Carve VFM: CARVING SPECIALISTS OUT OF GENERALISTS, AUTOMATIC TASK-SPECIFIC COMPRESSION OF VISION FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision foundation models (VFMs) provide transferable visual representations, but their inference cost limits deployment in many settings. Fine-tuning specializes a VFM without shrinking it; distillation shrinks it, but through expensive teacher-guided training. Structured pruning offers a more direct route by extracting a compact subnetwork from the VFM. However, as compression gets aggressive, existing methods fail because they treat subnetwork selection and recovery as separate axes. Across layers, capacity must follow task sensitivity rather than spread uniformly. Within a layer, a removal cannot be judged by importance alone; its impact also depends on how well the lost contribution can be recovered from what remains. Selection and recovery must therefore be decided jointly. We introduce *Carve VFM*, which resolves this interdependence. Given a VFM, a calibration set, and a parameter budget, it (i) searches for task-specific capacity allocations across layers; (ii) guides the search via an adaptive recovery-aware fitness criterion; and (iii) reconstructs removed behavior through iterative closed-form low-rank corrections merged into the surviving weights. The result is a dense encoder produced in minutes on a single GPU without backbone retraining. Across twelve benchmarks and four models from DINOv3 and MetaCLIP-2, *Carve VFM* achieves the highest reported mean accuracy in seven of eight backbone–budget configurations among the evaluated dense structured-pruning methods. Further fine-tuning surpasses the strongest evaluated edge model (**88.1%** vs. 85.8%), and carved models cut compute and memory while delivering **1.3–2.0**$\times$ higher GPU throughput than the uncompressed one. These results show that the compact model you need is inside the VFM; it only has to be *carved* out.

1 INTRODUCTION

Vision foundation models (VFMs) derive their value from generality: their representations transfer well across tasks and domains (Vemulapalli et al., 2024). In deployment, however, this generality can become a practical liability (Aduvuku et al., 2026). A model trained to represent a broad visual world may ultimately serve a narrow task such as distinguishing cars or tissues. We call resolving this mismatch *VFM specialization*: given a pretrained VFM, a target task, and a parameter budget, derive the most accurate task-specific model that fits within that budget.

Pretraining is done once and amortized over many uses, whereas inference is paid for on every input for the model's lifetime. In production, inference indeed accounts for a large share of ML energy use (Patterson et al., 2022).

Fine-tuning techniques fit the VFM to the target task but preserve its computational cost. Parameter-efficient updates reduce training cost but still execute the full model at inference (Hu et al., 2022). On the other side, knowledge distillation can yield compact student models that run faster, but it requires a costly teacher-guided training process (Hinton et al., 2015; Beyer et al., 2022).

Structured pruning offers a more direct route by removing entire components, such as MLP channels or attention heads, to produce a smaller dense model (Ahsaei & Raji, 2026). Beyond efficiency, pruning can also improve transfer learning by eliminating pretrained components that are detrimental to a target task (Liu et al., 2021). These observations suggest that a VFM contains task-specific

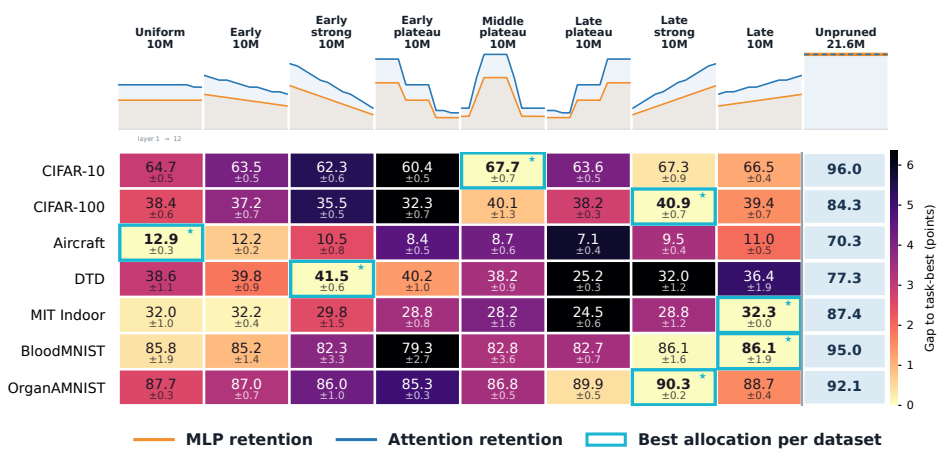


Figure 1: Task accuracy after pruning ViT-S from 21.6M to 10M parameters under different layer-wise allocations. Results are mean±std over 3 seeds. The unpruned model is included for reference.

subnetworks hidden within its general-purpose architecture, a hypothesis supported by recent work that successfully extracts such subnetworks (Zheng et al., 2022; Shi et al., 2023; Simoncini et al., 2025). Local importance scores can rank structures within a layer, yet they provide little guidance on how model capacity should be distributed across layers or which pruning operator is suitable for each layer. As Figure 1 illustrates, under the same 10M parameter budget, the optimal allocation varies across tasks, while poorly allocated removals can degrade accuracy. A first challenge is therefore *allocation*; specialization requires discovering a task-specific allocation to find a good subnetwork within the VFM.

At mild compression, pruning can concentrate on redundant or detrimental structures, often preserving and occasionally improving accuracy. As the budget tightens, however, useful structures can also be removed. At this point, a removal depends not only on the importance of its contribution, but also on how well that contribution can be reconstructed from the remaining structures. Recovery methods exploit this redundancy by compensating output bias or fitting low-rank corrections to the compression residual (An et al., 2024; Liu et al., 2024), yet they are typically applied only after subnetwork selection. As a result, two equal-budget subnetworks may incur similar degradation after pruning, yet differ substantially in how faithfully their lost behavior can be reconstructed. The second gap is therefore *recoverability*: subnetwork selection should favor structures whose lost contributions can be recovered from the surviving model, rather than simply the least important ones.

Ideally, the search would rank subnetworks by their recovered task accuracy. In practice, this would require repeatedly extracting, repairing, and evaluating thousands of candidate subnetworks, making exhaustive evaluation infeasible. Cheap local criteria enable efficient exploration but cannot capture cross-layer interactions or the preservation of the target task's decision boundaries. Conversely, feature reconstruction alone is not enough, as faithfully reproducing pretrained representations does not guarantee task discrimination (Liu et al., 2022). The third gap is therefore *evaluation*: the search requires a proxy that is both inexpensive for exploration and predictive of the recovered task accuracy that ultimately determines deployment performance.

We address these challenges with *Carve VFM*, which jointly searches capacity allocations and attention removal operators under a parameter budget, using recoverability-aware local-global evaluation. The selected subnetwork is carved and repaired into a dense task-specific model. Our main contributions are:

- **Task-specific allocation.** We formulate VFM specialization as allocation under a parameter budget and introduce a unified search space over layer-wise MLP and attention capacity together with attention removal operators.

- **Adaptive local-to-global selection.** We introduce a recoverability-aware fitness in which cached conditional-recovery estimates enable inexpensive exploration, while a held-out task audit detects when the local ordering is unreliable and adaptively guides the search.

- **Mergeable iterative repair.** We develop a closed-form low-rank recovery that iteratively transfers useful behavior from removed structures into the surviving weights and leaves no additional inference branch.

2 Related Work

Compression of pretrained transformers. Compression reduces deployment cost while trying to preserve the pretrained knowledge. Unstructured pruning (Frantar & Alistarh, 2022; Sun et al., 2024) and quantization (Yuan et al., 2022) compress weights or numerical precision, but typically preserve tensor shapes and therefore rely on sparse or low-precision hardware for efficient execution. Structured pruning instead removes components, such as channels or blocks, directly producing smaller dense models. Recent methods have improved structure selection through joint importance estimation (Zheng et al., 2022), progressive mask learning (Shi et al., 2023), self-supervised subnetworks (Simoncini et al., 2025), elastic architectures (Zhu et al., 2025), and activation-aware pruning (An et al., 2024). Collectively, these works demonstrate that pretrained VFMs contain compact subnetworks with strong downstream performance.

Allocation and recovery. The remaining challenge is identifying which subnetwork to extract under a given budget. Budget-aware methods such as AMC (He et al., 2018) and NetAdapt (Yang et al., 2018) search layer-wise capacity allocations, while most pruning methods rank candidate removals using importance or reconstruction error. Recovery methods such as EoRA (Liu et al., 2024) then compensate the selected compressed model through low-rank reconstruction. In existing approaches, however, architecture selection and recovery remain largely separate stages. As a result, pruning decisions are made without considering how well removed structure can be recovered.

3 Methodology: Carve VFM

Problem setup. Let f_θ denote a pretrained VFM with L transformer layers and let $\mathcal{C} = \{(x_i, y_i)\}_{i=1}^{n}$ be a small calibration set for a target task. Given a parameter budget B, the objective is to extract a dense subnetwork g that satisfies the deployment budget while maximizing downstream performance,

$$g^\star = \arg \max_{P(g) \le B} \mathrm{Acc}_{\mathrm{val}}(g). \tag{1}$$

Direct optimization of Equation (1) is intractable because the space of budget-feasible subnetworks grows combinatorially with model size, while evaluating a single candidate requires extracting, repairing, and assessing a separate model.

Previous approaches typically approximate this objective by ranking channels, or other structures according to their individual importance and pruning the least important ones. Under aggressive compression, however, importance is insufficient. The quality of the extracted subnetwork depends jointly on how capacity is allocated across layers, which removal operators are applied, and how well the discarded channels can be recovered from the surviving ones. Because these decisions are strongly coupled, optimizing them independently generally leads to suboptimal results.

We address this interdependence with *Carve VFM*, a budget-constrained architecture extraction framework that jointly optimizes allocation, removal structure, and recoverability. Figure 2 summarizes its three-stage pipeline. Stage 1 estimates task-dependent removal statistics, Stage 2 searches candidate architectures using an adaptive local-global fitness, and Stage 3 physically carves and repairs the selected architecture. The resulting model is a small dense model within the target budget, optionally fine-tuned for better performance.

Stage 1: Task-related statistics. A single forward/backward pass of the calibration set $\mathcal{C}$ through the frozen VFM caches the task-dependent statistics (Figure 2, box 1). These include second-order input statistics for conditional reconstruction (Frantar & Alistarh, 2022), task-gradient importance for identifying directions that most influence the downstream objective, and reference activations for subsequent repair and held-out task evaluation. These quantities are computed only once and reused later throughout the search.

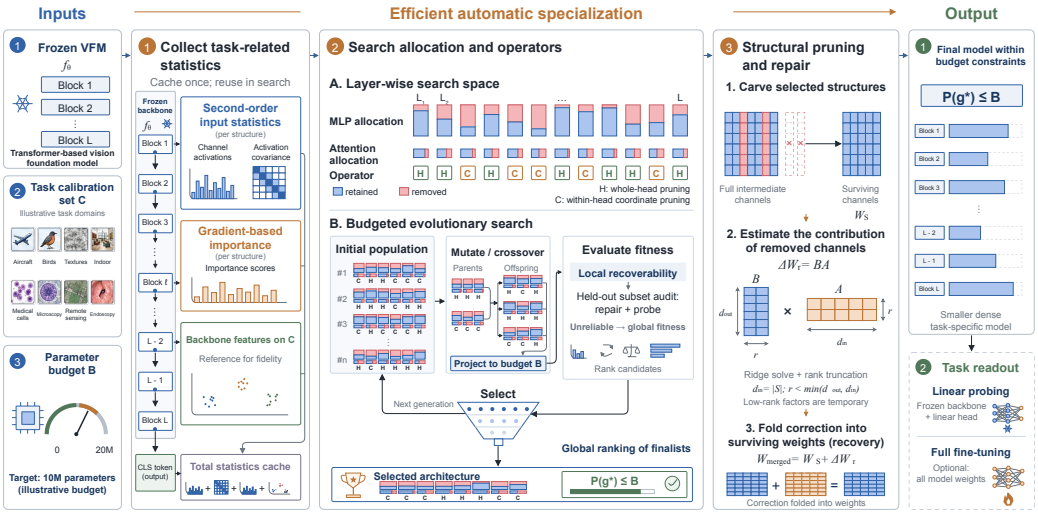


Figure 2: Overview of *Carve VFM*. From a VFM, a calibration set, and a parameter budget B, the method (1) estimates removal damage, (2) searches layer-wise MLP retention and attention operators, and (3) carves and repairs the selected architecture. The output is a standard dense task-specific model within B.

Stage 2: Budgeted subnetwork search. Stage 2 searches for the best budget-feasible subnetwork by jointly optimizing capacity allocation across layers and the attention removal operator used in each layer. The resulting search space is explored through an evolutionary optimization process, while an adaptive local-global fitness is used for efficient evaluation.

Search space. A candidate architecture assigns each transformer layer an independent MLP retention ratio s_l^{mlp}, an attention retention ratio s_l^{att}, and an attention removal operator $o_l \in \{\mathrm{H}, \mathrm{C}\}$, while the overall parameter count is constrained by the target budget B. The MLP ratio reduces the hidden dimension by removing hidden channels, implemented by deleting the corresponding rows of the MLP up-projection W_{up} and matching columns of the down-projection W_{down}.

Attention supports two complementary removal operators. **H** removes complete attention heads. **C** instead preserves every head while removing matched coordinates within the query/key pair (W_Q, W_K) and within the value/output pair (W_V, W_O), with separate retained sets for the two pairs. Consequently, H concentrates a given reduction in fewer heads, whereas C distributes the same parameter reduction across all heads, allowing compression even when no complete head is expendable.

Search procedure. This mixed continuous–discrete space is explored using an evolutionary algorithm (Figure 2, box 2B). Candidate subnetworks are initialized from diverse allocations, evolved through mutation and crossover, and projected onto the budget B before evaluation. To make evaluations efficient, an adaptive local-global fitness is used to guide the search. Local recoverability first ranks the population, after which held-out accuracy audits a small rank-stratified subset. If the audit preserves the local ranking, exploration continues under the local fitness. Otherwise, if the rankings diverge or the audit discovers a substantially better candidate than predicted locally, the local score is deemed unreliable and held-out accuracy guides a capped continuation. In both cases, the global score ranks the final candidates.

Local fitness: recoverable damage. The local fitness estimates whether refitted surviving weights can reconstruct removed functionality, unlike magnitude- or gradient-based importance. Held-out evaluations of repaired candidates audit this search-time proxy.

