LOCALPROP: NEURO-LOCALIZED MEMORY-EFFICIENT BACKPROPAGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The current deep learning training paradigm employs end-to-end backpropagation, regardless of the training stage, i.e. pre-training or fine-tuning. However, backpropagating through the entire model is neither biologically plausible nor memory efficient, since learning inside the brain is highly localized. Therefore, we propose LocalProp, a training procedure that locally updates the weights of a model. Our neuro-localized weight updates follow the "pre-training then fine-tuning" paradigm, where the pre-training is based on I-JEPA. After locally updating the weights, a pruning operation is performed, followed by a short final fine-tuning phase. Pruning helps by sending the learning signal from higher blocks to lower blocks. We perform experiments on several datasets, including large-scale benchmarks such as ImageNet, and empirically show that LocalProp reaches good performance at a fraction of GPU peak memory. By varying the number of jointly optimized blocks, we identify gradient-propagation span as a practical control over the accuracy–memory trade-off. We publicly release our code at `https://see.supplementary`.

1 INTRODUCTION

The development arc of the computer vision domain has been defined by the drive to learn increasingly robust and generalizable visual representations (Dosovitskiy et al., 2021; Liu et al., 2021). To build and deploy state-of-the-art approaches for tasks such as detection (Carion et al., 2020), segmentation (Kirillov et al., 2023), or 3D reconstruction (Mildenhall et al., 2020), the community has converged towards a multi-stage recipe: large-scale self-supervised pre-training (Caron et al., 2021; Chen et al., 2020; Grill et al., 2020; He et al., 2022; 2020), followed by a task-specific supervised fine-tuning, and optionally concluding with structured pruning (Han et al., 2015; Li et al., 2017) or distillation (Hinton et al., 2015; Romero et al., 2015; Zhang et al., 2019; Zhao et al., 2023) to meet inference constraints. Yet, across these stages, optimization relies on end-to-end backpropagation (Rumelhart et al., 1986). Although undeniably successful, the global backward pass introduces obvious bottlenecks: it couples all layers under a single unified objective and typically requires storing or recomputing intermediate activations across the entire network depth (Chen et al., 2016).

However, end-to-end backpropagation is not biologically plausible. Neural circuits in the brain do not freeze their forward processing to wait for explicit, global error signals over an entire architecture (Lillicrap et al., 2020). Instead, learning and structural adaptation inside a living brain (*in vivo*) are fundamentally decoupled and highly localized. Recent studies reveal that individual biological neurons simultaneously employ multiple distinct synaptic plasticity rules across different dendritic compartments, functioning entirely independently of a backward pass (Wright et al., 2025). Furthermore, the biological counterpart of neural network pruning, experience-dependent synapse elimination in the adult brain, can be predicted from local synaptic co-activity and timing related to postsynaptic output (Hedrick et al., 2024). Inspired by these findings, we propose treating gradient propagation not as a strict requirement, but as a flexible design dimension.

Several learning methods that do not use backpropagation have recently been proposed. The Forward-Forward algorithm (Hinton, 2022) uses two forward passes to update the weights of the model, while NoProp (Li et al., 2026) trains the network by learning a local denoising process. However, previous works (Hinton, 2022; Li et al., 2026) tested their methods on small neural network architectures (e.g. Forward-Forward (Hinton, 2022) uses three hidden layers) and small datasets (e.g. Forward-Forward (Hinton, 2022) was tested on MNIST (LeCun et al., 2010) and CIFAR-10

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.

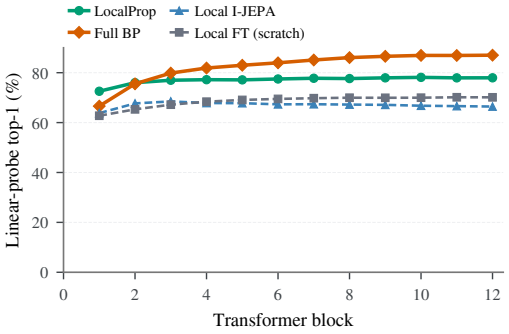


Figure 1: Representation development across ViT-S depth. Linear-probe top-1 accuracy at each transformer block for local I-JEPA pre-training, LocalProp, full backpropagation fine-tuning, and local fine-tuning from scratch on CIFAR-10. LocalProp rapidly improves feature separability in early blocks but then plateaus, while the full-BP upper bound continues to improve with depth.

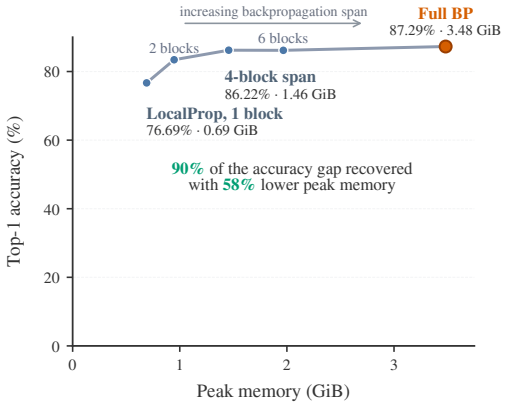


Figure 2: Accuracy-memory trade-off for CIFAR-10 fine-tuning as the local backpropagation window increases. Each point reports top-1 accuracy and peak allocated GPU memory when jointly optimizing 1, 2, 4, 6 blocks via LocalProp vs. 12 blocks (full backpropagation). The four-block setting achieves 86.22% accuracy at 1.46 GiB, using 58% less memory than full backpropagation (87.29% at 3.48 GiB) for a 1.07 percentage-point difference.

(Krizhevsky & Hinton, 2009), while NoProp (Li et al., 2026) added CIFAR-100 (Krizhevsky & Hinton, 2009)). Moreover, the aforementioned works are employed only during supervised training, but do not directly address the complete pipeline of self-supervised pre-training, supervised fine-tuning, and structured pruning considered here.

We propose LocalProp, a minimal-backpropagation framework for vision architectures that applies localized optimization during both pre-training and fine-tuning and extends the same gradient-locality constraint to pruning and recovery. Our approach unifies these stages under a constrained gradient budget. First, we decouple the network into isolated modules (transformer blocks or convolutional parameter groups) and optimize them using local error signals, detaching gradients at module boundaries. This allows us to circumvent the memory costs of standard full-depth backpropagation during large-scale self-supervised pre-training and task-specific fine-tuning. Second, rather than relying on a global loss controller for model compression, we extend this localized methodology to structural pruning. By ablating candidate components and evaluating their effect using a classifier, our method is motivated by the biological mechanism of experience-dependent synapse elimination. We emphasize that, while localized, block-wise learning has been studied previously, its viability across the full life-cycle of modern vision models, spanning pre-training, fine-tuning, and pruning has not been extensively explored.

