DOES THE VGGT FAMILY NEED ALL ITS LAYERS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Which layers of a feed-forward geometry model are needed to preserve both camera poses and dense 3D structure? We study layer redundancy in VGGT, π^3, and VGGT-Ω: 3,018 pruned configurations, scored on seven camera-pose and dense-geometry metrics across four indoor and outdoor datasets. Four findings follow: (i) Removable layers cluster in two redundancy regions: a dominant early region and a narrower late one, while deletions spanning the intervening layers are consistently more disruptive. This recurring pattern holds across models, datasets, and metrics, and contrasts with the middle-to-late redundancy commonly reported in the literature. (ii) Within these regions, we observe that the joint degradation from deleting two intervals is approximately the sum of their individual degradations, reducing the number of model evaluations for pruning search from $O(L^4)$ to $O(L^2)$, where L is the aggregator depth. (iii) We find that CKA provides a cheaper representation-based proxy for interval degradation, offering a practical trade-off between pruning quality and calibration cost. (iv) Closed-form linear calibration recovers accuracy after pruning without end-to-end retraining. A least-squares analysis shows that using a shared map for special and patch tokens generally incurs excess reconstruction loss, motivating token-aware recovery. Recovery maps fitted on just 100 calibration scenes generalize to held-out scenes and unseen datasets. The resulting models reduce aggregator parameters by up to 44% while maintaining accuracy comparable to their intact counterparts. All code is provided in the supplementary materials and will be released publicly.

1 INTRODUCTION

Feed-forward geometry models jointly recover camera poses and dense 3D structure from multi-view images in a single forward pass. VGGT (Wang et al., 2025) and its successors, including π^3 (Wang et al., 2026c) and VGGT-Ω (Wang et al., 2026a) (collectively, the *VGGT family*), have established a common framework for this task, demonstrating performance gains from scaling backbone depth and width. Yet, these benefits do not necessarily reveal *how much of a trained model's depth is actually needed at inference*. Understanding layer redundancy can therefore inform the design of shallower and more efficient models and complement existing acceleration methods such as token pruning or merging (Shen et al., 2025) and post-training quantization (Feng et al., 2025).

Studies of layer redundancy in LLMs often identify removable computation in *middle-to-late* layers (Gromov et al., 2025; Sun et al., 2025; He et al., 2024), although the inferred pattern can change substantially with the calibration criterion (Kim et al., 2026). Analyses of some ViTs also report feature stabilization in deeper blocks (Zhou et al., 2021; Barreiro et al., 2026), while cross-layer representation patterns vary with architecture and pretraining strategy (Park & Kim, 2022; Xie et al., 2023; Walmer et al., 2023). These findings provide useful starting points but offer no universal prescription for where to remove layers. This motivates us to ask: *Does the VGGT family exhibit consistent patterns of layer redundancy across models, datasets, and metrics, and if so, how can these patterns guide layer pruning and post-pruning recovery?*

We systematically study redundancy by removing contiguous intervals from the multi-view aggregator, which accounts for the largest share of parameters in each model. For an aggregator with L layers, we evaluate all $\binom{L+1}{2}$ single-interval deletions, covering 3,018 configurations across the three backbones. We measure their effects on camera trajectories and dense reconstruction across four indoor and outdoor datasets. The resulting *interval-deletion landscapes* reveal where layers can be removed with negligible loss of accuracy and provide the basis for studying joint deletions, efficient pruning selection, and calibration-based recovery. Our analysis yields four findings, summarized in Fig. 1:

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.

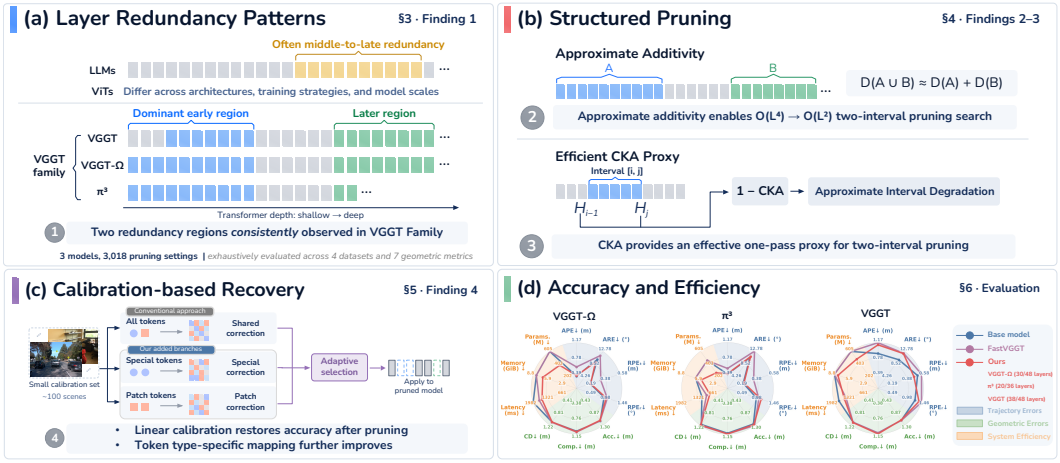

Figure 1: **Overview of our analysis and pruning framework.** (a) Layer-redundancy patterns in the VGGT family. (b) Structured pruning based on approximate additivity and CKA. (c) Calibration-based recovery after pruning. (d) Accuracy and efficiency of the resulting compressed models.

