BPAW-VIT: BLOCK PRUNING AND ADAPTIVE WIDTH FOR EFFICIENT VISION TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

In recent times, there has been a paradigm shift towards fine-tuned models rather than full-scale pre-training, owing to the availability of large corpora and pre-trained models on platforms like Hugging Face. The aim is to save significantly on computational costs during training. While Vision Transformers (ViTs) achieve remarkable performance, their deployment is hindered by high computational costs. Existing depth-Pruning methods such as block-dropping techniques like Entrodrop disrupt feature spaces, inducing severe loss in accuracy. Other depth-pruning methods, such as HeDEPT and NOSE, replace pruned layers with parameter-heavy linear substitutions. To achieve true parameter sparsity without architectural distortion, we propose a BPAW-ViT framework that unifies Alpha-Gated Progressive Layer Pruning (depth) with slimmable feed-forward networks (Width Slicing/Adaptive Width). Guided by an entropy-alignment criterion and a dual-width training recipe, our framework dynamically reduces redundant blocks to pure identity mappings or slices intermediate capacities, requiring zero extra parameters. Finally, as a natural extension, we integrate this framework with Early-Exit architectures (EnViT). Because alpha-gated pruning preserves monotonic hidden states and intact 2D spatial resolution, our joint approach enables us to achieve a higher speedup (up to 2.45×) with a significantly reduced memory footprint.

1 INTRODUCTION

There has been a paradigm shift in applied deep learning, away from training models from scratch towards finetuning Kornblith et al. (2019). Platforms such as Hugging Face (Wolf et al., 2020) now host hundreds of thousands of pre-trained Vision Transformer checkpoints spanning diverse architectures—such as ViT by Dosovitskiy et al. (2020), DeiT by Touvron et al. (2021), Swin by Liu et al. (2021), BEiT by Bao et al. (2022), and DINOv2 by Oquab et al. (2023)—as well as varied pretraining corpora. For the overwhelming majority of downstream applications—such as medical imaging (Raghu et al., 2019), industrial inspection, satellite imagery, and retail analytics—practitioners begin not with random initialisation, but with a checkpoint pre-trained on a large, general-purpose corpus such as ImageNet-21k (Deng et al., 2009; Ridnik et al., 2021) or JFT (Sun et al., 2017; Zhai et al., 2022), which is then fine-tuned on a smaller, task-specific dataset. As established by Yosinski et al. (2014) and Kornblith et al. (2019), this practice is now the de facto standard because it is dramatically more compute- and data-efficient.

Vision Transformers (ViTs) have established a new paradigm for visual recognition tasks, offering unprecedented representation power. However, due to their design intricacies and massive parameter counts, which impose severe computational bottlenecks, precluding their deployment in resource-constrained or latency-sensitive environments (Vaswani et al., 2017; Dosovitskiy et al., 2020). While hierarchical designs (Liu et al., 2021),(Wang et al., 2021) address some of these spatial inefficiencies, post-training compression paradigms remain essential for adapting frontier models to strict hardware budgets.

Block pruning—removing entire transformer blocks—is a highly effective compression strategy for deep Large Language Models (LLMs), often utilising stochastic block-dropping techniques such as Entrodrop or Cosine (He, 2025; Fan et al., 2020). However, we observe in Section 3.1 that directly adapting these heuristics to ViTs yields suboptimal results. Since ViTs are structurally

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.

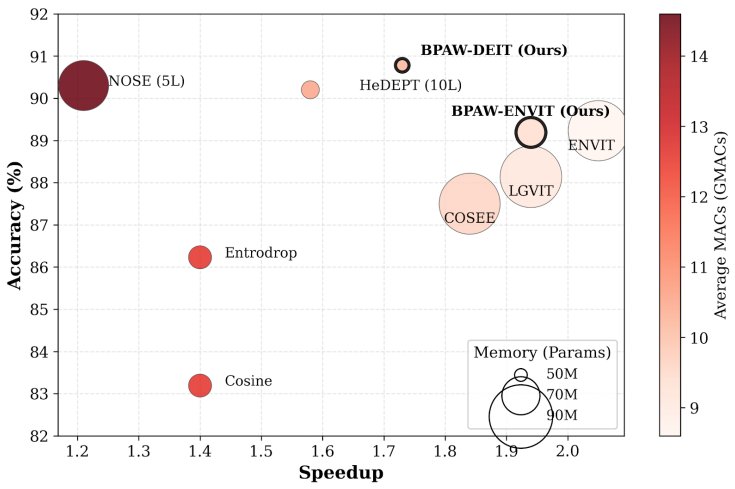


Figure 1: 4-D Bubble of ViT compression/speedup strategies. A comprehensive trade-off analysis comparing inference speedup, Top-1 accuracy, memory footprint (bubble size), and average computational cost in GMACs (colour intensity). Our proposed BPAW-DEIT architecture establishes a new state-of-the-art, outperforming other depth-pruning methods in accuracy, speedup, and memory reduction. By combining with EnViT and maximising inference acceleration through early exits, our model matches EnViT's accuracy while maintaining a competitive speedup and using only a fraction of the memory footprint.

shallower than LLMs, each layer's contribution to spatial token mixing is highly critical. Rigid, heuristic block dropping disrupts these continuous feature transformations, inevitably leading to severe representational collapse. Conversely, alternative ViT-based depth-pruning methods such as HetDEPT(Su et al., 2026) and NOSE(Lin et al., 2024) attempt to bridge the gap left by dropped blocks by replacing them with compensatory linear layers. This approach introduces critical architectural constraints: it incurs millions of additional trainable parameters (e.g. up to 2.5M for HetDEPT), destroys the unified transformer structure by creating disjointed feature spaces, and distorts the monotonic hidden-state distributions required for downstream adaptations.