We evaluate our LocalProp framework across the entire model life-cycle on high-resolution benchmarks, such as ImageNet (Deng et al., 2009), as well as CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009). We employ two types of vision architectures: convolutional networks and transformers. We apply LocalProp in the pre-training stage, by modifying the I-JEPA (Assran et al., 2023) algorithm to local updates. We compare the representations learned at sub-network level isolated by our local-gradient constraints against standard end-to-end approaches and same-size counterparts trained from random initialization, showing that our approach preserves critical structural priors and obtains highly competitive performance. Furthermore, we conduct a hardware efficiency analysis to demonstrate that confining optimization strictly to active blocks brings substantial benefits. As illustrated in Figure 2, a four-block LocalProp reaches strong performance at a fraction (58% less memory) of the peak GPU memory when compared to the full back-propagation algorithm.

In summary, our study makes the following main contributions:

- We show that self-supervised pre-training, supervised fine-tuning, and structured pruning can all be performed under the same locality constraint.

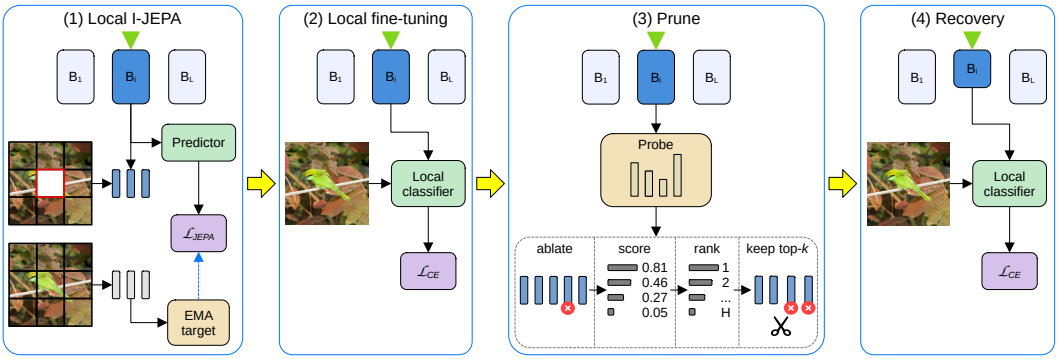


Figure 3: Overview of the LocalProp training pipeline. Our method applies local optimization throughout I-JEPA pre-training, supervised fine-tuning, structured pruning, and local recovery. Only the selected block is optimized and scored at each step, avoiding end-to-end backpropagation. During pruning, downstream task performance is used to score lower-level components, allowing information from higher blocks to guide structural decisions without backpropagation. Best viewed in color.

- We score attention heads or convolutional channels to obtain effective pruning paradigms. The obtained models outperform their same-size from-scratch counterparts, indicating that local pre-training and pruning preserve useful structure rather than merely selecting a smaller architecture.
- By restricting the backward graph, block-wise fine-tuning reduces peak memory per optimization step relative to full fine-tuning. These per-step savings are largest at ImageNet resolution, where avoiding full-depth activation storage substantially lowers memory usage.

2 Related Work

We position LocalProp with respect to three closely related directions: localized optimization, multi-stage vision training, and pruning strategies. While previous works have investigated these domains in isolation, LocalProp demonstrates that a single, restricted-gradient budget can successfully unify them under one cohesive paradigm.

Localized optimization. Modern neural network training relies on backpropagation (LeCun et al., 2015; Rumelhart et al., 1986; Schmidhuber, 2015). Despite its success, it brings substantial activation-memory costs and update locking, requiring error signals to propagate through the network in a way that does not closely reflect biological learning mechanisms (Lillicrap et al., 2016; 2020). This tension has motivated learning rules that reduce, replace, or localize global backpropagation (Lee et al., 2015; Lillicrap et al., 2016; Nøkland & Eidnes, 2019). The *Forward-Forward* algorithm (Hinton, 2022) replaces the backward pass with local goodness objectives. This paradigm has been further expanded to convolutional architectures via channel-wise competitive learning (Papachristodoulou et al., 2024) and prototypical representations (Sarode et al., 2026). Similarly, *NoProp* (Li et al., 2026) focuses on training networks by assigning local denoising objectives to individual blocks, while *Greedy InfoMax* (Löwe et al., 2019) has each layer independently learn to preserve the most salient structural information from the layer directly before it. These methods show that useful representations can emerge even when learning signals are not shared by the whole network.

Other approaches retain some gradient-based optimization, choosing to restrict the scope over which gradients are propagated. Local-error methods attach auxiliary objectives to group representations (Nøkland & Eidnes, 2019), while *Decoupled Greedy Learning* splits models into separately optimized modules to reduce update locking and enable more parallel training (Belilovsky et al., 2020). *SEDONA* (Pyeon et al., 2021) further searches for effective network partitions to be locally trained, highlighting that the design of block-wise learning systems in itself is not trivial. Siddiqui et al. (2024) train ResNet-50 blocks with local Barlow Twins objectives, while AugLocal (Ma et al., 2024) uses auxiliary networks constructed from selected downstream layers to improve supervised

local learning. These works complement DGL by demonstrating the scalability of local objectives in self-supervised and supervised settings.

LocalProp follows this line of work, but takes a middle ground aimed at practical vision models. We do not eliminate backpropagation entirely and instead ask how far a vision model life-cycle can be carried out when backpropagation is confined to small local regions of the network. In our setting, gradients are restricted to the currently optimized block, while the surrounding computation is detached. This preserves the convenience of standard full-depth optimization, but avoids the memory and dependency cost.

Multi-stage vision pipelines. State-of-the-art vision models commonly rely on large self-supervised pre-training, followed by supervised adaptation. Contrastive methods (Chen et al., 2020; He et al., 2020) learn representations by comparing global network outputs across augmented views, while self-distillation approaches (Caron et al., 2021; Grill et al., 2020) avoid explicit negative pairs. Masked-image methods instead predict missing content, as in Masked Autoencoders (MAE) (He et al., 2022; Hondru et al., 2025; Madan et al., 2024). Building upon this, Joint-Embedding Predictive Architectures (I-JEPA) (Assran et al., 2023) avoid pixel-level reconstruction, training with the objective of predicting the latent representations of target patches from a visible context region. Once pre-training is complete, the foundational representations are specialized for downstream tasks through supervised fine-tuning (Yosinski et al., 2014). To meet the hardware constraints of deployment, this pipeline sometimes concludes with pruning (Han et al., 2015; Li et al., 2017) or distillation (Grigore et al., 2025) to eliminate excess architectural capacity.

These stages are commonly implemented using uninterrupted, end-to-end global backward passes for optimization. While prior localized learning methods attempt to remove this dependency, they are almost exclusively evaluated in isolation, addressing either self-supervised pre-training or adaptation, but rarely both. We instead consider the full training pipeline under a limited gradient budget, applying localized optimization from I-JEPA pre-training, through downstream fine-tuning, pruning, and recovery.

