SIMPLER: Efficient Foundation Model Adaptation via Similarity-Guided Layer Pruning for Earth Observation

Víctor Barreiro[1,2], Johannes Jakubik[3], Francisco Argüello[2], and Dora B. Heras[1,2]

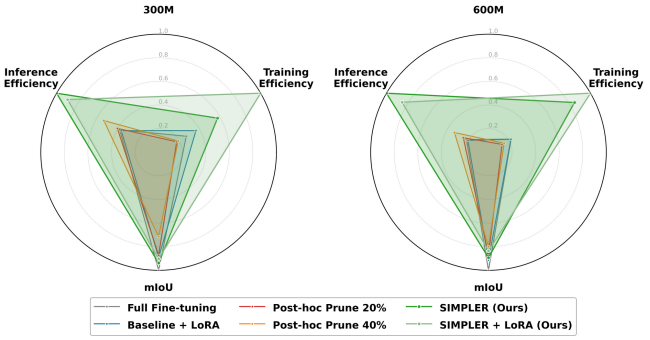
[1] Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela, Spain
[2] Departmento de Electrónica e Computación, Universidade de Santiago de Compostela, Spain
[3] IBM Research Europe, Zurich, Switzerland

Abstract. Fine-tuning foundation models for Earth Observation is computationally expensive, with high training time and memory demands for both training and deployment. Parameter-efficient methods reduce training cost but retain full inference complexity, while post-hoc compression optimizes inference only after costly full fine-tuning. We introduce SIMPLER, a pre–fine-tuning architecture selection method that reduces inference and deployment costs by identifying an effective model depth before adaptation. SIMPLER exploits stabilization of representations in deeper layers of pre-trained vision transformers: it computes layer-wise representation similarity on unlabeled task data and applies an automated scoring function to select redundant layers, with no gradients, magnitude heuristics, or hyperparameter tuning required. On Prithvi-EO-2, SIMPLER prunes up to 79% of parameters while retaining 94% of baseline performance, yielding a 2.1× training speedup and 2.6× inference speedup. The method generalizes to TerraMind (a multimodal EO foundation model) and ImageNet-pretrained ViT-MAE, demonstrating applicability across tasks, architectures, and spectral modalities. Code is available at `https://gitlab.citius.gal/hpc4rs/simpler`.

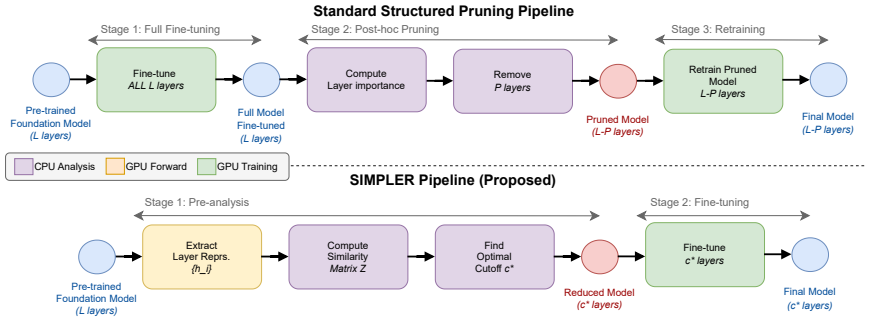
Keywords: Representation Similarity · Model Compression · Foundation Models · Efficient Fine-tuning · Earth Observation · Vision Transformers · Structured Pruning

1 Introduction

Training and deploying large-scale Earth Observation (EO) foundation models imposes substantial computational costs [34]. For example, fine-tuning Prithvi-EO-2 [35] (ViT [7] model, 300M parameters) on BigEarthNetv2 dataset crop mapping requires nearly 2.81 hours on a 4-GPU H200 cluster with 128 GB-VRAM total memory, while inference costs scale linearly with model depth, limiting deployment on satellites, drones, and edge devices that are critical for



(a) Trade-off between training/inference efficiency and mIoU on MADOS for Prithvi-EO-2. Training and inference efficiency are computed as the reciprocal of their respective times (1/time), normalized to [0,1] Higher values indicate better efficiency.



(b) Complete pipeline from layer representation extraction to fine-tuning.

Fig. 1: Overview of the proposed SIMPLER method. The upper radar plot summarizes the trade-off between training/inference efficiency and mIoU performance on the MADOS dataset. The lower panel illustrates the complete pipeline, from layer representation extraction to fine-tuning of the reduced model.

applications including disaster response and precision agriculture [22]. These constraints motivate efficiency research: reducing both training and inference costs without sacrificing the generalization capabilities that make foundation models [2] valuable for Earth observation tasks [6, 41] (Fig. 1a).

Existing solutions address only training or inference cost. Parameter-efficient fine-tuning methods constrain updates to low-rank subspaces or adapter modules, dramatically reducing training memory and time while leaving inference complexity unchanged; all transformer blocks remain active during deployment. Conversely, structured pruning compresses models for faster inference but operates post-hoc: practitioners must complete expensive full fine-tuning, analyze task-specific weight statistics, iteratively prune and retrain, and finally deploy (Fig. 1b). This sequential workflow incurs high computational cost before delivering efficiency gains and relies on post-adaptation parameters rather than

pre-trained representational structure. No existing method reduces both training and inference costs in a unified manner.

We propose SIMPLER (**SIM**ilarity-based **P**arameter **L**ightweight **E**fficient **R**eduction) based on exploiting a key observation: deep layers in pre-trained vision transformers produce nearly identical representations when processing downstream task samples [31], revealing redundancy before any adaptation occurs. SIMPLER (Fig. 1b) selects optimal architecture depth by computing layer-wise representation similarity on unlabeled task samples from the pre-trained model. Using Centered Kernel Alignment [21], we measure how each layer transforms input distributions, partition the similarity matrix at candidate cutoff points, and apply a scoring function that balances representational diversity in retained layers against stability in pruned layers. This automatically identifies the cutoff without magnitude thresholds, gradient computation, or hyperparameter search, enabling architecture selection *before* fine-tuning begins.

In particular, we make three contributions:

- We show that representation similarity on pre-trained features predicts post-fine-tuning layer importance. This fact is validated by ablation studies showing that pruned architectures retain full capacity when trained from scratch while pre-training provides gains of 42-43%.
- The proposed automated scoring criterion identifies optimal depth without hyperparameter tuning, with CKA-selected cutoffs (5 blocks, 94% performance) substantially outperforming alternative metrics (2 blocks, 76% performance).
- The approach generalizes across foundation models (Prithvi-EO-2, TerraMind, ViT-MAE [12]), task types (segmentation, classification, time series), and spectral modalities (multispectral EO, RGB natural images).

2 Related Work

Parameter-efficient fine-tuning (PEFT) methods like LoRA [15], adapters [4, 14], and visual prompts [18] constrain updates to low-rank subspaces or bottleneck modules, dramatically reducing training costs while maintaining task performance. However, these methods operate in weight space, assuming all layers contribute meaningfully. At inference, the full model depth remains active, offering no reduction in deployment costs.

