∼44G	82.65 / 87.9	84.50 / 90.6 (43.1G)	$83.98_{\pm 0.05}$ / 88.7 (44.7G)
∼39.5G	80.44 / 94.3	83.56 / 96.2	$82.25_{\pm 0.07}$ / 102.5
∼36G	76.46 / 107.0	82.02 / 108.9	$81.10_{\pm 0.04}$ / 111.3

Reading Table 10. Both importance-aware methods widen over TOME monotonically (ADAMERGE $+0.53$, $+1.33$, $+1.81$, $+4.64$; PiToMe $+0.93$, $+1.85$, $+3.12$, $+5.56$), which is the prediction of Proposition 2 confirmed through a method we did not design. Within the pair, PiToMe leads on accuracy by 0.40–1.31 pp on this checkpoint and ADAMERGE leads on throughput at three of four tiers. The two act on different steps (partition vs. aggregation and budget); Section 8 states what the analysis predicts for their combination.

Token Adapter in our protocol (Table 11). Hourglass (Liang et al., 2022a) and Token Adapter (Su et al., 2024) report speedups for dense prediction and, for ImageNet, on SWAG ViT-L/16 at 512^2 (1,024 tokens); neither reports the 196-token 224^2 protocol. We reimplemented Token Adapter's layer-adapter mechanism (layers 3/7, unique-retention 0.2) on ViT-B/16 `augreg2`, picking the closest tier from below at each budget. Its published characteristic reproduces, trailing TOME at the milder tier and overtaking it at the more aggressive one, and ADAMERGE leads it by $+0.71$ and $+1.27$ pp.

Table 11: Token Adapter reimplemented at 224^2 on ViT-B/16 `augreg2`.

Tier	Token Adapter	ToMe	PiToMe	AdaMerge
∼14.9G	83.78% (14.46G)	84.12%	84.29%	84.49%
∼13.4G	82.86% (13.35G)	82.46%	83.74%	84.13%

D Fine-Tuning

ViT-B/16 is fine-tuned jointly with the merging module: AdamW, learning rate 5×10^{-6}, 5 warm-up epochs then cosine decay, 30 epochs, batch 256 on 3 GPUs (DDP), weight decay 0.05; AdaMerge ($r_{\max} \in \{12, 14, 17\}$) against ToMe ($r \in \{4, 5, 6\}$) at matched FLOPs. DSM and PiToMe are excluded because their extra hyperparameters prevent matched conditions.

Table 12: Fine-tuned accuracy at matched FLOPs (30 epochs).

FLOPs tier	Method	Top-1 Acc	Δ Acc	Δ vs. ToMe
∼15.4G	AdaMerge ($r_{\max}$=12)	83.92%	−1.27%	+0.42
	ToMe (r=4)	83.50%	−1.69%	–
∼15.0G	AdaMerge ($r_{\max}$=14)	83.92%	−1.27%	+0.57
	ToMe (r=5)	83.35%	−1.84%	–
∼14.4G	AdaMerge ($r_{\max}$=17)	83.90%	−1.29%	+0.72
	ToMe (r=6)	83.18%	−2.01%	–

The margin widens with compression ($0.42 \to 0.57 \to 0.72$ pp), matching the training-free pattern at the same tiers ($0.04 \to 0.37 \to 0.61$ pp), and the three AdaMerge runs land within 0.03 pp of one another while ToMe degrades by 0.16 pp per step of r. Both methods end below their training-free accuracy (∼83.9 vs. ∼84.5%); the schedule is conservative by design and is not a recipe for the reduced-token regime.

E Ablations

E.1 Why the Budget Rule Needs the Influence (Coupling)

Table 13: Leave-one-out ablation on ViT-B/16 (ToMe r=5, $r_{\max}$=14, ∼15.0G). SW: influence-weighted similarity and aggregation; Adp: adaptive budget. †Adaptive variants merge a variable number of tokens; the $r_{\max}$=24 row matches the merge count of the budget-only variant.

Configuration	SW	Adp	Top-1 Acc	Δ vs. ToMe	Avg. merges
ToMe (r=5)	✗	✗	84.10%	–	60.0
SW only (r=5)	✓	✗	83.95%	−0.15	60.0
Adp only†	✗	✓	83.62%	−0.48	68.8
AdaMerge ($r_{\max}$=14)†	✓	✓	**84.43%**	+0.33	47.2
AdaMerge ($r_{\max}$=24)†	✓	✓	84.13%	+0.03	70.7

Table 13 gives the numbers. The budget-only variant computes $\bar{S}_l$ from unweighted cosine similarity. By Eq. 6 the weighted statistic is $\bar{\hat{s}} - \frac{1}{2}\overline{U}$: mean retained influence minus half the mean cost bound. Dropping the weight replaces it with $1 - \frac{1}{2}\overline{\|x_i - x_{\pi(i)}\|^2}$, which no longer knows whether the similar pairs carry influence. Layers where influential tokens have only distant partners but unimportant tokens have many near-duplicates then look redundant, the gate opens, and the merge budget is spent on pairs with large $J^\star$. This is why the variant over-merges (68.8 tokens) and still loses accuracy, and why full AdaMerge at a larger $r_{\max}$ can merge more tokens (70.7) and lead it by 0.51 pp: the difference is in which pairs, not how many.

E.2 Influence Proxy Swap

Only the influence signal is replaced; $r_{\max}$=17 and everything else are fixed. FLOPs differ across rows because the budget rule reacts to the signal it is given.

Table 14: Proxy swap at $r_{\max}$=17 on ViT-B/16. Affinity centrality reuses $\hat{\mathbf{A}}$; the alternatives need additional activations or a backward pass.

Proxy	Top-1 Acc	FLOPs (G)	Δ vs. ours	Extra access
Affinity centrality (ours, Eq. 3)	$84.14_{\pm 0.02}\%$	14.42	–	none (reuses $\hat{\mathbf{A}}$)
[CLS]-attention (EViT-style, prev. layer)	$75.55_{\pm 0.07}\%$	13.47	-8.59	QKV projection
Feature ℓ_2 norm	$80.94_{\pm 0.03}\%$	12.41	-3.20	$\mathcal{O}(Nd)$
Gradient saliency	n/a	–	–	backward pass

At the FLOPs-matched 13.4G tier the [CLS] proxy trails AdaMerge by 8.58 pp (75.55 vs. 84.13%) and uniform ToMe (82.46%) beats both proxies. Same-layer attention is unavailable because AdaMerge runs before the block's attention; the previous-layer [CLS] row is the realizable alternative, and it ranks tokens by relevance to one query rather than by the mass the whole sequence places on them. Corollary 1 makes the outcome expected: a weight computed from a proxy that disagrees with the influence in ordering lands farther from $w^\star$ than $\frac{1}{2}$ does.