Pruning. Standard pruning methods remove complete channels, attention heads, or other components according to an importance criterion. Common approaches estimate this saliency using magnitude or loss-sensitivity (Li et al., 2017; Molchanov et al., 2019), while dependency-based methods model couplings across layers (Fang et al., 2023). In vision transformers, pruning is usually followed by end-to-end fine-tuning to recover performance (Yu et al., 2022). Our framework uses gradient-free ablation scoring to assess structural importance through downstream classification performance. By ablating a channel and observing the drop in the downstream objective, we allow information from the network's final output to guide structural decisions. After pruning, recovery uses block-local supervised updates, while keeping the pruning masks enforced. This allows us to evaluate structured pruning and recovery under the same gradient-locality constraint used during pre-training and fine-tuning.

3 METHOD

Overview. In Figure 3, we illustrate the entire pipeline of LocalProp, composed of four stages: local I-JEPA pre-training, local supervised fine-tuning, pruning, and another short round of local fine-tuning to recover after pruning.

Local I-JEPA pre-training. We pre-train the neural model using a local variant of I-JEPA (Assran et al., 2023). For each training image, we draw one augmented view and partition it into non-overlapping patches. For each image, we then sample one context mask and four target masks as sets of patch indices. The context mask determines which patch tokens remain visible to the online encoder, while the target masks specify the latent representations that must be predicted.

At a selected neural block i, the online encoder processes the visible context only up to that depth. The target encoder processes the complete, unmasked token sequence through the same block and is evaluated without gradients. Its representations at the masked target positions provide the prediction targets. A block-specific predictor maps the context embeddings into a predictor space, appends learned masked tokens at the target positions, adds positional information, and predicts the target-encoder embeddings for those positions. For example, in vision transformers, each transformer block has its own predictor of exactly one transformer layer.

For any target token t, the local pre-training objective at depth i is the Smooth L_1 distance between the predictor output $z_t^{(i)}$ and the detached target-encoder representation $h_t^{(i)}$:

$$\mathcal{L}_{\text{JEPA}}^{(i)} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \text{SmoothL1}\left(z_t^{(i)}, \text{sg}\left[h_t^{(i)}\right]\right), \tag{1}$$

where $\mathcal{T}$ denotes the sampled target patch positions and sg denotes the stop-gradient operation. Image labels are not used during pre-training.

The defining constraint enabling *local* backpropagation training is that only one transformer block is optimized at a time. At each update, the training loop chooses an active block using a round-robin schedule and computes the I-JEPA objective at that depth. Earlier blocks are evaluated without gradients, and blocks following it are not evaluated. Consequently, gradients are restricted to the active block and its corresponding predictor. The shared patch embedding is updated during the processing of the first transformer block. Similarly to I-JEPA, the target block and target patch embedding are updated by exponential moving average (EMA) from their online counterparts rather than by gradient descent.

Local supervised fine-tuning. After self-supervised pre-training, we adapt the encoder to a labeled downstream dataset using block-local supervised losses. At each optimizer step, one single neural block is selected and optimized for a few updates, after which the schedule advances to the next block in round-robin order. The output patch tokens are average-pooled, and a block-specific linear classifier predicts the target. For a selected block i, the base fine-tuning objective is the cross-entropy with label smoothing:

$$\mathcal{L}_{\text{sup}}^{(i)} = \text{CE}\left(c_i\left(\text{pool}\left(h^{(i)}\right)\right), y\right), \tag{2}$$

where $h^{(i)}$ is the token sequence emitted by block i, c_i is its local classifier, and y is the ground-truth class.

Only the active block and its corresponding classifier are optimized during an update. The shared patch embedding is additionally optimized during the update burst assigned to the first transformer block. Applying the same training procedure to a randomly initialized encoder gives the local supervised-from-scratch baseline. Final predictions use the classifier attached to the last transformer block.

Local structured pruning. Pruning begins from a local-wise fine-tuned neural model and processes candidate blocks from deepest to shallowest. Before pruning, we load or fit a lightweight classifier on the pooled output tokens of the final block, while keeping the backbone frozen. The classifier provides a common downstream objective for scoring components throughout the network. For a candidate component u in block i, we temporarily suppress its output and evaluate the resulting final classification loss. Let $\mathcal{L}_{\text{final}}$ denote the baseline loss and let $\mathcal{L}_{\text{final},-u}^{(i)}$ denote the loss after suppressing component u in block i. We define its non-negative importance score as:

$$s_u^{(i)} = \max\left(0, \mathcal{L}_{\text{final},-u}^{(i)} - \mathcal{L}_{\text{final}}\right). \tag{3}$$

A large score indicates that removing the component degrades the final prediction. Scoring requires a forward pass through the complete encoder, but does not require gradient computation or backpropagation. For a partial-retention budget, we keep the highest-scoring components. When the prescribed budget removes an entire branch, all its components are removed and no ranking is required.

For attention-head pruning, suppressing a head removes both sides of its computation. We zero the corresponding rows of the combined query, key, and value projection and the matching input columns from the attention output projection. When the desired number of heads $H_{\text{keep}} > 0$, the surviving output-projection columns are rescaled by $\sqrt{H/H_{\text{keep}}}$, where H is the original number of heads. This approximately preserves the attention-output scale after pruning and limits disruption to the residual stream.

After pruning a block, we fit a classifier to its local representation and optimize only that block against the corresponding local supervised objective. The accumulated structured mask remains enforced in recovery mode. After processing all candidates, we perform a final local cleanup stage over the complete encoder under the final mask, using the same schedule described in the fine-tuning stage.

Table 1: Effect of LocalProp training phases across architectures and datasets. Top-1 accuracy on CIFAR-10, CIFAR-100, and ImageNet-1K for ViT-S and ResNet-50 under multiple configurations of local pre-training, fine-tuning, pruning, and recovery. Reduced-size variants use the same parameter budget as the pruned models. Local pre-training improves subsequent local fine-tuning, while recovery restores much of the performance lost during pruning.

Setting	Local I-JEPA	Pruning	Recovery	Param. factor	ViT-S			ResNet-50		
					CIFAR-10	CIFAR-100	ImageNet-1K	CIFAR-10	CIFAR-100	ImageNet-1K
Local FT (scratch)	✗	✗	✗	1.2×	68.63	42.53	33.29	87.75	65.31	47.36
LocalProp (no pruning)	✓	✗	✗	1.2×	76.69	50.53	37.12	89.79	67.07	55.75
LocalProp (no recovery)	✓	✓	✗	1.0×	52.21	27.62	24.96	88.23	52.94	37.83
LocalProp	✓	✓	✓	1.0×	75.40	47.48	34.81	88.84	67.26	52.19
Local I-JEPA probe	✓	✗	✗	1.2×	49.46	19.16	8.27	32.80	11.09	11.89
Scratch (reduced)	✗	✗	✗	1.0×	66.85	39.17	21.48	85.87	62.71	47.68
LocalProp	✓	✗	✗	1.0×	70.71	44.00	6.09	87.59	65.60	1.49
Full BP (reference)	–	–	–	1.0×	87.29	85.70	76.87	97.40	85.20	76.13

4 Experiments

4.1 Datasets

We evaluate LocalProp on CIFAR-10, CIFAR-100 (Krizhevsky & Hinton, 2009), and ImageNet-1K (Deng et al., 2009), three general-purpose benchmarks covering increasingly challenging settings, from a low-resolution of 32×32 pixels to natural images of 224×224 pixels. CIFAR-10 and CIFAR-100 each contain $50,000$ training and $10,000$ test images, while ImageNet-1K contains $1,281,167$ training and $50,000$ validation images across $1,000$ classes. We include ImageNet-1K as an explicit scalability test, since several recent approaches that avoid or localize backpropagation are evaluated primarily on small, CIFAR-scale benchmarks. This allows us to study whether local optimization remains effective when both dataset scale and input resolution increase substantially.