Model compression addresses inference costs through complementary strategies. Structured pruning [9, 28, 43, 45] removes entire architectural components (layers, attention heads, channels), achieving substantial compression while maintaining competitive performance on standard hardware. The critical limitation is timing [36]: structured pruning operates post-hoc, after completing expensive full fine-tuning. Practitioners must analyze task-specific weight statistics, iteratively prune and retrain, incurring full training costs upfront before efficiency gains are realized. Magnitude-based criteria can misidentify layer importance, removing layers with small weights that perform valuable transformations [10].

Knowledge distillation [13, 38, 40, 44] transfers knowledge from large teacher models to compact students, achieving impressive compression ratios but requiring training the new models from scratch. Quantization [26, 27, 46] reduces numerical precision, compressing models and accelerating inference through specialized hardware. While effective and orthogonal to our approach, quantization does not address architectural depth or training efficiency.

Adaptive depth methods enable flexible model capacity at deployment. LayerDrop [8] applies structured dropout during training, allowing arbitrary depth sub-networks to be extracted at inference without additional fine-tuning. Early exiting methods [1, 37, 42] attach auxiliary classifiers to intermediate layers, enabling samples to terminate inference early once a confidence criterion is met. While these approaches offer deployment flexibility, and potential inference savings, they increase training complexity and do not explicitly identify a single optimal architecture based on pre-trained representations prior to fine-tuning.

Representation similarity metrics provide tools for analyzing neural network internals. Centered Kernel Alignment (CKA) [21] measures similarity through kernel methods with invariance to orthogonal transformations. SVCCA [30] applies singular value decomposition before canonical correlation analysis. Jaccard similarity [20] quantifies overlap in k-nearest neighbor graphs. These metrics reveal that vision transformers produce increasingly uniform representations in deeper layers [3, 16, 31], a deep-layer redundancy also reported for large language models [11], as self-attention enables early global information aggregation while residual connections propagate features across depth. However, prior work uses these metrics exclusively for post-hoc analysis of trained models [29]. The crucial distinction is the stage of analysis: representation similarity patterns visible in pre-trained models on downstream task data are never exploited to inform architecture decisions before fine-tuning begins.

Table 1: Comparison of adaptation strategies and computational cost.

Method	**Stage**	**Train Cost**	**Inf. Cost**
LoRA	During	**Low**	Baseline
Post-hoc Pruning	After	High	**Low**
SIMPLER	**Before**	**Low**	**Low**

Adaptation Stage and Cost Comparison. As shown in Table 1, we compare methods according to the stage at which adaptation occurs (before, during, or after fine-tuning) and their training and inference costs; SIMPLER acts in pre-trained representation space, yielding low train and inference cost while avoiding layer-level and weight-specific constraints.

3 Methodology

SIMPLER selects optimal architecture depth for foundation model adaptation through representation similarity analysis before fine-tuning begins.

3.1 Problem Formulation

Given pre-trained model $\mathcal{F}_{\text{pre}}$ with L layers and representations $\mathbf{h}_\ell$, downstream task $\mathcal{T}$ with dataset $\mathcal{D}$, we identify optimal cutoff $c^* \in \{2, \ldots, L-2\}$ minimizing depth while preserving data-relevant features:

$$c^* = \underset{c \in \{2,\ldots,L-2\}}{\arg\max} \ \text{score}(c; \mathcal{S}, \mathcal{F}_{\text{pre}}) \tag{1}$$

where $\mathcal{S} \subset \mathcal{D}$ is a small unlabeled sample set. We then fine-tune $\mathcal{F}_{c^*}$ (layers 1 to c^*) with data head $\mathcal{H}_{\mathcal{D}}$, avoiding full model adaptation costs.

3.2 Representation Similarity Metrics

We compute layer similarity matrix $\mathbf{Z} \in \mathbb{R}^{L \times L}$ where $Z_{i,j}$ quantifies redundancy between layers i and j. We evaluate three metrics: **CKA** [21] (invariant to orthogonal transforms, $\mathcal{O}(N^2D^2)$ complexity); **Jaccard** [20] (nearest-neighbor overlap with parameter k); **SVCCA** [30] (SVD-based canonical correlation, $\mathcal{O}(ND^2)$ complexity). While our approach is metric-agnostic, CKA provides the most consistent results across datasets and architectures (see Suppl. Mat. Sec. A for detailed analysis).

Motivation. Pre-trained representation similarity predicts post-fine-tuning layer importance through two complementary mechanisms. When consecutive layers produce highly similar representations on downstream task data, their transformations are redundant for that distribution. During fine-tuning via gradient descent, the loss landscape exhibits flat directions along these aligned representations, making it difficult to differentiate them through task-specific learning. Consequently, one layer captures the essential transformation while other layers contribute marginally. We empirically validate this hypothesis in Sec. 4 through ablations showing that pruned architectures retain full capacity when trained from scratch (Tab. 5).

This contrasts with magnitude-based pruning, which risks discarding layers with small weights that perform valuable transformations, and gradient-based pruning, which requires expensive backpropagation and estimates task-specific loss sensitivity rather than intrinsic representational redundancy.

3.3 Automated Layer Selection

We partition similarity matrix $\mathbf{Z}$ at cutoff c into $\mathbf{Z}_{TL} = \mathbf{Z}[0{:}c-1, 0{:}c-1]$ (retained layers, $c \times c$) and $\mathbf{Z}_{BR} = \mathbf{Z}[c{:}L, c{:}L]$ (pruned layers, $(L-c) \times (L-c)$). For a square block $M \in \mathbb{R}^{k \times k}$, let $\delta(M) = \frac{1}{(k-1)\,k} \sum_{i=0}^{k-2} \sum_{j=0}^{k-1} |M_{i,j} - M_{i+1,j}|$

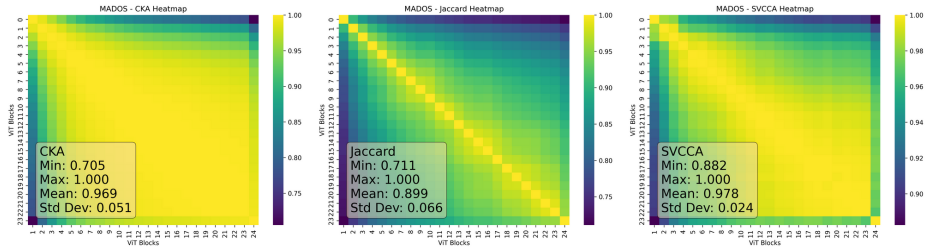


Fig. 2: Similarity metrics comparison (CKA, Jaccard, SVCCA) on MADOS dataset (weak semantic segmentation) with Prithvi-EO-2 300M. Higher values (yellow) indicate greater similarity between layer representations, while lower values (blue) indicate greater divergence.

denote the mean absolute consecutive-row difference. Variability measures are then:

$$\Delta_{TL} = \delta(\mathbf{Z}_{TL}), \qquad \Delta_{BR} = \delta(\mathbf{Z}_{BR}) \tag{2}$$

where Δ_{TL} captures diversity in retained layers (high indicates rich features) and Δ_{BR} captures stability in pruned layers (low indicates redundancy). The optimal cutoff maximizes $c^* = \arg\max_{c \in \{2,\ldots,L-2\}}(\Delta_{TL} - \Delta_{BR})$, requiring no hyperparameter tuning. **Procedure:** (1) Extract representations $\{\mathbf{h}_1, \ldots, \mathbf{h}_L\}$ from $\mathcal{S}$; (2) Compute $\mathbf{Z}$; (3) Select c^* maximizing score; (4) Fine-tune $\mathcal{F}_{c^*}$ with head $\mathcal{H}_{\mathcal{D}}$. Complete pseudocode is provided in Suppl. Mat. Algorithm S1.