- The VGGT family exhibits two separated redundancy regions: a broad early region and a narrower late one. This structure is remarkably consistent across models, datasets, and evaluation metrics, motivating a restriction of the pruning search space from arbitrary layer combinations to configurations comprising up to two separated intervals.
- Degradation is approximately additive for deletions within the two redundancy regions. Estimating joint degradation by summing individual-interval degradations reduces the required model evaluations for pruning search from $O(L^4)$ to $O(L^2)$. The resulting two-interval strategy preserves accuracy better than single-interval pruning and the compared methods in nearly all settings.
- Centered Kernel Alignment (CKA) (Kornblith et al., 2019) provides a practical proxy for pruning search using one intact-model forward pass per calibration input. It generally performs best among the compared proxies, offering a practical trade-off between calibration cost and pruning quality.
- Closed-form linear calibration effectively recovers accuracy after pruning without end-to-end training. A least-squares analysis reveals that special and patch tokens generally benefit from separate corrections, motivating token-aware calibration, which provides further accuracy gains.

Using only 100 calibration scenes without end-to-end retraining, the resulting models maintain accuracy comparable to their intact counterparts while removing up to 44% of aggregator layers, substantially reducing parameter counts and inference latency. The approach generalizes well across scenes and sequence lengths and remains complementary to token reduction and quantization.

2 Probing Layer Redundancy in the VGGT Family

2.1 Models and Layer Redundancy

We study the multi-view *aggregators* of pretrained VGGT (Wang et al., 2025), π^3 (Wang et al., 2026c), and VGGT-Ω (Wang et al., 2026a), which account for the largest share of parameters in each model. A *layer* is one complete transformer block, including its attention and feed-forward sublayers; within-view and cross-view blocks are counted separately. *Depth* denotes the number of aggregator layers, with $L = 48$, 36, and 48 for VGGT, π^3, and VGGT-Ω, respectively. We assess *layer redundancy* through the effect of layer removal on geometric prediction accuracy under a specified evaluation setting. For an intact model F, F_{-I} denotes the model obtained by bypassing the blocks indexed by $I \subseteq \{1, \ldots, L\}$ while retaining the image encoder and prediction heads. A contiguous interval of layers is denoted by $I = [i, j] = \{i, \ldots, j\}$, and the pruning budget is $b = |I|$.

2.2 Datasets and Evaluation Metrics

We use 30 frames per scene from the *indoor* datasets ScanNet (Dai et al., 2017) and 7Scenes (Shotton et al., 2013) and the *outdoor* datasets nuScenes (Caesar et al., 2020) and Waymo (Sun et al., 2020).

We use disjoint calibration and evaluation splits of 50 scenes each, except for 7Scenes, where we use the official splits. Trajectories are evaluated using absolute position and rotation errors (APE, ARE) and relative translation and rotation errors (RPE_t, RPE_r); dense reconstruction is evaluated using accuracy (Acc.), completeness (Comp.), and Chamfer distance (CD). All metrics are *lower-is-better*, with translation and reconstruction errors measured in metres and rotation errors in degrees.

2.3 Measuring geometric degradation

For a fixed model and dataset, let $P_m(\cdot)$ denote the error under metric m, averaged over the scenes in the relevant split. We define the degradation caused by deleting I as $D_m(I) = P_m(F_{-I}) - P_m(F)$, where positive values indicate degradation. This quantity measures a change in error relative to ground truth, rather than a representation distance or a deviation from the intact model's outputs. Because the seven metrics differ by orders of magnitude, we normalize their degradation before aggregating them into *a single scalar score* that measures the degradation caused by deleting I. For a set of evaluation metrics $\mathcal{M}$, we define $D_{\mathcal{M}}(I) = |\mathcal{M}|^{-1} \sum_{m \in \mathcal{M}} D_m(I)/\alpha_m$, where $|\mathcal{M}|$ is the number of metrics and α_m is the largest absolute degradation over all single-interval deletions. For each model and dataset, these normalization scales are computed on the calibration split and held fixed across pruning budgets and evaluation splits. When multiple datasets are considered, we use $D_{\mathcal{M}}$ to denote the equally weighted average of their individually normalized scores. Appendix J reports the unnormalized raw degradation for all seven metrics.

3 Where Are Layers Redundant?

For each model and dataset, we evaluate all $\binom{L+1}{2}$ single-interval deletions on the calibration split. In Fig. 2, each entry of the resulting *interval-deletion landscape* records $D_{\mathcal{M}}([i, j])$, the normalized degradation after bypassing blocks i through j. Each panel averages normalized degradation equally across its two datasets. Indexing by both endpoints distinguishes removal location from interval length: intervals with a fixed pruning budget b form a diagonal slice, $D_{\mathcal{M}}([s, s + b - 1])$.

3.1 Two redundancy regions of different sizes

Fig. 2 shows two separated regions of low degradation: a broad early region and a narrower late region. Intervals that span the intervening blocks are consistently more disruptive. The early region tolerates long contiguous removals, forming a broad square of low degradation in the mirrored heatmap. The late region forms a narrower band near the diagonal, indicating greater sensitivity to the number of consecutively removed layers. These observations suggest that a substantial portion of early aggregation after the image encoder can be skipped, whereas fewer consecutive layers can be removed near the output. The dominance of early redundancy contrasts with the middle-to-late redundancy often reported in LLMs (Men et al., 2025; Gromov et al., 2025; Sun et al., 2025; He et al., 2024) and the high similarity between deep-layer representations observed in some ViT studies (Zhou et al., 2021; Barreiro et al., 2026). The two-region structure motivates restricting the layer-pruning search space from arbitrary layer combinations to configurations comprising up to two separated intervals. The joint effect of two intervals is examined in § 4.