4.2 Implementation Details

As underlying models, we consider ViT-S/16, containing 12 blocks, 384-dimensional embeddings, and 6 attention heads, as well as ResNet-50, which we divide into 14 local groups following its bottleneck structure. For the convolutional network, we increase the capacity of the later stages using bottleneck width multipliers $[1.0, 1.0, 1.5, 1.5]$, corresponding to internal widths $[64, 128, 384, 768]$. ViT-S operates on 224×224 inputs and is locally pre-trained on ImageNet-1K for 600 epochs using AdamW, an effective batch size of 4096, and a cosine learning rate schedule starting at 10^{-4}. We use one context and four target masks, and update one transformer block per step in round-robin order. For the ResNet experiments on CIFAR-10/100, local I-JEPA pre-training lasts 200 epochs with batch size 128, groups are selected using round-robin, and optimized using detached inter-group activations. All pre-training experiments use BF16 and EMA target encoders.

Local supervised fine-tuning uses independent classifiers and losses for each block or group. The transformer is fine-tuned for 120 epochs with AdamW and a learning rate of 5×10^{-5} for a batch size of 64, updating one block at a time, whereas the convolutional network is trained for 500 epochs with detached group boundaries. An epoch denotes one pass through the training dataset overall, rather than a separate pass per block. In pruning, we use late-block attention-head pruning and channel pruning. Component importance is scored using only the training split of the respective dataset. Recovery employs the same local supervised objectives, while enforcing structural masks throughout optimization. Reported active parameter counts exclude masked weights. Tensors are not physically compacted in these experiments.

4.3 Results

LocalProp performance. In Table 1, we summarize the performance of LocalProp training phases: local pre-training, local adaptation, and pruning. The benefit of local self-supervised pre-training strongly depends on the architecture. For ViT-S, local I-JEPA pre-training provides a consistently better initialization for local fine-tuning, improving over local training from scratch on CIFAR-10, CIFAR-100, and ImageNet-1K. In contrast, ResNet-50 already performs strongly under purely local

Table 2: Peak allocated memory for ViT-S is measured on an RTX 3090. Full updates backpropagate through the complete encoder, whereas local updates optimize one transformer block per step.

Input	Setting	Peak memory	Memory reduction	Top-1 accuracy
CIFAR-10	Full	3.48 GiB	0.0%	87.29%
	Local (1 block)	0.69 GiB	80.2%	76.69%
ImageNet-1K	Full	3.40 GiB	0.0%	76.87%
	Local (1 block)	0.74 GiB	78.2%	37.12%

Table 3: Comparison with alternatives to full backpropagation. Top-1 accuracy of LocalProp, NoProp, Greedy InfoMax, Forward-Forward, and DGL on CIFAR-10, CIFAR-100, and ImageNet-1K using ResNet-50 and ViT-S backbones. Results unavailable in the cited works are obtained from our reproductions. NV (no variant) denotes a backbone variant unavailable in the evaluated implementation. EB (exceeding budget) denotes a configuration whose full-run training time exceeded our per-run budget of 72 GPU-hours (3 days) on an NVIDIA RTX 3090.

Method	Parameter factor	ResNet-50			ViT-S		
		CIFAR-10	CIFAR-100	ImageNet	CIFAR-10	CIFAR-100	ImageNet
LocalProp (no pruning)	1.2×	89.79	67.07	55.75	76.69	50.53	37.12
LocalProp	1.0×	88.84	67.26	52.19	75.40	47.48	34.81
NoProp (Li et al., 2026)	1.2×	83.91	52.02	EB	NV	NV	NV
Greedy InfoMax (Löwe et al., 2019)	1.2×	72.24	41.02	3.11	42.58	15.55	18.81
FF (Papachristodoulou et al., 2024; Sarode et al., 2026)	1.2×	78.11	51.23	25.00	NV	NV	NV
DGL (Belilovsky et al., 2020)	1.2×	83.40	57.60	57.14	78.36	47.02	36.88

supervised training, and the contribution of pre-training is most evident on ImageNet-1K, suggesting that convolutional inductive biases make the architecture less dependent on the self-supervised initialization on smaller datasets. We offer a full-backpropagation comparison from existing published results (Chen et al., 2022; Mo & Tong, 2024; TorchVision Contributors, 2022).

Our pruning strategy has slightly different effects on the two architectures. For ViT-S, our main late-block pruning configuration reduces the active parameter count by 21.8%. For ResNet-50, channel pruning removes 24.3%. Despite similar parameter-reduction ratios, the architectures respond differently to pruning. ViT-S suffers a substantial immediate drop after pruning (37.12% to 24.96% on ImageNet-1K), but the second supervised fine-tuning restores the accuracy toward that of the full-size model (from 24.96% to 34.81% on ImageNet-1K), indicating that the remaining blocks can locally reorganize after components are removed. ResNet-50 is robust immediately after pruning (drop from 89.79% to 88.23% on CIFAR-10), while the benefit of recovery is less uniform across datasets. This suggests that the usefulness of local recovery depends not only on the amount of pruning, but also on how redundancy is distributed across architectures and datasets.

Importantly, the recovered pruned models outperform their same-size counterparts (parameter factor $1.0\times$) initialized from scratch in every comparison, e.g. 75.40% vs. 66.85% for ViT-S on CIFAR-10, and 34.81% vs. 21.48% for ViT-S on ImageNet-1K. Thus, their performance cannot be explained solely by the capacity of the resulting smaller architecture: useful information learned before pruning survives the structural modification and can be recovered using local updates. The advantage diminishes, but remains substantial (75.40% vs. 70.71% for ViT-S on CIFAR-10), when the same-size architecture (parameter factor $1.0\times$) is itself locally pre-trained with I-JEPA.

Backpropagation budget. In Figure 2, we vary the number of jointly optimized ViT blocks from 1 to 12, treating gradient propagation as a tunable coordination budget. Larger windows improve accuracy, but require retaining more activations for backpropagation. The graph in Figure 2 shows that most of the accuracy benefit of full backpropagation is recovered well before reaching the full model depth (12 blocks). A four-block window recovers 90% of the accuracy gap between one-block training and full fine-tuning (86.22% vs. 87.29%), while using 58% less peak memory. These results suggest that gradient propagation is not always a binary choice between local and global learning. Instead, the propagation span acts as a tunable coordination budget, with larger windows allowing optimization to coordinate across more blocks, while shorter windows reduce the cost of training. This could also be employed to train very deep transformers without splitting the model across GPUs, therefore decreasing the communication time while retaining the performance.