3.4 Fine-Tuning Strategies

Once optimal architecture $\mathcal{F}_{c^*}$ has been selected via representation similarity analysis, any fine-tuning strategy can be applied to adapt the reduced model to the downstream task. SIMPLER operates in architecture space (layer selection) rather than weight space (parameter updates), making it orthogonal and complementary to parameter-efficient fine-tuning methods. In our experiments, we evaluate two strategies: **(1) Full fine-tuning** trains all parameters of $\mathcal{F}_{c^*}$ and task head $\mathcal{H}_{\mathcal{T}}$ to maximize performance on the downstream task; **(2) SIMPLER with LoRA** applies low-rank adaptation [15] to the reduced encoder $\mathcal{F}_{c^*}$, achieving compound efficiency gains by reducing both architectural depth (fewer layers) and trainable parameters (low-rank updates). SIMPLER is compatible with other PEFT methods such as adapters [14], prefix tuning [25], or prompt tuning [23], as architecture selection does not constrain weight-space optimization strategies.

An important practical advantage over post-hoc structured pruning is that SIMPLER produces standard dense models avoiding the need for specialized sparse inference libraries. The reduced architecture can be deployed directly on any standard PyTorch or TensorFlow runtimes without any modification, facilitating integration across HPC, cloud, and edge environments.

4 Experiments

We conduct comprehensive experiments to evaluate SIMPLER across multiple downstream tasks, datasets, and model configurations. Our experimental design addresses three key objectives: (1) confirming that similarity patterns in EO foundation models exhibit the representation stabilization observed in prior work; (2) benchmarking SIMPLER against existing techniques in terms of performance-efficiency trade-offs; and (3) validating the generalization of the approach across different foundation model architectures.

4.1 Experimental Setup

Tasks and Datasets. We evaluate SIMPLER across three diverse tasks: *semantic segmentation* on MADOS [19] for Marine Debris/Oil Spill detection with weak supervision; *multi-label classification* on BigEarthNetv2 [5] (19 co-occurring land cover classes); *time series analysis* on Sen4Map [32] for crop type mapping. We primarily use Prithvi-EO-2 [35] (300M and 600M parameters). The results are validated on TerraMind [17] for generalizability.

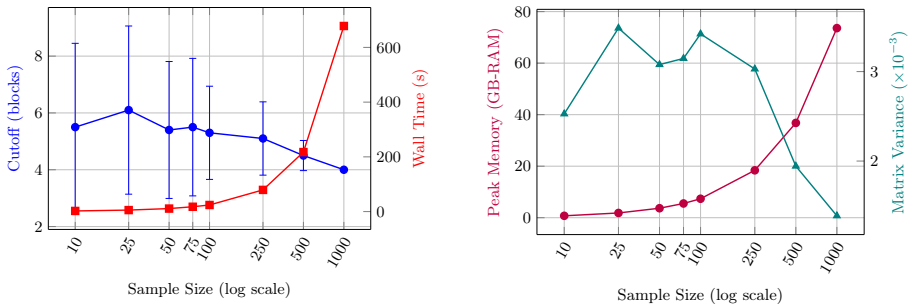
Baselines. We compare against: (1) *Full Fine-tuning* (all parameters); (2) *LoRA* [15] (encoder frozen; decoder and LoRA parameters trained; configuration details in Suppl. Mat. Sec. F); (3) *Post-hoc Structured Pruning* [9, 24] (20% and 40% compression, magnitude-based, with retraining; reported times include full fine-tuning + pruning + retraining); (4) *SIMPLER with LoRA* (combining both approaches, only LoRA parameters and decoder are fine-tuned). An adaptive-depth baseline (LayerDrop [8]) is also reported in Suppl. Mat., Sec. G.

Implementation Details. We extract representations from 500 sampled images per dataset (see sample size analysis in Fig. 3). Jaccard uses $k = 20$ neighbors. Cutoff selection maximizes $\text{score}(c) = \Delta_{TL} - \Delta_{BR}$ without hyperparameter tuning. Complete training hyperparameters, LoRA configurations, and computational requirements are provided in Suppl. Mat. Sec. F.

Evaluation Metrics. We report task-specific performance (mIoU for segmentation, accuracy for classification) and efficiency metrics (total/trainable parameters, training/inference speedup, FLOPs), measured on identical hardware.

4.2 Analysis of Similarity Metrics

We begin by investigating whether representation stabilization patterns documented in vision transformers models [31] manifest in EO foundation models. All three similarity metrics (CKA, Jaccard, SVCCA) reveal distinct progressive stabilization of representations in Prithvi-EO-2 when computed on MADOS samples from the pre-trained model (Fig. 2). Early layers (0-5) exhibit low self-similarity and high inter-layer variability (indicated by the darker off-diagonal regions), reflecting the rapid evolution of features as the model processes low-level visual information. In contrast, the middle layers (6-15) show a gradual increase in similarity, serving as a transitional phase. Crucially, the deep layers (16-24) display high block-diagonal similarity (bright yellow regions in CKA



(a) Cutoff stability and computation time

(b) Memory and CKA matrix variance

Fig. 3: Sample size sensitivity analysis for CKA computation on MADOS (Prithvi-300M). Left: Cutoff selection stabilizes at 500 samples (std=0.53, 5.6× reduction from 10 samples) with acceptable computation time (218s vs. 678s for 1000 samples). Right: Memory consumption scales linearly (36.8GB at 500 samples), while CKA variance remains stable. The 500-sample configuration provides optimal balance between cutoff stability, computational efficiency, and memory footprint. Notice that the computation time is in CPU and RAM, not VRAM

and SVCCA), indicating that representations have stabilized and consecutive layers are performing redundant transformations. This varying redundancy profile across depth confirms the hypothesis of representation stabilization in EO foundation models and justifies the use of our layer selection strategy.

Sample size selection for CKA analysis. We investigate the sensitivity of cutoff selection to the number of samples used for CKA computation (Fig. 3). With only 10 samples, the selected cutoff exhibits high variance (mean=5.5, std=2.95 blocks), leading to unstable layer selection across random draws. Increasing to 500 samples dramatically stabilizes selection (mean=4.5, std=0.53 blocks, 5.6× variance reduction), while maintaining practical computational cost (218s) and memory consumption (36.8GB RAM). Further increasing to 1000 samples yields full convergence (std=0.00) at 3.1× higher computational cost (678s) and 2× memory consumption (74GB-RAM). The matrix variance metric (right panel) remains consistent across sample sizes (mean $\approx$ 0.003), confirming that CKA similarity patterns converge with sufficient samples. Based on this empirical analysis, we use 500 samples for all experiments, achieving stable automated layer selection without excessive computational overhead.