E.3 Direct Measurement of the Bound

Lemma 1 and Proposition 1 are statements about one forward pass, so we checked them on one, with no training and no baseline. On the 256-image calibration set (train split, seed 42) we run ViT-B/16 with the paper's schedule at $r_{\max}$=17 and, at every block, for every pair the matching of Eq. 5 selects (52,190 pairs in total), record the realized perturbation $\|\Delta\mathbf{Y}\|_F^2$ of the affinity aggregate and the bound J_{ij} for three merge rules applied to the same pair: the uniform average, the proportional average with raw influence s (Proposition 1), and the proportional average with the rescaled $\hat{s}$ that the implementation uses. Separately, we evaluate ImageNet-1k Top-1 for the three rules with matching and budget held fixed. The Top-1 column was obtained on a second machine (RTX-class GPU, batch 64) on which the default configuration reproduces at 84.13% against 84.28% on the A6000 of Table 1; the machine offset is 0.15 pp and the within-machine comparison is what the column supports.

Table 15: Realized downstream perturbation and bound per merged pair (mean over 52,190 pairs, 12 blocks, 256 images; ViT-B/16, $r_{\max}$=17). "Bound holds" is the fraction of pairs with $\|\Delta\mathbf{Y}\|_F^2 \le J_{ij}$; ratio is $\bar{J}/\bar{J}^\star$ with $J^\star$ the closed-form minimum. Top-1: ImageNet-1k with the given aggregation and everything else fixed (second machine; see text). The raw–default Top-1 difference is not significant (McNemar p=0.52).

Aggregation	$\|\Delta\mathbf{Y}\|_F^2$	J_{ij}	Bound holds	$\|\Delta\mathbf{Y}\|_F^2/J$	Top-1
Uniform (w=$\frac{1}{2}$)	207.9	210.8	100.0%	0.987	84.07%
Proportional, raw s (Prop. 1)	91.4	92.9	100.0%	0.984	84.07%
Proportional, rescaled $\hat{s}$ (default)	91.7	93.1	100.0%	0.984	84.13%
Ratio uniform / proportional	2.27	2.27	–	–	-0.06 pp

What the measurement establishes. (i) *The bound is real and tight.* It holds on every pair for both rules, and the realized perturbation is within 1.6% of it on average and within 8% at the loosest block (block 1). The first-order surrogate $\hat{\mathbf{A}}\mathbf{X}$ is therefore not a loose proxy for what the merge does to the aggregate. (ii) *Uniform averaging pays, and pays in depth.* The realized perturbation of the uniform average is 2.27× that of the proportional average overall; by block the ratio is 1.00–1.02 at blocks 0, 2, and 3 and 2.7–3.1 at blocks 7, 8, and 11 (Table 16). The last three rows of the table say why: $\hat{\mathbf{A}} = \mathrm{softmax}(\mathbf{X}\mathbf{X}^\top)$ is close to the identity in the shallow blocks (column sums 1.00 ± 0.08 at blocks 0 and 2, pair asymmetry 0.01), so the influence is nearly uniform there and the weight of the average is immaterial; with depth the rows spread and a few tokens absorb most of the mass (block 11: column sums from 0 to 116, asymmetry 3.2), which is where the weight matters. The rescaled $\hat{s}$ used by the matching therefore carries little information in the shallow blocks and tracks s closely in the deep ones (Pearson 0.2–0.3 at blocks 0–3, 0.7–0.8 at blocks 7–11). The tightness of the bound has the same origin: rows that attend to at most one member of a pair leave no Cauchy–Schwarz slack. The median pair has a ratio of 1.000: half of the selected pairs have nearly equal influence and cost the same under either rule, and the mean is carried by the asymmetric pairs, which sit in the deep blocks. Read together with Table 21, this is the mechanism of the widening: a fixed schedule keeps merging r pairs per block in the depths where each pair costs most, and the budget rule stops there first. (iii) *Monotone, not convex, in our order.* $p_k > 0$ on 99.8% of pairs and $p_k = 0$ on the rest, so $P(r)$ is non-decreasing as Proposition 2 states. In the

Table 16: The same measurement by block. $\bar{J}^\star$: mean minimum pair cost; ratio: $\bar{J}_{\text{uniform}}/\bar{J}^\star$; tightness: realized perturbation over bound for the uniform and the proportional rule; $\overline{\cos}$: mean cosine of the selected pairs; last three rows: spread, maximum, and pair asymmetry of the raw influence s (column sums of $\hat{\mathbf{A}}$; the mean is 1.0 by construction up to the number of tokens). Absolute costs are in feature units, which grow with depth; the ratio does not depend on scale.

Block	0	1	2	3	4	5	6	7	8	9	10	11
$\bar{J}^\star$	43.8	51.7	20.8	21.0	26.8	31.8	55.5	69.5	95.8	118	194	388
Ratio	1.01	1.19	1.00	1.02	1.19	1.23	1.82	2.91	2.73	1.79	1.69	3.13
Tightness (unif.)	0.998	0.922	0.995	0.992	0.992	0.994	0.975	0.969	0.978	0.994	0.987	0.993
Tightness (prop.)	0.999	0.933	0.995	0.994	0.996	0.996	0.964	0.952	0.988	0.994	0.990	0.989
$\overline{\cos}$	0.72	0.75	0.67	0.64	0.64	0.64	0.67	0.67	0.69	0.73	0.76	0.79
$\mathrm{std}(s_i)$	0.08	1.01	0.06	0.29	0.54	0.37	0.54	0.77	0.84	0.65	1.23	6.70
$\max(s_i)$	3.3	32.6	4.2	7.7	9.8	7.6	9.7	13.0	17.6	21.5	25.4	116
$\overline{\lvert s_i - s_j \rvert}$	0.01	0.62	0.01	0.04	0.10	0.07	0.13	0.16	0.16	0.17	0.63	3.24

order the matching score $\hat{s}_i \cos$ produces, p_k decreases with k in all twelve blocks: the score admits the most influential sources first, and those are the most asymmetric pairs, so the penalty our aggregation removes is front-loaded. This does not test (A2), which concerns ToMe's cosine order. Re-sorting the same selected pairs by cosine, the mean increment correlates positively with rank in 11 of 12 blocks (Pearson 0.03–0.92; block 5: -0.11; Figure 1), which is the direction (A2) requires but on our pairs rather than ToMe's. We leave the convexity claim as stated in Section 5: proven monotone, unproven convex. (iv) *The raw-s and rescaled-$\hat{s}$ rules coincide.* Their perturbations differ by 0.3% and their Top-1 by 0.06 pp (not significant), so the rescaling is a numerical convenience and the implementation is the rule of Proposition 1. (v) *The classifier absorbs most of the perturbation on ViT-B.* Uniform aggregation under our matching and budget costs 0.06 pp, one to two standard deviations of the five-run spread, while more than doubling the perturbation. The accuracy of AdaMerge over ToMe (+0.61 pp at this tier) is therefore earned by the matching and the budget, and Table 13 says the budget is the larger part. Whether the aggregation matters more where the margin is larger (ViT-L, +4.64 pp) we did not measure.