An alternative line of work for speeding up during inference, gaining popularity in recent times, is the Early-Exit paradigm (He et al., 2025; Xu et al., 2023). In methods such as EnViT (Dong, 2026), auxiliary classification heads are attached to intermediate layers of the network. A confidence- or entropy-based criterion determines, on a per-sample basis, whether inference can terminate before reaching the final layer. This allows *easy* inputs to be classified from the earlier layers themselves, skipping a substantial fraction of the forward pass. However, this efficiency gain comes with a structural limitation: All layers, along with the auxiliary exit heads, must remain resident in memory at all times, since any given input may require traversal of the full network depth. Therefore, an early-exit model, despite its favourable average-case latency, still requires the full, uncompressed network's memory footprint to be loaded and available at all times. This is a critical limitation when considering deployment in edge and embedded deployment settings. In contrast, depth/block pruning approaches permanently remove components, thereby reducing the memory footprint.

In this work, we propose Alpha-Gated-ViT (a Block-Dropping method), which addresses the problem at its core. Our method identifies, progressively during fine-tuning and on a per-block basis, how critical each block's transformation is to the token representations that downstream layers depend on. It then uses this signal to drive progressive block dropping during the fine-tuning process itself—the setting in which the overwhelming majority of ViT deployments actually originate (Section 1). Weak layers are progressively reduced to pure identity mappings, and their parameters are frozen, completely bypassing the depth computation while leaving the 2-D spatial resolution intact and preserving critical spatial semantics. This mechanism guarantees zero extra parameter overhead and maintains a stable feature space. Grounding pruning decisions in dynamic, block-level contribution metrics—rather than predefined heuristics—enables our framework to circumvent representational collapse. Furthermore, our method eliminates the need for compensatory linear corrections,

thereby avoiding the structural overhead associated with auxiliary layers while preserving both the unified transformer architecture and the monotonic hidden-state behaviour required for downstream adaptation.

We further extend this Block-pruning foundation into a highly granular, block-pruned, adjustable-width Vision Transformer(BPAW-ViT). Drawing inspiration from MoSE-style (Tastan et al., 2026) width slicing, we replace the standard feed-forward networks (FFNs) inside the alpha-gated blocks with slimmable subnetworks. By employing a dual-width training recipe—where the network jointly learns full-capacity representations alongside randomly sampled sliced capacities—we decouple depth (active layers) from width (intermediate FFN capacity) into orthogonal axes of compression. This elasticity allows a single trained checkpoint to dynamically scale along a continuous Accuracy-MACs Pareto frontier at inference time, satisfying arbitrary computational budgets without requiring separate fine-tuning or distillation (Hao et al., 2022; Wu et al., 2022).

The symmetry of the entire architecture is thus preserved. Leveraging this symmetry argument, we introduce a unified approach that integrates early exiting with block pruning and width slicing, achieving competitive speedups and accuracy while substantially lowering memory footprints for edge deployment. We are combining the genuine, worst-case memory and compute reduction of block pruning with the additional average-case latency gains of confidence-based early termination. This yields an approach that, to our knowledge, is the first to address the heterogeneity problem in ViT depth pruning, avoid the parameter and structural overhead, and remain compatible with early-exit acceleration.

Our main contributions are summarised as follows:

- **Alpha-Gated Progressive Block Pruning:** We introduce an entropy-aligned depth compression technique, designed natively for the fine-tuning setting. It progressively ranks the blocks' importance based on their real-time contribution to the output signal, rather than relying on heuristics. It then removes ViT blocks completely without needing to compensate using additional trainable layers. This directly addresses the layer-heterogeneity problem left unresolved by uniform heuristic dropping (Entrodrop, Cosine), while avoiding the parameter overhead and disjointed feature spaces introduced by compensatory-layer approaches (HeDEPT, NOSE). The result is a worst-case memory reduction suited to edge deployment, achieved with zero architectural overhead.

- **Width Slicing (BPAW-ViT)**: We make the FFNs of the remaining blocks Slimmable, via a dual-width training recipe to construct a continuous inference-time Pareto frontier over both depth and width, allowing a single fine-tuned model to be dynamically reconfigured to the accuracy–efficiency trade-off best suited to a target device, without retraining. Rather than committing to a single pruned architecture, practitioners can select the best-accuracy configuration for a given compression budget directly from this frontier.

- **Synergy with Early-Exit Methods (EnViT):** We show that because our pruning preserves the inherent symmetry of the architecture and the monotonic hidden-state behaviour required for downstream adaptation, our compressed models are compatible with existing early-exit routing approaches such as EnViT, for classification tasks. This demonstrates that our block-dropping technique is not mutually exclusive with early-exit acceleration: the two can be combined, combining the genuine worst-case memory reduction of our pruning with the additional average-case latency gains that EnViT's confidence-based routing provides on top of it.

- **State-of-the-Art Performance:** We observe that our approach consistently outperforms established baselines (Entrodrop, Cosine, and HeDEPT), delivering better accuracy and speedups at lower memory footprint (BPAW-ViT and BPAW-DEiT). Upon combining with Envit, we are able to match Envit's accuracy at competitive speed-ups, only utilising a fraction of their memory footprint, as is evident from Figure 1 $2.45\times$ inference speedup on diverse benchmarks including CIFAR-100, Food-101, Stanford Cars, and Flowers-102 datasets(Krizhevsky et al., 2009; Bossard et al., 2014; Krause et al., 2013; Nilsback & Zisserman, 2008).

2 METHODOLOGY

2.1 OVERVIEW OF BPAW-VIT

Our method prunes ViT-based architectures based on a combination of 2 different techniques: (1) **Alpha-Gated Pruning**: A block-dropping technique based on re-weighting transformer blocks based on their contribution to the final output using learnable scalar gates, and (2) **Width-Slicing**: Width slicing dynamically executes only a fraction of the transformers' intermediate FFN dimension, which reduces computational footprint. Instead of maintaining separate isolated weights for different target model sizes, the architecture utilises a nested sub-network structure.

2.2 ALPHA-GATED TRANSFORMER BLOCK

This is a two-phase fine-tuning framework that progressively prunes redundant layers of a Vision Transformer (ViT) through learnable scalar gates. It is based on 2 key design principles: (1) removal of an entire transformer block, which contributes roughly(8.5%) of the model, can cause disruption at the feature level. Hence, it is important to reduce the block's weightage in the final output signal. (2) After a block is dropped, significant recovery time has to be provided for the feature-level disruptions to recover before a block can be dropped again.