3.2 Shared structure across models, datasets, and metrics

The two-region structure is shared across all three models and remains stable across indoor and outdoor datasets and trajectory and geometry metrics, despite substantial architectural and training differences within the VGGT family. Relative to VGGT, π^3 removes camera and register tokens, shortens the aggregator, and uses two-stage training (Wang et al., 2026c); VGGT-Ω introduces register attention, replaces DINOv2 (Oquab et al., 2024) with DINOv3 (Siméoni et al., 2025) in the image encoder, and incorporates large-scale unlabeled data into its three-stage training (Wang et al., 2026a). Only minor variations appear within this shared pattern. The early redundancy region is slightly shorter in VGGT than in π^3 and VGGT-Ω, while the late region is slightly broader under trajectory metrics than under geometry metrics. Despite these differences, all three models retain the overall two-region pattern, with similar locations and relative strengths. This consistency contrasts with the dependence of LLM redundancy on calibration objectives (Kim et al., 2026) and with variations in ViT representation similarity across architectures (Park & Kim, 2022), pretraining strategies (Xie et al., 2023), and model scales (Walmer et al., 2023). It supports structured pruning across scenes and datasets without implying that redundancy is independent of data or evaluation objectives.

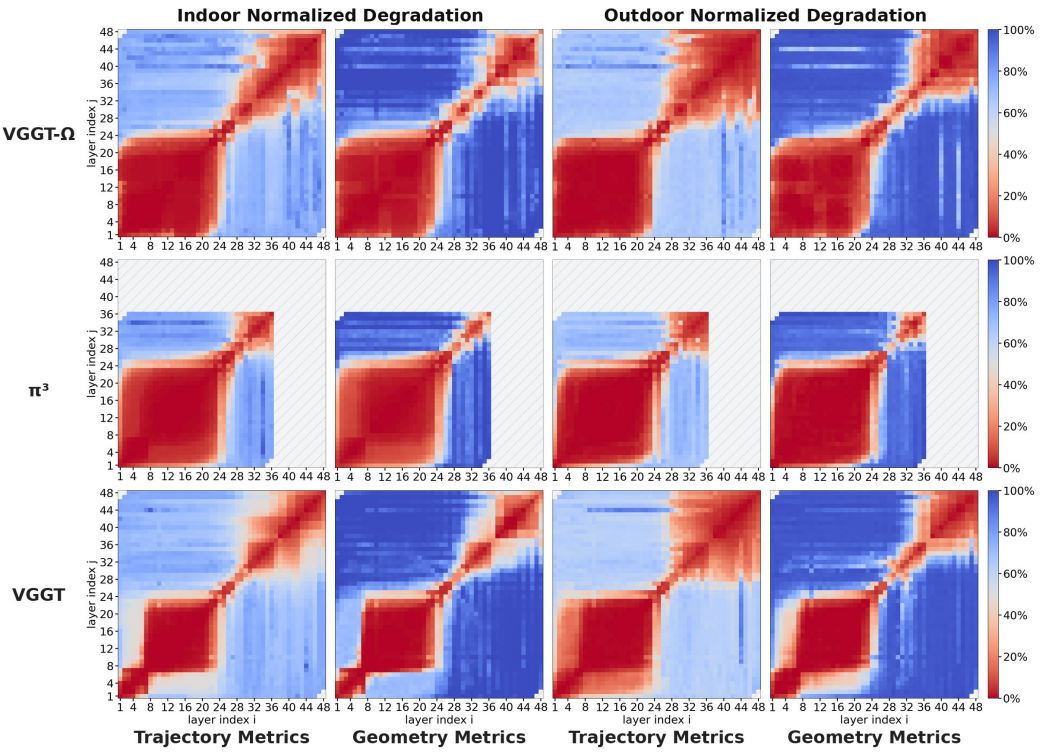

Figure 2: **Two separated low-degradation regions consistently appear across models, domains and metrics, with the dominant region occurring early in the aggregator.** Each cell reports normalized degradation after deleting the contiguous interval $[i, j]$ of aggregator layers. Scores are mirrored across the diagonal, with π^3 aligned to the first 36 layers and the remaining 12 positions hatched. Trajectory and geometry panels average over their four and three metrics, respectively. Indoor and outdoor panels average equally over ScanNet/7Scenes and nuScenes/Waymo, respectively.

Finding I: *The VGGT family exhibits two separated redundancy regions in the aggregator: a broad early region and a narrower late region. This structure is remarkably consistent across models, datasets, and evaluation metrics.*

4 How Can the Two-Region Structure Guide Pruning?

The observed two-region structure motivates restricting the layer-pruning search space from arbitrary layer combinations to configurations comprising up to two separated contiguous intervals. Even under this restriction, exhaustive evaluation remains expensive. Across all pruning budgets, there are $\binom{L+1}{2}$ single-interval configurations and $\binom{L+1}{4}$ two-interval configurations with at least one retained layer between the intervals. This gives $\binom{L+1}{2} + \binom{L+1}{4} = O(L^4)$ unique non-empty configurations, or 213,052 candidates for $L = 48$. At an illustrative cost of one second per candidate per scene for inference and evaluation, exhaustive evaluation would require approximately 5,920 GPU hours just for a 100-scene calibration set. We therefore approximate candidate degradation to efficiently explore the performance–compression frontier across budgets.