E.4 Sigmoid Temperature

T rescales z_l before the gate. Lower T sharpens the decision between redundant and non-redundant layers; T=0.5 gains 0.40 pp over the default at equivalent FLOPs and $T \geq 2$ loses about 1 pp. We keep T=1 throughout so that the main results carry no tuned knob beyond $r_{\max}$.

Table 17: Sensitivity to T ($r_{\max}$=17, converged statistics).

Temperature T	0.5	**1.0 (default)**	2.0	4.0
Top-1 Acc	84.80%	**84.40%**	83.49%	83.10%
Δ vs. default	+0.40	–	−0.91	−1.30

F Calibration

F.1 Per-Layer Fixed-Point Statistics

Table 18 lists $\theta_l^\star = (\mu_l, \sigma_l)$ at convergence. The input layer is invariant to the compression regime ($\mu_0 \approx 0.295$, $\sigma_0 \approx 0.036$ for every $r_{\max}$): the redundancy of raw patch embeddings is a property of the model and data, not of the schedule. The last block is not: as $r_{\max}$ grows from 8 to 23, μ_{11} rises from 0.019 to 0.267, because heavy shallow merging changes the token population that reaches it. The fixed point of Eq. 10 therefore differs by regime, and calibrating per $r_{\max}$ is what lets the deep gates compare an image against the population they will actually see.

F.2 Convergence and Initialization

Figure 3 tracks $\|\theta^{(k)} - \theta^{(k-1)}\|_2$ over ten iterations from three initializations (A: μ=0.5, σ=0.10, the default; B: 0.3, 0.05; C: 0.7, 0.15). Every trajectory contracts by more than 90% within two iterations and then oscillates

Table 18: Converged per-layer statistics (μ_l, σ_l), ViT-B/16.

		$r_{\max}$				
Layer	Stat	8	12	16	20	23
$l = 0$	μ_l	0.296	0.295	0.295	0.295	0.295
	σ_l	0.037	0.037	0.036	0.036	0.036
$l = 5$	μ_l	0.138	0.148	0.159	0.169	0.177
	σ_l	0.036	0.037	0.036	0.036	0.036
$l = 11$	μ_l	0.019	0.027	0.050	0.239	0.267
	σ_l	0.019	0.024	0.037	0.122	0.135

in a neighbourhood of width < 0.005; the minima occur at k=5–6 (0.008, 0.019, 0.028). The terminal points agree to 0.0047 between A and B and between A and C, and μ_0 agrees to 0.001 (0.293 for all three). Two iterations capture the contraction; a third can overshoot and cost about 1 pp, so two is the protocol.

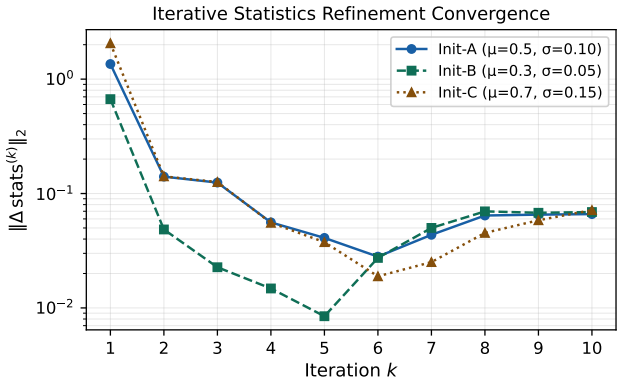


Figure 3: Fixed-point iteration $\theta^{(k+1)} = \mathcal{T}(\theta^{(k)})$ from three initializations (log scale): rapid contraction, then oscillation within $\|\Delta\theta\|_2 < 0.005$.

F.3 Data Efficiency

Table 19: Calibration-set size, 0.5%–5% of ImageNet-1k train ($r_{\max}$=17, two-pass calibration).

Subset size	0.5%	**1% (default)**	2%	5%
# images	6,405	**12,811**	25,623	64,058
Top-1 Acc	84.30%	**84.29%**	84.28%	84.21%
FLOPs (G)	14.34	14.47	14.40	14.40

Accuracy moves by at most 0.09 pp across the 0.5%–5% range and by 0.04 pp between 256 images and the full 1%; the worst 64-image draw costs 0.40 pp. The data dependence of AdaMerge is therefore a few hundred unlabeled images and two forward passes, which separates it from the fine-tuned adaptive methods AdaViT, Evo-ViT, and SPViT (Meng et al., 2022; Xu et al., 2022; Kong et al., 2022).

F.4 No-Calibration Control

Running Eq. 9 with the uninformed prior $\theta^{(0)} = (0.5, 0.1)$ and no calibration pass produces zero merges at every layer on ViT-B/16: the true scale of $\bar{S}_l$ (0.02–0.30, Table 18) sits several prior standard deviations below 0.5, so $\sigma(\alpha z_l) \approx 0$ and $r_l = 0$. The output equals the uncompressed model (85.19%, 17.45 G). Calibration is thus load-bearing, and its cost is the two passes of Appendix F.3.

Table 20: Sub-percent calibration sets ($r_{\max}$=17, matched ∼13.4–13.6G). Ranges are over 3 disjoint seeds at the two smallest sizes.