Phase 1 fine-tunes a standard pretrained ViT on the target dataset for a few epochs before pruning starts. Given the importance of blocks in a task, it is important that the model has some knowledge of the task at hand. Phase 2 re-initialises the gates from an entropy-based criterion and applies a structured progressive pruning schedule that drops the weakest layer whenever its gate falls significantly below the running mean of the active gates.

2.3 PROBLEM FORMULATION:

Let f_θ denote a ViT with L transformer blocks $\{B_1, ..., B_L\}$. We wrap each block B_l in an alpha-gated container that introduces a scalar gate $\alpha_l \in (0, 1)$ parameterised via a learnable logit $\alpha_{raw,l} \in \mathbb{R}$ through the sigmoid:

$$\alpha_l = \sigma(\alpha_{raw,l}), \quad \sigma(x) = \frac{1}{1 + e^{-x}}. \tag{1}$$

The forward propagation through the l-th gated block is a convex interpolation between the identity map and the original block:

$$z^l = (1 - \alpha_l)z^{l-1} + \alpha_l B_l(z^{l-1}), \tag{2}$$

where z^{l-1} and z^l denote the input and output activations of the l-th block. When $\alpha_l \to 1$, the gated block reduces to the standard transformer operation, while $\alpha_l \to 0$ reduces it to a pure identity. To prevent the gate from saturating, standard settings, we clamp $\alpha_{raw,l}$ to $[-6, 6]$ after every optimisation step, bounding α_l to the approximate range $[0.0025, 0.9975]$.

A block is considered dropped once its gate is permanently deactivated. Concretely, dropping short-circuits the forward pass to $z^l = z^{l-1}$ and freezes every parameter of B_l and $\alpha_{raw,l}$, so that the dropped block consumes no gradient memory. The set of active blocks $\mathcal{A} = \{l : \text{block}_l \text{ is active}\}$ defines the current effective depth of the network. We denote the number of active layers by $|\mathcal{A}|$.

We perform the 2-phase finetuning finetuning using a pruning aware training objective and entropy initialised gates. Please refer to section A.3 for details on the Phase1 and Phase 2 finetuning,

2.3.1 ENTROPY-BASED GATE INITIALISATION(PHASE2)

Rather than initialising all gates to a fixed value, we derive the initial gate magnitude from the information-theoretic contribution of each block as introduced Entro-dropHe (2025). Let $H(Z^l)$ denote the differential entropy of the activation tensor Z^l at the output of block l, estimated via a 256-bin histogram entropy over a small calibration set of $B = 10$ training batches. We compute the per-layer entropy delta:

$$\Delta H_l = H(Z^l) - H(Z^{l-1}) \tag{3}$$

and define S_{start} as the first layer at which ΔH_l becomes positive, marking the transition from early "Stage-1" feature extraction to deeper "Stage-2" refinement. Layers with index $< S_{start}$ are forced

to $\alpha_l = 0.95$ and effectively excluded from pruning. For the remaining layers, ΔH_l is linearly normalised into $[0.4, 0.95]$:

$$\alpha_l^{init} = 0.4 + 0.55 \cdot \frac{\Delta H_l - \min_{l' \geq S} \Delta H_{l'}}{\max_{l' \geq S} \Delta H_{l'} - \min_{l' \geq S} \Delta H_{l'} + \epsilon}, \tag{4}$$

and the corresponding logit $\alpha_{raw,l} = \text{logit}(\alpha_l^{init})$ is written into the parameter. Layers that increase the activation entropy (ΔH_l large) are thus initialised near unity and protected from early pruning, while layers that barely perturb the entropy are initialised close to zero and become the primary pruning candidates.

2.3.2 Pruning-Aware Training Objective

We apply a bimodal pressure that encourages gates to commit to either 0 or 1:

$$\mathcal{L}_{bim} = \frac{1}{|\mathcal{A}|} \sum_{l \in \mathcal{A}} \alpha_l (1 - \alpha_l). \tag{5}$$

Minimising $\alpha_l(1 - \alpha_l)$ pushes each gate away from 0.5 toward the extremes of its range, producing the bimodal distribution that the pruning criterion relies on.

We also introduce a budget-pressure term that forces the alphas downward. This is the highest for the alphas away from the mean. The primary objective is to drive importance parameters α that fall significantly below the mean past the pruning threshold, thereby enabling the structural elimination of their corresponding layers

$$\mathcal{L}_{budget} = \left(\max \left(0, \frac{1}{L} \sum_{i=1}^{L} \alpha_i - T_{keep} \right) \right)^2 \tag{6}$$

The full Phase-2 objective comprises two regimes, depending on whether the active-layer budget has been reached. λ_2 denotes the Phase-2 gate regularisation weight . When the number of active layers is above the minimum budget L_{min}, we optimise:

$$\mathcal{L}_{P2}^{drop} = \mathcal{L}_{CE} + \lambda_2 \mathcal{L}_{bim} + \mathcal{L}_{budget}. \tag{7}$$

Once $|\mathcal{A}| = L_{\min}$, the pruner stops dropping, and the rest of the fine-tuning process goes on as it is.

2.3.3 Progressive Drop Criterion

Every K optimisation steps, the pruner inspects the current gate distribution of the active blocks and decides whether to drop the weakest one. Let $\overline{\alpha}$, s_α and α_{min} denote the mean, standard deviation, and minimum of the active gates. A drop is triggered only if all of the following hold: (i) the number of active layers exceeds the minimum budget L_{min}; (ii) a certain number of cooldown of C steps has elapsed since the last drop; (iii) $s_\alpha > s_{min}$, ensuring the gate distribution has separated sufficiently to be bimodal; (iv) $\alpha_{min} < \rho\overline{\alpha}$, a relative threshold against the running mean; and (v) $\overline{\alpha} - \alpha_{min} > g_{min}$, an absolute gap. After each drop, the corresponding block is frozen in place, so the remaining training updates only the surviving block and its gates.