Table 4: Ablation of interface objective for ViT-S on CIFAR-10. Top-1 accuracy obtained with different detached interface targets. Representation-level cosine matching performs best, with little difference between neighboring and final block targets.

Interface objective	Target	C10
Pooled cosine	Neighbor block	77.20
Pooled cosine	Final block	**77.25**
Token cosine with delay/ramp	Final block	77.08
Logit KL with delay/ramp	Final block	76.23

Table 5: Ablation of pruning strategies for ViT-S. Comparison of uniform pruning, global head pruning, block dropping and late-block head pruning on CIFAR-10 and CIFAR-100.

Setting	#Params	C10	C100
LocalProp (no pruning)	21.66M	76.69	50.53
Global head pruning	16.94M	69.02	41.49
Late-block head pruning	16.93M	**75.40**	**47.48**
Every third block removed	14.57M	73.55	43.88
Uniform 50% retention	11.03M	66.55	37.35

Understanding local backpropagation. The results shown in Figure 1 indicate that LocalProp achieves strong linear separability in early blocks but saturates later, while full backpropagation continues to improve. The supplementary analysis shows positively aligned local and full-backpropagation gradients with similar magnitudes. These observations motivate our extended gradient-span experiments: although strict local training learns useful representations, independently optimizing each block limits how the model benefits from increasing depth. The qualitative feature maps in Figure 4 show more spatially concentrated activations for LocalProp than for local training from scratch, particularly in later blocks, while the pretrained variants retain clearer spatial structure.

Structured pruning and recovery. The results in Table 5 show that pruning performance depends on where capacity is removed. At nearly the same active-parameter budget, concentrating head pruning in later ViT blocks substantially outperforms global head pruning, indicating that attention capacity in the earlier parts of the model is more sensitive to removal under local training. Dropping every third transformer block performs better than global head pruning, but still remains below late-block head pruning, suggesting that preserving network depth is preferable to eliminating entire blocks. Finally, aggressive uniform pruning produces the largest degradation, showing that compression quality depends on both the amount and structural location of removed capacity.

Comparison with backpropagation alternatives. Table 3 compares LocalProp with approaches that restrict or avoid global backpropagation. The clearest trend is that the gap between LocalProp and competing methods grows as the problem becomes more challenging. On CIFAR, several approaches retain reasonable performance, whereas the gaps over Greedy InfoMax and Forward-Forward become substantially larger on ImageNet-1K suggesting that maintaining useful signals becomes more difficult as dataset scale and class diversity increase. DGL, which uses auxiliary prediction networks during training (Belilovsky et al., 2020), remains competitive, with mixed results across settings.

Across architectures, Greedy InfoMax degrades sharply when transferred from ResNet-50 to ViT-S, while LocalProp remains effective for both convolutional and transformer backbones. The pruned LocalProp models also remain competitive, while operating with roughly one quarter fewer active parameters. Overall, these results indicate that the advantage of our method is most pronounced in settings where local optimization is hardest.

Efficiency. The measurements reported in Tables 2 and 6 show that the computational advantage of local optimization becomes more pronounced as input resolution increases. While localized updates already reduce memory at CIFAR scale, the gap becomes substantially larger on ImageNet, where full backpropagation must retain high-resolution activations throughout the entire network. This indicates that the main savings come from shortening the backward graph, making LocalProp increasingly attractive as training cost scales with model and input size. The per-step savings quantify the benefit of optimizing one block rather than all of them. Total training time additionally depends on the update schedule and the accuracy target.

Interface objective. We further explore whether locally optimized layers can communicate through a detached representation-matching signal, without restoring end-to-end backpropagation. Table 4 shows that representation-level cosine matching performs best, with little difference between neighboring-block and final-block targets, while logit-level KL is less effective. These results sug-

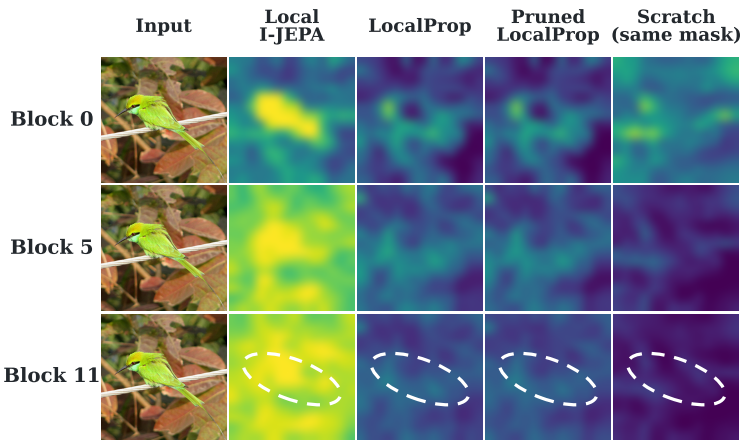


Figure 4: Feature visualizations on an ImageNet validation image. Feature magnitude is shown at transformer blocks 0, 5, and 11 for local I-JEPA, LocalProp, pruned LocalProp, and the same-mask architecture trained from scratch. Colors are jointly normalized across methods within each block row, and the dashed contour marks the same object region in all final maps. LocalProp develops a clearer object-aligned late-layer response than the scratch model, which is largely preserved after pruning.

Table 6: Training cost of block-wise local fine-tuning vs. end-to-end fine-tuning for ViT-S/16. Backward operations are reported per optimization step, and VRAM is the peak allocated memory on an RTX 3090. GPU-hours are measured for block-wise ImageNet fine-tuning and full fine-tuning over 120 epochs.

Method	#blocks	TFLOPs	GiB	GPU-h
Local FT	1/12	0.195	0.74	27.8
Full FT	12/12	2.340	3.40	75.1
Reduction	12.0×	12.0×	4.6×	47.3 saved

gest that additional information from cross-block supervision can help, but is not a major source of performance gain.

5 Conclusion

In this paper, we introduced LocalProp, a novel minimal-backpropagation framework based on localized gradient updates. Our approach applies the same locality principle throughout self-supervised pre-training, supervised fine-tuning and structured pruning with recovery. We conducted experiments on CIFAR-10, CIFAR-100, and ImageNet-1K using transformers and convolutional networks, and considered different gradient-propagation spans and pruning settings. The results show that LocalProp compares favorably with existing local optimization approaches, while substantially reducing peak training memory. Our analyses show that local objectives provide useful and comparably scaled learning signals, but miss out on the coordination across depth. These findings support viewing gradient propagation as a tunable coordination budget and are consistent with the biological motivation that effective plasticity can arise from locally available signals without network-wide information transport.

AI use statement

We used generative AI tools for coding assistance and LaTeX formatting of the tables and figures. The authors reviewed all AI-assisted contributions and take responsibility for the final content of this work, including its code, text, claims, and reported results.