# Calibration images	64	256	1,024	6,400	12,811 (full 1%)
Top-1 Acc	83.75–84.37%	84.13–84.19%	84.04%	84.07%	84.15%

G WHERE THE BUDGET GOES

The schedule of Eq. 9 was given no per-layer prior, so where it spends the budget is a measurement. Over the 50,000 validation images it settles into three regimes (Table 21, Figure 4). At $r_{\max} \leq 11$ the last block merges most ($\mathbb{E}[r_{11}] = 5.18$ at $r_{\max}$=9); at $r_{\max} \in \{14, 17\}$ blocks 0–1 become the most active ($\mathbb{E}[r_1] = 4.71$–5.87) and block 11 contracts ($3.12 \to 0.84$); at $r_{\max} \geq 20$ blocks $l \geq 10$ ($l \geq 9$ at $r_{\max}$=23) merge nothing, because shallow merging has removed the near-duplicates before they reach them. Block 2 merges least in all six settings. Whether deep-layer inactivity is a correct allocation is answered by intervention in Section 7; the per-image side of the same adaptivity, at batch size 1, is visible in Figure 2a and Appendix J.

G.1 PER-BLOCK MERGE COUNTS

Table 21: Per-block $\mathbb{E}[r_l]$ over the 50,000 ImageNet-1k validation images. **Bold**: maximum per row; –: $\mathbb{E}[r_l] = 0$.

$r_{\max}$	l=0	l=1	l=2	l=3	l=4	l=5	l=6	l=7	l=8	l=9	l=10	l=11
9	2.57	3.01	1.03	1.82	2.26	2.56	3.08	3.10	2.91	2.86	2.66	**5.18**
11	3.33	3.59	1.46	2.63	2.60	2.98	3.74	3.52	3.55	3.51	3.20	**5.03**
14	4.45	**4.71**	2.25	3.62	2.98	3.35	4.30	3.89	4.20	4.53	3.99	3.12
17	5.59	**5.87**	3.20	4.69	3.39	3.60	4.80	3.85	4.66	5.04	4.01	0.84
20	6.74	**6.96**	4.28	5.96	3.47	3.35	4.66	3.25	3.83	2.54	–	–
23	7.67	**8.83**	5.39	7.50	3.97	3.37	4.57	2.35	1.31	–	–	–

The three regimes are visible row by row: the maximum sits at l=11 for $r_{\max} \leq 11$, moves to l=1 for $r_{\max} \geq 14$, and the deep blocks reach zero at $r_{\max} \geq 20$. Block 2 has the smallest active count in every row. The regime shift is what Eq. 10 predicts from Table 18: once shallow merging removes the near-duplicates, the deep statistic $\bar{S}_{11}$ of a typical image falls below the calibrated μ_{11}, and the gate closes.

G.2 FORCED-ACTIVATION CONTROL

Table 22: Forcing deep-layer merges at $r_{\max}$=23 (ViT-B/16, mean ± std over 3 runs). The natural schedule assigns $r_l \approx 0$ for $l \geq 9$.

Schedule	Top-1 Acc	FLOPs (G)
Natural ($r_{\max}$=23)	$84.15_{\pm 0.02}$%	13.50
+ forced $r_l \geq 1$, $l \geq 9$	$84.07_{\pm 0.03}$%	13.47
+ forced $r_l \geq 2$, $l \geq 9$	$84.04_{\pm 0.03}$%	13.44

Accuracy falls monotonically as the floor rises while FLOPs fall. The exchange rate is what matters: the forced merges give up 0.11 pp for 0.06 G (1.8 pp per GFLOP), whereas ADAMERGE's own curve between $r_{\max}$=20 and 23 gives up 0.12 pp for 0.52 G (0.23 pp per GFLOP). Within the three-run spread of both measurements the forced rate lies between 1.2 and 2.5 pp per GFLOP, five to eleven times the schedule's marginal rate, so the deep blocks' inactivity is a correct allocation.

G.3 CLASS-STRATIFIED ACCURACY

The dog block loses 1.80 pp under ADAMERGE against 4.51 under TOME, the birds 0.36 pp, and the disproportionality is less than half of TOME's. Across tiers the dogs index moves from −0.25 to −0.81 for ADAMERGE and from +0.38 to −1.87 for TOME. The worst individual classes drop 8–12 pp and are dispersed across breeds rather than concentrated. Fine-grained classes are the ones whose discriminative tokens carry the most influence, and they are the ones the influence-weighted merge protects.

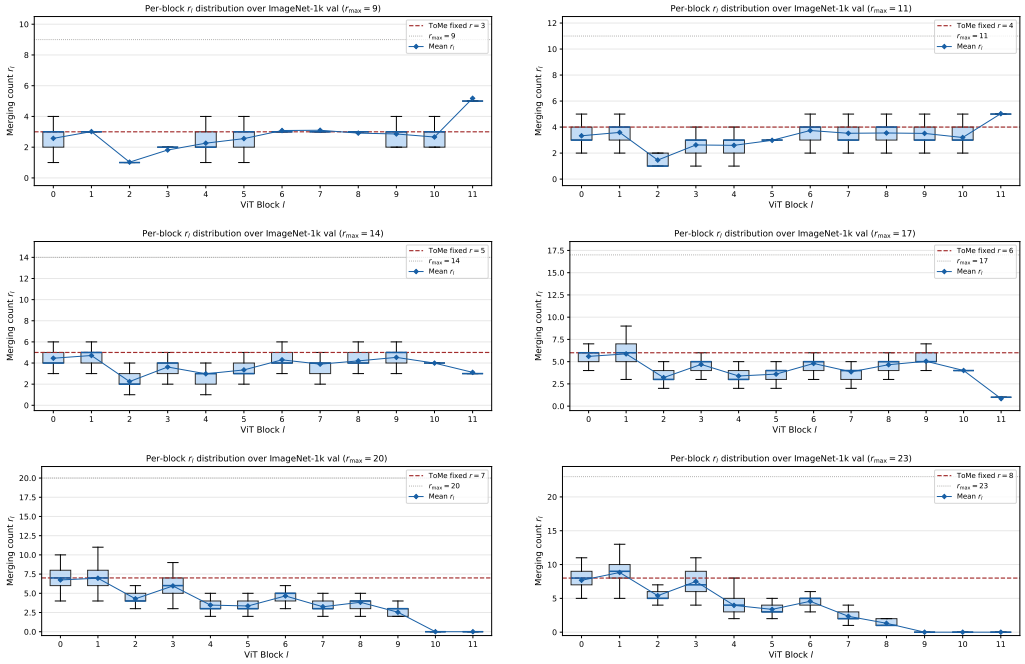


Figure 4: Per-block r_l distributions (n=50,000) for $r_{\max} \in \{9, 11, 14, 17, 20, 23\}$, left to right, top to bottom. Boxes: interquartile range; whiskers: 5th/95th percentile; diamonds: mean; dashed red: the matched ToMe fixed r; dotted grey: $r_{\max}$.