2.4 Joint Depth-Width Compression via Slimmable Blocks

To bridge macro-level depth pruning with micro-level capacity allocation, we extend the alpha-gated framework by introducing a global, continuous width multiplier $w \in [w_{min}, 0.95]$.

2.4.1 Slimmable Transformer Blocks

We replace the standard Multi-Layer Perceptron (MLP) within each transformer block B_l with a slimmable subnetwork Tastan et al. (2026). The forward propagation for the l-th gated block is subsequently reformulated to integrate both the layer-specific depth gate α_l and the global width multiplier w:

$$z^l = (1 - \alpha_l) z^{l-1} + \alpha_l B_l(z^{l-1}; w). \tag{8}$$

2.4.2 TWO-WIDTH TRAINING RECIPE

To ensure the shared parameters remain robust across the full width spectrum and to prevent catastrophic forgetting (Wang et al., 2024) for smaller widths, we modify the training objectives in both Phase 1 and Phase 2 using a two-width sampling strategy Tastan et al. (2026). For every mini-batch, the model undergoes two distinct forward passes:

- **Full-Width Pass:** The model is executed at maximum capacity ($w = 1.0$), yielding $\mathcal{L}_{full}$.
- **Slim-Width Pass:** The model is executed at a uniformly sampled random width $w \sim \mathcal{U}(0.25, 95)$ yielding $\mathcal{L}_{slim}$.

The final accumulated loss used to update the optimiser is the average of the two forward passes:

$$\mathcal{L}_{total} = \frac{1}{2}(\mathcal{L}_{full} + \mathcal{L}_{slim}). \tag{9}$$

This dual-pass objective forces the internal representation to learn hierarchy and redundancy, ensuring that the truncated weight matrices function effectively even when a significant portion of the channels is removed. Additionally, we have also observed regularisation effects from the second forward pass discussed in Section 3.2.

2.4.3 ORTHOGONAL INFERENCE

Crucially, the progressive depth pruner evaluates the gate distribution α_l completely independently of the execution width w. While Phase 2 structurally drops layers to reduce depth, the two-width forward pass allows the retained layers to operate at variable widths during inference. Consequently, a single trained Phase 2 checkpoint yields a continuous 2-D Pareto frontier at inference time 3.4, allowing users to dynamically scale both $|\mathcal{A}|$ (active layers) and w (global width) to meet strict hardware budget requirements at desirable accuracy without retraining.

3 RESULTS

3.1 BLOCK-LEVEL PARAMETER PRUNING DURING VIT FINE TUNING

As discussed in Section 1, recent state-of-the-art pruning methods for ViTs appear to have moved away from Block Dropping. Hence, we first adopt block-dropping methods, such as Entrodrop (He, 2025) and Cosine (Fan et al., 2020), which have proven successful in large language models, to Vision Transformers (ViTs). As shown in Table 1, our method Alpha-Gated Pruning vastly outperforms them across all datasets. On CIFAR-100, our method achieves an accuracy of 91.85% compared to just 86.23% by Entrodrop. On the Food-101 datasets, our method beats the most obvious Cosine, the second-best-performing method, by a similar margin, achieving 91.37% compared to 86.12% by Cosine. On the Stanford Cars and Flowers datasets, the same trend is observed, with our method beating the corresponding second-best method by 4.05% and 2%, respectively.

This highlights the benefits of re-weighting transformer blocks based on importance criteria and progressive pruning rather than in one go. ViTs are structurally shallow, making each layer's contribution to spatial token mixing highly critical. Block dropping disrupts the flow of information and prevents continuous feature transformations, leading to severe representational collapse. Instead our Alpha-gated framework re-weights the blocks based on their importance, and drop the block with least weightage(also least important), minimizing the representational disruption.

Table 1: Comparison of our Alpha-Gated pruning method with established block dropping methods using VIT-B architecture.

Method	CIFAR-100	Food-101	Stanford Cars	Flowers-102
Entrodrop He (2025)	86.23%	83.37%	81.32%	92.34%
Cosine Fan et al. (2020)	83.19%	86.12%	81.14%	93.28%
OURS	**91.85%**	**90.78%**	**85.47%**	**98.92%**

3.2 JOINT BLOCK PRUNING AND ADAPTIVE WIDTH (BPAW-VIT)

As shown in Figure 2, our method achieves 91.16% accuracy on CIFAR-100 with just 9 blocks and 75% width, representing a 37.5% reduction in MACs. Vanilla alpha-gated pruning can reduce MACs by $\sim$25%. However, dropping a fourth block triggers severe representational collapse—accuracy plummets by 4.58% to 87.70% with a MACs reduction of 8.5%. Instead, our width-slicing method across 9 active blocks yields an additional 12.5% drop in MACs with minimal loss in accuracy (0.7%). In addition, we have observed that our 2-width sampled forward pass also acts as a regularizer, boosting the full-width accuracy by 0.6 %, which is visible in Figure 2.

We attribute this accuracy gain to two primary reasons: Firstly, the weight matrices are randomly truncated from a fixed starting index rather than randomly masked. Low hidden channels receive gradient updates every step, but the high hidden channels only do when a high width is sampled. This forces the network to concentrate robust, generalizable features within a shared low-width core, pushing instance-specific requirements into high-instance channels. This hierarchical organisation reduces co-adaptation among the feed-forward units. Secondly, because each optimisation step averages gradients from two distinct effective sub-networks. This penalises the shared parameters for relying on minima that are sharp with respect to structural perturbations. This pushes the trajectory towards flatter regions of the loss landscape.

Moreover, we can observe from Figure 2 that after a certain reduction in a certain number of Blocks, which happens to be 9 for ViT-B, reducing the width on surviving blocks yields a much gentler degradation trajectory than discarding entire Blocks. For 10 active blocks, stepping w from $1.00 \rightarrow 0.75 \rightarrow 0.50$ scales computation smoothly from 14.65 G $\rightarrow$ 12.31 G $\rightarrow$ 10.01 G, while preserving 92.68%, 91.64%, and 89.91% accuracy, respectively (a minor drop of $\sim$1.0% per 2.3 GFLOP reduction). A catastrophic drop is only triggered when the FFN is over-sliced to $w = 0.25$ (83.65% at 7.68 GFLOPs), pinpointing $w = 0.50$ as the practical operating boundary for linear slicing.