Let Z be a projection input, W its downstream weight, and $G = Z^\top Z/n + \lambda I$ the regularized Gram matrix. For a removable group $\mathcal{I}$ (an MLP channel or attention head), its local damage is

$$d(\mathcal{I}) = \mathrm{tr}\big(W_{:\mathcal{I}}[G_{\mathcal{I}\mathcal{I}}^{-1}]^{-1}W_{:\mathcal{I}}^\top\big)\,. \tag{2}$$

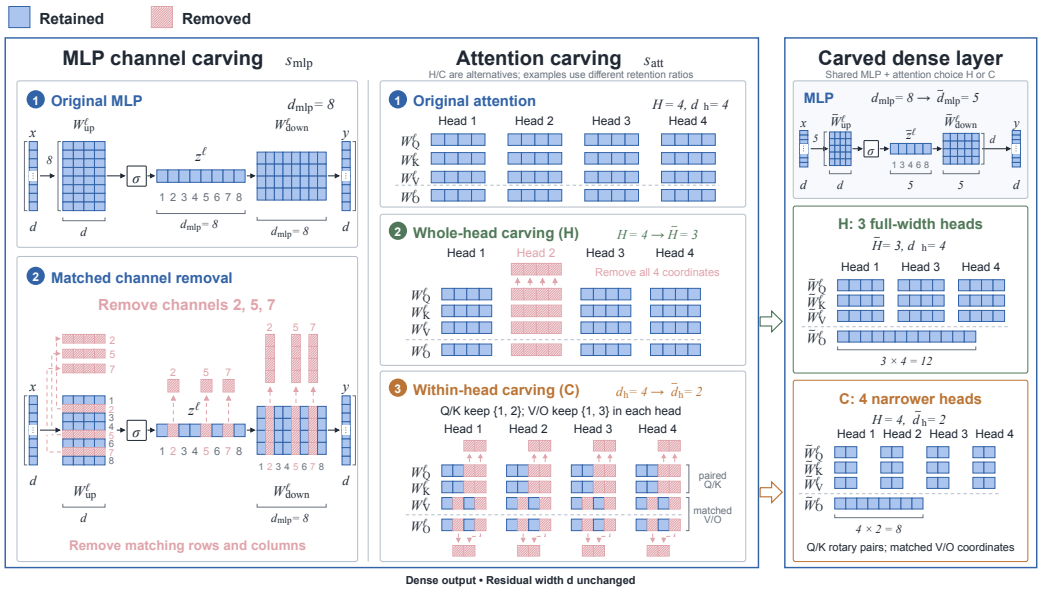


Figure 3: Structured pruning used in *Carve VFM*. MLP carving removes hidden channels with matching rows and columns from its two projections. Attention removes either complete heads (H) or some channels within each head (C).

Here $[G^{-1}_{\mathcal{I}\mathcal{I}}]^{-1}$ is the conditional covariance of the removed group given the surviving inputs. Following OBC (Frantar & Alistarh, 2022), groups are removed iteratively, the surviving weights are refit, and G^{-1} is updated by a Schur complement. This produces nested removal orders and cumulative damage curves $E^o_l(k)$ for every layer and operator, cached for MLP channels, attention heads, and V/O coordinates. Q/K rotary pairs are ranked separately by their perturbation of calibration logits, leaving their bilinear distortion to the global fitness. A candidate architecture is scored by Equation (3), requiring neither model construction nor a forward pass.

$$\mathcal{Q}_{\text{local}}(\mathbf{a}) = -\sum_l E^{o_l}_l(k_l). \tag{3}$$

Global fitness: task discriminability. Because the local score is additive across layers, it cannot capture cross-layer interactions or the preservation of task discriminability. We therefore periodically audit a small, rank-stratified subset of candidates using held-out accuracy. After applying the repairs, we fit a regularized linear probe on one part of each of T stratified calibration splits of C and evaluate it on the others. This held-out accuracy is the search-time proxy for Equation (1).

Stage 3: Carving and repair. Stage 2 specifies a budget-feasible subnetwork; Stage 3 realizes its smaller tensor dimensions and repairs the surviving weights without restoring any removed dimension. The result therefore remains a dense model within the target budget B.

Carving: making the architecture smaller. A structural feature is usually produced by one projection and consumed by another. Carving therefore removes the same index from both sides of this producer–consumer path. For an MLP layer, keeping the hidden-channel set $\mathcal{S}^{\text{mlp}}_l$ gives an ordinary MLP with smaller hidden width $|\mathcal{S}^{\text{mlp}}_l|$ as shown below.

$$\widetilde{W}^l_{\text{up}} = W^l_{\text{up}}[\mathcal{S}^{\text{mlp}}_l, :], \qquad \widetilde{W}^l_{\text{down}} = W^l_{\text{down}}[:, \mathcal{S}^{\text{mlp}}_l]. \tag{4}$$

Attention follows the same paired-indexing principle, with the survivor sets determined by the selected operator o_l:

$$\begin{aligned} \widetilde{W}^l_p &= W^l_p[\mathcal{S}^l_p, :], \quad p \in \{Q, K, V\}, \\ \widetilde{W}^l_O &= W^l_O[:, \mathcal{S}^l_O]. \end{aligned} \tag{5}$$

For whole-head carving ($o_l = \mathrm{H}$), all projections share the same survivor set, corresponding to complete retained heads. For column carving ($o_l = \mathrm{C}$), Q/K and V/O use separate survivor sets, preserving rotary pairs and matched value/output coordinates, respectively. See Figure 3 for an illustration of the two carving operators. In all cases, the biases are sliced accordingly, producing physically smaller dense projections.

Repair: recovering removed behavior. Carving removes entire structures and therefore part of each layer's output. We recover this lost behavior by predicting the removed output from the surviving activations. Similar to calibration-based low-rank recovery (Liu et al., 2024), but formulated directly in the reduced coordinate space, the resulting correction can be merged exactly into the carved model.

Partition a linear projection into surviving indices $\mathcal{S}$ and removed indices $\mathcal{I}$. The output lost by removing $\mathcal{I}$ is $Y_{\mathcal{I}} = Z_{\mathcal{I}} W_{\mathcal{I}}^{\top}$. We reconstruct this contribution by solving

$$\begin{aligned} \Delta W^{\star} &= \arg\min_{\Delta W} \|Z_{\mathcal{S}}(\Delta W)^{\top} - Y_{\mathcal{I}}\|_F^2 + \lambda \|\Delta W\|_F^2 \\ &= Y_{\mathcal{I}}^{\top} Z_{\mathcal{S}} (Z_{\mathcal{S}}^{\top} Z_{\mathcal{S}} + \lambda I)^{-1}, \qquad \Delta W_r = \mathrm{SVD}_r(\Delta W^{\star}). \end{aligned} \tag{6}$$

The rank-r reconstruction $\Delta W_r = U_r \Sigma_r V_r^{\top}$, using the leading singular triplets, regularizes the solve without changing the reduced shape. The correction merges as $W_{\mathrm{merged}} = W_{\mathcal{S}} + \Delta W_r$, with no additional inference parameters or operations. The same formulation repairs the MLP down-projection, the V/O projections under operator C, and the retained-head output projection under operator H; residual means are folded into the existing biases. Q/K under operator C is repaired separately because its effect is bilinear in the attention logits.

Unlike post-hoc recovery methods such as EoRA (Liu et al., 2024), Carve VFM integrates repair into architecture search. Each cached curve refits the surviving weights and updates G^{-1} between group removals. The curves guide Stage 2 local ranking; held-out evaluations of repaired candidates audit that ranking before Stage 3 carves and repairs the selected architecture.

Output: dense subnetwork and optional fine-tuning. After search, pruning and recovery, the resulting carved model satisfies $P(\mathbf{a}^{\star}) \leq B$ (Figure 2, output). The primary protocol freezes this carved model and trains a linear head, while the full-fine-tuning variant optimizes it end-to-end. Both yield an ordinary dense transformer with task-specific widths.

4 Experiments

Experimental Setup

Benchmarks and source models. We evaluate four source models: DINOv3 ViT-S (21.6M), ViT-B (85.7M), and ViT-L (303.1M) (Siméoni et al., 2025), and the image–text-pretrained MetaCLIP-2 ViT-B (85.8M) (Chuang et al., 2025). The benchmark spans twelve tasks: CIFAR-10, CIFAR-100, and Caltech-256 for general recognition; FGVC-Aircraft, CUB-200, Stanford Cars, and Oxford-IIIT Pets for fine-grained recognition; DTD and MIT Indoor-67 for texture/scene recognition; and BloodMNIST, PneumoniaMNIST, and OrganAMNIST for medical recognition (Krizhevsky, 2009; Griffin et al., 2007; Maji et al., 2013; Wah et al., 2011; Krause et al., 2013; Parkhi et al., 2012; Cimpoi et al., 2014; Quattoni & Torralba, 2009; Yang et al., 2023). We use standard train/test partitions, except for Caltech-256, where each seed defines an 80/20 split, and report accuracy.

Protocol and comparisons. For each task, 512 training samples form the calibration set; the test set is never used for architecture selection. After carving and repair, the backbone is frozen, and a fresh linear classifier is trained on the full training set for ten epochs. Results are mean and standard deviation over three seeds. The historical comparison uses common source checkpoints, data splits, calibration sets, and parameter budgets, with Random, BlockDrop (Wu et al., 2018), SNIP-Magnitude (Lee et al., 2019), SnapViT (Simoncini et al., 2025), FLAP (An et al., 2024), SVD-LLM (Wang et al., 2025), and CORP (Zhang & Yang, 2026). Table 1 additionally includes SliceGPT (Ashkboos et al., 2024) and available OSSCAR (Meng et al., 2024) results at the mild and tight budgets. Additional scoring baselines and intermediate-budget results are provided in the supplement, together with search, recovery, optimization, and hardware details.

Table 1: Linear probing accuracy (%) at mild and tight compression ratios, mean over three seeds. Shading (low mid high) is normalized within each block; gray: full model or unavailable; **bold**: best per source and budget (ties included). Standard deviations, additional budgets, and method details are reported in the appendix.