Table 23: Accuracy by semantic granularity at the ∼13.4G tier. Disproportionality = Δ(dogs) − Δ(rest); more negative means the fine-grained block is hurt more than the remainder.

Method	Dogs (118 cls)	Birds (39)	Rest (843)	All	Disproportionality
Baseline	85.15%	96.21%	84.69%	85.20%	–
ToMe (r=8)	80.64%	95.69%	82.05%	82.42%	−1.87
AdaMerge ($r_{\max}$=23)	83.36%	95.85%	83.71%	84.14%	−0.81

H Wall-Clock Across Scale

H.1 Measurements

Matched-FLOPs throughput was measured across 39 speedup-matched settings on ViT-S/16, DeiT-B/16, and ViT-L/16 (RTX 3090, batch 64, median of 5 runs under sustained load, baseline re-checked at the end of each session, ±2%). These runs are on a different GPU than the ViT-B/16 measurements of Table 1 (A6000), so absolute throughputs are not comparable across the two tables; the relative overhead is.

Table 24: Throughput overhead of AdaMerge relative to ToMe at matched FLOPs. ViT-L/16 absolute numbers are in Table 10.

Backbone	Params	d	Overhead vs. ToMe	Direction
ViT-S/16	22M	384	6–9%	AdaMerge slower
DeiT-B/16	86M	768	0–1%	parity
ViT-B/16 (A6000, Section 6.3)	86M	768	5–9%	AdaMerge slower
ViT-L/16	304M	1024	≈ −7%	AdaMerge faster

H.2 Overhead Model

Per block, a ViT with N tokens and width d spends about $4Nd^2$ on the QKV and output projections, $8Nd^2$ on the MLP, and $2N^2d$ on the attention scores and their application. AdaMerge adds the $N \times N$ affinity (N^2d), its softmax and column sum ($\mathcal{O}(N^2)$), and the same $\frac{N}{2} \times \frac{N}{2}$ cosine block that ToMe computes. The relative overhead of the extra term, Eq. 11, is $N/(12d + 2N)$: $3.9\%, 2.0\%, 1.5\%$ at $d = 384, 768, 1024$. The measured overhead exceeds this on ViT-S/B by a few points because the extra kernels are launched once per layer and the adaptive decision synchronizes the device once per layer; that part is a constant per layer and independent of d. Attention savings from merging r_l tokens scale as $(2Nr_l - r_l^2)d$ and the projection and MLP savings as $12r_ld^2$, both growing with d, which is why the sign of the overhead flips between ViT-B and ViT-L. The two d=768 rows were measured on different GPUs (A6000 vs. RTX 3090) as well as different checkpoints, so we do not read their 5–9% vs. 0–1% difference as a recipe effect; the clean comparison is the ViT-S $\rightarrow$ ViT-L sign flip on one device.

I Multimodal Transfer: LLaVA-1.5-7B

Setup follows PiToMe's LLaVA protocol: CLIP ViT-L/14 tower, 576 visual tokens, POPE (F1, 8,910 questions) and GQA (accuracy, 12,578 questions), all three methods at matched token counts; ToMe and PiToMe at fixed r bracket AdaMerge's adaptive average.

Table 25: LLaVA-1.5-7B (baseline: 576 tokens, POPE 85.86, GQA 61.93).

Tier	Method	Tokens	POPE F1 (Δ)	GQA (Δ)
Moderate	ToMe (r=5)	461	85.51 (-0.35)	61.50 (-0.43)
	PiToMe (r=5)	461	85.57 (-0.29)	61.57 (-0.36)
	AdaMerge ($r_{\max}$=14)	451.7	85.68 (-0.18)	61.73 (-0.20)
Aggressive	ToMe (r=14)	254	86.72 ($+0.86$)	60.73 (-1.20)
	PiToMe (r=14)	254	86.93 ($+1.07$)	60.14 (-1.79)
	AdaMerge ($r_{\max}$=36)	238.7	85.55 (-0.31)	60.31 (-1.62)

Moderate tier. AdaMerge leads on both benchmarks with the fewest tokens; the margins (0.11–0.23 pp) are within sampling error ($\sim$0.25 pp on GQA, $\sim$0.4 pp on POPE). We read this as transfer without regression.

Aggressive tier. On GQA the three methods lie within 0.05 pp once token-matched (the fixed-r rows bracket AdaMerge's 238.7 tokens; the interpolated value is $\approx$60.3). On POPE both fixed-schedule methods exceed the uncompressed model, which a fidelity metric cannot do at 55–60% token removal unless another channel is open. Decomposing F1 locates it: ToMe's yes-ratio moves $0.418 \rightarrow 0.444$ and recall $0.791 \rightarrow 0.823$; PiToMe's move further (0.449/0.457 and 0.819/0.828 at r=14/15). AdaMerge's yes-ratio moves by $+0.006$ and recall by $+0.003$, and its F1 declines monotonically ($85.68 \rightarrow 85.55$). The inflation is thus a property of heavy fixed-schedule merging under POPE's balanced yes/no design, reproduced by two independently designed methods, and the schedule-adaptive method is the one that does not exhibit it.

Scope. The classification widening does not appear here. The bound of Lemma 1 is a statement about the vision tower; the 7B language model downstream can absorb encoder-side perturbations, which we offer as a hypothesis. What transfers is plug-and-play use on a contrastively pretrained tower with no regression against ToMe at either tier.

J Qualitative Maps

Figure 6 shows survival maps for AdaMerge ($r_{\max}$=17) and ToMe (r=6) on three validation images. ToMe keeps exactly $124/196$ tokens on every image. AdaMerge keeps $58/196$ on the polar bear, whose ice-and-water background is nearly constant, and 139–$142/196$ on the two cluttered scenes, and the surviving tokens sit on the animals. Figure 7 shows the influence field at four depths for five more images: the mass moves from texture at layer 0 to the bumblebee's wing, the dog's eyes and nose, and the telescope body by layer 11.