While depth-only pruning permanently drops transformer blocks to reduce computation, combining alpha-gated pruning with MoSE-style (Tastan et al., 2026) width slicing/adaptive width unlocks a granular, two-dimensional elastic architecture. By treating depth (active layers) and width (intermediate FFN capacity) as orthogonal axes of compression, a single trained checkpoint can dynamically scale to meet strict hardware MAC budgets during fine-tuning.

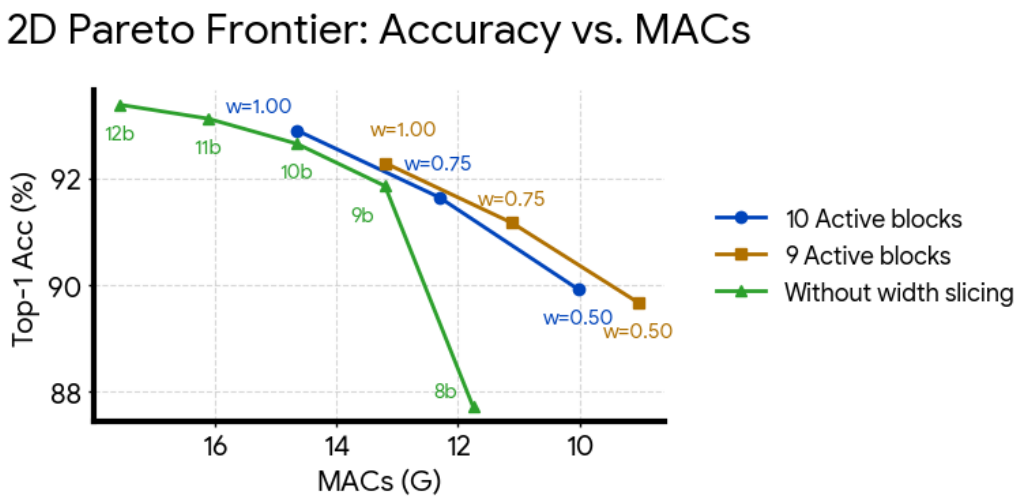


Figure 2: 2D Pareto frontier illustrating the trade-off between Top-1 Accuracy on CIFAR-100 and computation (MACs). The plot compares our Alpha-Gated Pruning trajectory without Width slicing against our BPAW-ViT(Alpha-Gated pruning and Width slicing) operating at 9 and 10 active blocks across varying FFN widths ($w \in \{0.50, 0.75, 1.00\}$).

3.3 COMPARISON WITH DEPTH PRUNING AND DEPTH WIDTH PRUNING METHODS ON VIT

As shown in Table 2, our joint Alpha gating and Width slicing method (BPAW-DEiT) outperforms the recently proposed state-of-the-art HetDPT (Su et al., 2026) and other relevant depth pruning or Depth and Width pruning methods on CIFAR-100. For DEIT-B, we achieve 90.78 % accuracy upon dropping 4 blocks (equivalent to 8 layers) and 80% width, thus significantly outperforming HetDEPT by 0.58% under the 10 layers dropped configuration and still saves 3.5 million parameters compared to baseline. It is also worth noting that methods like Het-DEPT and NOSE do not completely remove layers; instead, they replace the FFN with compensatory linear layers, which add another 2.5 million parameters based on the standard DEiT dimensions and the number of layers pruned.

Further, component-wise linear replacement introduces heterogeneity in the Transformer structure, breaking the symmetry, Early-Exit frameworks rely heavily on symmetry in ViT architectures to effectively train multiple student sub-networks. As a result, HetDEPT's symmetry distortion may preclude its straightforward extension to early-exit architectures.

Table 2: CIFAR-100 fine-tuning performance (3 runs avg.) comparison with state-of-the-art pruning methods on vision transformers. We include the calculated extra parameter overhead incurred by methods substituting transformer blocks with linear layers.

Method	Fine-tuning (%)	Linear probing (%)	Extra Params (M)
DeiT-B (Dense)	90.5	80.6	0.0
Evo-ViT	90.1	79.1	0.0
EVIT	90.0	80.2	0.0
TPS	90.1	76.5	0.0
NOSE (5 layers)	90.3	81.3	20.6
NOSE (6 layers)	90.2	80.6	18.3
HeDEPT (8 layers) Su et al. (2026)	90.4	81.2	10.9
HeDEPT (10 layers) Su et al. (2026)	90.2	80.6	3.5
BPAW-DEIT(ours)	**90.78**	-	**0.0**

3.4 SYNERGY WITH EARLY EXIT ARCHITECTURES (ENVIT)

Samples that are easy to classify should require fewer neural network layers compared to harder ones. Early exit networks exploit this principle to reduce overall inference latency, thereby also reducing energy consumption (by reducing MACs). Hence, early-exit network designs have recently garnered significant attention. However, a closer look at the methods reveals that these designs never focus on reducing model memory, which is achieved by the pruning strategies discussed above. From Table 1, we observe that on combining with Envit, our method(BPAW-ENVIT), which involves joint pruning and Early Exit, nearly matches the accuracy of ENVIT (89.19% vs 89.23%) with competitive speedup values (1.94X vs 2.05X). Above all, in this setting, it has just 10 active blocks and 80% width blocks and vastly reduced memory footprint(27%) thus making it more suitable for edge devices than the vanilla ENVIT. In addition, on the other fine-tuning datasets, BPAW-VIT consistently outperforms ENVIT and other early-exit methods, achieving 89.03% on Food-101, 88.67% on Stanford Cars, and 96.04% on the Flowers-102 dataset. In terms of accuracy, while maintaining competitive speedups.