4.1 Approximate Additivity of Interval-Wise Degradation

Let A and B denote two separated intervals. The marginal raw degradation measures the incremental effect of removing B after A has been removed: $D_m(B \mid A) = P_m(F_{-(A\cup B)}) - P_m(F_{-A})$. Under the same normalization and aggregation, $D_{\mathcal{M}}(A \cup B) = D_{\mathcal{M}}(A) + D_{\mathcal{M}}(B \mid A)$ holds exactly. We observe approximate additivity: $D_{\mathcal{M}}(B \mid A) \approx D_{\mathcal{M}}(B)$, meaning that after removing A, the marginal degradation landscape for B remains close to the original landscape. Fig. 3 illustrates this agreement using scores averaged over all four datasets: mean absolute error is 0.02–0.05, with Pearson and Spearman correlations of at least 0.97, indicating close agreement in both values and candidate rankings. We examine when this approximation holds in Appendix B.

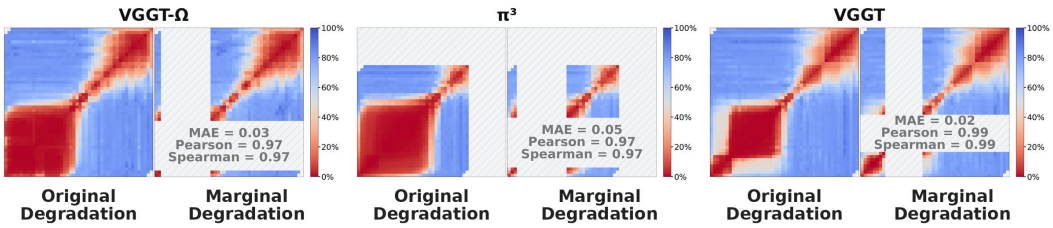

Figure 3: **Approximate additivity of interval-wise degradation.** For each backbone, the intact-model landscape is compared with the marginal landscape after removing an interval A within the early redundancy region. The marginal normalized degradation $D_{\mathcal{M}}(B \mid A)$ closely matches the original $D_{\mathcal{M}}(B)$ in both values and rankings.

This approximation lets us estimate joint degradation as $\widetilde{D}_{\mathcal{M}}(A \cup B) = D_{\mathcal{M}}(A) + D_{\mathcal{M}}(B)$ using only single-interval results. It reduces the required model evaluations for two-interval pruning search from $O(L^4)$ to $O(L^2)$, while two-interval candidates are scored without additional model evaluations. For $L = 48$, the number of candidate evaluations falls from 213,052 to 1,176, reducing the illustrative cost for 100 calibration scenes from approximately 5,920 hours to less than 33 hours.

We next evaluate whether this approximation supports effective pruning across budgets. For budget b, let $\mathcal{C}_b$ contain all configurations of one or two separated intervals removing exactly b layers. For each backbone, we use only the ScanNet and nuScenes calibration sets to select $\widetilde{I}_b = \arg\min_{I \in \mathcal{C}_b} \widetilde{D}_{\mathcal{M}}(I)$, using measured degradation for single intervals and the additive estimate for two intervals. We compare this selection with a search restricted to single intervals. Each configuration is then applied unchanged to their disjoint held-out sets for *same-dataset generalization* and the unseen datasets 7Scenes and Waymo for *cross-dataset generalization*. We also compare against ShortGPT (Men et al., 2025), Gardener (Xiang et al., 2026), ReplaceMe (Shopkhoev et al., 2025), and SIMPLER (Barreiro et al., 2026), with implementation details provided in Appendix C.

Fig. 4 shows that allowing two intervals generally yields lower degradation than single-interval selection at larger budgets, when selected configurations begin to draw on both redundancy regions. Gains are strongest for VGGT and VGGT-Ω, whose second redundancy regions are larger, and more modest for π^3, whose second region is much smaller (Fig. 2). Our pruning selection achieves the lowest degradation in nearly all evaluated settings, supporting approximate additivity as a practical basis for pruning selection. Configurations selected on the calibration sets also perform well on held-out and unseen datasets, supporting their transferability and generalization.

Finding II: *Degradation is approximately **additive** for deletions within the two redundancy regions, enabling effective two-interval pruning search with $O(L^2)$ model evaluations.*

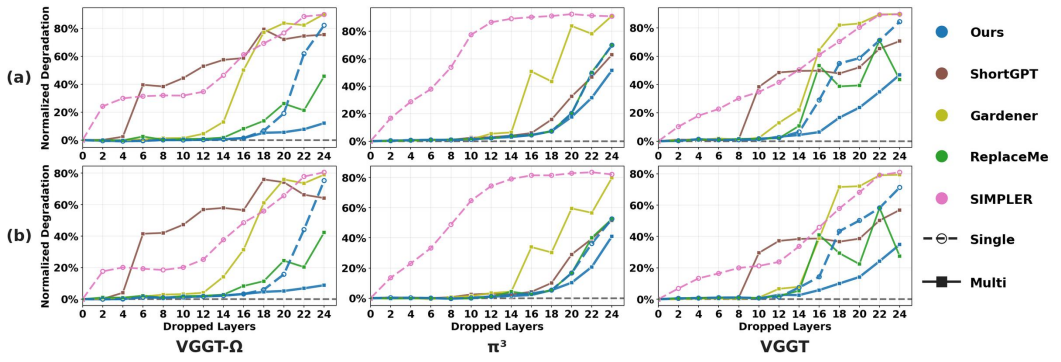

Figure 4: **Pruning quality and generalization.** For each method and budget, one pruning configuration is selected based only on the combined 100-scene calibration set from ScanNet and nuScenes and applied unchanged to: (a) the held-out evaluation sets, and (b) the unseen 7Scenes and Waymo datasets. "Single" restricts removal to one contiguous interval, whereas "Multi" allows multiple intervals according to each method's search space; our method allows up to two separated intervals.