4.3 Performance-Efficiency Trade-offs

We evaluate SIMPLER's effectiveness in balancing task performance with computational efficiency across three downstream tasks, demonstrating consistent compression with competitive accuracy.

Semantic Segmentation (MADOS). SIMPLER (Tab. 2) retains 94% baseline performance (mIoU 62.8% vs 66.9%) while reducing the 300M Prithvi-EO-2 model to just 64.57M parameters (79% reduction), achieving 2.1× training

Table 2: Task 1 - Semantic Segmentation: Results on MADOS dataset comparing SIMPLER against baseline methods. Results show mean ± std over 5 runs. Best results highlighted with light green, second best with light gray.

			Training Cost			Inference Cost			Performance	
Model	**Method**	Params (M)	Train (M)	Time (min)	Mem (GB)	FLOPs (G)	Thr. (img/s)	Inf (s)	mIoU (%)	Acc (%)
300M	Baseline	303.90	303.90	15.90±4.80	11.70±0.47	238.50	33.02±1.94	3.04±0.18	66.9±2.5	95.3±1.2
	Baseline with LoRA	306.32	2.42	11.77±2.17	8.76±0.50	238.50	31.62±1.13	3.17±0.11	59.6±1.5	94.1±0.6
	Baseline + Prune 20%	240.92	240.92	24.34±5.09	11.70±0.47	189.07	35.33±4.02	2.87±0.36	58.4±1.6	93.2±2.0
	Baseline + Prune 40%	177.94	177.94	22.51±4.83	11.70±0.47	139.64	47.03±6.03	2.16±0.25	47.9±3.7	87.2±3.6
	SIMPLER (Ours)	64.57	64.57	7.46±1.62	2.83±0.08	50.67	88.72±15.04	1.16±0.21	62.8±1.2	94.2±1.1
	SIMPLER with LoRA (Ours)	65.12	0.55	4.31±0.28	2.46±0.10	50.67	79.51±14.44	1.30±0.21	60.4±1.4	91.8±1.2
600M	Baseline	631.21	631.21	29.20±8.11	25.06±1.09	646.85	16.28±0.31	6.15±0.12	69.6±2.2	94.1±1.5
	Baseline with LoRA	635.20	3.99	29.55±4.85	17.99±1.05	646.85	15.22±0.63	6.58±0.28	63.5±1.5	95.1±0.3
	Baseline + Prune 20%	513.14	513.14	48.75±10.08	25.06±1.09	525.86	18.62±1.15	5.39±0.34	60.8±5.5	92.8±2.7
	Baseline + Prune 40%	375.40	375.40	42.34±8.84	25.06±1.09	384.70	25.08±0.90	3.99±0.14	55.6±2.0	90.2±3.5
	SIMPLER (Ours)	80.24	80.24	7.70±1.90	3.97±0.16	82.22	77.26±11.27	1.33±0.24	62.2±2.0	90.5±3.4
	SIMPLER with LoRA (Ours)	80.79	0.55	6.49±1.18	3.27±0.12	82.22	64.92±9.56	1.57±0.19	58.1±1.3	88.8±2.1

Table 3: Task 2 - Multi-label Classification: Results on BigEarthNetv2 dataset comparing SIMPLER against baseline methods. Results show mean ± std over 5 runs. Best results are highlighted with light green, second best with light gray.

			Training Cost			Inference Cost			Performance		
Model	**Method**	Params (M)	Train (M)	Time (min)	Mem (GB)	FLOPs (G)	Thr. (img/s)	Inf (s)	mAP (%)	F1-macro (%)	F1-micro (%)
300M	Baseline	303.91	303.91	168.87±6.60	127.84±0.76	238.49	37.67±0.04	2.65±0.00	73.4±0.5	66.8±0.7	77.7±0.2
	Baseline with LoRA	306.33	2.42	230.65±17.90	93.8±0.88	240.38	34.89±0.08	2.87±0.01	72.2±0.4	66.3±0.7	77.6±0.1
	SIMPLER (Ours)	51.98	51.98	40.66±1.64	23.76±0.00	40.78	110.05±0.36	0.91±0.00	71.2±0.3	65.3±0.3	76.4±0.2
	SIMPLER with LoRA (Ours)	52.43	0.45	92.29±16.33	19.04±0.40	41.12	105.11±0.42	0.95±0.00	70.1±0.2	64.2±0.4	76.5±0.1

speedup and 2.6× inference speedup. Compared to LoRA (which reduces trainable parameters but maintains full inference architecture), SIMPLER provides 2.7× inference speedup at comparable performance. Combining SIMPLER with LoRA provides both benefits: 0.55M trainable parameters (0.2% of original), fastest training (4.31 min), 90% baseline performance.

Post-hoc pruning requires full fine-tuning + retraining (22-24 min for 300M vs SIMPLER's 7.46 min). While 40% pruning achieves 72% of baseline (mIoU 47.9%), it shows higher variance (±3.7), suggesting sensitivity to magnitude-based criteria. For 600M, SIMPLER retains 89% of baseline (mIoU 62.2% vs 69.6%) with 87% parameter reduction (80.24M) and 3.8× training speedup, demonstrating scalability and the advantage of pre-fine-tuning architecture selection.

Multi-label Classification (BigEarthNetv2). On the more complex multi-label classification task with 19 co-occurring classes, SIMPLER achieves 83% compression (51.98M parameters) while retaining 97% baseline mAP (71.2% vs 73.4%), with 4.2× training speedup and 2.9× inference speedup (Tab. 3).

For BigEarthNetv2 (300M), SIMPLER reduces the number of parameters to 17% (51.98M) while retaining 97% of baseline mAP (71.2% vs 73.4%), with 2.9× of inference speedup and 4.2× of training speedup. The slightly larger performance gap vs MADOS reflects increased task complexity (19 co-occurring

Table 4: Task 3 - Time Series Classification: Results on Sen4Map dataset comparing SIMPLER against baseline methods. Results show mean ± std over 5 runs. Best results are highlighted with light green, second best with light gray.

		Training Cost				Inference Cost			Performance			
Model	**Method**	Params (M)	Train (M)	Time (min)	Mem (GB)	FLOPs (G)	Thr. (img/s)	Inf (s)	OA (%)	AA (%)	F1-macro (%)	Kappa (%)
300M	Baseline	303.90	303.90	133.81±8.48	75.72±2.88	238.49	4.72±0.00	21.16±0.01	75.4±0.2	65.0±0.4	66.6±0.3	69.7±0.3
	Baseline with LoRA	306.31	2.41	930.42±257.38	53.60±2.04	240.38	3.79±0.09	26.40±0.65	75.8±0.3	65.1±0.7	66.9±0.6	70.1±0.4
	SIMPLER (Ours)	89.76	89.76	55.62±3.83	23.28±1.08	70.44	15.55±0.03	6.43±0.01	73.6±0.1	61.8±0.7	63.8±0.4	67.4±0.1
	SIMPLER with LoRA (Ours)	90.50	0.74	398.17±40.71	17.72±0.56	71.01	12.32±0.08	8.12±0.05	73.9±0.2	61.5±0.6	63.8±0.4	67.6±0.2

land-cover classes). Combining SIMPLER with LoRA achieves 0.45M trainable parameters (0.1% of original) with comparable performance (mAP 70.1%), validating compatibility with parameter-efficient fine-tuning and automatically adapting model capacity to task complexity.