Model	Size	Method	General			Fine-grained				Texture/scene		Medical			Mean
			C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	Org.A	
DINOv3 ViT-S/16	21.6M	Full	96.0	84.3	92.6	70.3	83.6	88.8	94.0	77.3	87.4	95.0	84.8	92.1	87.2
	21.6M ↓ 14M	BlockDrop	65.6	40.6	44.3	14.5	12.1	11.1	21.2	42.7	40.2	91.6	81.6	89.3	46.2
		SNIP–Mag.	46.4	14.9	9.8	4.1	2.2	1.8	5.5	15.5	8.6	85.1	81.6	82.6	29.8
		SnapViT	52.5	26.3	31.3	9.2	6.2	3.9	13.9	28.0	27.2	86.7	82.4	87.9	38.0
		FLAP	71.1	52.1	57.6	33.2	25.2	39.7	53.2	54.9	51.7	94.3	83.9	92.7	59.1
		SVD-LLM	92.5	75.0	76.4	51.2	61.6	63.0	79.4	65.1	74.7	95.7	88.9	92.7	76.3
		CORP	94.7	**81.1**	85.5	**70.9**	72.9	83.0	90.3	57.9	81.9	**96.8**	88.0	**93.9**	83.1
		SliceGPT	92.4	75.7	41.0	7.9	7.5	5.4	26.5	59.5	51.1	95.3	88.3	92.7	53.6
		OSSCAR	91.3	73.5	75.1	57.5	59.7	75.6	81.6	69.8	69.1	96.1	88.9	93.0	77.6
		Carve VFM	**94.8**	80.5	**86.5**	70.1	**79.1**	**88.7**	**91.3**	**75.2**	**83.9**	96.7	**89.2**	93.6	**85.8**
	21.6M ↓ 6M	BlockDrop	51.9	25.7	19.8	7.5	5.4	3.8	12.3	19.1	19.3	69.9	**85.4**	85.2	33.8
		SNIP–Mag.	42.0	14.2	9.1	2.4	1.3	1.0	4.4	15.1	10.2	76.1	81.4	74.2	27.6
		SnapViT	44.0	18.6	16.5	4.5	3.8	2.2	7.1	17.7	15.8	72.4	82.3	80.4	30.4
		FLAP	53.8	26.7	16.9	4.8	4.6	2.2	7.7	23.2	15.3	73.8	80.0	83.1	32.7
		SVD-LLM	75.1	48.0	19.3	4.4	4.6	2.5	10.7	28.0	21.1	90.1	83.5	89.4	39.7
		CORP	57.1	29.7	20.4	5.8	5.8	2.7	10.1	16.0	14.6	79.7	82.1	86.5	34.2
		SliceGPT	**78.4**	**52.0**	26.4	7.6	4.3	3.0	19.5	25.2	18.7	80.4	80.7	90.1	40.5
		OSSCAR	57.7	31.4	27.7	12.4	10.9	5.4	18.5	41.6	24.5	89.7	82.2	86.2	40.7
		Carve VFM	77.1	51.3	**38.4**	**21.4**	**18.5**	**10.2**	**31.1**	**43.8**	**34.8**	**93.2**	83.9	**90.4**	**49.5**
DINOv3 ViT-B/16	85.7M	Full	98.2	88.9	95.9	81.8	89.2	94.3	95.9	80.0	89.7	95.7	83.8	93.7	90.6
	85.7M ↓ 60M	BlockDrop	73.7	48.5	55.1	26.8	12.6	15.9	23.3	55.4	54.7	94.8	84.0	89.8	52.9
		SNIP–Mag.	74.3	46.3	47.9	22.9	14.5	10.3	22.6	43.8	44.5	94.4	88.1	92.9	50.2
		SnapViT	75.7	52.1	60.1	20.8	13.5	15.7	33.5	45.5	45.2	95.2	87.0	92.0	53.0
		FLAP	83.9	65.5	71.4	66.5	59.0	82.0	74.4	71.4	74.7	96.8	84.6	94.0	77.0
		SVD-LLM	97.0	84.9	93.2	80.5	86.5	93.7	95.3	79.1	88.5	97.1	87.2	93.7	89.7
		CORP	96.4	85.3	93.7	81.7	80.7	88.2	94.6	59.0	89.0	**97.3**	87.0	94.2	87.3
		SliceGPT	96.6	83.3	89.6	78.1	83.8	91.8	93.4	76.6	87.2	96.0	87.3	93.5	88.1
		OSSCAR	97.2	85.3	91.6	76.8	84.1	91.5	94.8	77.7	87.2	96.3	86.5	94.1	88.6
		Carve VFM	**97.3**	**86.2**	**93.8**	**82.9**	**87.9**	**94.0**	**95.7**	**81.3**	**89.9**	**97.3**	**89.2**	**94.5**	**90.8**
	85.7M ↓ 20M	BlockDrop	32.7	10.0	17.9	5.2	2.4	2.8	5.3	11.9	13.5	70.8	80.6	66.8	26.7
		SNIP–Mag.	48.1	18.8	16.0	4.6	4.6	2.2	7.3	17.9	16.6	76.6	83.0	78.0	31.1
		SnapViT	46.1	16.5	15.2	4.0	3.2	1.8	6.0	15.1	13.6	75.7	79.4	73.8	29.2
		FLAP	57.3	29.8	18.6	5.4	3.8	3.1	7.5	18.2	17.7	79.9	81.9	88.8	34.3
		SVD-LLM	84.3	59.9	42.7	17.9	13.3	12.1	26.6	44.4	38.7	**94.1**	85.0	92.6	51.0
		CORP	46.0	22.1	24.3	6.9	5.1	3.4	9.5	21.2	20.3	86.2	80.6	83.9	34.1
		SliceGPT	**84.8**	**60.2**	27.3	6.1	4.2	3.4	12.9	22.9	19.3	92.3	**88.2**	92.2	42.8
		OSSCAR	66.3	32.8	37.0	21.6	12.5	6.4	26.8	46.2	34.0	90.1	85.6	87.3	45.5
		Carve VFM	82.6	59.0	**45.0**	**32.4**	**17.3**	**12.7**	**39.0**	**50.1**	**46.2**	93.1	81.2	**92.8**	**54.3**
MetaCLIP-2 ViT-B/16	85.8M	Full	97.5	86.4	96.0	69.5	81.1	91.8	92.0	81.0	92.6	94.7	83.9	88.1	87.9
	85.8M ↓ 60M	BlockDrop	81.0	56.3	45.8	4.7	7.4	6.4	17.1	33.5	39.8	89.0	82.9	87.7	46.0
		SNIP–Mag.	63.7	36.2	29.1	8.9	5.5	4.3	12.9	18.8	18.9	73.9	83.7	86.7	36.9
		SnapViT	55.1	28.8	28.7	8.1	6.4	4.6	12.0	22.4	25.0	83.5	82.3	87.4	37.0
		FLAP	84.0	59.7	43.3	24.6	27.4	42.7	52.5	52.7	49.3	92.4	86.7	88.2	58.6
		SVD-LLM	94.8	78.8	82.2	52.0	65.3	77.4	85.3	73.2	83.8	95.0	**88.2**	**91.1**	80.6
		CORP	96.8	83.2	92.2	**65.0**	71.7	84.9	88.7	60.1	89.5	**95.4**	85.4	90.0	83.6
		SliceGPT	96.8	82.0	91.1	59.9	74.7	88.9	90.3	77.4	88.7	93.8	83.8	88.1	84.6
		OSSCAR	96.5	82.0	90.9	62.4	74.0	87.3	90.4	76.2	87.7	93.6	87.7	89.2	84.8
		Carve VFM	**97.2**	**84.1**	**93.2**	64.8	**77.7**	**89.2**	**91.7**	**78.2**	**90.1**	**95.4**	88.1	89.6	**86.6**
	85.8M ↓ 20M	BlockDrop	54.0	26.0	20.8	2.9	1.7	2.5	6.8	12.2	18.1	78.4	79.1	85.5	32.3
		SNIP–Mag.	56.2	26.1	21.3	6.2	4.7	3.8	9.7	20.5	20.7	83.7	84.5	86.2	35.3
		SnapViT	51.4	25.6	22.3	5.5	5.2	3.8	9.6	19.7	20.7	87.2	84.1	87.0	35.2
		FLAP	63.2	35.7	25.6	7.1	4.9	4.7	13.1	28.9	25.1	89.9	**86.6**	90.1	39.6
		SVD-LLM	77.7	50.9	31.9	9.5	6.6	4.3	12.7	33.1	28.6	90.1	85.4	90.7	43.5
		CORP	62.2	34.1	25.4	6.3	5.4	3.0	10.5	20.2	21.9	90.2	84.5	89.9	37.8
		SliceGPT	**85.8**	**56.0**	**43.7**	**19.4**	**22.8**	**27.4**	**53.1**	**43.7**	35.2	88.2	82.1	84.0	**53.4**
		OSSCAR	66.9	38.5	28.2	10.7	5.3	5.2	20.3	40.1	29.2	83.8	81.0	86.6	41.3
		Carve VFM	76.6	42.5	36.6	16.5	15.9	7.8	34.5	42.3	**43.9**	**91.1**	85.1	**90.8**	48.6
DINOv3 ViT-L/16	303.1M	Full	99.1	93.1	97.3	80.4	90.6	94.4	96.7	82.7	90.7	95.9	87.3	92.3	91.7
	303.1M ↓ 200M	BlockDrop	77.8	56.4	76.5	41.8	46.7	58.6	65.9	64.8	70.1	95.6	86.1	90.8	69.3
		SNIP–Mag.	69.6	44.5	50.3	25.8	15.3	17.2	26.7	40.4	47.9	94.6	86.4	92.1	50.9
		SnapViT	72.2	46.5	61.8	26.8	15.9	18.6	28.0	45.8	55.5	93.1	85.2	92.9	53.5
		FLAP	93.4	77.5	88.7	73.7	81.2	90.7	92.6	78.5	88.7	95.7	86.2	91.6	86.6
		SVD-LLM	**98.9**	90.7	96.1	80.6	89.7	**94.1**	**96.5**	83.2	**91.0**	96.2	**89.9**	93.1	**91.7**
		CORP	**98.9**	89.5	**96.5**	79.8	83.1	88.7	95.8	63.8	90.6	**97.0**	88.2	93.0	88.8
		SliceGPT	98.7	89.2	94.0	78.5	87.4	93.2	95.5	80.3	89.5	94.4	89.2	92.1	90.2
		OSSCAR	97.8	86.6	92.1	71.0	74.2	82.0	92.4	75.0	89.0	95.3	86.6	92.2	86.2
		Carve VFM	**98.9**	**90.9**	96.3	**81.6**	**89.8**	94.0	**96.5**	**83.3**	90.8	96.5	87.9	**93.6**	**91.7**
	303.1M ↓ 100M	BlockDrop	54.6	32.8	25.6	8.4	7.6	5.4	14.6	25.1	15.6	90.8	84.7	85.0	37.5
		SNIP–Mag.	49.2	22.7	20.5	5.6	6.4	3.5	10.0	18.4	23.1	76.5	81.0	82.2	33.3
		SnapViT	47.5	19.4	18.7	4.4	3.7	2.6	7.3	16.5	17.2	68.8	81.7	82.4	30.8
		FLAP	77.4	52.4	45.4	29.3	11.4	12.5	28.9	51.5	40.8	92.9	84.0	92.8	51.6
		SVD-LLM	**96.1**	**80.0**	78.5	**74.0**	**81.5**	**91.8**	**92.0**	69.8	**82.1**	**94.9**	**86.1**	92.0	**84.9**
		CORP	71.0	37.1	57.9	36.9	33.5	59.6	58.1	45.7	57.1	91.8	80.4	89.5	59.9
		SliceGPT	95.8	77.5	67.1	68.2	58.1	86.4	86.4	47.0	78.7	92.1	84.1	91.3	77.7
		OSSCAR	84.9	62.0	61.3	47.2	27.7	56.5	69.9	63.4	56.2	92.4	84.4	91.5	66.4
		Carve VFM	94.2	76.9	**80.1**	73.6	76.4	89.0	90.6	**74.0**	80.3	92.7	84.2	**93.1**	83.7

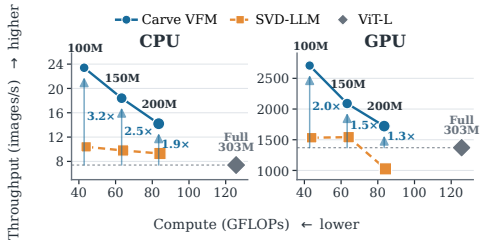


Figure 4: ViT-L efficiency of Carve VFM, uncompressed ViT-L, and SVD-LLM on CPU (AMD EPYC 9754) and GPU (RTX PRO 6000 Blackwell). Marker area denotes peak memory.

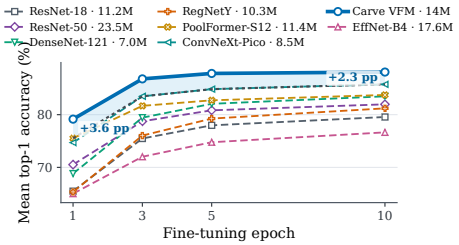


Figure 5: Average full-fine-tuning accuracy over twelve tasks. Model sizes in the legend; shading marks the gap between Carve VFM and the strongest baseline at each checkpoint.

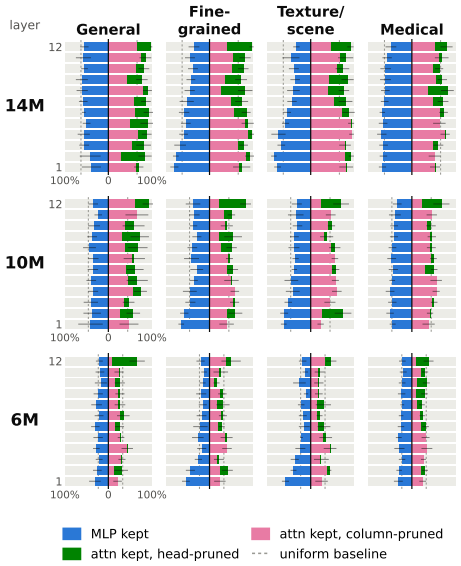


Figure 6: Architectural patterns discovered by Carve VFM across task categories. Rows are encoder layers; bars show retained MLP and attention, separated into head and column pruning. Dashed lines are uniform allocation baseline, and whiskers show variation across tasks and seeds.

Recoverability becomes critical under aggressive compression. Among dense structured-pruning methods with complete results in Table 1, Carve VFM has the highest reported twelve-task mean in seven of eight source–budget settings. The exception is MetaCLIP-2 at 20M parameters, where SliceGPT reaches 53.4% versus 48.6% for Carve VFM. Checkpoint-selection protocols for SliceGPT and OSSCAR differ from those of the other baselines, so these rankings are descriptive. The supplement reports the complete twelve-setting grid for the other baselines. As budgets tighten, useful structures may also be removed; performance then depends on preserving or reconstructing their task-relevant contribution. These comparisons motivate considering allocation and recoverability jointly, but do not isolate their individual contributions.

The benefit transfers across representations and domains. The advantage is consistent across both self-supervised DINOv3 and image–text-pretrained MetaCLIP-2, and from ViT-S through ViT-L, suggesting that recoverability is largely independent of the pretrained representation. The gains are largest on fine-grained and texture-rich recognition tasks, where small semantic differences require preserving highly task-specific capacity. The gains are smaller and more variable on the medical benchmarks, where the source models transfer less consistently.

Carve VFM vs. factorization. SVD-LLM offers a complementary approach through low-rank factorization. Across the twelve source–budget settings in the supplement, Carve VFM has the higher reported twelve-task mean in nine, SVD-LLM in two, and the two tie in one at the reported one-decimal precision. SVD-LLM leads for DINOv3 ViT-B at 35M and ViT-L at 100M; the tie occurs for ViT-L at 200M. Carve VFM returns conventional dense projections, whereas SVD-LLM executes low-rank factors, motivating the deployment comparison below.

Deployment efficiency. Figure 4 compares *Carve VFM* with the original ViT-L and the parameter-matched low-rank baseline SVD-LLM. Other structured pruning methods already produce conventional dense networks with the same execution pattern as *Carve VFM* and therefore do not provide a distinct deployment comparison.

Carve VFM consistently shifts the deployment trade-off toward higher throughput, lower compute, and lower memory. Compared with the original ViT-L, it improves CPU throughput by 1.9–3.2× and GPU throughput by 1.3–2.0×, while reducing peak GPU memory by 26–51%. At matched parameter budgets, it also delivers 1.5–2.3× higher CPU throughput and 1.4–1.8× higher GPU throughput than SVD-LLM, while using 7–17% less GPU memory. This advantage stems from the compression mechanism itself: SVD-LLM replaces each projection with two serial low-rank operations, whereas *Carve VFM* physically narrows the dense operators, reducing both arithmetic and memory traffic without introducing additional layers.

Full fine-tuning and convergence. We fully fine-tune the carved 14M ViT-S specialist for each task and compare it with natively pretrained compact architectures under a ten-epoch schedule. Table 10 reports a best-checkpoint macro-average of 88.1% for the 14M specialist and 85.8% for ConvNeXt-Pico, the strongest evaluated native compact model, a difference of 2.3 percentage points. These models have different parameter counts: 14M and 8.5M, respectively. Figure 5 compares the reported learning curves. Full optimization settings, per-dataset results, and comparisons with all eight compact baselines are provided in the supplement.

Architecture patterns discovered by Carve VFM. Figure 6 shows that Carve VFM consistently discovers task-dependent subnetworks rather than uniformly distributing the capacity. Texture and scene recognition retain more early-layer MLP capacity, with fine-grained tasks exhibiting the same tendency more moderately, indicating that these domains benefit from preserving richer low-level features. In contrast, medical specialists remain much closer to a uniform allocation, suggesting that task-relevant information is distributed more evenly across depth. A second pattern is remarkably consistent across domains: most parameter savings come from reducing MLP capacity, while attention is compressed more conservatively. In particular, whole-head removal is rare in the early layers, where Carve VFM instead favors column pruning, distributing the reduction across all heads rather than eliminating complete ones. These results reveal that Carve VFM discovers distinct compression regimes across depth.

When is global fitness needed? The adaptive audit provides a complementary diagnostic. For DINOv3 ViT-S/B, the search switched from local recoverability to cross-validated task accuracy in 26% of mild-budget runs and 51% of tight-budget runs; this rose to 79% for MetaCLIP-2 and 89% for ViT-L. Cached local statistics therefore guide exploration efficiently when their ranking is stable, while global task evidence becomes more important as compression, representation shift, or model scale makes that ranking unreliable.

Decoupling search from repair. To determine whether Carve's search improves performance beyond repair alone, we compare uniform allocations and FLAP subnetworks with and without the same repair against full Carve VFM. We evaluate DINOv3-S at 14M and 6M parameters on CIFAR-100, CUB-200, and BloodMNIST. At these budgets, repair raises mean accuracy from 31.7/31.5% to 78.2/47.7% for uniform allocation and from 28.2/26.6% to 63.5/36.7% for FLAP subnetworks. Across these datasets, the corresponding Carve entries in Table 1 average 85.4/54.3%, 7.2/6.6 percentage points above repaired uniform allocation. Thus, repair provides most of the gain, while searching allocations and attention operators adds further improvements on average.

5 Conclusion

We introduced *Carve VFM*, a framework for extracting compact, task-specific models from pretrained VFMs under an explicit parameter budget, without backbone retraining. A single search jointly decides how capacity is allocated across layers, which structures to remove, and how to recover their lost contribution, producing standard dense models competitive with evaluated compression methods and compact architectures, with gains varying by source, task, and budget. Our evaluation centers on image classification. Natural next steps include dense prediction and multimodal foundation models, dynamic deployment budgets that adapt to shifting device constraints, and continual carving that re-allocates capacity as new tasks arrive. Relaxing the requirement for labeled calibration data through self-supervised surrogates would further broaden applicability.

AI USE STATEMENT

Generative AI tools were used only to assist with manuscript drafting and grammatical and language editing. They were not used to develop the method, generate data, implement the research methods, execute experiments, or analyze experimental results. The authors take responsibility for the final manuscript and all reported claims and results.

ETHICS STATEMENT

Carve VFM is evaluated on existing image-classification benchmarks, including three medical-image benchmarks. The experiments assess classification accuracy and computational efficiency. These benchmark results do not establish clinical safety or effectiveness; compressed models would require application-specific validation before clinical use. The reported evaluation does not measure demographic fairness or privacy, so accuracy and efficiency gains should not be interpreted as evidence of improvements in those properties.

REPRODUCIBILITY STATEMENT

Section 3 describes the architecture search, recoverability scoring, and closed-form repair procedure. Appendix A reports the source checkpoints, datasets, data splits, calibration sampling, and shared evaluation protocol. Appendix C provides the search and repair settings, random seeds, software versions, and computing hardware. Appendix B reports per-dataset results with standard deviations across three seeds. The full-fine-tuning and deployment measurement protocols are provided in Appendices E and F, respectively.