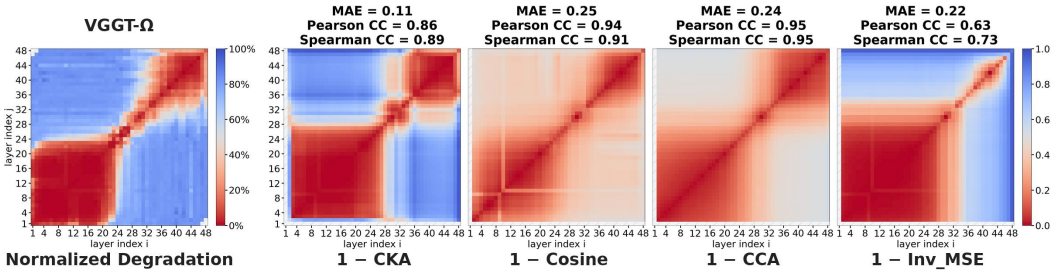

Figure 5: **Representation-based proxies versus measured normalized degradation on VGGT-Ω.** CKA achieves the lowest MAE, while cosine similarity and CCA yield higher correlations.

4.2 One-Forward-Pass Representation Proxies

We further examine whether single-interval degradation admits a cheaper representation-based proxy. Let H_t denote the representations after layer t, with H_0 denoting the aggregator input. For each interval $I = [i, j]$, we compare its input H_{i-1} and output H_j using CKA (Kornblith et al., 2019), cosine similarity, CCA, and inverse MSE. Each measure q is normalized to produce a similarity score $s_q(i, j) \in [0, 1]$, giving the degradation proxy $\widetilde{D}_q(I) = 1 - s_q(i, j)$.

Fig. 5 compares these proxies with measured normalized degradation for VGGT-Ω; results for the other two backbones appear in the appendix (Fig. 11). All four broadly reflect the early and later redundancy regions, but differ in their boundaries and agreement with measured degradation. Cosine similarity and CCA achieve Pearson and Spearman correlations above 0.90, with MAEs around 0.25. CKA has slightly lower correlations around 0.9, yet achieves the lowest MAE of 0.12. Inverse MSE shows weaker correlations and an MAE of 0.22.

We then replace measured single-interval degradation with each proxy, keeping the pruning search space and evaluation protocol unchanged. For two separated intervals $A = [i, j]$ and $B = [k, l]$, we use the additive estimate $\widetilde{D}_q(A \cup B) = \widetilde{D}_q(A) + \widetilde{D}_q(B) = 2 - s_q(i, j) - s_q(k, l)$. Thus, representation-based selection approximates both single-interval degradation and the joint effect of two deletions. Fig. 6 shows that CKA is not always the best criterion for single-interval selection, but generally performs best among the four proxies when allowing up to two intervals. It closely matches our measured-degradation criterion on VGGT-Ω and π^3, although a gap emerges on VGGT beyond 16 removed layers. CKA's lower MAE on VGGT-Ω may help explain its effectiveness compared with cosine similarity and CCA under additive scoring, where numerical agreement with measured degradation matters beyond correlation alone. Further details are provided in Appendix D.

CKA itself requires only one intact-model forward pass per calibration input, followed by $O(L^2)$ low-cost similarity computations on cached features. For a 48-layer aggregator and 100 calibration scenes, this reduces calibration from approximately 33 GPU hours under the preceding cost assumption to less than one hour, offering a practical trade-off between calibration cost and pruning quality.

Finding III: *CKA provides a practical proxy for effective two-interval pruning, requiring only one intact-model forward pass per calibration input.*

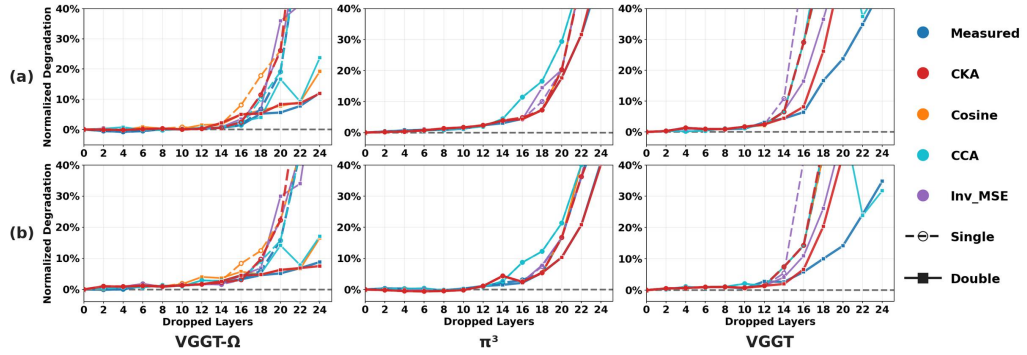

Figure 6: **Pruning quality and generalization with representation-based proxies.** The protocol and row definitions follow Fig. 4.

5 What Can Calibration Recover?

Removing an interval creates a representation mismatch at its output boundary. A natural response is to use knowledge distillation (Muralidharan et al., 2024) or parameter-efficient fine-tuning (Ma et al., 2023). However, these approaches still require iterative training, with recovery quality depending on the available training data and compute. We instead investigate whether lightweight mappings fitted on a small calibration set can recover accuracy without end-to-end retraining. We identify a limitation of shared linear calibration in the VGGT family: heterogeneous token types may require different recovery mappings. We theoretically characterize the resulting reconstruction penalty and develop an adaptive token-type-aware recovery method. Experiments show that separate mappings with a shared fallback improve recovery on VGGT and VGGT-Ω.