Time Series Analysis (Sen4Map). For time series crop type mapping, SIMPLER compresses the model by 70% (89.76M parameters) while retaining 96% baseline F1-macro (63.8% vs 66.6%), demonstrating 3.3× inference speedup and 2.4× training speedup (Tab. 4).

For Sen4Map (300M), SIMPLER reduces parameters by 70% (89.76M) while retaining 96% of baseline F1-macro (63.8% vs 66.6%), with 3.3× inference speedup and 2.4× training speedup. The results demonstrate consistent parameter reduction (70-83%) with 94-97% performance retention across all three task types (segmentation, classification, time series), validating SIMPLER's broad applicability to diverse EO applications.

Performance Trade-offs: For the time series task (Sen4Map), the baseline with LoRA achieves slightly superior performance (66.9% F1-macro) compared to SIMPLER (63.8% F1). However, this comes at the cost of 16.7× slower training (930.42 min vs 55.62 min) and offers *zero* reduction in inference cost (FLOPs remain 240.38G). In contrast, SIMPLER provides a substantial reduction in both training time (2.4× speedup) and deployment costs (FLOPs reduced to 70.44G, a 3.4× decrease) while retaining 96% of baseline performance. This trade-off reflects different optimization priorities: time series tasks with complex temporal dependencies may benefit from full-depth architectures when accuracy is paramount and inference constraints are relaxed, while SIMPLER is better suited for deployment-critical scenarios requiring both training efficiency and low-latency inference, such as on-device processing for disaster response or real-time edge monitoring. For resource-constrained scenarios requiring both efficient training and low-latency edge deployment, SIMPLER's combined efficiency advantages are decisive.

4.4 Ablation Studies

Similarity Metric Comparison. CKA outperforms Jaccard and SVCCA for layer selection (Tab. 5). While Jaccard/SVCCA select aggressive cutoff (layer 2, 9% parameters), they severely degrade performance to 76% baseline mIoU. CKA selects conservative cutoff (layer 5, 21% parameters) while preserving 94%

Table 5: Ablation Studies on MADOS (300M): (Top) Similarity metrics; (Bottom) Pre-training impact.

Experiment	Configuration	Cutoff	Params (M)	mIoU (%)	Acc (%)
Metric	Baseline (Full)	24 bl.	303.90	66.9±2.5	95.3±1.2
	SIMPLER (Jaccard/SVCCA)	2 bl.	26.78	50.7±3.4	84.6±0.8
	SIMPLER (CKA)	5 bl.	64.57	62.8±1.2	94.2±1.1
Pre-training	Baseline (From Scratch)	24 bl.	303.90	46.7±2.4	83.0±0.8
	Baseline (Fine-tuning)	24 bl.	303.90	66.9±2.5	95.3±1.2
	Baseline (LoRA)	24 bl.	306.32	59.6±1.5	94.1±0.6
	SIMPLER (From Scratch)	5 bl.	64.57	44.1±1.9	81.9±1.3
	SIMPLER (Fine-tuning)	5 bl.	64.57	62.8±1.2	94.2±1.1
	SIMPLER (LoRA)	5 bl.	65.12	60.4±1.4	91.8±1.2

of baseline performance, demonstrating superior task-relevant depth identification. The 3-layer difference yields 18% performance gap, validating CKA as default metric. CKA's advantage arises from its invariance to orthogonal transformations and smooth gradient properties with respect to layer depth, enabling robust identification of semantic similarity rather than superficial feature alignment. In contrast, Jaccard's discrete nearest-neighbor structure and SVCCA's sensitivity to low-variance dimensions can lead to overly aggressive pruning that removes layers still contributing meaningful transformations. CKA's consistent performance across three model families (Prithvi-EO-2, TerraMind, ViT-MAE) and four downstream tasks demonstrates its generality for pre-fine-tuning layer selection. Linear probing analysis (Suppl. Mat. Sec. E) provides complementary evidence: while Block 2 (Jaccard/SVCCA) achieves 33.00% probe mIoU vs. Block 5 (CKA) at 30.26%, the fine-tuning results demonstrate that CKA's selection better captures task-relevant features during adaptation.

Pre-training Impact Analysis. SIMPLER's 5-block architecture (Tab. 5) achieves nearly identical from-scratch performance as the full 24-block baseline (mIoU 44.1% vs 46.7%), indicating removed layers contribute minimal architectural capacity. However, pre-training provides substantial gains (43% and 42% relative improvement), confirming SIMPLER preserves the most valuable pre-trained features. Linear probing on frozen representations (Suppl. Mat. Sec. E) corroborates this finding: Block 6 achieves peak performance (34.03% mIoU), validating that SIMPLER's automated selection of Block 5 captures semantically optimal depth without gradient-based search.

Random Data Ablation. Representation stabilization arises from learned features, not architectural artifacts (Fig. 4). Random noise inputs produce uniformly high similarity (0.998-1.000) with 115× narrower range than real data (0.77-1.00), confirming SIMPLER exploits genuine learned hierarchies requiring downstream task samples.

Block-Selection Ablation. At matched depth, selecting random or last-k blocks instead of the early blocks chosen by SIMPLER degrades performance to the from-scratch range, confirming the gains stem from retaining the transferable early layers rather than from depth reduction alone (Suppl. Mat. Sec. G).

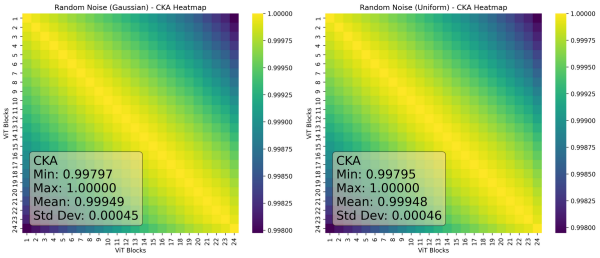


Fig. 4: CKA similarity for random noise (Gaussian, Uniform). Uniformly high similarity (0.998-1.000) with 115× narrower range vs. real data confirms learned features, not architectural artifacts.

Table 6: Task 1 - Semantic Segmentation: Results on MADOS dataset using TerraMind-Large, TerraMind-Small and TerraMind-Tiny models comparing SIMPLER against baseline methods. Results show mean ± std over 5 runs. Best results are highlighted with light green, second best with light gray.