REFERENCES

Vincent Aduvuku, Geoffrey Andogah, and Taban Habibu. Bridging the deployment gap: A decision oriented survey of efficient on-device foundation models for vision. *Journal of Future Artificial Intelligence and Technologies*, 3(1):99–131, 2026.

Setareh Ahsaei and Mohsen Raji. Pruning strategies of vision transformers: A specialized survey. *IEEE Open Journal of the Computer Society*, 7:1083–1105, 2026. doi: 10.1109/OJCS.2026.3692927.

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. FLAP: Fluctuation-based adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 10865–10873, 2024. doi: 10.1609/aaai.v38i10.28960.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vXxardq6db.

Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10925–10934, 2022.

Yung-Sung Chuang, Yang Li, Dong Wang, Ching-Feng Yeh, Kehan Lyu, Ramya Raghavendra, James Glass, Lifei Huang, Jason Weston, Luke Zettlemoyer, Xinlei Chen, Zhuang Liu, Saining Xie, Wen-tau Yih, Shang-Wen Li, and Hu Xu. Meta clip 2: A worldwide scaling recipe. In *Advances in Neural Information Processing Systems*, 2025.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3606–3613, 2014.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488, 2022.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical Report CNS-TR-2007-001, California Institute of Technology, 2007.

Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. AMC: AutoML for model compression and acceleration on mobile devices. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 784–800, 2018.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

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

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Jaeho Lee, Sejun Park, Sangwoo Mo, Sungsoo Ahn, and Jinwoo Shin. Layer-adaptive sparsity for the magnitude-based pruning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=H6ATjJ0TKdf.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1VZqjAcYX.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rJqFGTslg.

Bingyan Liu, Yifeng Cai, Yao Guo, and Xiangqun Chen. Transtailor: Pruning the pre-trained model for improved transfer learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8627–8634, 2021. doi: 10.1609/aaai.v35i10.17046.

Jing Liu, Bohan Zhuang, Zhuangwei Zhuang, Yong Guo, Junzhou Huang, Jinhui Zhu, and Mingkui Tan. Discrimination-aware network pruning for deep model compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(8):4035–4051, 2022. doi: 10.1109/TPAMI.2021.3066410.

Shih-Yang Liu, Huck Yang, Chien-Yi Wang, Nai Chit Fung, Hongxu Yin, Charbel Sakr, Saurav Muralidharan, Kwang-Ting Cheng, Jan Kautz, Yu-Chiang Frank Wang, Pavlo Molchanov, and Min-Hung Chen. EoRA: Training-free compensation for compressed LLM with eigenspace low-rank approximation. *arXiv preprint arXiv:2410.21271*, 2024.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft, 2013. URL https://arxiv.org/abs/1306.5151.

Xiang Meng, Shibal Ibrahim, Kayhan Behdin, Hussein Hazimeh, Natalia Ponomareva, and Rahul Mazumder. OSSCAR: One-shot structured pruning in vision and language models with combinatorial optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35354–35377. PMLR, 2024. URL https://proceedings.mlr.press/v235/meng24a.html.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11264–11272, 2019.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3498–3505, 2012.

David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R. So, Maud Texier, and Jeff Dean. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 55(7):18–28, 2022. doi: 10.1109/MC.2022.3148714.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 413–420, 2009.

Dachuan Shi, Chaofan Tao, Ying Jin, Zhendong Yang, Chun Yuan, and Jiaqi Wang. UPop: Unified and progressive pruning for compressing vision-language transformers. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31292–31311. PMLR, 2023.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Walter Simoncini, Michael Dorkenwald, Tijmen Blankevoort, Cees G. M. Snoek, and Yuki M. Asano. Elastic ViTs from pretrained models without retraining. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://openreview.net/forum?id=OU6FXkSIe0.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PxoFut3dWW.

Raviteja Vemulapalli, Hadi Pouransari, Fartash Faghri, Sachin Mehta, Mehrdad Farajtabar, Mohammad Rastegari, and Oncel Tuzel. Knowledge transfer from vision foundation models for efficient training of small task-specific models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 49345–49367. PMLR, 2024. URL https://proceedings.mlr.press/v235/vemulapalli24a.html.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Xin Wang, Samiul Alam, Zhongwei Wan, Hui Shen, and Mi Zhang. SVD-LLM v2: Optimizing singular value truncation for large language model compression. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4287–4296, 2025.

Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. BlockDrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8817–8826, 2018.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bin Ke, Hanspeter Pfister, Bing Ni, and Bilian Liu. MedMNIST v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(41), 2023.

Tien-Ju Yang, Andrew Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze, and Hartwig Adam. NetAdapt: Platform-aware neural network adaptation for mobile applications. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 285–300, 2018.

Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. PTQ4ViT: Post-training quantization for vision transformers with twin uniform quantization. In *European conference on computer vision*, pp. 191–207. Springer, 2022.

Boxiang Zhang and Baijian Yang. CORP: Closed-form one-shot representation-preserving structured pruning for vision transformers. *arXiv preprint arXiv:2602.05243v1*, 2026. URL https://arxiv.org/abs/2602.05243v1.

Chuanyang Zheng, Zheyang Li, Kai Zhang, Zhi Yang, Wenming Tan, Jun Xiao, Ye Ren, and Shiliang Pu. SAViT: Structure-aware vision transformer pruning via collaborative optimization. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Chen Zhu, Wangbo Zhao, Huiwen Zhang, Yuhao Zhou, Weidong Tang, Shuo Wang, Zhihang Yuan, Yuzhang Shang, Xiaojiang Peng, Kai Wang, et al. EA-ViT: Efficient adaptation for elastic vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1038–1047, 2025.

APPENDICES

These appendices report results and implementation details omitted from the main paper for space: complete three-budget, per-dataset comparisons for every source model; the full fine-tuning comparison; documented search and repair settings; measured wall-clock and deployment costs; and limitations and future work. Unless stated otherwise, accuracy is reported as mean±sample standard deviation over seeds 17, 42, and 91.

A EVALUATION PROTOCOL

A.1 SOURCE MODELS, TASKS, AND BUDGETS

We evaluate DINOv3 ViT-S/16 (21.6M parameters), ViT-B/16 (85.7M), and ViT-L/16 (303.1M), and MetaCLIP-2 Worldwide ViT-B/16 (85.8M). The DINOv3 models use the LVD-1689M pretrained checkpoints; MetaCLIP-2 uses its released Worldwide B/16 checkpoint. Table 2 lists the three dense parameter budgets evaluated for each source.

Table 2: Source models, dense parameter budgets, and final repair rank.

Source	Budgets (M)			Repair rank
	Mild	Intermediate	Tight	r
DINOv3 ViT-S/16	14	10	6	128
DINOv3 ViT-B/16	60	35	20	128
MetaCLIP-2 ViT-B/16	60	35	20	256
DINOv3 ViT-L/16	200	150	100	256

The twelve tasks are CIFAR-10, CIFAR-100, and Caltech-256 (general recognition); FGVC-Aircraft, CUB-200, Stanford Cars, and Oxford-IIIT Pets (fine-grained recognition); DTD and MIT Indoor-67 (texture and scene recognition); and BloodMNIST, PneumoniaMNIST, and OrganAMNIST (medical recognition). We use the standard train/test split for every dataset except Caltech-256, for which each seed defines an 80/20 split. All inputs are resized to 224×224 using the preprocessing associated with the source checkpoint.

A.2 CALIBRATION AND MODEL SELECTION

For each task and seed, 512 class-balanced examples are drawn from the training split to form the compression calibration set. All matched methods share the same examples and data split. Carve VFM further partitions this set internally when held-out task accuracy is required; no test image is used to guide search or choose an architecture.

After compression, the backbone is frozen and a new linear classifier is trained on the full training split for ten epochs. AdamW uses batch size 32, learning rate 10^{-3}, and weight decay 10^{-4}. We evaluate epochs 1, 3, 5, and 10. SliceGPT and OSSCAR use a different checkpoint-selection protocol from the protocol-matched baseline grid; their downstream training budget remains ten epochs, and their per-dataset results are reported separately in Table 7.

A.3 BASELINES

The common comparison contains seven structured methods. Random removes budget-matched structures uniformly at random. BlockDrop (Wu et al., 2018) removes complete transformer blocks. SNIP–Magnitude (Lee et al., 2019) ranks structures using sensitivity at initialization followed by magnitude aggregation. SnapViT (Simoncini et al., 2025) applies its structured ViT scoring rule. FLAP (An et al., 2024) uses activation fluctuation to allocate structured removals. SVD-LLM (Wang et al., 2025) replaces projections with budget-matched low-rank factors. CORP (Zhang & Yang, 2026) uses compensated one-shot structured pruning. All methods start from the same source checkpoint and use identical calibration data, seeds, dense parameter budgets, and frozen-backbone linear-probing protocol.

Where a complete protocol-matched grid exists, the source-specific tables also include scoring baselines omitted from the main paper: Magnitude (Li et al., 2017), LAMP (Lee et al., 2021), Wanda (Sun et al., 2024), and Taylor (Molchanov et al., 2019) for DINOv3 ViT-S/B, and Magnitude and LAMP for MetaCLIP-2 ViT-B. We omit UPop and EA-ViT because no completed three-seed, twelve-task grid is available under the current checkpoint, budget, and linear-probing protocol. This rule avoids mixing results obtained under different training regimes with the controlled comparison.

B Complete Per-Dataset Results

The main paper displays only the mild and tight budgets and a compact baseline set. Tables 3–6 restore the intermediate budget and every completed protocol-matched baseline. "Mean" gives equal weight to the twelve dataset means rather than to their numbers of examples. Values are mean$\pm$sample standard deviation over seeds 17, 42, and 91. Accuracies and standard deviations are displayed to one decimal, while macro means are computed from the unrounded per-task values. Each source table includes only methods with a complete protocol-matched grid; bold marks the best result within each budget.

Table 7 reports the separate per-dataset results for SliceGPT and OSSCAR, including the standard deviations omitted from the main table.

Table 3: Complete per-dataset frozen-backbone linear-probing accuracy (%) for DINOv3 ViT-S/16 at three dense parameter budgets.