3.5 TOPOLOGICAL ANALYSIS OF TRI-AXIS DYNAMIC SYNERGY

Unifying structural block dropping, width slicing, and early-exit confidence thresholds provides three orthogonal control axes that span a dense global Pareto frontier. As is evident from Figure 3, retaining more active blocks tends to naturally boost accuracy numbers but provide lesser speedups, as we can observe by comparing Figure3A against Figure3B. We also observe that, for a given number of dropped blocks and FFN width, the model tends to achieve rapid speedups with minimal loss of accuracy by reducing the confidence threshold to 0.7; beyond this threshold, the model quickly degrades.

Table 3: Comparison of our BPAW-ENVIT with other early exit-based pruning methods.

Methods	CIFAR-100		Food-101		Stanford Cars		Flowers-102	
	Acc.	Speed-up	Acc.	Speed-up	Acc.	Speed-up	Acc.	Speed-up
ViT-B/16	91.66%	1.00x	91.19%	1.00x	87.75%	1.00x	97.67%	1.00x
VIT-EE	87.63%	1.83x	88.85%	1.75x	83.41%	2.04x	94.26%	2.13x
SDN	88.12%	1.86x	88.45%	2.12x	85.45%	2.01x	95.07%	2.15x
BYOT	87.20%	1.82x	87.93%	2.03x	86.05%	2.01x	94.89%	2.00x
LGVIT	88.14%	1.94x	88.46%	1.97x	79.11%	1.45x	93.14%	1.63x
BEEM	88.87%	1.66x	89.15%	2.06x	83.70%	1.70x	93.04%	2.14x
COSEE	87.50%	1.84x	87.78%	1.89x	80.94%	1.72x	91.96%	1.48x
ENVIT Dong (2026)	89.23%	2.05x	89.15%	2.20x	87.03%	2.05x	95.33%	2.23x
BPAW-ENVIT(ours)	**89.19%**	**1.94x**	**89.03%**	**2.47x**	**88.67%**	**1.94x**	**96.04%**	**2.47x**

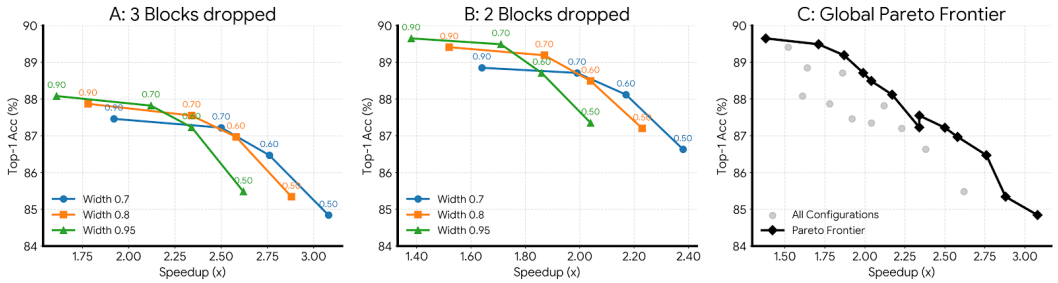


Figure 3: A and B display the Top-1 Accuracy against inference speedup for configurations with 3 and 2 blocks dropped respectively, parameterised by FFN widths and varying early-exit confidence thresholds. Panel C aggregates these operating points to map the global Pareto frontier.

The topological analysis reveals that optimal intermediate operating points are dominated by mixed-dimension configurations rather than single-axis extremes. For situations demanding high target accuracy, the optimal strategy would be the 2-block-dropped topology with moderate-width slicing and conservative exit thresholds. Conversely, for strict latency budgets demanding speedups $\geq 2.25\times$, the optimal configuration is to stick to the 3-block dropped topology coupled with narrower FFN widths and aggressive early-exit classifications depending on the latency budget. By aggregating all configurations into the global Pareto frontier, we establish a definitive set of optimal configurations for any desired speedup requirement. This frontier serves as a practical recipe, empowering users to easily select the best combination of Depth, Width, and Exit thresholds tailored to their specific latency and accuracy constraints.

4 Discussion

Figure 1 summarises our central result: across the four axes that jointly determine deployability-speedup, accuracy, memory (bubble size), and GMACs (colour)-our method(BPAW-DEiT) dominates competing depth-pruning approaches rather than trading one axis for another. Purely heuristic block dropping techniques like (Entrodrop, Cosine) demonstrate low accuracy and speedup, with larger model memory, confirming the representational collapse from ignoring block heterogeneity. Compensatory-layer pruning methods (HeDEPT, NOSE) recover accuracy and reduce model memory while not aiming to achieve significant speedups. Precisely, the memory cost pruning is meant to eliminate. Our BPAW-DEiT outperforms HetDEPT in both accuracy and speedup while maintaining a reduced memory footprint.

On the other hand, our joint pruning and early-exit method BPAW-ENVIT matches EnViTs accuracy (89.19% vs. 89.23%) and even exceeds its speedup on two of four benchmarks, achieving a competitive speedup with a significantly reduced memory footprint ($\sim$27% less). This happens because alpha-gated pruning permanently removes blocks rather than routing around them, whereas EnViT must keep the full network resident. The depth–width results (Figure3) further show that width slicing degrades more gracefully than additional block drops beyond 9 active blocks, motivating our treatment of depth and width as orthogonal, jointly tunable axes. This allows users to select the best configuration based on their hardware capabilities, latency budget and accuracy requirements.

AI USE STATEMENT

AI tools were employed to assist with code implementation, debugging, and language editing of this manuscript. The authors have reviewed all AI-assisted content and assume full responsibility for the accuracy and integrity of the final text.

ETHICS STATEMENT

This work concerns architectural efficiency for Vision Transformers and involves no human subjects or personally identifiable data; all experiments use standard public benchmarks (CIFAR-100, Food-101, Stanford Cars, Flowers-102). By reducing ViT memory footprint and inference cost without retraining from scratch, our method can lower the energy and hardware requirements of deploying vision models, improving accessibility on resource-constrained edge devices. We note, however, that this same edge efficiency could facilitate the deployment of vision models for unauthorised surveillance on low-power devices (e.g., cameras, drones); we encourage downstream users to consider applicable privacy regulations when deploying such compressed models. This dual-use risk is inherent to the underlying vision task rather than specific to our compression technique. We are not aware of any conflicts of interest.