		Training Cost				Inference Cost			Performance	
Model	**Method**	Params (M)	Train (M)	Time (min)	Mem (GB)	FLOPs (G)	Thr. (img/s)	Inf (s)	mIoU (%)	Acc (%)
TerraMind-L	Baseline	304.89	304.89	8.84±1.76	8.38±0.51	61.69	124.91±10.20	3.18±1.08	70.1±2.7	97.1±0.3
	Baseline with LoRA	320.82	15.93	10.34±1.37	6.11±0.50	64.81	69.73±3.10	3.10±0.30	66.1±2.9	96.3±0.3
	SIMPLER (Ours)	53.23	53.23	4.79±0.54	1.62±0.09	10.76	661.24±19.94	1.05±0.63	58.8±2.1	92.7±2.6
	SIMPLER with LoRA (Ours)	56.05	2.82	4.25±0.89	1.24±0.08	11.31	386.94±7.23	0.99±0.47	59.0±2.7	94.2±0.7
TerraMind-S	Baseline	22.37	22.37	6.48±2.01	1.02±0.04	4.74	279.23±6.25	1.06±0.37	53.0±1.1	87.9±1.9
	Baseline with LoRA	26.07	3.70	10.57±3.18	0.95±0.04	5.47	158.57±0.60	1.14±0.14	40.4±7.2	86.7±4.3
	SIMPLER (Ours)	9.96	9.96	6.06±1.15	0.55±0.02	2.10	588.62±7.43	0.76±0.20	53.6±3.7	88.9±1.6
	SIMPLER with LoRA (Ours)	11.59	1.63	8.18±1.87	0.52±0.02	2.42	338.56±3.46	0.81±0.19	41.3±4.7	85.7±4.0
TerraMind-T	Baseline	5.88	5.88	5.15±0.48	0.53±0.01	1.33	236.13±30.85	8.14±5.83	56.3±2.5	91.4±2.3
	Baseline with LoRA	7.79	1.92	10.16±3.56	0.54±0.02	1.71	151.15±12.20	1.92±0.43	53.5±4.1	93.1±1.3
	SIMPLER (Ours)	2.32	2.32	4.85±0.90	0.32±0.01	0.52	762.76±7.53	0.54±0.13	53.8±4.0	89.5±1.4
	SIMPLER with LoRA (Ours)	3.06	0.74	7.75±1.91	0.32±0.00	0.66	433.44±6.89	0.66±0.25	50.5±5.0	85.8±0.6

4.5 Generalization Across Foundation Models

To validate generalization beyond Prithvi-EO-2, we evaluate SIMPLER on TerraMind [17], a multimodal EO foundation model, across three scales: Large, Small, and Tiny on MADOS (Tab. 6).

SIMPLER consistently achieves 55–83% parameter reduction across all three TerraMind scales while retaining 84–101% of baseline mIoU (Tab. 6). Two findings stand out. First, for TerraMind-Small, SIMPLER *improves* over the baseline (mIoU 53.6% vs 53.0%) despite halving the parameters. Second, cross-scale comparison reveals that reducing a larger model with SIMPLER outperforms natively smaller architectures: TerraMind-Large reduced to 53.23M parameters (mIoU 58.8%) surpasses TerraMind-Small baseline (22.37M, mIoU 53.0%) by 5.8 points, indicating that richer representations learned during large-scale pretraining are preserved after layer selection. At the smallest scale, SIMPLER enables ultra-lightweight deployment (2.32M parameters, 762 img/s) while retaining 96% baseline mIoU, opening practical avenues for satellite on-board processing. These results advocate for a "reduce large" strategy, selecting depth from a

Table 7: Generalization validation on ViT-MAE (pre-trained on ImageNet) for CIFAR-100 classification. Results show mean ± std over 5 runs. Best results highlighted with light green, second best with light gray. Training from scratch (without pre-training) results are provided in Suppl. Mat. Sec. B for comparison.

	Training Cost				Inference Cost			Performance
Method	Params (M)	Train (M)	Time (min)	Mem (GB-VRAM)	FLOPs (G)	Thr. (img/s)	Inf (s)	Accuracy (%)
Baseline	303.40	303.40	49.31±5.37	49.59±1.46	59.70	119.11±1.19	1.02±0.15	88.8±0.4
Baseline with LoRA	309.72	6.42	116.50±12.61	57.75±0.59	60.94	88.78±0.49	1.49±0.26	91.8±0.2
SIMPLER (Ours)	38.88	38.88	42.97±1.37	9.06±1.02	7.60	822.51±3.48	0.22±0.01	72.8±0.3
SIMPLER with LoRA (Ours)	40.51	1.73	28.51±11.57	9.62±0.13	7.92	583.11±26.54	0.78±0.73	67.9±2.1

single well-trained foundation model, over training multiple smaller models independently. Beyond the optical modality, SIMPLER also generalizes to Sentinel-1 SAR (BigEarthNet-S1 with TerraMind-Large), retaining 93% of baseline mAP at ∼12× fewer parameters (Suppl. Mat. Sec. G).

Validation on RGB-Only Foundation Models (ViT-MAE + CIFAR-100). Beyond multispectral EO data, SIMPLER generalizes to standard RGB vision transformers. On ViT-MAE (ImageNet pre-trained, CIFAR-100 fine-tuned), SIMPLER achieves 87% parameter reduction (38.88M vs 303.40M) while retaining 82% baseline accuracy, with 81.7% memory reduction (9.06 GB-VRAM vs 49.59 GB-VRAM), 1.1× training speedup, 7.9× FLOPs reduction, and 6.9× inference throughput improvement (Tab. 7).

For ViT-MAE on CIFAR-100, SIMPLER achieves 87% parameter reduction while retaining 82% baseline accuracy (72.8% vs 88.8%), with 81.7% memory reduction and 6.9× inference throughput improvement. The from-scratch comparison (Suppl. Mat. Sec. B) confirms SIMPLER preserves layers most enriched by pre-training rather than selecting based on architectural capacity alone. These results demonstrate that SIMPLER's representation-based layer selection generalizes beyond multispectral EO data to standard RGB vision transformers.

4.6 Discussion

SIMPLER exploits a key property of pre-trained vision transformers: consecutive layers producing highly similar representations on downstream task data perform redundant transformations for that distribution. Our ablation studies (Tab. 5) validate this principle empirically. The pruned 5-block architecture achieves nearly identical from-scratch performance as the full 24-block baseline (mIoU 44.1% vs 46.7%), confirming that removed layers contribute minimal architectural capacity. However, pre-training provides substantial gains (42-43% relative improvement for both architectures), demonstrating that SIMPLER preserves layers most enriched with pre-trained features rather than merely selecting based on capacity.

The multi-scale TerraMind evaluation (Tab. 6) reinforces this finding: at every scale, a SIMPLER-reduced model retains or even surpasses the baseline of the next-smaller architecture, while inheriting the richer representations of the

larger pre-training. This consistent cross-scale pattern suggests that investing in a single large-scale pre-training and deriving task-specific reduced models via SIMPLER is more cost-effective than training multiple smaller foundation models independently.

The method delivers greatest benefits when deployment efficiency is critical for edge devices, satellites, or resource-constrained environments requiring low-latency inference and both training and inference costs matter simultaneously. For scenarios prioritizing absolute accuracy where inference costs are unconstrained (e.g., cloud-based batch processing), parameter-efficient methods like LoRA may be preferable as they maintain full model depth. SIMPLER addresses the common case in Earth observation applications requiring on-board processing, where no existing method simultaneously reduces both training and deployment costs. These gains transfer to embedded hardware: on an NVIDIA Jetson Orin, SIMPLER delivers a $\sim$3.9$\times$ on-device inference speedup (Suppl. Mat. Sec. G).