Budget	Method	General			Fine-grained				Texture/scene		Medical			**Mean**
		C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	OrganA	
21.6M	Linear probe	96.0 ±0.1	84.3 ±0.2	92.6 ±0.2	70.3 ±0.1	83.6 ±0.1	88.8 ±0.3	94.0 ±0.4	77.3 ±0.1	87.4 ±0.2	95.0 ±0.1	84.8 ±0.2	92.1 ±0.1	87.2 ±0.0
14M	Random	57.6 ±1.9	30.9 ±1.9	29.1 ±4.3	7.7 ±1.2	6.0 ±1.1	3.9 ±0.9	13.1 ±2.4	25.1 ±3.1	26.0 ±2.6	84.8 ±1.3	83.4 ±1.5	87.9 ±1.5	38.0 ±2.0
	Magnitude	42.6 ±0.1	17.3 ±0.6	14.4 ±1.0	3.9 ±0.2	2.5 ±0.1	1.9 ±0.1	7.3 ±0.3	13.3 ±0.3	12.2 ±0.5	79.3 ±0.2	82.5 ±0.8	78.2 ±0.2	29.6 ±0.4
	LAMP	44.4 ±0.2	19.4 ±0.2	19.7 ±1.0	5.5 ±0.0	4.1 ±0.2	2.8 ±0.1	9.6 ±0.4	17.2 ±0.1	16.5 ±0.3	85.5 ±0.2	82.3 ±1.3	83.5 ±0.2	32.5 ±0.4
	SNIP–Mag.	46.4 ±0.8	14.9 ±0.5	9.8 ±0.9	4.1 ±0.7	2.2 ±0.5	1.8 ±0.3	5.5 ±0.8	15.5 ±1.7	8.6 ±1.5	85.1 ±0.7	81.6 ±0.3	82.6 ±1.9	29.8 ±0.9
	Wanda	70.6 ±0.6	47.1 ±1.9	52.2 ±2.4	19.1 ±0.8	16.6 ±0.2	11.1 ±1.1	31.0 ±1.1	43.7 ±0.1	42.4 ±0.9	92.4 ±0.1	85.1 ±1.7	93.3 ±0.1	50.4 ±0.9
	Taylor	59.5 ±13.2	39.3 ±0.2	40.1 ±2.1	7.6 ±7.1	5.3 ±4.5	5.6 ±4.4	18.9 ±4.5	32.2 ±6.6	16.9 ±8.9	79.6 ±16.8	83.7 ±2.6	88.9 ±3.4	39.8 ±6.2
	SnapViT	52.5 ±0.3	26.3 ±0.4	31.3 ±1.1	9.2 ±0.2	6.2 ±0.1	3.9 ±0.2	13.9 ±0.4	28.0 ±0.5	27.2 ±0.6	86.7 ±0.4	82.4 ±0.5	87.9 ±0.1	38.0 ±0.4
	BlockDrop	65.6 ±0.1	40.6 ±0.5	44.3 ±1.1	14.5 ±0.3	12.1 ±2.7	11.1 ±0.2	21.2 ±0.5	42.7 ±0.3	40.2 ±3.1	91.6 ±0.1	81.6 ±0.9	89.3 ±0.2	46.2 ±0.3
	FLAP	71.1 ±0.1	52.1 ±1.2	57.6 ±0.6	33.2 ±3.2	25.2 ±0.3	39.7 ±1.2	53.2 ±2.3	54.9 ±0.8	51.7 ±1.7	94.3 ±0.2	83.9 ±1.8	92.7 ±0.2	59.1 ±0.1
	SVD-LLM	92.5 ±0.1	75.0 ±0.2	76.4 ±1.1	51.2 ±0.9	61.6 ±1.0	63.0 ±4.3	79.4 ±0.8	65.1 ±0.8	74.7 ±1.6	95.7 ±0.1	88.9 ±1.7	92.7 ±0.2	76.3 ±0.4
	CORP	94.7 ±0.1	**81.1 ±0.2**	85.5 ±0.4	**70.9 ±0.3**	72.9 ±0.5	83.0 ±0.5	90.3 ±0.0	57.9 ±0.2	81.9 ±0.3	**96.8 ±0.1**	88.0 ±1.0	**93.9 ±0.2**	83.1 ±0.0
	Carve VFM	**94.8 ±0.5**	80.5 ±0.5	**86.5 ±0.9**	70.1 ±0.4	**79.1 ±0.9**	**88.7 ±0.5**	**91.3 ±0.3**	**75.2 ±0.3**	**83.9 ±0.4**	96.7 ±0.2	**89.2 ±1.6**	93.6 ±0.0	**85.8 ±0.1**
10M	Random	41.6 ±1.3	17.0 ±1.8	13.7 ±2.9	3.7 ±0.6	2.8 ±0.5	1.8 ±0.5	6.8 ±1.2	14.0 ±1.1	11.0 ±2.9	71.2 ±10.4	78.9 ±2.6	77.8 ±1.4	28.4 ±2.3
	Magnitude	38.9 ±0.4	14.3 ±0.5	12.7 ±0.8	3.7 ±0.3	2.8 ±0.1	1.7 ±0.1	6.4 ±0.3	12.8 ±0.8	11.5 ±0.8	75.2 ±0.2	82.7 ±0.6	70.1 ±0.6	27.7 ±0.5
	LAMP	41.8 ±0.2	16.5 ±0.5	17.4 ±1.0	4.6 ±0.2	3.3 ±0.2	2.5 ±0.3	8.2 ±0.1	16.1 ±0.5	14.3 ±0.4	81.9 ±0.3	83.2 ±0.4	79.9 ±0.6	30.8 ±0.4
	SNIP–Mag.	43.5 ±1.3	14.8 ±1.9	11.6 ±0.4	3.7 ±0.9	1.7 ±0.2	1.1 ±0.4	5.2 ±0.4	14.4 ±0.2	10.0 ±2.5	80.6 ±1.0	81.5 ±1.5	80.3 ±1.4	29.0 ±1.0
	Wanda	62.2 ±0.2	35.1 ±0.9	30.5 ±0.8	10.3 ±0.2	7.8 ±0.4	5.0 ±0.1	14.7 ±0.5	26.2 ±0.3	29.6 ±0.7	86.5 ±0.5	83.1 ±0.9	87.5 ±0.4	39.9 ±0.5
	Taylor	44.3 ±11.3	25.5 ±3.6	22.5 ±7.4	5.1 ±1.8	3.6 ±0.5	2.7 ±0.3	9.7 ±3.4	20.4 ±7.7	14.9 ±7.2	77.6 ±2.2	79.3 ±5.9	83.4 ±8.4	32.4 ±5.0
	SnapViT	42.1 ±0.3	16.5 ±0.4	21.4 ±0.7	6.3 ±0.4	4.3 ±0.1	2.9 ±0.1	10.2 ±0.4	22.4 ±0.3	19.9 ±0.5	77.2 ±0.5	81.5 ±0.8	82.6 ±0.2	32.3 ±0.4
	BlockDrop	57.2 ±0.1	29.5 ±2.1	29.9 ±0.5	10.5 ±0.2	8.2 ±0.2	5.8 ±0.1	17.1 ±0.3	25.6 ±0.4	26.5 ±0.6	87.0 ±0.2	78.3 ±1.0	87.1 ±0.1	38.6 ±0.1
	FLAP	65.1 ±0.3	39.4 ±0.1	32.5 ±0.8	15.4 ±0.2	9.7 ±0.2	6.5 ±0.1	25.8 ±0.7	40.5 ±0.9	31.7 ±0.8	89.7 ±0.4	80.4 ±0.6	91.7 ±0.2	44.0 ±0.1
	SVD-LLM	87.7 ±0.0	66.2 ±0.2	32.0 ±0.5	17.9 ±0.6	7.9 ±0.3	4.6 ±0.3	41.6 ±3.4	46.2 ±1.2	36.3 ±0.5	95.0 ±0.3	86.8 ±2.4	93.2 ±0.1	51.3 ±0.4
	CORP	87.1 ±0.8	66.4 ±0.7	54.7 ±0.9	48.1 ±0.4	37.5 ±5.5	33.1 ±1.6	51.4 ±3.1	41.9 ±0.3	59.3 ±0.5	94.5 ±0.3	85.1 ±1.1	92.8 ±0.4	62.6 ±0.4
	Carve VFM	**91.1 ±1.0**	**73.6 ±1.4**	**70.7 ±1.0**	**52.5 ±2.7**	**63.3 ±1.9**	**74.6 ±0.9**	**80.3 ±0.7**	**67.0 ±1.8**	**70.4 ±3.9**	**96.2 ±0.2**	**87.4 ±1.1**	**93.5 ±0.5**	**76.7 ±0.8**
6M	Random	38.6 ±1.8	14.0 ±0.9	12.3 ±0.9	3.0 ±0.2	2.4 ±0.1	1.7 ±0.1	6.4 ±0.3	11.9 ±0.8	10.0 ±0.9	66.3 ±8.7	78.0 ±2.0	74.1 ±3.2	26.6 ±1.7
	Magnitude	40.3 ±0.3	14.5 ±0.3	14.0 ±0.1	4.3 ±0.2	2.7 ±0.1	2.0 ±0.1	6.7 ±0.5	14.3 ±0.0	12.4 ±1.0	72.2 ±0.5	79.8 ±1.1	75.9 ±0.8	28.3 ±0.4
	LAMP	41.9 ±0.4	17.5 ±0.3	17.1 ±0.9	5.2 ±0.1	3.8 ±0.1	2.3 ±0.1	7.5 ±0.3	16.0 ±0.2	14.3 ±0.7	81.5 ±0.2	81.1 ±2.0	80.5 ±0.3	30.7 ±0.5
	SNIP–Mag.	42.0 ±2.5	14.2 ±0.2	9.1 ±0.8	2.4 ±0.3	1.3 ±0.0	1.0 ±0.1	4.4 ±0.6	15.1 ±3.2	10.2 ±3.8	76.1 ±2.5	81.4 ±1.8	74.2 ±2.3	27.6 ±1.5
	Wanda	55.1 ±0.2	27.6 ±0.5	18.6 ±0.7	4.8 ±0.2	4.8 ±0.2	3.1 ±0.2	9.3 ±0.4	17.2 ±0.3	18.2 ±0.9	79.5 ±0.9	81.4 ±0.8	83.0 ±0.5	33.5 ±0.5
	Taylor	42.0 ±8.7	18.1 ±5.8	17.4 ±2.5	3.7 ±0.7	3.0 ±1.1	1.7 ±0.5	7.8 ±1.2	12.7 ±4.4	16.5 ±0.3	67.1 ±5.1	78.7 ±5.0	80.9 ±3.7	29.1 ±3.2
	SnapViT	44.0 ±0.9	18.6 ±0.4	16.5 ±0.7	4.5 ±0.4	3.8 ±0.2	2.2 ±0.1	7.1 ±0.4	17.7 ±0.1	15.8 ±0.8	72.4 ±0.4	82.3 ±1.0	80.4 ±0.3	30.4 ±0.5
	BlockDrop	51.9 ±0.4	25.7 ±0.2	19.8 ±0.5	7.5 ±0.2	5.4 ±0.2	3.8 ±0.2	12.3 ±0.1	19.1 ±0.0	19.3 ±2.5	69.9 ±0.7	**85.4 ±0.7**	85.2 ±0.2	33.8 ±0.1
	FLAP	53.8 ±0.7	26.7 ±0.2	16.9 ±0.5	4.8 ±0.8	4.6 ±0.1	2.2 ±0.2	7.7 ±1.2	23.2 ±0.5	15.3 ±0.4	73.8 ±0.9	80.0 ±0.6	83.1 ±0.7	32.7 ±0.0
	SVD-LLM	75.1 ±0.4	48.0 ±0.3	19.3 ±0.3	4.4 ±0.2	4.6 ±0.1	2.5 ±0.0	10.7 ±0.6	28.0 ±1.2	21.1 ±0.3	90.1 ±0.5	83.5 ±2.1	89.4 ±0.2	39.7 ±0.3
	CORP	57.1 ±0.5	29.7 ±0.4	20.4 ±0.6	5.8 ±0.4	5.8 ±0.3	2.7 ±0.1	10.1 ±0.2	16.0 ±0.7	14.6 ±1.2	79.7 ±1.0	82.1 ±2.8	86.5 ±0.9	34.2 ±0.2
	Carve VFM	**77.1 ±2.0**	**51.3 ±6.0**	**38.4 ±1.2**	**21.4 ±2.9**	**18.5 ±4.8**	**10.2 ±1.7**	**31.1 ±7.1**	**43.8 ±1.7**	**34.8 ±1.8**	**93.2 ±0.9**	83.9 ±0.5	**90.4 ±1.0**	**49.5 ±0.7**

Table 4: Complete per-dataset frozen-backbone linear-probing accuracy (%) for DINOv3 ViT-B/16 at three dense parameter budgets.

Budget	Method	General			Fine-grained				Texture/scene		Medical			**Mean**
		C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	OrganA	
85.7M	Linear probe	98.2 ±0.1	88.9 ±0.1	95.9 ±0.3	81.8 ±0.2	89.2 ±0.1	94.3 ±0.2	95.9 ±0.2	80.0 ±0.2	89.7 ±0.5	95.7 ±0.1	83.8 ±0.2	93.7 ±0.1	90.6 ±0.0
60M	Random	65.9 ±6.0	38.7 ±6.1	52.6 ±9.5	17.8 ±3.8	13.0 ±3.5	12.7 ±5.2	28.3 ±8.7	45.2 ±4.7	47.9 ±8.0	93.0 ±1.6	86.4 ±1.8	89.7 ±1.3	49.3 ±5.0
	Magnitude	54.4 ±0.8	26.9 ±0.4	35.6 ±1.0	9.6 ±0.3	5.6 ±0.1	5.7 ±0.2	14.4 ±0.2	35.7 ±0.4	38.3 ±0.6	89.1 ±0.9	85.8 ±0.7	87.7 ±0.2	40.7 ±0.5
	LAMP	56.1 ±0.3	28.6 ±0.3	39.1 ±0.5	12.9 ±0.1	7.4 ±0.3	7.3 ±0.2	15.3 ±0.6	37.4 ±0.2	40.4 ±1.0	85.2 ±0.8	84.3 ±1.3	88.0 ±0.3	41.8 ±0.5
	SNIP–Mag.	74.3 ±1.9	46.3 ±1.3	47.9 ±2.6	22.9 ±1.4	14.5 ±1.3	10.3 ±0.4	22.6 ±4.6	43.8 ±2.0	44.5 ±4.4	94.4 ±0.9	88.1 ±0.9	92.9 ±0.3	50.2 ±1.8
	Wanda	74.5 ±0.9	51.2 ±1.6	54.9 ±0.5	22.8 ±1.0	14.9 ±0.2	12.3 ±0.8	27.9 ±0.1	49.8 ±0.6	50.1 ±0.9	94.0 ±0.2	87.2 ±1.5	93.3 ±0.8	52.7 ±0.8
	Taylor	78.1 ±4.6	49.1 ±10.1	60.6 ±6.6	36.0 ±15.1	29.3 ±10.6	38.3 ±8.9	33.4 ±6.2	51.9 ±2.0	52.1 ±4.6	95.4 ±0.1	87.4 ±2.3	92.4 ±0.3	58.7 ±5.9
	SnapViT	75.7 ±0.4	52.1 ±0.3	60.1 ±0.8	20.8 ±0.2	13.5 ±0.3	15.7 ±0.2	33.5 ±0.1	45.5 ±0.6	45.2 ±1.3	95.2 ±0.3	87.0 ±0.7	92.0 ±0.3	53.0 ±0.4
	BlockDrop	73.7 ±0.4	48.5 ±0.5	55.1 ±0.3	26.8 ±0.4	12.6 ±0.2	15.9 ±0.3	23.3 ±0.1	55.4 ±5.1	54.7 ±0.3	94.8 ±0.3	84.0 ±1.5	89.8 ±0.2	52.9 ±0.4
	FLAP	83.9 ±0.4	65.5 ±0.0	71.4 ±0.4	66.5 ±0.7	59.0 ±2.5	82.0 ±0.7	74.4 ±0.2	71.4 ±1.0	74.7 ±1.2	96.8 ±0.4	84.6 ±3.8	94.0 ±0.3	77.0 ±0.3
	SVD-LLM	97.0 ±0.2	84.9 ±0.2	93.2 ±0.1	80.5 ±0.3	86.5 ±0.2	93.7 ±0.2	95.3 ±0.2	79.1 ±0.3	88.5 ±0.4	97.1 ±0.2	87.2 ±2.7	93.7 ±0.1	89.7 ±0.2
	CORP	96.4 ±1.2	85.3 ±0.3	93.7 ±0.3	81.7 ±0.5	80.7 ±0.2	88.2 ±0.1	94.6 ±0.2	59.0 ±4.8	89.0 ±0.3	**97.3 ±0.3**	87.0 ±3.2	94.2 ±0.2	87.3 ±0.7
	Carve VFM	**97.3 ±0.4**	**86.2 ±0.4**	**93.8 ±0.6**	**82.9 ±0.7**	**87.9 ±0.4**	**94.0 ±0.1**	**95.7 ±0.0**	**81.3 ±0.5**	**89.9 ±0.2**	**97.3 ±0.0**	**89.2 ±0.8**	**94.5 ±0.4**	**90.8 ±0.2**
35M	Random	49.7 ±1.3	21.1 ±0.7	22.7 ±1.5	5.9 ±0.3	4.8 ±0.2	3.1 ±0.1	8.7 ±0.5	21.1 ±1.4	20.7 ±3.0	80.1 ±2.4	82.7 ±1.2	82.5 ±1.3	33.6 ±1.1
	Magnitude	42.8 ±0.8	17.9 ±0.4	22.1 ±0.7	7.1 ±0.0	4.8 ±0.1	3.5 ±0.2	9.0 ±0.3	19.3 ±0.5	22.0 ±0.7	85.4 ±0.6	84.7 ±0.5	80.4 ±0.7	33.3 ±0.5
	LAMP	45.1 ±1.1	17.6 ±0.6	22.1 ±0.8	5.7 ±0.1	4.6 ±0.4	3.4 ±0.2	8.7 ±0.3	19.3 ±0.5	22.5 ±0.2	79.7 ±0.7	81.9 ±1.0	81.2 ±1.5	32.7 ±0.6
	SNIP–Mag.	58.2 ±1.3	28.0 ±1.7	25.8 ±2.5	7.8 ±0.1	7.2 ±0.1	4.1 ±0.1	11.4 ±1.0	27.5 ±1.5	26.2 ±0.6	87.4 ±1.5	84.5 ±1.3	86.0 ±0.8	37.8 ±1.0
	Wanda	58.4 ±0.3	29.2 ±1.1	22.8 ±0.9	6.6 ±0.4	5.1 ±0.3	2.9 ±0.2	9.2 ±0.6	21.3 ±1.0	21.5 ±0.2	83.9 ±1.2	83.1 ±0.5	86.6 ±1.1	35.9 ±0.6
	Taylor	54.5 ±1.1	25.9 ±2.2	25.1 ±2.1	7.6 ±1.6	6.0 ±0.2	3.4 ±0.3	9.8 ±1.2	26.0 ±0.4	25.2 ±1.2	85.4 ±1.6	82.4 ±0.3	85.6 ±1.2	36.4 ±1.1
	SnapViT	56.5 ±0.2	27.5 ±1.0	26.6 ±0.9	7.0 ±0.1	5.6 ±0.3	3.3 ±0.1	9.6 ±0.4	24.7 ±0.5	23.1 ±0.7	85.4 ±0.1	81.7 ±1.5	83.2 ±0.7	36.2 ±0.5
	BlockDrop	33.9 ±0.8	12.4 ±0.2	19.7 ±0.4	6.5 ±0.4	1.7 ±0.2	4.1 ±0.2	5.3 ±0.7	30.4 ±0.3	29.8 ±0.6	61.5 ±1.1	82.6 ±0.9	69.3 ±0.6	29.8 ±0.1
	FLAP	71.9 ±0.2	47.3 ±1.1	31.6 ±1.2	15.9 ±0.7	7.3 ±0.1	6.4 ±0.2	15.2 ±0.9	37.0 ±2.4	27.5 ±0.5	93.8 ±0.1	84.4 ±3.0	93.2 ±0.4	44.3 ±0.3
	SVD-LLM	**93.9 ±0.1**	**76.0 ±0.3**	**73.6 ±1.3**	67.9 ±0.4	**72.7 ±0.2**	**85.9 ±0.4**	**87.8 ±0.2**	68.7 ±0.8	76.6 ±0.2	96.1 ±0.6	**86.7 ±2.7**	93.6 ±0.3	**81.6 ±0.2**
	CORP	80.2 ±10.8	73.0 ±0.9	67.9 ±0.6	66.7 ±0.2	44.0 ±17.0	79.6 ±0.3	51.1 ±24.2	48.9 ±0.3	64.7 ±3.8	**96.2 ±0.3**	85.7 ±2.3	**94.7 ±0.2**	71.1 ±0.4
	Carve VFM	93.3 ±1.0	71.4 ±4.9	69.4 ±3.5	**69.0 ±1.0**	69.8 ±2.1	85.3 ±4.5	83.3 ±4.3	**69.9 ±2.7**	**77.6 ±2.3**	95.8 ±0.9	85.7 ±1.3	94.4 ±0.3	80.4 ±1.4
20M	Random	41.7 ±1.3	14.6 ±1.0	15.0 ±2.1	3.7 ±0.5	3.4 ±0.7	2.0 ±0.4	6.0 ±0.7	14.2 ±1.6	14.2 ±1.3	69.1 ±2.9	80.5 ±1.2	75.5 ±0.7	28.3 ±1.2
	Magnitude	41.2 ±1.1	15.5 ±0.3	17.6 ±1.0	4.6 ±0.1	3.6 ±0.5	2.4 ±0.1	7.2 ±0.3	15.7 ±0.9	15.5 ±0.6	77.9 ±0.8	82.0 ±1.1	73.2 ±0.3	29.7 ±0.6
	LAMP	40.4 ±1.2	13.6 ±0.7	15.3 ±1.0	4.0 ±0.2	3.3 ±0.2	2.1 ±0.2	6.6 ±0.3	14.8 ±0.7	14.8 ±0.6	66.7 ±1.2	80.3 ±0.5	76.0 ±1.4	28.2 ±0.7
	SNIP–Mag.	48.1 ±1.3	18.8 ±0.7	16.0 ±2.2	4.6 ±0.3	4.6 ±0.2	2.2 ±0.1	7.3 ±0.5	17.9 ±1.0	16.6 ±0.5	76.6 ±1.3	83.0 ±0.6	78.0 ±1.1	31.1 ±0.8
	Wanda	47.9 ±0.8	17.6 ±0.7	15.0 ±0.8	3.8 ±0.2	2.9 ±0.2	1.9 ±0.2	6.4 ±0.4	13.2 ±0.4	15.0 ±0.6	68.3 ±0.3	82.3 ±0.9	77.1 ±1.4	29.3 ±0.6
	Taylor	44.7 ±1.3	15.3 ±2.0	15.1 ±1.1	4.3 ±0.7	3.9 ±0.2	1.9 ±0.4	6.6 ±0.6	16.4 ±0.7	14.7 ±0.5	72.1 ±0.6	81.7 ±0.2	78.4 ±2.2	29.6 ±0.9
	SnapViT	46.1 ±0.8	16.5 ±0.7	15.2 ±0.9	4.0 ±0.2	3.2 ±0.2	1.8 ±0.1	6.0 ±0.2	15.1 ±1.0	13.6 ±1.0	75.7 ±0.7	79.4 ±0.6	73.8 ±0.9	29.2 ±0.6
	BlockDrop	32.7 ±0.9	10.0 ±0.4	17.9 ±0.3	5.2 ±0.2	2.4 ±0.0	2.8 ±0.2	5.3 ±0.3	11.9 ±0.3	13.5 ±0.4	70.8 ±0.1	80.6 ±1.4	66.8 ±0.4	26.7 ±0.1
	FLAP	57.3 ±0.7	29.8 ±0.4	18.6 ±0.9	5.4 ±0.2	3.8 ±0.4	3.1 ±0.2	7.5 ±0.7	18.2 ±0.2	17.7 ±0.8	79.9 ±0.5	81.9 ±1.3	88.8 ±0.8	34.3 ±0.2
	SVD-LLM	**84.3 ±0.2**	**59.9 ±0.5**	42.7 ±1.1	17.9 ±1.0	13.3 ±0.4	12.1 ±0.7	26.6 ±0.4	44.4 ±0.8	38.7 ±1.3	**94.1 ±0.1**	**85.0 ±2.1**	92.6 ±0.3	51.0 ±0.1
	CORP	46.0 ±5.6	22.1 ±0.6	24.3 ±1.0	6.9 ±0.2	5.1 ±1.2	3.4 ±0.3	9.5 ±1.3	21.2 ±0.6	20.3 ±1.8	86.2 ±1.8	80.6 ±2.3	83.9 ±0.3	34.1 ±0.7
	Carve VFM	82.6 ±1.6	59.0 ±1.3	**45.0 ±1.9**	**32.4 ±5.4**	**17.3 ±7.5**	**12.7 ±6.2**	**39.0 ±6.2**	**50.1 ±2.0**	**46.2 ±3.7**	93.1 ±0.8	81.2 ±2.7	**92.8 ±0.8**	**54.3 ±1.1**