We did not use generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, and interpret results.

Ethics statement

This work studies memory-efficient training of vision models using established benchmarks: CIFAR-10, CIFAR-100, and ImageNet-1K. Reducing memory requirements may improve access to model training on limited hardware. The resulting models may nevertheless inherit biases from their training data, and deployment in sensitive applications requires additional assessment of fairness, privacy, and potential misuse.

Reproducibility statement

Section 3 describes the local learning objectives, block-selection schedule, and pruning and recovery procedures. Section 4 provides the datasets, model configurations, and main training hyperparameters. Additional implementation details, experimental settings, and analyses are provided in the appendix. Source code is included in the supplementary material to support reproduction of the experiments.

References

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of CVPR*, pp. 15619–15629, 2023.

Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Decoupled greedy learning of CNNs. In *Proceedings of ICML*, volume 119, pp. 736–745, 2020.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of ECCV*, pp. 213–229, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of ICCV*, pp. 9650–9660, 2021.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of ICML*, pp. 1597–1607, 2020.

Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When Vision Transformers Outperform ResNets without Pre-Training or Strong Data Augmentations. In *Proceedings of ICLR*, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of CVPR*, pp. 248–255, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of ICLR*, 2021.

Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *Proceedings of CVPR*, pp. 16091–16101, 2023.

Diana-Nicoleta Grigore, Mariana-Iuliana Georgescu, Jon Alvarez Justo, Tor Johansen, Andreea Iuliana Ionescu, and Radu Tudor Ionescu. Weight copy and low-rank adaptation for few-shot distillation of vision transformers. In *Proceedings of WACV*, pp. 7368–7378, 2025.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Proceedings of NeurIPS*, volume 33, 2020.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Proceedings of NeurIPS*, volume 28, 2015.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of CVPR*, pp. 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of CVPR*, pp. 16000–16009, 2022.

Nathan G. Hedrick, William J. Wright, and Takaki Komiyama. Local and global predictors of synapse elimination during motor learning. *Science Advances*, 10(11), 2024.

Geoffrey Hinton. The Forward-Forward Algorithm: Some Preliminary Investigations. *arXiv preprint arXiv:2212.13345*, 2022.

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

Vlad Hondru, Florinel Alin Croitoru, Shervin Minaee, Radu Tudor Ionescu, and Nicu Sebe. Masked image modeling: A survey. *International Journal of Computer Vision*, 133(10):7154–7200, 2025.

András Kalapos and Bálint Gyires-Tóth. CNN-JEPA: Self-Supervised Pretraining Convolutional Neural Networks Using Joint Embedding Predictive Architecture. In *Proceedings of ICMLA*, pp. 1111–1114, 2024.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of ICCV*, pp. 4015–4026, 2023.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Department of Computer Science, University of Toronto, 2009.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. Mnist handwritten digit database. Available: `http://yann.lecun.com/exdb/mnist/`, 2010.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Proceedings of ECML-PKDD*, pp. 498–515, 2015.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *Proceedings of ICLR*, 2017.

Qinyu Li, Yee Whye Teh, and Razvan Pascanu. Noprop: Training neural networks without back-propagation or forward-propagation. In *Proceedings of CoLLAs*, pp. 525–544, 2026.

Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7 (13276), 2016.

Timothy P. Lillicrap, Adam Santoro, Luke Marris, Colin J. Akerman, and Geoffrey Hinton. Back-propagation and the brain. *Nature Reviews Neuroscience*, 21:335–346, 2020.

Zheyuan Liu, Cristian Rodriguez-Opazo, Damien Teney, and Stephen Gould. Image retrieval on real-life images with pre-trained vision-and-language models. In *Proceedings of ICCV*, pp. 2125–2134, October 2021.

Sindy Löwe, Peter O'Connor, and Bastiaan S. Veeling. Putting an end to end-to-end: Gradient-isolated learning of representations. In *Proceedings of NeurIPS*, volume 32, 2019.

Chenxiang Ma, Jibin Wu, Chenyang Si, and Kay Chen Tan. Scaling supervised local learning with augmented auxiliary networks. In *Proceedings of ICLR*, 2024.

Neelu Madan, Nicolae-Cătălin Ristea, Kamal Nasrollahi, Thomas B Moeslund, and Radu Tudor Ionescu. CL-MAE: Curriculum-learned masked autoencoders. In *Proceedings of WACV*, pp. 2480–2490, 2024.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of ECCV*, pp. 405–421, 2020.

Shentong Mo and Shengbang Tong. Connecting joint-embedding predictive architecture with contrastive self-supervised learning. In *Proceedings of NeurIPS*, volume 37, 2024.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of CVPR*, pp. 11264–11272, 2019.

Arild Nøkland and Lars Hiller Eidnes. Training neural networks with local error signals. In *Proceedings of ICML*, pp. 4839–4850, 2019.

Andreas Papachristodoulou, Christos Kyrkou, Stelios Timotheou, and Theocharis Theocharides. Convolutional channel-wise competitive learning for the forward-forward algorithm. In *Proceedings of AAAI*, volume 38, pp. 14536–14544, 2024.

Myeongjang Pyeon, Jihwan Moon, Taeyoung Hahn, and Gunhee Kim. SEDONA: Search for decoupled neural networks toward greedy block-wise learning. In *Proceedings of ICLR*, 2021.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for Thin Deep Nets. In *Proceedings of ICLR*, 2015.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

Shalini Sarode, Brian Moser, Joachim Folz, Federico Raue, Tobias Nauen, Stanislav Frolov, and Andreas Dengel. Hyperspherical forward-forward with prototypical representations. *arXiv preprint arXiv:2605.00082*, 2026.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

Shoaib Ahmed Siddiqui, David Krueger, Yann LeCun, and Stéphane Deny. Blockwise self-supervised learning at scale. *Transactions on Machine Learning Research*, 2024.

TorchVision Contributors. ResNet-50 pretrained weights, 2022. IMAGENET1K_V1: 76.13% ImageNet-1K top-1 accuracy.

William J. Wright, Nathan G. Hedrick, and Takaki Komiyama. Distinct synaptic plasticity rules operate across dendritic compartments in vivo during learning. *Science*, 388, 2025.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Proceedings of NeurIPS*, volume 27, 2014.

Shixing Yu, Tianlong Zhao, Jiwen Yin, Hui Liu, and Jian Yang. A unified framework for compression and acceleration of vision transformers. In *Proceedings of CVPR*, pp. 12124–12133, 2022.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of ICCV*, pp. 3713–3722, 2019.

Borui Zhao, Renjie Song, and Jiajun Liang. Cumulative spatial knowledge distillation for vision transformers. In *Proceedings of ICCV*, pp. 6146–6155, 2023.

Table 7: Key hyperparameters for Local I-JEPA pre-training. The active transformer block or ResNet group is selected in round-robin order. For ViT-S, the learning-rate entry denotes the initial learning rate before warmup and cosine decay. The three ResNet-50 rates correspond to group 0, groups 1–12, and group 13, respectively.