Representation stabilization naturally emerges in foundation models trained via masked-autoencoding self-supervised objectives. However, architectures employing contrastive (e.g., SoftCon [39]) or explicit collapse-prevention (e.g., DINOv3 [33] with KoLeo regularization) objectives may instead exhibit oscillating or non-stabilizing similarity patterns (Suppl. Mat. Sec. C). Practitioners can verify applicability via CKA heatmap analysis on 500 task samples: clear block-diagonal structure in deep layers indicates suitability, while uniform or oscillating patterns suggest alternative approaches.

5 Conclusion

SIMPLER introduces a paradigm shift in foundation model compression: rather than pruning task-adapted weights post-hoc or assuming all layers are necessary during training, we analyze pre-trained representations on downstream task data to select architecture depth before fine-tuning begins. By computing layer-wise similarity through Centered Kernel Alignment, our automated scoring criterion identifies redundant depth without setting any hyperparameters, achieving 55–87% parameter reduction while retaining 82–101% performance across semantic segmentation, multi-label classification, and time series analysis on multispectral EO data, and RGB classification on natural images. Validation across diverse foundation models (Prithvi-EO-2, TerraMind, ViT-MAE) and datasets (MADOS, BigEarthNetv2, Sen4Map, CIFAR-100) establishes representation similarity as a predictive signal for layer importance that simultaneously reduces training costs, inference costs, and memory footprint, enabling deployment on resource-constrained hardware.

Multi-scale evaluation on TerraMind (Large, Small, Tiny) shows that SIMPLER-reduced models consistently match or outperform natively smaller architectures, with depth reduction even acting as implicit regularization at smaller scales. These findings advocate for a "reduce once" paradigm: investing in a single large-scale pre-training and deriving task-specific reduced architectures via

similarity-based layer selection, rather than training multiple smaller foundation models independently.

SIMPLER generalizes to foundation models trained with MAE-style self-supervised objectives, which naturally exhibit the representation stabilization the method exploits. For models employing alternative pre-training strategies, practitioners can verify applicability through CKA heatmap analysis as detailed in Suppl. Mat. Sec. C. Having validated SIMPLER across architectures, sensing modalities (multispectral, RGB, SAR), and task types, we leave its extension to paradigms without deep-layer stabilization, such as contrastive and collapse-prevention models, as the principal direction for future work (Suppl. Mat., Sec. C).

Acknowledgements

This work was supported in part by grants PID2022–141623NB–I00 funded by MCIN/AEI/10.13039/501100011033 and by "European Union NextGenerationEU/PRTR". It was also supported by Xunta de Galicia - Consellería de Cultura, Educación, Formación Profesional e Universidades [Centro de investigación de Galicia accreditation 2024-2027 ED431G-2023/04 and Reference Competitive Group accreditation, ED431C-2022/16], and by "ERDF/EU". The work of Víctor Barreiro was supported by the Ministerio de Universidades, under a FPU Grant [grant number FPU2022-04364].

References

1. Bakhtiarnia, A., Zhang, Q., Iosifidis, A.: Single-layer vision transformers for more accurate early exits with less overhead. Neural Networks **153**, 461–473 (2022). https://doi.org/10.1016/j.neunet.2022.06.038
2. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021). https://doi.org/10.48550/arXiv.2108.07258
3. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9650–9660 (2021). https://doi.org/10.1109/ICCV48922.2021.00951
4. Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., Luo, P.: AdaptFormer: Adapting vision transformers for scalable visual recognition. In: Advances in Neural Information Processing Systems (NeurIPS) (2022), https://arxiv.org/abs/2205.13535
5. Clasen, K.N., Hackel, L., Burgert, T., Sumbul, G., Demir, B., Markl, V.: reBEN: Refined BigEarthNet dataset for remote sensing image analysis. In: IEEE International Geoscience and Remote Sensing Symposium (IGARSS) (2025). https://doi.org/10.5281/zenodo.10891137
6. Cong, Y., Khanna, S., Meng, C., Liu, P., Rozi, E., He, Y., Burke, M., Lobell, D.B., Ermon, S.: Satmae: pre-training transformers for temporal and multi-spectral satellite imagery. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. NIPS '22 (2022)

7. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (ICLR) (2021), https://openreview.net/forum?id=YicbFdNTTy
8. Fan, A., Grave, E., Joulin, A.: Reducing transformer depth on demand with structured dropout. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=SylO2yStDr
9. Fang, G., Ma, X., Song, M., Mi, M.B., Wang, X.: DepGraph: Towards any structural pruning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16091–16101 (2023). https://doi.org/10.1109/CVPR52729.2023.01544
10. Frankle, J., Carbin, M.: The lottery ticket hypothesis: Finding sparse, trainable neural networks. In: International Conference on Learning Representations (ICLR) (2019), https://openreview.net/forum?id=rJl-b3RcF7
11. Gromov, A., Tirumala, K., Shapourian, H., Glorioso, P., Roberts, D.A.: The unreasonable ineffectiveness of the deeper layers (2024)
12. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15979–15986 (2022). https://doi.org/10.1109/CVPR52688.2022.01553
13. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015). https://doi.org/10.48550/arXiv.1503.02531
14. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., Gelly, S.: Parameter-efficient transfer learning for NLP. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 2790–2799. PMLR (09–15 Jun 2019), https://proceedings.mlr.press/v97/houlsby19a.html
15. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. In: International Conference on Learning Representations (ICLR) (2022), https://openreview.net/forum?id=nZeVKeeFYf9
16. Huh, M., Cheung, B., Wang, T., Isola, P.: Position: The platonic representation hypothesis. In: International Conference on Machine Learning (ICML) (2024), https://proceedings.mlr.press/v235/huh24a.html
17. Jakubik, J., Yang, F., Blumenstiel, B., Scheurer, E., Sedona, R., Maurogiovanni, S., Bosmans, J., Dionelis, N., Marsocci, V., Kopp, N., et al.: TerraMind: Large-scale generative multimodality for earth observation. In: IEEE/CVF International Conference on Computer Vision (ICCV) (2025)
18. Jia, M., Tang, L., Chen, B.C., Cardie, C., Belongie, S., Hariharan, B., Lim, S.N.: Visual prompt tuning. In: European Conference on Computer Vision (ECCV). pp. 709–727. Springer (2022). https://doi.org/10.1007/978-3-031-19827-4_41
19. Kikaki, K., Kakogeorgiou, I., Hoteit, I., Karantzalos, K.: Detecting marine pollutants and sea surface features with deep learning in Sentinel-2 imagery. ISPRS Journal of Photogrammetry and Remote Sensing **210**, 39–57 (2024). https://doi.org/10.1016/j.isprsjprs.2024.02.017, mADOS dataset available at https://marine-pollution.github.io/