Table 5: Complete per-dataset frozen-backbone linear-probing accuracy (%) for MetaCLIP-2 ViT-B/16 at three dense parameter budgets.

Budget	Method	General			Fine-grained				Texture/scene		Medical			**Mean**
		C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	OrganA	
85.8M	Linear probe	97.5 ±0.1	86.4 ±0.0	96.0 ±0.3	69.5 ±0.2	81.1 ±0.2	91.8 ±0.2	92.0 ±0.3	81.0 ±0.4	92.6 ±0.1	94.7 ±0.3	83.9 ±1.0	88.1 ±0.5	87.9 ±0.0
60M	Random	55.4 ±2.7	28.9 ±2.4	31.4 ±3.5	10.6 ±3.7	7.8 ±2.2	5.5 ±1.7	14.1 ±1.7	26.7 ±3.0	28.4 ±3.9	90.2 ±2.6	83.4 ±2.4	84.4 ±1.8	38.9 ±2.1
	Magnitude	59.5 ±0.2	34.9 ±0.7	33.8 ±1.2	11.5 ±0.6	7.0 ±0.1	6.0 ±0.1	17.1 ±0.8	30.8 ±0.1	34.6 ±0.3	87.9 ±1.5	83.8 ±1.0	88.1 ±0.5	41.2 ±0.3
	LAMP	60.5 ±0.1	35.5 ±0.7	32.3 ±0.8	10.8 ±0.0	7.3 ±0.3	7.3 ±0.2	19.2 ±0.2	30.6 ±1.2	36.9 ±0.8	91.3 ±0.9	87.5 ±1.7	89.3 ±0.3	42.4 ±0.1
	SNIP–Mag.	63.7 ±1.0	36.2 ±4.7	29.1 ±4.6	8.9 ±1.2	5.5 ±0.4	4.3 ±0.6	12.9 ±3.9	18.8 ±0.9	18.9 ±0.4	73.9 ±8.9	83.7 ±1.5	86.7 ±1.3	36.9 ±2.5
	SnapViT	55.1 ±1.3	28.8 ±0.6	28.7 ±0.9	8.1 ±0.4	6.4 ±0.1	4.6 ±0.0	12.0 ±0.5	22.4 ±0.2	25.0 ±0.7	83.5 ±1.9	82.3 ±0.2	87.4 ±0.7	37.0 ±0.0
	BlockDrop	81.0 ±0.1	56.3 ±0.4	45.8 ±0.2	4.7 ±1.0	7.4 ±0.2	6.4 ±0.0	17.1 ±0.3	33.5 ±0.4	39.8 ±0.7	89.0 ±0.5	82.9 ±1.9	87.7 ±0.2	46.0 ±0.2
	FLAP	84.0 ±0.6	59.7 ±1.1	43.3 ±1.0	24.6 ±2.0	27.4 ±3.4	42.7 ±5.3	52.5 ±0.9	52.7 ±0.3	49.3 ±0.6	92.4 ±1.1	86.7 ±1.7	88.2 ±0.3	58.6 ±0.6
	SVD-LLM	94.8 ±0.2	78.8 ±0.4	82.2 ±0.5	52.0 ±0.3	65.3 ±0.5	77.4 ±0.0	85.3 ±0.5	73.2 ±0.3	83.8 ±0.2	95.0 ±0.1	**88.2 ±1.1**	**91.1 ±0.1**	80.6 ±0.2
	CORP	96.8 ±0.3	83.2 ±0.7	92.2 ±0.4	**65.0 ±0.5**	71.7 ±0.2	84.9 ±0.4	88.7 ±0.3	60.1 ±0.1	89.5 ±0.2	**95.4 ±0.5**	85.4 ±3.1	90.0 ±0.3	83.6 ±0.4
	Carve VFM	**97.2 ±0.1**	**84.1 ±0.4**	**93.2 ±0.4**	64.8 ±1.2	**77.7 ±0.8**	**89.2 ±0.1**	**91.7 ±0.3**	**78.2 ±0.9**	**90.1 ±0.4**	**95.4 ±0.0**	88.1 ±2.0	89.6 ±0.6	**86.6 ±0.2**
35M	Random	57.3 ±1.7	31.7 ±2.3	27.3 ±1.7	7.6 ±0.6	5.9 ±0.4	4.7 ±0.3	13.2 ±1.2	25.4 ±1.2	26.3 ±1.0	87.7 ±3.0	84.8 ±1.4	88.9 ±0.4	38.4 ±0.6
	Magnitude	56.9 ±0.8	30.7 ±0.5	27.6 ±1.4	9.3 ±0.3	5.6 ±0.2	5.0 ±0.0	12.5 ±0.8	25.3 ±1.0	27.5 ±1.0	85.4 ±1.4	85.3 ±0.5	88.7 ±0.3	38.3 ±0.2
	LAMP	58.5 ±1.0	31.0 ±0.7	27.0 ±0.7	8.7 ±0.1	5.4 ±0.1	4.9 ±0.3	13.4 ±0.7	25.8 ±0.4	27.5 ±2.0	88.8 ±1.3	87.1 ±1.3	89.3 ±0.7	39.0 ±0.1
	SNIP–Mag.	59.0 ±1.1	31.6 ±1.8	22.6 ±2.9	6.2 ±0.4	4.4 ±0.2	5.0 ±0.2	9.8 ±1.3	21.1 ±0.9	21.1 ±4.0	79.0 ±5.2	84.9 ±0.8	88.0 ±2.0	36.1 ±1.7
	SnapViT	52.9 ±0.3	26.8 ±0.6	23.7 ±0.9	7.1 ±0.3	6.1 ±0.2	4.3 ±0.2	10.9 ±0.5	23.3 ±0.5	22.0 ±1.0	85.9 ±1.6	84.2 ±0.7	86.6 ±0.2	36.1 ±0.2
	BlockDrop	64.6 ±0.1	37.9 ±0.3	31.6 ±0.8	4.2 ±0.1	4.8 ±0.2	3.8 ±0.4	7.9 ±3.5	24.8 ±9.8	30.2 ±0.1	89.3 ±0.4	79.0 ±1.5	86.6 ±0.9	38.7 ±1.0
	FLAP	63.2 ±0.6	39.4 ±0.8	27.5 ±0.7	11.0 ±0.7	6.5 ±0.4	4.2 ±0.4	16.1 ±1.6	32.4 ±0.8	23.3 ±0.4	86.8 ±1.2	**88.0 ±0.3**	87.6 ±0.3	40.5 ±0.2
	SVD-LLM	89.2 ±0.4	66.9 ±0.7	48.3 ±1.2	21.7 ±0.7	21.3 ±1.5	10.3 ±0.3	38.2 ±0.8	50.1 ±1.7	49.5 ±2.3	**94.8 ±0.4**	83.7 ±2.0	91.3 ±0.8	55.4 ±0.5
	CORP	89.0 ±0.1	66.1 ±0.4	56.8 ±0.9	33.5 ±0.6	42.1 ±0.2	44.5 ±1.0	59.9 ±0.5	44.2 ±1.1	56.3 ±2.7	94.6 ±0.7	86.9 ±4.0	91.4 ±0.7	63.8 ±0.2
	Carve VFM	**92.0 ±0.5**	**70.4 ±0.8**	**68.6 ±2.0**	**39.5 ±0.9**	**52.3 ±4.1**	**60.4 ±6.2**	**75.0 ±0.1**	**54.4 ±9.7**	**74.1 ±1.2**	93.9 ±0.0	85.4 ±2.4	**92.2 ±0.2**	**71.5 ±1.0**
20M	Random	52.6 ±1.1	25.1 ±1.0	20.7 ±1.8	5.2 ±0.2	3.9 ±0.2	2.9 ±0.4	9.5 ±1.0	19.5 ±0.4	19.3 ±1.4	80.8 ±2.6	82.5 ±1.5	86.8 ±1.8	34.1 ±0.8
	Magnitude	54.1 ±0.2	26.5 ±0.1	23.1 ±1.0	6.8 ±0.2	4.4 ±0.2	3.4 ±0.2	10.8 ±0.6	20.8 ±0.4	21.9 ±0.9	80.3 ±1.6	82.7 ±0.4	87.6 ±0.4	35.2 ±0.2
	LAMP	54.7 ±0.2	27.3 ±0.6	22.8 ±0.6	6.5 ±0.2	4.5 ±0.2	3.6 ±0.4	10.8 ±0.4	20.7 ±0.5	22.1 ±1.2	81.1 ±1.5	82.9 ±0.8	87.4 ±0.8	35.4 ±0.1
	SNIP–Mag.	56.2 ±0.5	26.1 ±0.6	21.3 ±1.0	6.2 ±0.2	4.7 ±0.2	3.8 ±0.1	9.7 ±1.0	20.5 ±0.6	20.7 ±1.7	83.7 ±3.2	84.5 ±1.2	86.2 ±0.8	35.3 ±0.9
	SnapViT	51.4 ±0.8	25.6 ±0.6	22.3 ±0.5	5.5 ±0.2	5.2 ±0.1	3.8 ±0.2	9.6 ±0.7	19.7 ±0.4	20.7 ±0.7	87.2 ±1.3	84.1 ±0.7	87.0 ±0.6	35.2 ±0.1
	BlockDrop	54.0 ±1.0	26.0 ±0.2	20.8 ±0.9	2.9 ±0.4	1.7 ±0.5	2.5 ±0.1	6.8 ±2.4	12.2 ±7.6	18.1 ±0.3	78.4 ±0.4	79.1 ±2.0	85.5 ±1.2	32.3 ±0.8
	FLAP	63.2 ±0.9	35.7 ±0.5	25.6 ±1.2	7.1 ±0.5	4.9 ±0.1	4.7 ±0.1	13.1 ±0.0	28.9 ±0.9	25.1 ±1.2	89.9 ±1.1	**86.6 ±0.1**	90.1 ±0.4	39.6 ±0.1
	SVD-LLM	**77.7 ±0.1**	**50.9 ±0.6**	31.9 ±0.8	9.5 ±0.5	6.6 ±0.5	4.3 ±0.5	12.7 ±0.1	33.1 ±0.8	28.6 ±0.9	90.1 ±0.6	85.4 ±2.5	90.7 ±0.9	43.5 ±0.3
	CORP	62.2 ±0.7	34.1 ±0.4	25.4 ±0.3	6.3 ±0.1	5.4 ±0.3	3.0 ±0.2	10.5 ±1.0	20.2 ±0.8	21.9 ±1.1	90.2 ±0.2	84.5 ±0.8	89.9 ±0.9	37.8 ±0.1
	Carve VFM	76.6 ±1.3	42.5 ±5.1	**36.6 ±1.1**	**16.5 ±0.9**	**15.9 ±1.0**	**7.8 ±1.9**	**34.5 ±1.4**	**42.3 ±4.4**	**43.9 ±1.2**	**91.1 ±0.3**	85.1 ±2.4	**90.8 ±2.2**	**48.6 ±0.5**

Table 6: Complete per-dataset frozen-backbone linear-probing accuracy (%) for DINOv3 ViT-L/16 at three dense parameter budgets.