5.1 Why Distinguish Token Types?

VGGT and VGGT-Ω contain camera and register tokens serving global or auxiliary roles, alongside patch tokens representing dense visual features. We refer to camera and register tokens collectively as *special tokens*. Patch tokens substantially outnumber special tokens and can therefore dominate a reconstruction objective that weights all tokens equally. Beyond this imbalance, the two token types may require *different* recovery mappings. In that case, a shared mapping must compromise between them, even when their contributions to the objective are balanced.

We quantify this compromise by comparing the best shared linear mapping with the best separate mappings. For a fixed interval and calibration context, let $r \in \{s, p\}$ index special and patch tokens, respectively, and let x_r and y_r denote input and target output row vectors. Define the expected reconstruction loss, its optimal linear mapping, and the input second-moment matrix as

$$L_r(W) = \mathbb{E}\|x_r W - y_r\|_2^2, \qquad T_r = \arg\min_W L_r(W), \qquad \Sigma_r = \mathbb{E}[x_r^\top x_r], \tag{1}$$

where the expectation is over input–target pairs of token type r. Assume finite second moments and $\Sigma_r \succ 0$. Let $\pi_s, \pi_p > 0$ with $\pi_s + \pi_p = 1$ weight the two token types, and set $Q_r = \pi_r \Sigma_r$ and $\Delta_T = T_s - T_p$. The excess reconstruction loss of the best shared mapping over separate mappings is

$$\min_W \sum_{r\in\{s,p\}} \pi_r L_r(W) - \sum_{r\in\{s,p\}} \pi_r L_r(T_r) = \operatorname{tr}\left[\Delta_T^\top (Q_s^{-1} + Q_p^{-1})^{-1} \Delta_T\right]. \tag{2}$$

This penalty is strictly positive whenever $T_s \neq T_p$, even with balanced weights $\pi_s = \pi_p = 1/2$. Thus, balancing the contributions of the two token types does not eliminate the reconstruction penalty of a shared mapping. This result concerns unregularized population reconstruction loss and motivates type-specific recovery. The proof of equation 2 is provided in Appendix F.

5.2 Adaptive Token-Type-Aware Recovery

For each interval $[i, j]$, let $X_{\text{all}} = \text{cat}(X_s, X_p)$ and $Y_{\text{all}} = \text{cat}(Y_s, Y_p)$ denote its input and output activation matrices, concatenated along the token dimension. We fit mappings $W_g = \mathbf{I} + \Delta_g$ for $g \in \{\text{all}, s, p\}$, where $\mathbf{I}$ is the identity matrix. Following the residual least-squares formulation of Ghost (Yun et al., 2026), each fit minimizes $\|X_g W_g - Y_g\|_F^2 + \lambda \|W_g - \mathbf{I}\|_F^2$ by solving

$$(X_g^\top X_g + \lambda \mathbf{I})\Delta_g = X_g^\top (Y_g - X_g), \qquad \lambda = 10^{-6}. \tag{3}$$

The type-specific recovery applies $Y'_{\text{all}} = \text{cat}(X_s W_s, X_p W_p)$ with tokens restored to their original positions. Intervals are processed sequentially from shallow to deep. For each interval, input activations are collected from the model with all preceding pruning and recovery operations applied, while target output activations are collected from the intact model.

For VGGT and VGGT-Ω, special-token fits use only 5 and 17 tokens per frame, respectively, far fewer than patch-token fits. In our experiments, the special-token Gram matrix $X_s^\top X_s$ sometimes exhibits a condition number on the order of 10^{16}, indicating severe ill-conditioning in the unregularized fitting problem. We therefore use the relative deviation $\rho = \|W_s - W_{\text{all}}\|_F / \|W_{\text{all}} - \mathbf{I}\|_F$ as a heuristic safeguard. This score measures the departure of the special-token mapping from the shared mapping relative to the magnitude of the shared correction. We use the type-specific mappings when $\rho < 20$ and otherwise apply W_{all} to all tokens. Since π^3 has only patch tokens, it uses one mapping per interval without adaptive selection.

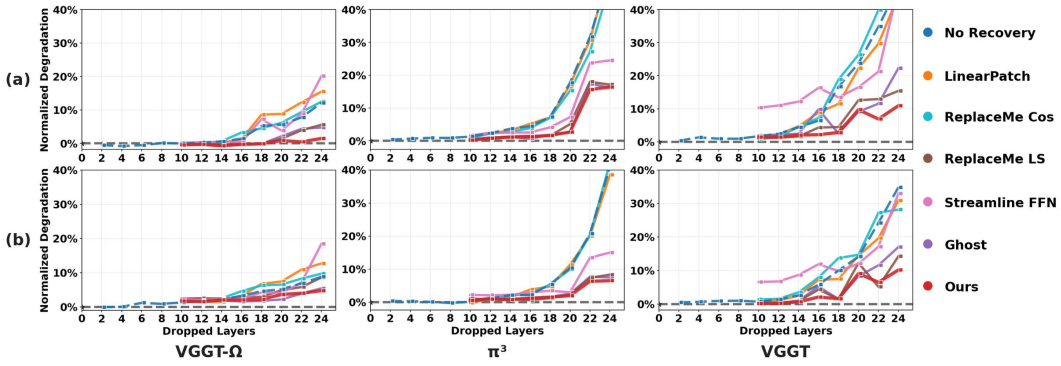

Figure 7: **Post-pruning recovery and generalization.** Recovery is fitted on 100 ScanNet+nuScenes calibration scenes and evaluated on (a) held-out and (b) unseen 7Scenes+Waymo data.