20. Klabunde, M., Schumacher, T., Strohmaier, M., Lemmerich, F.: Similarity of neural network models: A survey of functional and representational measures. ACM Comput. Surv. **57**(9) (May 2025). https://doi.org/10.1145/3728458, https://doi.org/10.1145/3728458
21. Kornblith, S., Norouzi, M., Lee, H., Hinton, G.: Similarity of neural network representations revisited. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 3519–3529. PMLR (09–15 Jun 2019), https://proceedings.mlr.press/v97/kornblith19a.html
22. Lacoste, A., Lehmann, N., Rodriguez, P., Sherwin, E.D., Kerner, H., Lütjens, B., Irvin, J.A., Dao, D., Alemohammad, H., Drouin, A., et al.: GEO-Bench: Toward foundation models for earth monitoring. In: Advances in Neural Information Processing Systems (NeurIPS) (2023), https://arxiv.org/abs/2306.03831
23. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 3045–3059. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (Nov 2021). https://doi.org/10.18653/v1/2021.emnlp-main.243, https://aclanthology.org/2021.emnlp-main.243/
24. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient ConvNets. In: International Conference on Learning Representations (2017), https://openreview.net/forum?id=rJqFGTslg
25. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. arXiv preprint arXiv:2101.00190 (2021), https://arxiv.org/abs/2101.00190
26. Li, Y., Xu, S., Zhang, B., Cao, X., Gao, P., Guo, G.: Q-ViT: Accurate and fully quantized low-bit vision transformer. In: Advances in Neural Information Processing Systems (2022), https://proceedings.neurips.cc/paper_files/paper/2022/hash/deb921bff461a7b0a5c344a4871e7101-Abstract-Conference.html
27. Lin, Y., Zhang, T., Sun, P., Li, Z., Zhou, S.: FQ-ViT: Post-training quantization for fully quantized vision transformer. In: Raedt, L.D. (ed.) Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22. pp. 1173–1179. International Joint Conferences on Artificial Intelligence Organization (7 2022). https://doi.org/10.24963/ijcai.2022/164, https://doi.org/10.24963/ijcai.2022/164, main Track
28. Michel, P., Levy, O., Neubig, G.: Are sixteen heads really better than one? In: Advances in Neural Information Processing Systems (NeurIPS). pp. 14014–14024 (2019), https://proceedings.neurips.cc/paper/2019/hash/2c601ad9d2ff9bc8b282670cdd54f69f-Abstract.html
29. Nguyen, T., Raghu, M., Kornblith, S.: Do wide and deep networks learn the same things? Uncovering how neural network representations vary with width and depth. In: International Conference on Learning Representations (ICLR) (2021), https://openreview.net/forum?id=KJNcAkY8tY4
30. Raghu, M., Gilmer, J., Yosinski, J., Sohl-Dickstein, J.N.: SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In: Neural Information Processing Systems (2017). https://doi.org/10.48550/arXiv.1706.05806
31. Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., Dosovitskiy, A.: Do vision transformers see like convolutional neural networks? In: Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems (2021), https://openreview.net/forum?id=Gl8FHfMVTZu

32. Sharma, S., Sedona, R., Riedel, M., Cavallaro, G., Paris, C.: Sen4Map: Advancing mapping with Sentinel-2 by providing detailed semantic descriptions and customizable land-use and land-cover data. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **17**, 13893–13907 (2024). https://doi.org/10.1109/JSTARS.2024.3435081
33. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., Massa, F., Haziza, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., Tolan, J., Brandt, J., Couprie, C., Mairal, J., Jégou, H., Labatut, P., Bojanowski, P.: DINOv3 (2025), https://arxiv.org/abs/2508.10104
34. Strubell, E., Ganesh, A., McCallum, A.: Energy and policy considerations for deep learning in NLP. arXiv preprint arXiv:1906.02243 (2019). https://doi.org/10.48550/arXiv.1906.02243, seminal work on carbon footprint of training large models
35. Szwarcman, D., Roy, S., Fraccaro, P., Þorsteinn Elí Gíslason, Blumenstiel, B., Ghosal, R., de Oliveira, P.H., de Sousa Almeida, J.L., Sedona, R., Kang, Y., Chakraborty, S., Wang, S., Gomes, C., Kumar, A., Truong, M., Godwin, D., Lee, H., Hsu, C.Y., Asanjan, A.A., Mujeci, B., Shidham, D., Keenan, T., Arevalo, P., Li, W., Alemohammad, H., Olofsson, P., Hain, C., Kennedy, R., Zadrozny, B., Bell, D., Cavallaro, G., Watson, C., Maskey, M., Ramachandran, R., Moreno, J.B.: Prithvi-eo-2.0: A versatile multi-temporal foundation model for earth observation applications (2025), https://arxiv.org/abs/2412.02732
36. Tay, Y., Dehghani, M., Bahri, D., Metzler, D.: Efficient transformers: A survey. ACM Computing Surveys **55**(6), 1–28 (2022). https://doi.org/10.1145/3530811
37. Teerapittayanon, S., McDanel, B., Kung, H.T.: BranchyNet: Fast inference via early exiting from deep neural networks. In: International Conference on Pattern Recognition (ICPR). pp. 2464–2469 (2016). https://doi.org/10.1109/ICPR.2016.7900006
38. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers and distillation through attention. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 10347–10357. PMLR (18–24 Jul 2021), https://proceedings.mlr.press/v139/touvron21a.html
39. Wanyan, X., Seneviratne, S., Shen, S., Kirley, M.: Extending global-local view alignment for self-supervised learning with remote sensing imagery. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 2443–2453 (2024)
40. Wu, K., Zhang, J., Peng, H., Liu, M., Xiao, B., Fu, J., Yuan, L.: TinyViT: Fast pretraining distillation for small vision transformers. In: European Conference on Computer Vision (ECCV). pp. 68–85. Springer (2022). https://doi.org/10.1007/978-3-031-19803-8_5
41. Xiao, A., Xuan, W., Wang, J., Huang, J., Tao, D., Lu, S., Yokoya, N.: Foundation models for remote sensing and earth observation: A survey (2025). https://doi.org/10.1109/MGRS.2025.3576766
42. Xu, G., Hao, J., Shen, L., Hu, H., Luo, Y., Lin, H., Shen, J.: LGViT: Dynamic early exiting for accelerating vision transformer. In: Proceedings of the 31st ACM International Conference on Multimedia. p. 9103–9114. MM '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3581783.3611762

43. Yang, H., Yin, H., Shen, M., Molchanov, P., Li, H., Kautz, J.: Global vision transformer pruning with hessian-aware saliency. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18547–18557 (June 2023)
44. Yang, Z., Li, Z., Zeng, A., Li, Z., Yuan, C., Li, Y.: ViTKD: Feature-based knowledge distillation for vision transformers. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1379–1388 (2024). https://doi.org/10.1109/CVPRW63382.2024.00145
45. Yu, L., Xiang, W.: X-pruner: explainable pruning for vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 24355–24363 (June 2023)
46. Yuan, Z., Xue, C., Chen, Y., Wu, Q., Sun, G.: PTQ4ViT: Post-training quantization for vision transformers with twin uniform quantization. In: European Conference on Computer Vision (ECCV). pp. 191–207. Springer (2022). https://doi.org/10.1007/978-3-031-19775-8_12