Budget	Method	General			Fine-grained				Texture/scene		Medical			**Mean**
		C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	OrganA	
303.1M	Linear probe	99.1 ±0.0	93.1 ±0.1	97.3 ±0.1	80.4 ±0.1	90.6 ±0.1	94.4 ±0.1	96.7 ±0.1	82.7 ±0.2	90.7 ±0.1	95.9 ±0.0	87.3 ±0.6	92.3 ±0.1	91.7 ±0.0
200M	Random	67.5 ±0.5	41.7 ±0.6	54.6 ±0.7	20.4 ±3.2	13.4 ±0.5	11.4 ±2.1	28.1 ±1.2	42.9 ±1.0	47.4 ±2.3	93.1 ±1.0	85.5 ±0.9	90.7 ±1.4	49.7 ±0.6
	SNIP–Mag.	69.6 ±0.7	44.5 ±1.3	50.3 ±3.1	25.8 ±1.0	15.3 ±1.2	17.2 ±1.3	26.7 ±4.6	40.4 ±1.6	47.9 ±3.0	94.6 ±0.3	86.4 ±0.7	92.1 ±0.8	50.9 ±0.6
	SnapViT	72.2 ±1.0	46.5 ±0.3	61.8 ±0.8	26.8 ±0.5	15.9 ±0.4	18.6 ±0.1	28.0 ±0.5	45.8 ±0.1	55.5 ±1.0	93.1 ±0.1	85.2 ±0.9	92.9 ±0.3	53.5 ±0.1
	BlockDrop	77.8 ±0.5	56.4 ±0.2	76.5 ±0.7	41.8 ±5.1	46.7 ±2.7	58.6 ±0.5	65.9 ±0.2	64.8 ±0.1	70.1 ±0.9	95.6 ±0.2	86.1 ±0.8	90.8 ±0.4	69.3 ±0.2
	FLAP	93.4 ±0.3	77.5 ±0.2	88.7 ±0.1	73.7 ±0.7	81.2 ±1.0	90.7 ±0.5	92.6 ±0.1	78.5 ±0.5	88.7 ±0.3	95.7 ±0.4	86.2 ±2.0	91.6 ±0.1	86.6 ±0.2
	SVD-LLM	**98.9 ±0.0**	90.7 ±0.1	96.1 ±0.1	80.6 ±1.2	89.7 ±0.1	**94.1 ±0.1**	**96.5 ±0.0**	83.2 ±0.3	**91.0 ±0.5**	96.2 ±0.4	**89.9 ±1.9**	93.1 ±0.2	**91.7 ±0.3**
	CORP	**98.9 ±0.0**	89.5 ±2.8	**96.5 ±0.1**	79.8 ±2.4	83.1 ±0.1	88.7 ±0.2	95.8 ±0.2	63.8 ±0.1	90.6 ±0.2	**97.0 ±0.3**	88.2 ±1.1	93.0 ±0.4	88.8 ±0.2
	Carve VFM	**98.9 ±0.1**	**90.9 ±0.3**	96.3 ±0.2	**81.6 ±0.6**	**89.8 ±0.7**	94.0 ±0.3	**96.5 ±0.1**	**83.3 ±0.9**	90.8 ±0.4	96.5 ±0.7	87.9 ±1.7	**93.6 ±0.2**	**91.7 ±0.5**
150M	Random	52.3 ±2.6	23.7 ±2.5	25.6 ±2.5	6.3 ±1.2	5.6 ±1.3	3.7 ±0.5	9.9 ±1.2	21.7 ±2.6	23.1 ±0.6	77.9 ±5.0	82.3 ±1.7	84.0 ±0.8	34.7 ±1.5
	SNIP–Mag.	60.2 ±0.6	31.7 ±1.5	33.1 ±1.0	11.7 ±1.0	11.2 ±0.7	7.2 ±0.2	14.2 ±0.9	29.1 ±0.8	33.5 ±1.6	90.3 ±0.9	83.4 ±1.5	89.5 ±0.8	41.3 ±0.2
	SnapViT	60.9 ±1.1	33.5 ±0.6	36.6 ±0.7	11.0 ±0.1	8.4 ±0.5	6.5 ±0.4	13.7 ±0.3	30.5 ±0.5	33.4 ±0.9	86.9 ±0.4	84.0 ±1.4	89.7 ±0.3	41.3 ±0.1
	BlockDrop	62.1 ±7.6	41.6 ±0.5	34.2 ±0.2	8.7 ±3.4	9.0 ±1.1	8.9 ±0.3	19.0 ±0.6	38.4 ±0.2	27.9 ±2.4	92.6 ±0.3	84.6 ±0.7	87.1 ±0.1	42.8 ±0.8
	FLAP	84.1 ±0.5	62.0 ±0.3	70.4 ±0.6	64.3 ±0.7	60.6 ±0.9	83.9 ±1.1	78.6 ±0.1	72.3 ±0.1	71.5 ±0.7	95.5 ±0.4	85.3 ±1.3	92.0 ±0.0	76.7 ±0.2
	SVD-LLM	**98.4 ±0.0**	88.1 ±0.1	93.6 ±0.3	79.5 ±1.4	88.2 ±0.3	**94.0 ±0.1**	**95.9 ±0.1**	80.2 ±0.5	90.2 ±0.2	95.8 ±0.7	**88.2 ±2.1**	92.8 ±0.2	90.4 ±0.3
	CORP	**98.4 ±0.1**	88.2 ±0.3	93.1 ±0.3	78.0 ±2.2	77.8 ±4.2	88.4 ±0.1	94.4 ±1.0	62.1 ±0.9	89.8 ±0.1	**96.4 ±0.2**	86.9 ±1.1	93.1 ±0.3	87.2 ±0.3
	Carve VFM	98.1 ±0.5	**88.3 ±1.5**	**93.7 ±0.5**	**81.2 ±0.7**	**88.5 ±0.4**	93.6 ±0.1	95.7 ±0.3	**82.2 ±0.6**	**90.4 ±0.6**	95.9 ±0.3	86.8 ±2.7	**93.6 ±0.4**	**90.7 ±0.7**
100M	Random	40.4 ±2.7	13.5 ±0.7	12.8 ±0.2	2.8 ±0.4	2.4 ±0.2	1.5 ±0.1	5.6 ±0.2	12.5 ±0.6	11.3 ±1.3	62.2 ±3.0	78.9 ±3.2	75.4 ±0.4	26.6 ±0.7
	SNIP–Mag.	49.2 ±1.3	22.7 ±1.1	20.5 ±1.3	5.6 ±0.2	6.4 ±0.4	3.5 ±0.2	10.0 ±0.2	18.4 ±0.6	23.1 ±1.7	76.5 ±5.4	81.0 ±1.0	82.2 ±0.3	33.3 ±0.3
	SnapViT	47.5 ±2.0	19.4 ±0.9	18.7 ±0.6	4.4 ±0.2	3.7 ±0.3	2.6 ±0.4	7.3 ±0.7	16.5 ±0.4	17.2 ±1.3	68.8 ±0.5	81.7 ±1.1	82.4 ±1.0	30.8 ±0.3
	BlockDrop	54.6 ±4.7	32.8 ±0.3	25.6 ±0.7	8.4 ±1.0	7.6 ±0.9	5.4 ±0.2	14.6 ±1.5	25.1 ±0.1	15.6 ±1.6	90.8 ±0.3	84.7 ±1.5	85.0 ±0.1	37.5 ±0.3
	FLAP	77.4 ±0.2	52.4 ±0.3	45.4 ±0.6	29.3 ±0.6	11.4 ±0.4	12.5 ±0.8	28.9 ±0.5	51.5 ±0.7	40.8 ±0.7	92.9 ±0.3	84.0 ±0.6	92.8 ±0.1	51.6 ±0.1
	SVD-LLM	**96.1 ±0.0**	**80.0 ±0.2**	78.5 ±0.5	**74.0 ±0.6**	**81.5 ±0.2**	**91.8 ±0.1**	**92.0 ±0.2**	69.8 ±1.0	**82.1 ±0.3**	**94.9 ±0.3**	**86.1 ±2.2**	92.0 ±0.4	**84.9 ±0.2**
	CORP	71.0 ±8.0	37.1 ±15.6	57.9 ±3.5	36.9 ±10.0	33.5 ±2.4	59.6 ±2.7	58.1 ±3.6	45.7 ±1.1	57.1 ±1.4	91.8 ±0.6	80.4 ±3.5	89.5 ±3.4	59.9 ±0.8
	Carve VFM	94.2 ±0.9	76.9 ±4.2	**80.1 ±1.3**	73.6 ±1.6	76.4 ±5.4	89.0 ±1.8	90.6 ±0.8	**74.0 ±1.0**	80.3 ±1.9	92.7 ±2.9	84.2 ±1.2	**93.1 ±0.4**	83.7 ±1.9

Table 7: Per-dataset linear-probing accuracy (%) for SliceGPT and OSSCAR at mild and tight budgets.

Budget	Method	General			Fine-grained				Texture/scene		Medical			**Mean**
		C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	OrganA	
DINOv3 ViT-S/16														
14M	SliceGPT	92.4 ±0.2	75.7 ±0.3	41.0 ±1.1	7.9 ±0.6	7.5 ±0.6	5.4 ±0.5	26.5 ±0.9	59.5 ±1.1	51.1 ±1.3	95.3 ±0.4	88.3 ±1.0	92.7 ±0.1	53.6 ±0.3
14M	OSSCAR	91.3 ±0.1	73.5 ±0.3	75.1 ±0.8	57.5 ±1.1	59.7 ±0.1	75.6 ±0.3	81.6 ±0.3	69.8 ±0.9	69.1 ±6.1	96.1 ±0.1	88.9 ±0.8	93.0 ±0.2	77.6 ±0.6
6M	SliceGPT	78.4 ±0.3	52.0 ±0.4	26.4 ±0.7	7.6 ±0.4	4.3 ±0.6	3.0 ±0.1	19.5 ±0.5	25.2 ±0.9	18.7 ±0.5	80.4 ±0.9	80.7 ±1.5	90.1 ±0.2	40.5 ±0.1
6M	OSSCAR	57.7 ±0.7	31.4 ±0.2	27.7 ±1.8	12.4 ±0.6	10.9 ±0.2	5.4 ±0.3	18.5 ±0.8	41.6 ±1.0	24.5 ±0.4	89.7 ±0.3	82.2 ±1.0	86.2 ±0.4	40.7 ±0.1
DINOv3 ViT-B/16														
60M	SliceGPT	96.6 ±0.1	83.3 ±0.3	89.6 ±0.3	78.1 ±0.7	83.8 ±0.3	91.8 ±0.0	93.4 ±0.2	76.6 ±0.3	87.2 ±1.0	96.0 ±0.2	87.3 ±2.5	93.5 ±0.2	88.1 ±0.2
60M	OSSCAR	97.2 ±0.1	85.3 ±0.1	91.6 ±0.2	76.8 ±0.7	84.1 ±0.5	91.5 ±0.4	94.8 ±0.2	77.7 ±0.9	87.2 ±0.4	96.3 ±0.0	86.5 ±2.0	94.1 ±0.2	88.6 ±0.1
20M	SliceGPT	84.8 ±0.4	60.2 ±0.1	27.3 ±0.4	6.1 ±0.3	4.2 ±0.3	3.4 ±0.2	12.9 ±0.4	22.9 ±1.6	19.3 ±0.7	92.3 ±0.2	88.2 ±0.5	92.2 ±0.2	42.8 ±0.2
20M	OSSCAR	66.3 ±1.3	32.8 ±0.1	37.0 ±0.4	21.6 ±0.4	12.5 ±0.7	6.4 ±0.8	26.8 ±0.4	46.2 ±0.2	34.0 ±0.8	90.1 ±0.9	85.6 ±1.4	87.3 ±0.4	45.5 ±0.3
MetaCLIP-2 ViT-B/16														
60M	SliceGPT	96.8 ±0.1	82.0 ±0.2	91.1 ±0.8	59.9 ±1.3	74.7 ±0.7	88.9 ±0.1	90.3 ±0.5	77.4 ±0.8	88.7 ±0.4	93.8 ±0.3	83.8 ±3.2	88.1 ±1.2	84.6 ±0.2
60M	OSSCAR	96.5 ±0.3	82.0 ±0.6	90.9 ±0.5	62.4 ±0.7	74.0 ±0.1	87.3 ±0.5	90.4 ±0.5	76.2 ±0.6	87.7 ±0.3	93.6 ±0.8	87.7 ±2.2	89.2 ±1.1	84.8 ±0.3
20M	SliceGPT	85.8 ±0.3	56.0 ±1.0	43.7 ±0.3	19.4 ±1.0	22.8 ±0.3	27.4 ±0.4	53.1 ±1.1	43.7 ±1.4	35.2 ±1.3	88.2 ±0.8	82.1 ±2.6	84.0 ±1.0	53.4 ±0.4
20M	OSSCAR	66.9 ±0.5	38.5 ±0.7	28.2 ±0.7	10.7 ±1.7	5.3 ±0.4	5.2 ±0.2	20.3 ±1.6	40.1 ±0.6	29.2 ±1.1	83.8 ±2.3	81.0 ±3.3	86.6 ±0.6	41.3 ±0.4
DINOv3 ViT-L/16														
200M	SliceGPT	98.7 ±0.1	89.2 ±0.3	94.0 ±0.3	78.5 ±0.5	87.4 ±0.2	93.2 ±0.3	95.5 ±0.0	80.3 ±0.1	89.5 ±0.2	94.4 ±0.4	89.2 ±0.9	92.1 ±0.6	90.2 ±0.1
200M	OSSCAR	97.8 ±0.2	86.6 ±0.2	92.1 ±0.4	71.0 ±0.7	74.2 ±3.9	82.0 ±1.0	92.4 ±0.8	75.0 ±0.5	89.0 ±0.4	95.3 ±0.5	86.6 ±0.1	92.2 ±0.2	86.2 ±0.5
100M	SliceGPT	95.8 ±0.2	77.5 ±0.1	67.1 ±0.8	68.2 ±1.0	58.1 ±4.8	86.4 ±0.5	86.4 ±0.2	47.0 ±0.4	78.7 ±1.3	92.1 ±0.4	84.1 ±0.7	91.3 ±0.7	77.7 ±0.3
100M	OSSCAR	84.9 ±0.5	62.0 ±0.1	61.3 ±0.8	47.2 ±1.3	27.7 ±2.8	56.5 ±3.8	69.9 ±4.7	63.4 ±0.5	56.2 ±2.4	92.4 ±0.1	84.4 ±1.4	91.5 ±0.5	66.4 ±0.5

C CARVE VFM IMPLEMENTATION DETAILS

C.1 SEARCH SPACE AND BUDGET PROJECTION

An architecture stores, for every transformer layer, an MLP removal ratio, an attention removal ratio, and an attention operator. These removal ratios are $1 - s_l^{\mathrm{mlp}}$ and $1 - s_l^{\mathrm{att}}$, the complements of the retention ratios in the main paper. The MLP decision removes matched intermediate channels from the up- and down-projections. The attention operator is either whole-head carving (H) or coupled column carving (C), which narrows every surviving head using paired Q/K coordinates and matched V/O coordinates. Every proposed architecture is projected onto the same parameter budget before scoring. This projection applies a common shift to its continuous removal ratios, clips them to $[0, 0.95]$ (retention in $[0.05, 1]$), solves the remaining budget discrepancy by bisection, and rounds attention structures to a feasible dense shape. Consequently, fitness cannot improve by silently retaining more parameters.