Fig. 7 compares our method with LinearPatch (Chen et al., 2025b), ReplaceMe (Shopkhoev et al., 2025), Streamline (Chen et al., 2025a), and Ghost (Yun et al., 2026), using pruning configurations selected by our measured-degradation criterion and 100 calibration scenes from ScanNet and nuScenes. For each recovery method, we report the better result between single- and two-interval pruning. Implementation details are in Appendix E. Streamline and ReplaceMe-Cosine often fail to improve over direct removal, whereas Ghost and ReplaceMe-LS substantially reduce degradation. Our adaptive token-type-aware recovery further improves VGGT and VGGT-Ω in nearly all evaluated settings. These gains persist on both held-out scenes and unseen datasets, showing that 100 calibration scenes suffice to fit recovery transformations that generalize beyond the calibration data.

Finding IV: *Linear calibration effectively recovers accuracy after pruning, with adaptive token-type-aware mappings providing further gains.*

6 From Findings to Compressed Models

Compression baselines. We combine our findings to construct final compressed models and evaluate their accuracy and efficiency under a deployment-oriented protocol. For each backbone, we use the calibration splits from all four datasets to select the pruning configuration and fit the recovery transformations, then freeze the model and evaluate it on their disjoint held-out splits. We compare against ReplaceMe (Shopkhoev et al., 2025), which also combines layer removal with linear recovery, at the same retained depth. We additionally compare and combine our method with FastVGGT (Shen et al., 2025) for token reduction and QuantVGGT (Feng et al., 2025) for quantization. QuantVGGT is evaluated only on VGGT, without latency or memory results, because its released calibration parameters are backbone-specific and its public implementation does not provide INT4 deployment.

Accuracy and efficiency. Table 1 and Fig. 8 provide quantitative and qualitative comparisons, respectively. For each backbone, we report a high pruning budget that preserves accuracy close to the intact model; at these budgets, measured-degradation and CKA-based selection choose the same pruning configuration. Removing 18/48, 16/36, and 10/48 aggregator blocks from VGGT-Ω, π^3, and VGGT reduces aggregator parameters by 20.8–44.5%, latency by 14.2–30.9%, and memory by 6.8–18.5% on 30-frame inputs, where parameter counts refer to the aggregator only, while latency and peak allocated GPU memory are measured over the full forward pass. In comparison, the token reduction and quantization methods considered here retain the original parameter count. These savings come with modest accuracy changes overall. Averaged equally across the three backbones and the absolute and relative metrics within each category, translation and rotation errors increase by approximately 0.04 m and 0.32°, respectively, while CD increases by 0.01 m. Our method outperforms ReplaceMe across almost all metrics and backbones at matched retained depths. Compared with FastVGGT and QuantVGGT, our method achieves the best accuracy in 18 of the 21 backbone–metric combinations.

The achievable compression is consistent with the deletion landscapes in Fig. 2. VGGT-Ω and π^3 exhibit broad early redundancy regions and tolerate removing 18 and 16 blocks, respectively. VGGT has a narrower early region and exhibits higher trajectory degradation even after removing only 10 blocks. Adding our layer pruning to FastVGGT further reduces latency while maintaining broadly comparable accuracy to FastVGGT alone. This supports their complementary effects on model depth and token computation, allowing both techniques to be used together.

Table 1: **Comparison averaged over the held-out splits of ScanNet, 7Scenes, nuScenes, and Waymo.** Bold error values indicate the best results among compressed variants of each backbone.

Method	Layers	Trajectory Errors				Geometric Errors			System Efficiency		
		APE ↓ (m)	ARE ↓ (°)	RPE_t ↓ (m)	RPE_r ↓ (°)	Acc. ↓ (m)	Comp. ↓ (m)	CD ↓ (m)	Params ↓ (M)	Latency ↓ (s)	Memory ↓ (GiB)
VGGT-Ω	48	0.31	9.99	0.18	0.93	1.15	1.08	1.11	605	1.98	8.2
+ FastVGGT	48	0.57	11.44	0.33	1.04	1.15	1.10	1.13	605	$1.70_{\downarrow 14.0\%}$	8.2
+ ReplaceMe	**30**	0.96	23.46	0.53	4.59	1.38	1.30	1.34	$378_{\downarrow 37.5\%}$	$1.57_{\downarrow 20.9\%}$	$6.9_{\downarrow 15.9\%}$
+ **Ours**	**30**	**0.33**	**9.07**	**0.19**	**1.00**	**1.13**	**1.09**	**1.11**	$378_{\downarrow 37.5\%}$	$1.58_{\downarrow 20.4\%}$	$6.9_{\downarrow 15.9\%}$
+ Ours + FastVGGT	**30**	0.64	11.24	0.38	1.15	1.14	**1.09**	1.12	$378_{\downarrow 37.5\%}$	$1.39_{\downarrow 29.5\%}$	$6.9_{\downarrow 15.9\%}$
π^3	36	0.36	10.59	0.21	0.96	1.25	1.12	1.18	454	1.16	6.5
+ FastVGGT	36	0.51	12.58	0.35	**1.05**	1.30	**1.14**	1.22	454	$0.95_{\downarrow 18.4\%}$	6.5
+ ReplaceMe	**20**	0.73	12.99	0.35	1.32	**1.27**	1.17	1.22	$252_{\downarrow 44.5\%}$	$0.79_{\downarrow 32.3\%}$	$5.3_{\downarrow 18.5\%}$
+ **Ours**	**20**	**0.37**	**10.92**	**0.22**	1.13	**1.27**	**1.14**	**1.20**	$252_{\downarrow 44.5\%}$	$0.80_{\downarrow 30.9\%}$	$5.3_{\downarrow 18.5\%}$
+ Ours + FastVGGT	**20**	0.66	12.05	0.46	1.20	1.33	1.15	1.24	$252_{\downarrow 44.5\%}$	$0.69_{\downarrow 40.9\%}$	$5.3_{\downarrow 18.5\%}$
VGGT	48	0.90	9.90	0.52	1.24	1.13	1.12	1.12	605	1.78	8.8
+ FastVGGT	48	1.14	12.06	0.56	1.46	1.13	**1.12**	**1.12**	605	$1.40_{\downarrow 18.8\%}$	8.8
+ QuantVGGT	48	1.12	14.25	0.79	1.56	1.16	**1.12**	1.14	605	–	–
+ ReplaceMe	**38**	1.23	12.27	0.51	**1.27**	**1.12**	1.14	1.13	$479_{\downarrow 20.8\%}$	$1.44_{\downarrow 19.2\%}$	$8.2_{\downarrow 6.8\%}$
+ **Ours**	**38**	1.12	**12.05**	**0.48**	1.38	**1.12**	1.15	1.13	$479_{\downarrow 20.8\%}$	$1.49_{\downarrow 14.2\%}$	$8.2_{\downarrow 6.8\%}$
+ Ours + FastVGGT	**38**	**1.06**	13.75	0.58	1.58	1.14	**1.12**	1.13	$479_{\downarrow 20.8\%}$	$1.23_{\downarrow 28.9\%}$	$8.2_{\downarrow 6.8\%}$
+ Ours + QuantVGGT	**38**	1.34	16.04	0.68	1.57	1.15	1.17	1.16	$479_{\downarrow 20.8\%}$	–	–