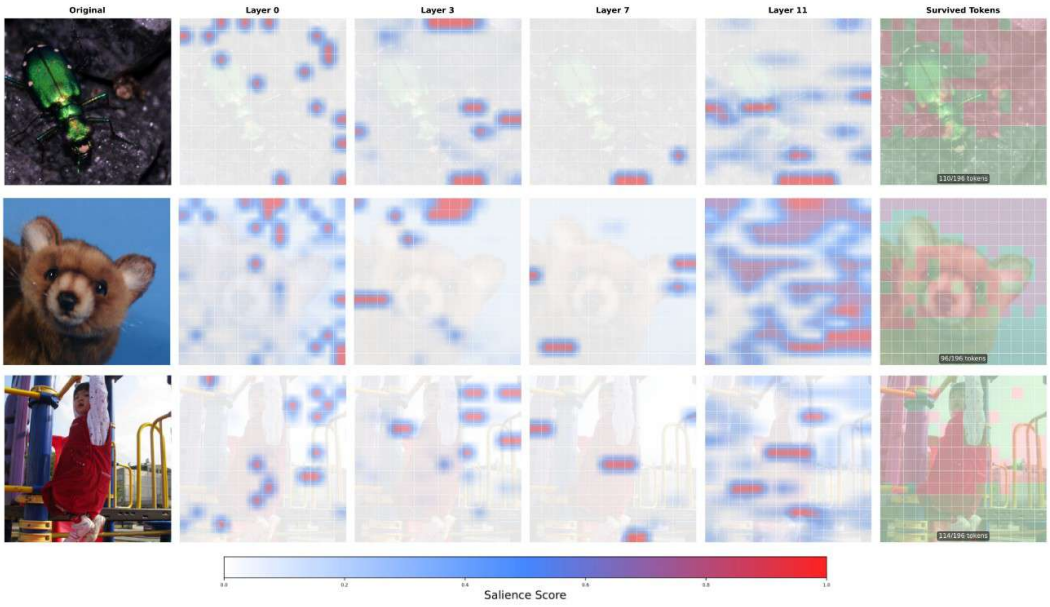


Figure 5: Layer-wise influence maps (column sums of the row-normalized affinity; warmer is higher) and surviving tokens (green kept, red merged) for AdaMerge at $r_{\max}=18$ on three ImageNet-1k images. The retained count, printed on each survival panel, varies with content; ToMe at the matched budget keeps the same number on every image.

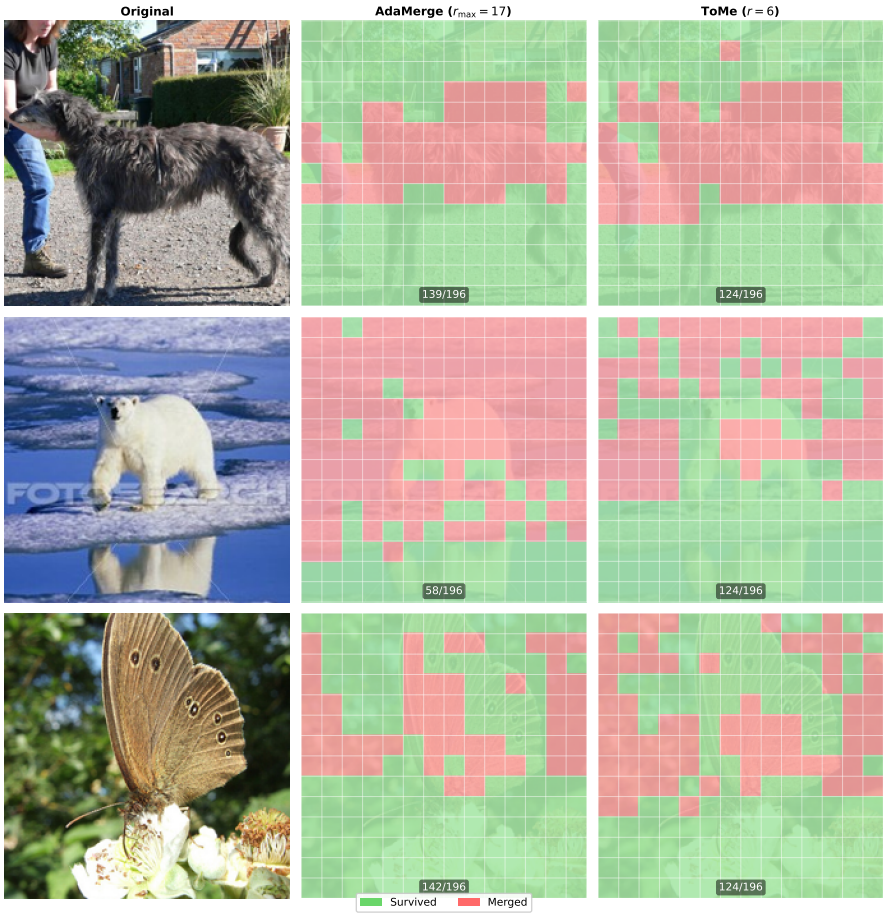


Figure 6: Token survival maps for AdaMerge ($r_{\max}$=17) and ToMe (r=6). Green: kept; red: merged. ToMe keeps $124/196$ tokens on every image; AdaMerge keeps $58/196$ on the uniform scene and 139–$142/196$ on the cluttered ones.