The evolutionary population contains 20 candidates, of which eight are retained as elites. Initial candidates include uniform and depth-biased allocations, MLP-only boundary cases, and systematic and random H/C patterns. Mutation perturbs continuous ratios with standard deviation 0.08 and flips an operator with probability 0.18. Crossover swaps complete layer decisions. Runs use at most 12 generations, stop after four generations without improvement, and cap the total number of evaluated architectures at 120 or 200 depending on the campaign.

C.2 LOCAL RECOVERABILITY AND ADAPTIVE GLOBAL AUDIT

Let Z be the input to a linear projection, W the downstream weight, and $G = Z^\top Z/n + \lambda I$ its regularized Gram matrix. For a removable coordinate group $\mathcal{I}$, the local damage score is

$$d(\mathcal{I}) = \mathrm{tr}\big(W_{:\mathcal{I}}[G^{-1}_{\mathcal{I}\mathcal{I}}]^{-1}W_{:\mathcal{I}}^\top\big)\,. \tag{7}$$

Groups are removed greedily. After each removal, the working weights are refitted and G^{-1} is updated by a Schur complement. This yields a nested removal order and cumulative damage curve for every layer and operator. Candidate fitness is then a sum of curve lookups, so a complete generation can be ranked without constructing or forwarding its models. Q/K rotary pairs, whose effect is bilinear, are ordered by their perturbation of calibration attention logits. These local quantities are used only for ranking; held-out evaluations of repaired candidate subnetworks then audit those rankings.

A local sum cannot represent all cross-layer interactions. We therefore audit a rank-stratified sample of 12 candidates with a global score: repeated stratified probe accuracy on held-out partitions of the calibration set. The search changes to this global fitness if the Spearman agreement with the local ranking is below 0.35 or if the locally preferred candidate incurs more than 0.03 normalized held-out regret. Otherwise, the inexpensive local score continues to guide exploration. In both cases, global accuracy re-ranks the 20 finalists over five repeated splits. This policy switched to global search in 26% of mild and 51% of tight DINOv3 ViT-S/B runs, 79% of MetaCLIP-2 runs, and 89% of DINOv3 ViT-L runs.

C.3 CLOSED-FORM REPAIR

For a linear path whose activation coordinates are divided into retained indices $\mathcal{S}$ and removed indices $\mathcal{I}$, the deleted output is $Y_{\mathcal{I}} = Z_{\mathcal{I}}W_{\mathcal{I}}^\top$. Carve VFM predicts this output from the retained activation:

$$\Delta W^\star = Y_{\mathcal{I}}^\top Z_{\mathcal{S}}\left(Z_{\mathcal{S}}^\top Z_{\mathcal{S}} + \lambda I\right)^{-1}, \qquad W_{\mathrm{merged}} = W_{\mathcal{S}} + \mathrm{SVD}_r(\Delta W^\star). \tag{8}$$

Here $\mathrm{SVD}_r(A) = U_r\Sigma_r V_r^\top$ is the reconstructed matrix from the leading r singular triplets of A, with the same dimensions as A. We use $\lambda = 10^{-4}$ and include a mean-residual bias correction. The low-rank factors exist only while estimating and truncating the correction; after addition, the deployed projection is a single smaller dense matrix. Equation (7) therefore supplies a search score, whereas Equation (8) specifies the correction used when a candidate is explicitly repaired. The solve repairs MLP down-projections and linearly recoverable attention paths. Q/K column carving uses alternating ridge updates against the original pre-softmax logits. This update is retained only when it lowers end-to-end calibration feature error; otherwise it is reverted, and up to the next two finalists are tried. No masks, adapters, low-rank branches, or sparse kernels remain at inference.

Table 8: Search, repair, and frozen-backbone linear-probing settings.

Search setting	Value	Repair/linear-probing setting	Value
Population	20	Calibration examples	512
Elites	8	Ridge λ	10^{-4}
Generations	≤ 12	DINOv3 S/B rank	128
Candidate cap	120–200	MetaCLIP/ViT-L rank	256
Mutation σ	0.08	Linear-head epochs	10
Operator flip	0.18	Linear-head learning rate	10^{-3}
Patience	4	Linear-head weight decay	10^{-4}
Finalists	20	Checkpoints	1, 3, 5, 10
Audit candidates	12	Input resolution	224^2
Audit ρ threshold	0.35	Seeds	17, 42, 91
Audit regret threshold	0.03		

C.4 Software and Hardware

Runs use Python 3.10.19, PyTorch 2.9.1 with CUDA 13.0, torchvision 0.24.1, and the Transformers 5.0 development branch. Each compression run uses one NVIDIA RTX PRO 6000 Blackwell Server Edition GPU with 96 GB of memory; independent task–seed runs are distributed across eight such devices. The host contains two AMD EPYC 9754 128-core processors and 2 TB of system memory. Unless a deployment benchmark explicitly states otherwise, reported training and search times exclude queueing between independent jobs but include model loading and data preparation performed inside the run.

D Wall-Clock Cost

Each JSON summary records elapsed wall time from model loading through compression and ten-epoch linear-head training. Table 9 aggregates those values over completed tasks, budgets, and seeds on one RTX PRO 6000 Blackwell GPU. The numbers therefore compare complete per-task runs, not only the score calculation inside each method.

Table 9: Wall-clock minutes per complete task run, reported as mean$\pm$sample standard deviation. A dash marks the incomplete archived DINOv3 S/B CORP timing grid.

Source	Carve VFM	BlockDrop	FLAP	SVD-LLM	CORP
DINOv3 ViT-S/B	5.3 $\pm$2.1	1.5 $\pm$1.1	0.9 $\pm$0.5	1.4 $\pm$0.7	–
MetaCLIP-2 ViT-B	7.6 $\pm$1.8	2.0 $\pm$1.3	1.2 $\pm$0.6	2.1 $\pm$0.5	2.5 $\pm$1.6
DINOv3 ViT-L	16.4 $\pm$2.7	7.2 $\pm$2.9	2.2 $\pm$0.8	4.7 $\pm$0.8	27.1 $\pm$7.4

Carve VFM costs more than most fixed one-shot rules because it evaluates an architecture population, but remains a minutes-scale procedure: 5.3 minutes for ViT-S/B, 7.6 for MetaCLIP-2 ViT-B, and 16.4 for ViT-L on average. The adaptive audit is important to this cost. When the local ordering is trustworthy, most candidates are ranked by cached table lookup rather than model construction. At ViT-L scale, Carve VFM is also faster than the iterative CORP implementation under the same end-to-end protocol.

E Full Fine-Tuning

The selected 14M and 10M DINOv3 ViT-S specialists are fine-tuned end to end without repeating search. AdamW uses learning rate 10^{-5} and weight decay 10^{-2} for the backbone, and learning rate 10^{-3} and weight decay 10^{-4} for the classifier. The batch size is 32 and the schedule lasts ten epochs. Every native compact model starts from its released pretrained weights and follows the identical data, augmentation, optimization, and checkpoint protocol.

Table 10: Full-fine-tuning accuracy (%) under a 10-epoch schedule. Native models use their published pretrained weights; our rows fine-tune the already selected 14M and 10M compressed ViT-S children without rerunning structure search. All rows use batch size 32 and the same checkpoint-selection rule over epochs 1, 3, 5, and 10. Parameter counts vary slightly with the dataset-specific classifier; representative counts are shown. Bold denotes the best fully fine-tuned result in each column (ties at the reported precision are all highlighted). Accuracies are displayed to one decimal.

Model	Params (M)	General			Fine-grained				Texture/scene		Medical			Mean
		C10	C100	Cal.	Air.	CUB	Cars	Pets	DTD	MIT	Blood	Pneu.	OrganA	
ResNet-18	11.2	93.7	77.0	82.0	61.1	69.6	63.1	90.4	64.3	68.7	98.7	90.5	95.4	79.6
ResNet-50	23.5	95.9	81.1	87.8	61.7	74.2	66.8	92.3	66.3	73.1	98.7	91.5	94.2	82.0
EfficientNet-B4	17.6	94.8	79.9	86.9	49.3	65.6	46.4	89.3	63.6	64.0	97.6	89.1	93.2	76.6
DeiT-Tiny	5.5	95.8	80.6	84.1	53.6	73.3	58.2	91.3	65.5	71.6	98.7	92.3	95.9	80.1
ConvNeXt-Pico	8.5	96.7	83.5	88.8	71.7	79.5	77.4	**93.5**	70.4	79.1	98.7	94.4	95.3	85.8
RegNetY-1.6GF	10.3	96.5	82.6	**89.0**	57.2	74.5	65.4	92.3	63.3	70.1	98.7	89.6	95.2	81.2
PoolFormer-S12	11.4	95.3	79.7	87.7	65.1	76.1	71.7	93.0	70.4	76.4	98.4	**95.2**	95.7	83.7
DenseNet-121	7.0	95.7	79.9	86.5	69.7	77.8	72.9	92.3	68.5	72.8	**98.8**	92.3	94.8	83.5
Carve VFM	14	**97.3**	**86.7**	88.8	**74.2**	**80.8**	**89.7**	91.5	**75.5**	**82.2**	**98.8**	94.2	**97.2**	**88.1**
Carve VFM	10	95.7	82.2	75.3	66.2	72.6	83.8	87.7	69.4	75.9	98.4	90.7	96.9	82.9

Against the best native result in each task, the 14M carved specialist wins on eight tasks and ties on one at the reported precision. Because the table selects each model's best checkpoint over epochs 1, 3, 5, and 10, it should not be interpreted as a fixed-epoch comparison.

The advantage depends on model capacity: the 10M specialist reaches 82.9%, below ConvNeXt-Pico (85.8%), PoolFormer-S12 (83.7%), and DenseNet-121 (83.5%). Their representative parameter counts are 8.5M, 11.4M, and 7.0M, respectively. Thus, the 14M result does not imply that every carved budget outperforms native compact models; these comparisons use different model sizes.

F Deployment Measurements

The main paper visualizes the operating points; Table 11 reports the underlying measurements. CPU experiments use FP32, batch 16, and 32 threads on an AMD EPYC 9754. GPU experiments use BF16 and batch 64 on an RTX PRO 6000 Blackwell. Batch-1 latency is averaged over 50 timed iterations and throughput over 30 batches after warm-up. Preprocessing and data loading are excluded. CPU memory is peak process resident set size and GPU memory is peak allocated device memory.

Higher batched throughput does not imply lower batch-1 latency. The full ViT-L takes 10.06 ms on GPU, compared with 10.82, 10.71, and 10.85 ms for the 200M, 150M, and 100M carved models: increases of 7.6%, 6.5%, and 7.9%, respectively. These measurements distinguish batched deployment gains from single-image response time; they do not identify the cause of the latency difference.

Table 11: Deployment efficiency of ViT-L/16 compression at 224^2 resolution. We report compute, batch-1 latency, throughput, and peak memory usage on CPU and GPU. Throughput uses batch 16 on CPU and batch 64 on GPU; memory is peak process RSS on CPU and peak allocated device memory on GPU. Timings are mean±standard deviation.

Device	Model	Budget (M)	GFLOPs↓	Lat. (ms)↓	img/s↑	Mem. (MiB)↓
CPU	ViT-L/16	Full	125.68	214.2 ±1.8	7.4 ±2.7	2246
	SVD-LLM	200	84.17	190.3 ±1.6	9.3 ±0.5	1837
	Carve VFM	200	**83.49**	**117.4 ±3.9**	**14.2 ±0.6**	**1769 ±18**
	SVD-LLM	150	64.11	182.7 ±1.3	9.8 ±0.6	1641
	Carve VFM	150	**63.30**	**105.6 ±6.6**	**18.4 ±0.7**	**1541 ±13**
	SVD-LLM	100	43.96	161.1 ±1.0	10.4 ±1.0	1453
	Carve VFM	100	**42.87**	**99.0 ±14.1**	**23.4 ±3.0**	**1331 ±12**
GPU	ViT-L/16	Full	125.68	10.06 ±0.04	1371 ±1	934
	SVD-LLM	200	84.17	13.24 ±0.04	1026 ±1	738
	Carve VFM	200	**83.49**	**10.82 ±0.20**	**1722 ±38**	**689 ±9**
	SVD-LLM	150	64.11	12.96 ±0.83	1542 ±2	642
	Carve VFM	150	**63.30**	**10.71 ±0.27**	**2090 ±65**	**557 ±12**
	SVD-LLM	100	43.96	13.43 ±0.82	1533 ±1	546
	Carve VFM	100	**42.87**	**10.85 ±0.24**	**2708 ±131**	**454 ±2**

G Limitations and Future Work

Our evaluation demonstrates Carve VFM across image classification, four source backbones, and 512 labeled calibration examples per task. Natural extensions include dense prediction, multimodal tasks, and unlabeled calibration. The medical benchmarks broaden the evaluated domain coverage; extending this evaluation to clinical settings will require dedicated validation.

Performance remains task- and budget-dependent. At 6M parameters, the ViT-S frozen-backbone results are 18.5% on CUB and 10.2% on Cars (Table 3), highlighting room to improve frozen representations under extreme compression. Some tight-budget estimates also vary substantially across seeds, including 17.3±7.5% on CUB for the 20M ViT-B (Table 4). Full fine-tuning currently covers the 14M and 10M ViT-S specialists; evaluating the 6M specialist is a natural next step for the most aggressive budget. Native compact models and factorized models remain competitive in some settings, as detailed in Appendices E and F.

Deployment measurements cover the stated hardware, precision, and batch sizes. The observed throughput and memory gains coexist with higher batch-1 GPU latency than the full model. Complete task runs, including model loading, compression, and ten-epoch linear-head training, also take longer than most fixed one-shot baselines (Table 9). These costs are paid during compression and should be considered against the recurring savings over the subsequent inference workload; measuring the break-even workload is an important direction for deployment analysis.