Hyperparameter	ViT-S/16	ResNet-50
Pre-training epochs	600	200
Optimizer	AdamW	AdamW
Learning rate	10^{-4}	$[1, 3, .75] \times 10^{-4}$
Warmup	40 ep.	5 ep.
Weight decay	0.04	10^{-4}
Batch size	4096 (effective)	128
Precision	BF16	BF16
EMA momentum	$.996 \rightarrow 1$	$.999 \rightarrow 1$
Update schedule	1 block/step	1 group/step

A Appendix

A.1 Additional Method Details

Local I-JEPA for ResNet-50. We provide the architectural specifics of the ResNet-50 instantiation of local I-JEPA (Assran et al., 2023). Adapting the objective from vision transformers is not straightforward because, unlike the fixed token grid used by ViT, ResNet-50 repeatedly downsamples its feature maps and increases their channel dimensionality across stages. As a result, context and target masks defined in the input space must be aligned with a different spatial resolution at each local group, while predicting the full target feature map would become increasingly expensive at deeper layers. CNN-JEPA (Kalapos & Gyires-Tóth, 2024) addresses these differences by aligning masks with the encoder's spatial downsampling and using a lightweight predictor built from depthwise-separable convolutions. Inspired by these architectural choices, we resize the input-space context and target masks to the resolution of each local group and use a similar predictor. Our predictive objective is instantiated independently at multiple local groups under the LocalProp gradient constraint.

We partition ResNet-50 into 14 local groups. The first contains the convolutional stem and the first bottleneck. Each remaining bottleneck through Stage 3 forms an individual group, while the final group contains the complete Stage 4. We chose this split to avoid introducing increasingly small local modules. Each local group is followed by a 1×1 projection and a predictor composed of three residual depthwise-separable convolutional blocks.

We use a resolution-agnostic group-level objective for the convolutional variant. Averaging over the target-mask region avoids requiring exact element-wise correspondence after repeated spatial downsampling, while ℓ_2 normalization and cosine distance reduce sensitivity to stage-dependent feature magnitudes. Consequently, the objective matches the semantic content of the masked region, whereas the ViT objective retains the finer token-wise Smooth L1 supervision:

$$\mathcal{L}^{(i)}_{\text{JEPA}} = 1 - \left\langle \widehat{p}^{(i)}, \text{sg}\left[\widehat{h}^{(i)}\right] \right\rangle, \tag{4}$$

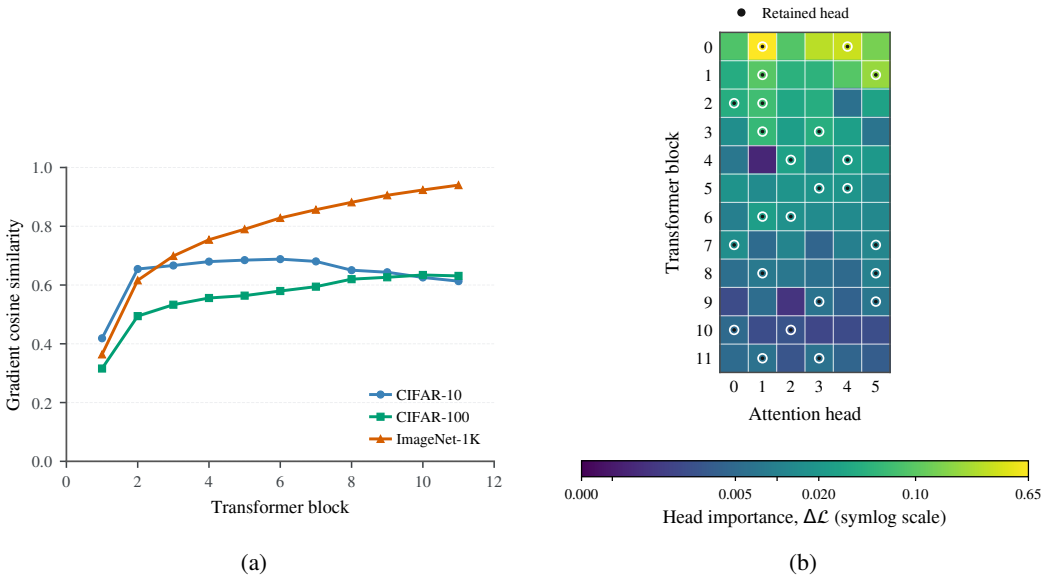
where $\widehat{p}^{(i)}$ and $\widehat{h}^{(i)}$ denote the normalized predicted and target representations for group i, respectively, and sg denotes stop-gradient.

Local supervised fine-tuning for ResNet-50. Unlike the round-robin ViT procedure, all 14 ResNet groups are evaluated and updated on every mini-batch. The input to each group is detached from the preceding group, and its output feature map is globally average-pooled and passed to a local linear classifier over the original dataset classes.

The 14 local losses are summed before backpropagation. Because activations are detached at every group boundary, each loss updates only its corresponding group and classifier. After adaptation, the local classifiers are discarded, and performance is evaluated by training a linear probe on the frozen final-group representations.

Table 8: Effect of pruning order on ResNet-50. CIFAR-10 top-1 accuracy after channel pruning and recovery. Both traversal orders use the same starting checkpoint, pruning budget, scoring procedure, and structured masked constraints, removing 24.29% of the parameters. The main configuration processes layers from deepest to shallowest.

Pruning order	Post-pruning	After recovery
Deepest to shallowest (main)	88.21%	88.84%
Shallowest to deepest	88.04%	89.13%



(a) (b)

Figure 5: **(a)** Cosine similarity between gradients induced by the block-local objective and those received by the same block under full backpropagation. Local gradients across depth have a mean norm ratio of 1.04 ± 0.02. **(b)** Each cell shows the increase in final classification loss after temporarily ablating one attention head. White markers indicate heads retained under an example partial-retention budget. Best viewed in color.

Channel pruning and local recovery for ResNet-50. For ResNet-50, channel importance is evaluated using a fixed linear classifier attached to the final network representation. Each candidate is temporarily ablated, and its importance is defined as the resulting increase in final classification loss. Channels with the highest scores are retained until the target width is reached. The resulting structured mask sets the removed convolutional filters and their associated batch-normalization parameters to zero. Recovery uses the same group-local supervised objective. The structured masks are applied to both parameters and gradients throughout optimization, ensuring that removed channels remain inactive.

A.2 Representation Analysis

Gradient analysis. In Figure 5(a), we compare the gradients induced by each block-local objective with those induced by the final classification objective, computed with respect to the same block parameters at identical model weights and on identical mini-batches. The cosine similarity remains positive across all blocks and datasets, showing that our local losses favor directions compatible with the end-to-end variant. The alignment is imperfect, particularly on CIFAR-100, indicating that LocalProp models do not receive the complete coordination signal available through full-backpropagation. At the same time, the mean gradient-norm ratio is 1.04 ± 0.02, showing that the difference primarily concerns direction rather than gradient scale.