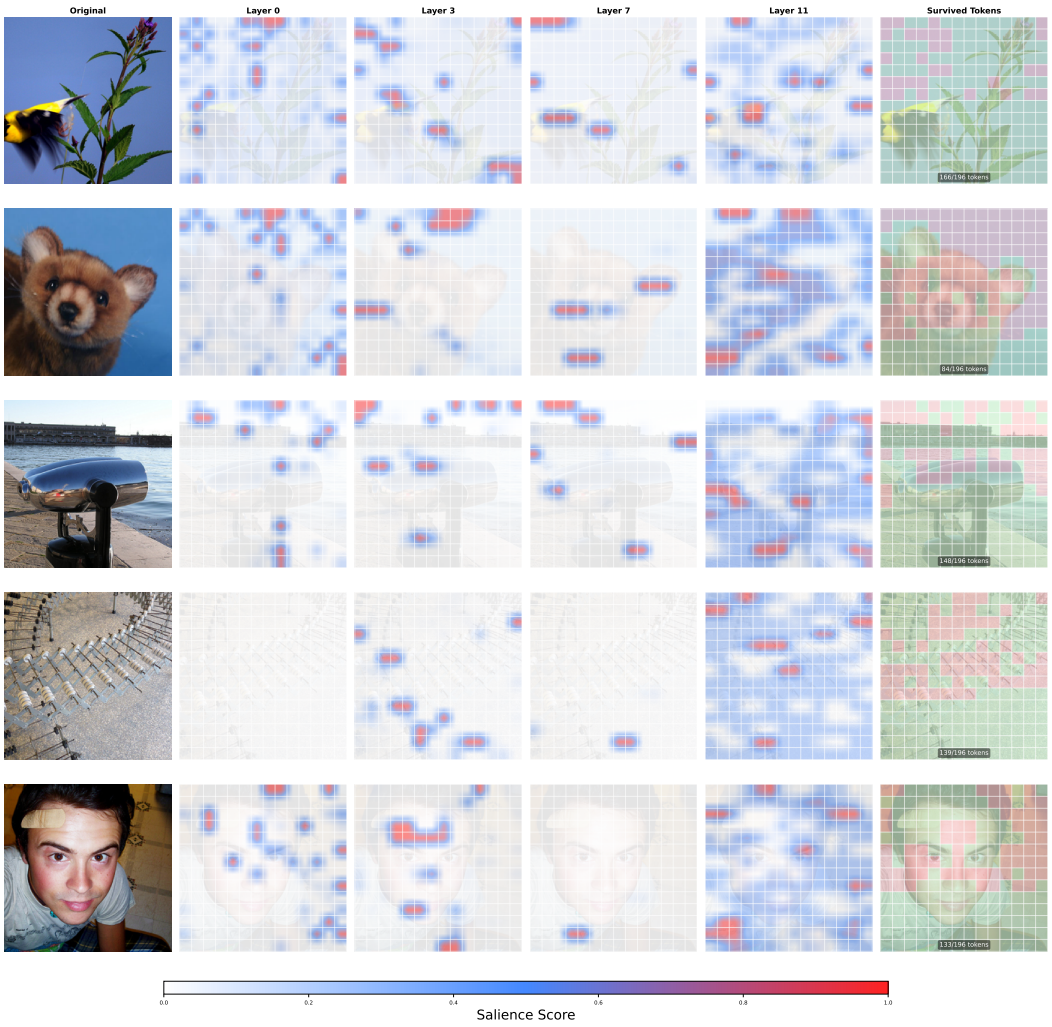


Figure 7: Influence maps at layers 0, 3, 7, 11 and surviving tokens (AdaMerge, $r_{\max}$=18) on five ImageNet-1k images. Warmer is higher; green kept, red merged.

K ALGORITHMS AND NOTATION

This appendix lists the procedure as implemented in the supplementary code and maps each step to the equations of Sections 3–4. One ADAMERGE call runs before every Transformer block (Algorithm 1) and consists of four steps. (i) INFLUENCEWEIGHTEDSIMILARITY (Algorithm 2) forms the row-normalized affinity $\hat{\mathbf{A}}$ of the current patch tokens, takes its column sums as the influence $\mathbf{s}$ (Eq. 3), rescales them per image to $[0, 1]$, and weights the cosine similarity between the two index halves by the source's influence (Eq. 5); the [CLS] token is set aside before this step and reattached after merging. Influence is recomputed from scratch at every layer, so no state is carried between layers other than the tokens themselves. (ii) BUDGET (Algorithm 3) averages the row maxima of $\tilde{\mathbf{S}}$ into the redundancy statistic $\bar{S}_l$ (Eq. 8), standardizes it against the calibrated moments (μ_l, σ_l), and passes the z-score through the sigmoid gate to obtain r_l (Eq. 9). In the released code the mean is taken over all images of the evaluation batch, so r_l is shared by the batch (Section 4.4); if σ_l is zero or the statistic is not finite the code falls back to $r_l = \lfloor r_{\max}/2 \rfloor$, which never occurs with the released statistics. (iii) MERGE (Algorithm 4) selects the r_l sources with the largest best-match score, merges each into its target with the salience-proportional weight (Eq. 7), and returns the $N - r_l$ surviving tokens; sources that share a target are written in sequence and the last write is kept. (iv) CALIBRATE (Algorithm 5) runs (i)–(iii) over the calibration set twice, each time under the moments of the previous pass, and stores the per-layer mean and standard deviation of $\bar{S}_l$ (Eq. 10); it is executed once per (backbone, $r_{\max}$) and its output is the only state ADAMERGE reads at inference. Table 26 collects the symbols.

Algorithm 1 ADAMERGEBLOCK (one call before each Transformer block l)

Require: tokens $[x_{\text{cls}}; \mathbf{X}]$, $\mathbf{X} \in \mathbb{R}^{N \times d}$; moments (μ_l, σ_l); $r_{\max}$, α, T

1: $(\tilde{\mathbf{S}}, \hat{\mathbf{s}}) \leftarrow$ INFLUENCEWEIGHTEDSIMILARITY$(\mathbf{X})$ ▷ Alg. 2
2: $r_l \leftarrow$ BUDGET$(\tilde{\mathbf{S}}, \mu_l, \sigma_l, r_{\max}, \alpha, T)$ ▷ Alg. 3
3: $\mathbf{X}' \leftarrow$ MERGE$(\mathbf{X}, \tilde{\mathbf{S}}, \hat{\mathbf{s}}, r_l)$ ▷ Alg. 4
4: **return** Block$_l([x_{\text{cls}}; \mathbf{X}'])$ ▷ $N - r_l$ patch tokens

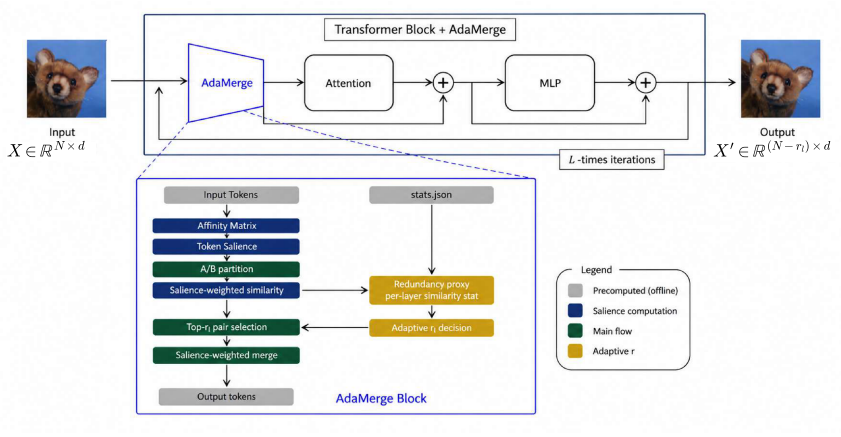


Figure 8: **One objective, two decisions.** *Top:* ADAMERGE reduces the sequence from N to $N - r_l$ tokens before each block. *Bottom:* the affinity $\hat{\mathbf{A}}$ yields the influence $\mathbf{s}$ and the influence-weighted similarity $\tilde{\mathbf{S}}$; the row maxima of $\tilde{\mathbf{S}}$ give the redundancy statistic $\bar{S}_l$ that sets r_l against calibrated statistics (μ_l, σ_l), and the selected pairs are combined by the salience-proportional average.