REPRODUCIBILITY STATEMENT

We describe the block pruning, adaptive width, and early-exit design strategies in the methods section for clear reproduction of this work. The code and other relevant details will also be shared publicly upon the paper's acceptance.

REFERENCES

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *International Conference on Learning Representations (ICLR)*, 2022.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, pp. 446–461, 2014.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009.

et al. Dong. Envit: Enhancing the performance of early-exit vision transformers via exit-aware structured dropout-enabled self-distillation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations*, 2020.

Zhiwei Hao, Jianyuan Guo, Ding Jia, Kai Han, Yehui Tang, Chao Zhang, Han Hu, and Yunhe Wang. Learning efficient vision transformers via fine-grained manifold distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 9164–9175, 2022.

et al. He. Entropy-based block pruning for efficient large language models. *arXiv preprint arXiv:2504.03794*, 2025.

Jianing He, Qi Zhang, Hongyun Zhang, Xuanjing Huang, Usman Naseem, and Duoqian Miao. COSEE: Consistency-oriented signal-based early exiting via calibrated sample weighting mechanism. In *AAAI Conference on Artificial Intelligence*, 2025.

Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better ImageNet models transfer better? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2661–2671, 2019.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 554–561, 2013.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

S. Lin, P. Lyu, D. Liu, T. Tang, X. Liang, A. Song, and X. Chang. MLP can be a good transformer learner. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19489–19498, 2024.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10012–10022, 2021.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Sixth Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP)*, pp. 722–729, 2008.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Nikhila Ravi, Benjamin Graham, Haoqi Fan, Timothée Johnson, Armand Joulin, David Lopez-Paz, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2023.

Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. ImageNet-21K pretraining for the masses. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Zhenfeng Su, Kang Zhao, Han Bao, Tao Yuan, Zhongzhe Hu, Xianzhi Yu, and Wenxuan Wang. Rethinking depth pruning for vision transformers: A heterogeneity-aware perspective. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306, 2026.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 843–852, 2017.

Nurbek Tastan, Stefanos Laskaridis, Karthik Nandakumar, and Samuel Horváth. MoSE: Mixture of slimmable experts for efficient and adaptive language models. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

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)*, pp. 10347–10357, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 46(8):5362–5383, 2024.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 568–578, 2021.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP)*, pp. 38–45, 2020.

Kan Wu, Jinnian Zhang, Houwen Peng, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. TinyViT: Fast pretraining distillation for small vision transformers. In *European Conference on Computer Vision (ECCV)*, pp. 68–85, 2022.

Guanyu Xu, Jiawei Hao, Li Shen, Han Hu, Yong Luo, Hui Lin, and Jialie Shen. LGVIT: Dynamic early exiting for accelerating vision transformer. In *Proceedings of the 31st ACM International Conference on Multimedia (ACM MM)*, pp. 9103–9114, 2023.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, 2014.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12104–12113, 2022.

A Appendix

A.1 Experimental Setup

All experiments were conducted on enterprise-grade workstation GPUs with hardware acceleration. The library stack was PyTorch 2.8.0+cu128, torchvision, and timm (PyTorch Image Models), with the `vit_base_patch16_224` backbone initialised from ImageNet-pretrained (`augreg2_in21k_ft_in1k`) weights.

A.2 Hyperparametres

Table 4: Best hyperparameters for joint block-pruning + width-slicing + early-exit (BPAW-ENVIT) training, per dataset.

Dataset	LR	α-LR	WD	Batch	Epochs	Min Layers	Keep Ratio	α-Thr	Cool-down
CIFAR-100	5e-5	1e-2	5e-3	64	30	9	0.70	0.50	800
Food-101	1e-4	1e-2	5e-3	64	30	9	0.60	0.70	800
Stanford Cars	5e-5	1e-2	5e-3	64	30	9	0.60	0.70	300
Flowers-102	5e-5	5e-3	5e-3	16	50	9	0.70	0.70	100

Shared hyperparameters (fixed across all datasets): exit-head placement interval (`exit_every`) = 2 blocks; early-exit distillation weight p_{max} = 0.3; distillation balance α = 0.5; distillation temperature τ = 4.0; inference-time exit confidence threshold = 0.5; initial gate value α_{init} = 0.9; bimodal regularisation weight λ_{bim} = 0.1; budget regularisation weight λ_{budget} = 0.5 (CIFAR-100,Food101) / 0.4 (Stanford Cars)/ 0.25(Flowers-102); pruning-check interval = 100 steps (20 for Flowers-102); width range $w \in [0.25, 1.0]$ with step 0.05, sampled uniformly under the two-width training recipe (full-width pass + one randomly sampled slim-width pass per batch).

A.3 PHASE 1: ACCURACY-FIRST FINE-TUNING

The goal of the first phase is to achieve a viable level of accuracy for the task at hand. This is to enable proper ranking of blocks using entropy-based initialisation. We start from an ImageNet-pretrained ViT-B, which is wrapped in an alpha-gated container with $\alpha_{raw,l}$ initialised so that $\alpha_l = 0.9$ for all l. The model is then trained using soft target cross entropy loss:

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}\tilde{y}_{ic}\log p_{ic}, \tag{10}$$

where $\tilde{y}$ is the mixed label and $p = \text{softmax}(\text{logits})$.

To prevent the gates from drifting to unity while not yet pruning, we add a very small regulariser on the mean gate magnitude:

$$\mathcal{L}_{P1} = \mathcal{L}_{CE} + \lambda_1 \cdot \frac{1}{|\mathcal{A}|}\sum_{l\in\mathcal{A}}\alpha_l, \tag{11}$$

with $\lambda_1 = 10^{-5}$. This near-zero weight ensures the gate magnitudes are free to respond to the task signal without prematurely dropping any layer. The best validation checkpoint is saved.