Table 9: Key hyperparameters for local supervised fine-tuning. All runs use AdamW and cross-entropy with label smoothing 0.1.

Model	Dataset(s)	Epochs	Batch size	Learning rate	Weight decay	LR schedule	Local Updates
ViT-S/16	CIFAR-10, CIFAR-100, ImageNet-1K	120	64	5×10^{-5}	0.05	Cosine	8 updates / block
ResNet-50	CIFAR-10, CIFAR-100	500	64	3×10^{-4}	0.01	Constant	all 14 groups
ResNet-50	ImageNet-1K	500	224	3×10^{-4}	0.01	Constant	all 14 groups

Table 10: Key pruning and recovery hyperparameters. Structured masks are enforced throughout local recovery by zeroing the corresponding attention projections for ViT-S and convolutional filters and normalization parameters for ResNet-50.

Hyperparameter	ViT-S/16	ResNet-50	
		CIFAR	ImageNet
Pruning unit	Attention heads	Output channels	Output channels
Pruning target	48/72 heads removed	$1/3$ of each width removed	
Scoring budget (batches)	8	8	8
Recovery budget (epochs)	10	10	10
Recovery batch size	64	64	224
Recovery LR	10^{-5} / 10^{-4}	10^{-5} / 10^{-4}	10^{-5} / 10^{-3}
Weight decay	0.05	10^{-4}	0.05
Label smoothing	0.1	0.1	0.1

Table 11: Top-1 accuracy across three training seeds. Variability is reported using the sample standard deviation.

Setting	Seed 0	Seed 1	Seed 42	$\mu \pm \sigma$
LocalProp	76.69	77.17	76.76	76.87 ± 0.26
Pruned	52.21	53.86	52.36	52.81 ± 0.91
Recovered	75.40	75.93	75.40	75.58 ± 0.31
Same-mask Scratch	66.85	66.46	66.68	66.66 ± 0.20

Pruning score heatmap. In Figure 5(b), we showcase the importance assigned to each attention head. Head importance is measured as $s_u^{(i)} = \max\left(0, \mathcal{L}_{\text{final},-u}^{(i)} - \mathcal{L}_{\text{final}}\right)$, where $\mathcal{L}_{\text{final},-u}^{(i)}$ is the final classifier loss after temporarily suppressing head u in block i. The symmetric-log color scale is linear near zero and logarithmic above $\Delta\mathcal{L} = 0.005$, revealing variation among low-scoring heads, while spanning the complete observed range from zero to the maximum score.

Pruning-order ablation. We examine whether sequential ResNet-50 pruning depends on the order in which the layers are processed. Both variants start from the same CIFAR-10-adapted LocalProp checkpoint and use identical target widths, scoring budgets, structured masks, and local recovery process. The deepest-to-shallowest variant processes layers from the output side toward earlier stages, while the reverse variant follows the opposite traversal. The differences are small and indicate that the final recovered performance is largely insensitive to traversal order.

Additional feature visualizations. In Figure 6, we extend the analysis from the main paper to three additional ImageNet examples. We display feature-magnitude at different blocks using the same protocol and observe that spatial structure is preserved across different object categories.

A.3 Reproducibility Details

Tables 7, 9, and 10 summarize the final configurations used to obtain the results reported in the main paper.

Seed sensitivity. Table 11 reports CIFAR-10 results across three seeds. The relative ordering of the configurations remains stable, with recovery restoring most of the pruning loss and outperforming the same-mask model trained from scratch.

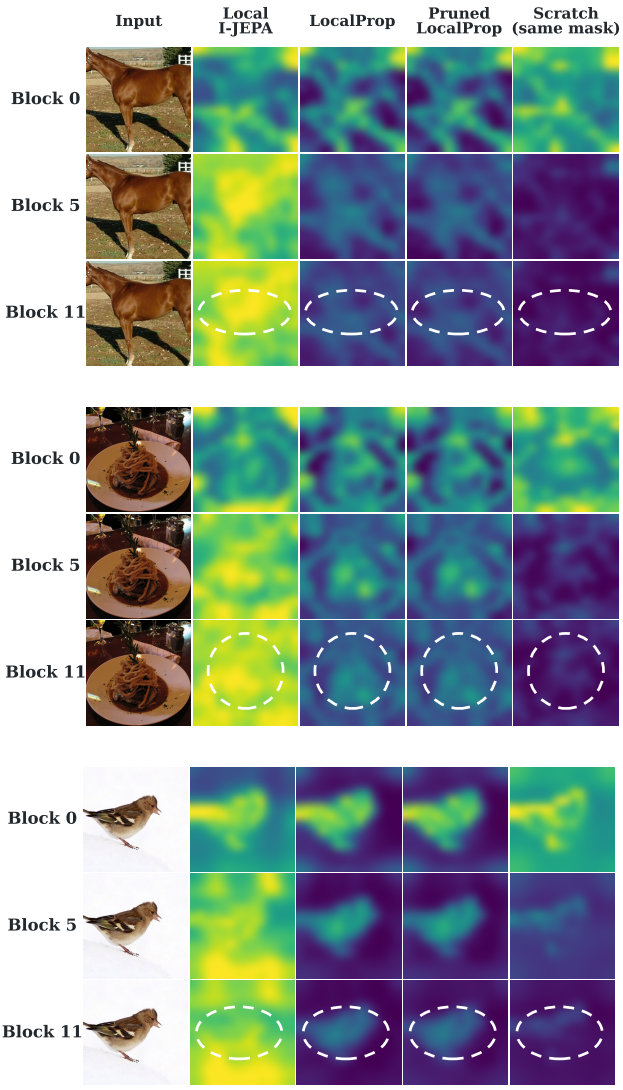


Figure 6: Additional layerwise feature-magnitude visualizations. Columns show the input, local I-JEPA, LocalProp, pruned LocalProp and the same-mask architecture trained from scratch. Rows show transformer blocks 0, 5, and 11 for three ImageNet-1K validation examples. Feature magnitudes are jointly normalized across methods, and dashed contours indicate the same object region. Best viewed in color.

Hardware, software, and randomness. Experiments ran on NVIDIA RTX 3090 and RTX 5090 GPUs, together with an Intel Core i9 CPU running Ubuntu 20.04.6. The software environment used Python 3.8, PyTorch 2.4.1. We seed Python, NumPy, PyTorch, and all CUDA devices before model initialization and data-loader construction. Each main-table entry corresponds to one complete training run. Table 11 separately reports three-seed results. Full version list is included with the submitted code.

Evaluation metrics. We report top-1 classification accuracy as the primary downstream performance metric. Efficiency is measured in the main paper using active parameter count, peak allocated GPU memory, and GPU-hours.

Code and model availability. The supplementary code archive contains implementations and configuration files required to reproduce local I-JEPA pre-training, local supervised fine-tuning, pruning and recovery for both architectures. Upon publication, the source code and trained models will be released under the CC BY-NC 4.0 license.