Algorithm 2 INFLUENCEWEIGHTEDSIMILARITY

Require: $\mathbf{X} \in \mathbb{R}^{N \times d}$, split $(\mathcal{A}, \mathcal{B})$

1: $\hat{\mathbf{A}} \leftarrow \text{softmax}(\mathbf{X}\mathbf{X}^\top)$ ▷ $\mathcal{O}(N^2 d)$
2: $\mathbf{s} \leftarrow \hat{\mathbf{A}}^\top \mathbf{1}$; $\hat{s}_i \leftarrow (s_i - \min_k s_k)/(\max_k s_k - \min_k s_k + \epsilon)$
3: $\tilde{S}_{ij} \leftarrow \hat{s}_i \langle x_i, x_j \rangle / (\|x_i\| \|x_j\|)$ for $i \in \mathcal{A}$, $j \in \mathcal{B}$
4: **return** $\tilde{\mathbf{S}}, \hat{\mathbf{s}}$

Table 26: Notation.

Symbol	Meaning
$\mathbf{X} \in \mathbb{R}^{N \times d}$	Patch tokens entering a block (N=196 for ViT-B/16 at layer 0)
$\hat{\mathbf{A}} = \mathrm{softmax}(\mathbf{X}\mathbf{X}^\top)$	Row-normalized affinity
$s_i = \sum_k \hat{A}_{ki}$, $\hat{s}_i \in [0,1]$	Influence (column sum), rescaled per image; recomputed at every layer
$\mathcal{A}, \mathcal{B}$	Index split $\mathbf{X}_{[:N/2]}$, $\mathbf{X}_{[N/2:]}$
$\tilde{S}_{ij} = \hat{s}_i \cos(x_i, x_j)$, $\pi(i)$	Influence-weighted similarity and best target of source i
J_{ij}, $J^\star_{ij}$, U_{ij}	Pair cost (Eq. 1), its minimum (Eq. 4), its bound (Eq. 6)
$\bar{S}_l$, (μ_l, σ_l), z_l	Redundancy statistic, its calibrated moments, its z-score
r_l, $r_{\max}$, α=3, T=1	Merge count, budget, gate sensitivity, temperature
$\mathcal{M}_l$	Selected pairs at layer l
$\mathcal{T}$, $\theta^\star$	Calibration operator and its fixed point (Eq. 10)

Algorithm 3 BUDGET

Require: $\tilde{\mathbf{S}}$, (μ_l, σ_l), $r_{\max}$, α, T

$\bar{S}_l \leftarrow \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \max_j \tilde{S}_{ij}$

$z_l \leftarrow (\bar{S}_l - \mu_l)/\sigma_l$

return $\mathrm{clip}\big(\lfloor r_{\max}\, \sigma(\alpha z_l / T) \rfloor,\ 0,\ r_{\max}\big)$

Algorithm 4 MERGE

Require: $\mathbf{X}$, $\tilde{\mathbf{S}}$, $\hat{\mathbf{s}}$, r_l, split $(\mathcal{A}, \mathcal{B})$

$\pi(i) \leftarrow \arg\max_j \tilde{S}_{ij}$; $\quad \mathcal{M}_l \leftarrow \text{Top-}r_l\{\tilde{S}_{i\pi(i)}\}_{i \in \mathcal{A}}$

for $i \in \mathcal{M}_l$ (vectorized; on a shared target the last write wins) **do**

  $w \leftarrow \hat{s}_i / (\hat{s}_i + \hat{s}_{\pi(i)} + \epsilon)$; $\quad x_{\pi(i)} \leftarrow w\, x_i + (1 - w)\, x_{\pi(i)}$

end for

return $[\mathbf{X}_{\mathcal{A} \setminus \mathcal{M}_l};\ \mathbf{X}_{\mathcal{B}}] \in \mathbb{R}^{(N - r_l) \times d}$

Algorithm 5 CALIBRATE (two applications of $\mathcal{T}$)

$\theta^{(0)} \leftarrow (\mu_l, \sigma_l) = (0.5, 0.1)$ for all l

for $k = 1, 2$ **do**

  run the network on $\mathcal{D}_c$ with the budget of Algorithm 3 under $\theta^{(k-1)}$; collect $\bar{S}_l$ per image

  $\theta_l^{(k)} \leftarrow (\mathrm{mean}, \mathrm{std})$ of the collected $\bar{S}_l$

end for

return $\theta^{(2)}$ (written to `stats.json`)

L REPRODUCIBILITY

Software. Python 3.10, PyTorch 2.1.0 (CUDA 12.1), `timm` 0.9.12, `fvcore` 0.1.5 for FLOPs, seed 42 for subset selection and loader order.

Hardware. ViT-B/16 results: one NVIDIA A6000; throughput at batch 64, 224^2, 50 warm-up and 200 timed batches, median of 5 runs. Cross-backbone wall-clock: RTX 3090 (Appendix H). Fine-tuning: 3 GPUs with DDP.

FLOPs conventions. FLOPs are multiply–accumulate counts from `fvcore` at 224^2; the reduction axis of Figure 2c uses the uncompressed counts 4.6 G (ViT-S/16), 17.45 G (ViT-B/16), and 61.6 G (ViT-L/16). The five-run standard deviations in Table 1 cover non-deterministic kernels and tie-breaking at a fixed calibration; the variation across calibration seeds is reported separately in Table 20.

Hyperparameters. Backbone `vit_base_patch16_224.augreg2_in21k_ft_in1k`; resize 256, centre-crop 224, ImageNet normalization; $\alpha = 3.0$, $T = 1.0$; calibration prior $(0.5, 0.1)$, two passes, $\varepsilon = 0.005$; 1% calibration subset (12,811 images) unless stated.

Baselines. Official ToME (https://github.com/facebookresearch/ToMe), reproduced within ± 0.05 pp of published numbers; official PiToMe with default energy threshold; DSM as released.

Code. The supplementary material contains the full pipeline: influence computation, matching, budget, calibration, ImageNet-1k evaluation, FLOPs and throughput measurement, and the three baseline evaluations, with the precomputed `stats.json` and commands reproducing every number in Table 1. Scripts for the appendix tables will be added to the public release.