A.4 PHASE 2: PROGRESSIVE LAYER PRUNING

After Phase 1 is complete, Phase 2 progressively removes less important layers during fine-tuning. We derive a per-layer gate initialisation from an entropy criterion, and train with a pruning-aware objective while a scheduler monitors the gate distribution and drops layers one at a time. By raising the pressure on the Alpha gates and dropping the weakest block whenever its gate falls significantly below the active mean. We derive a per-layer gate initialisation from an entropy criterion, and train with a pruning-aware objective while a scheduler monitors the gate distribution and drops layers one at a time.

B EXTENDED RELATED WORK

This section provides an extended discussion of prior work across the four areas most relevant to our contributions: depth (block) pruning, width/channel-level compression, early-exit inference, and the fine-tuning paradigm that motivates our design choices. Where possible, we relate each line of work back to the specific limitations that motivate BPAW-ViT.

B.1 DEPTH PRUNING AND BLOCK DROPPING

Reducing the effective depth of deep networks by removing or bypassing entire layers has a long history outside vision transformers. In the LLM literature, structured stochastic depth methods such as LayerDrop (Fan et al., 2020) randomly drop transformer sublayers during training so that the resulting network is robust to layer removal at inference time, while more recent entropy-guided approaches such as Entrodrop (He, 2025) use information-theoretic criteria to rank layer importance and drop low-contribution layers post hoc. These methods were designed for deep, highly redundant LLM stacks (often 30+ layers), where the marginal contribution of any single layer to the residual stream is small. As we show in Section3.1, naively transplanting these heuristics onto ViTs—which are substantially shallower and where each block performs comparatively more critical spatial token mixing—causes severe representational collapse rather than graceful degradation.

Within the vision-transformer-specific literature, several works have proposed depth-pruning strategies that attempt to preserve accuracy by compensating for dropped blocks rather than removing them outright. HetDEPT (Su et al., 2026) explicitly frames depth pruning through a heterogeneity-aware lens, acknowledging that ViT blocks contribute unevenly to the final representation, but addresses this by substituting pruned blocks with lightweight compensatory linear layers rather than adapting the pruning criterion itself. NOSE (Lin et al., 2024) similarly replaces transformer sublayers with MLP-based approximations to preserve downstream accuracy. Both approaches, while effective at mitigating accuracy loss, reintroduce nontrivial parameter counts (up to several million

additional parameters, as quantified in Table2) and, more importantly, break architectural uniformity by producing a network in which some blocks are standard transformer layers and others are linear approximations with different representational capacity. This disjointedness is particularly problematic for downstream applications that assume a symmetric, uniform block structure, including manifold-distillation-based efficiency methods (Hao et al., 2022) and, as we discuss below, early-exit training procedures. Our alpha-gated approach differs fundamentally in that dropped blocks are reduced to true identity mappings rather than being replaced, preserving both zero parameter overhead and full architectural symmetry.

B.2 Width and Channel-Level Compression

Orthogonal to depth reduction, a separate line of work compresses networks along the width dimension. Slimmable neural networks and their successors train a single network to operate at multiple channel-width configurations simultaneously, typically via switchable batch normalization or sandwich-rule training over a discrete set of widths. Closest to our width-slicing component is MoSE (Tastan et al., 2026), which introduces mixture-of-slimmable-experts training for efficient, adaptive language models via a dual-width sampling recipe. We adapt this dual-width training principle to the vision setting, applying it specifically to the feed-forward sublayers of the surviving (non-pruned) blocks in our alpha-gated architecture. Unlike prior slimmable-network approaches, which treat width as the sole compression axis, our contribution lies in demonstrating that width and depth can be treated as genuinely orthogonal, independently controllable axes within a single fine-tuned checkpoint, enabling the 2D Pareto frontier described in Figure3.

Other efficiency-oriented ViT architectures address computational cost through complementary mechanisms rather than explicit pruning: hierarchical designs such as Swin Transformer (Liu et al., 2021) and Pyramid Vision Transformer (Wang et al., 2021) reduce cost via windowed or pyramidal attention, while token-reduction methods such as EViT and TPS (evaluated as baselines in Table 2) reduce the number of spatial tokens processed rather than the number of layers or channels. These approaches are complementary to, rather than competitive with, block- and width-level pruning, and could in principle be composed with our method in future work.

B.3 Early-Exit Inference

Early-exit networks attach auxiliary classification heads at intermediate depths and use confidence- or entropy-based criteria to terminate inference early for easy inputs, reducing average-case latency without altering the underlying architecture. Representative methods include BranchyNet-style shallow-deep networks (SDN), You-Only-Train-Once BYOT-style self-distillation**??**, LGViT (Xu et al., 2023), which introduces heterogeneous exit-head designs tailored to ViT feature maps, COSEE (He et al., 2025), which calibrates exit confidence via consistency-oriented signals, and EnViT (Dong, 2026), which uses exit-aware structured dropout and self-distillation to train multiple student sub-networks sharing a backbone. A common thread across this family is reliance on architectural symmetry: because every sample may exit at any depth, the shared backbone must produce consistent, well-calibrated intermediate representations at every candidate exit point, which is typically achieved through distillation from the full-depth model into each shallower sub-network. As we argue in Section 3.4, this symmetry requirement is precisely what compensatory-layer pruning methods (HetDEPT, NOSE) are liable to violate, since their disjointed feature spaces make the intermediate representations at pruned positions structurally inconsistent with the distillation targets these methods assume. Our approach, by preserving both architectural symmetry and monotonic hidden-state behavior, is designed to remain compatible with this class of methods, which we verify empirically in Section 3.4.

Critically, and as emphasized throughout this paper, none of the early-exit methods above reduce the resident memory footprint of the deployed model: all backbone layers and all auxiliary exit heads must be loaded regardless of which exit a given sample ultimately uses. This makes early-exit methods, in isolation, a poor fit for strictly memory-constrained edge and embedded hardware, motivating our combination of early exiting with genuine, worst-case-reducing block pruning.