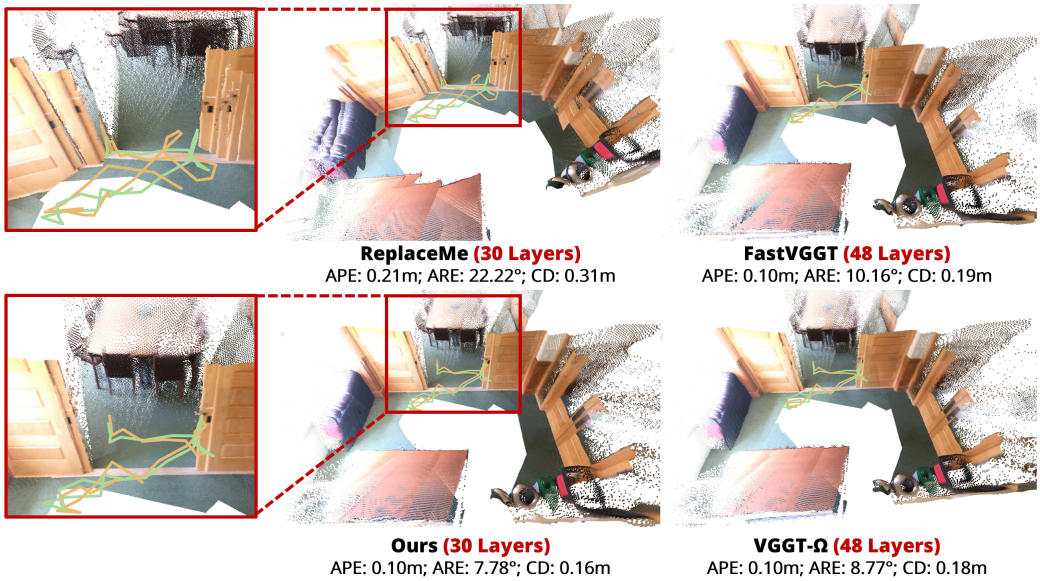

Figure 8: **Qualitative reconstruction comparison.** After removing 18 of 48 aggregator layers, ReplaceMe exhibits substantial geometric misalignment and trajectory errors, while our recovery preserves accurate geometry and trajectories, matching or even improving upon the intact VGGT-Ω.

7 Conclusion and Discussion

We identify two separated redundancy regions across the VGGT family, with a dominant early region and a narrower late one. This structure enables effective pruning: approximate additivity reduces the required model evaluations from $O(L^4)$ to $O(L^2)$, while a CKA proxy requires only one intact-model forward pass per calibration input. Together with adaptive token-type-aware recovery, the resulting compressed models substantially reduce parameter counts and inference latency while maintaining accuracy comparable to their intact counterparts across datasets and sequence lengths. The origin of this shared redundancy pattern remains unclear. Future work should distinguish among possible causes, including shared architecture design, multi-view geometric objectives, pretrained visual features, and optimization dynamics, ideally by tracking redundancy throughout training. Such understanding could guide adaptive depth allocation or the training of shallower models, extending redundancy analysis beyond post-training compression to reduce both training and inference costs.

AI USE STATEMENT

In this work, we used generative AI tools to assist with searching for information, implementing methods, adapting related work to our framework, analyzing experimental feedback, developing visualizations and dashboards, and drafting and polishing the manuscript. All AI-assisted text and code were reviewed by the authors, and the resulting implementations and visualizations were inspected and tested for correctness. We take responsibility for the final content of this work, including all text, claims, code, and other artifacts produced with the aid of AI.

REPRODUCIBILITY STATEMENT

We describe the proposed methodology, experimental protocols, datasets, evaluation metrics, and implementations of the compared methods in the main paper, with additional implementation and experimental details provided in the appendix. The supplementary material includes all code used in this work, complete text-based records of the experimental results, and an interactive dashboard for exploring and visualizing these results. Non-text intermediate artifacts are omitted to keep the supplementary package at a manageable size.

REFERENCES

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *Int. Conf. Learn. Represent.*, 2022.

Víctor Barreiro, Johannes Jakubik, Francisco Argüello, and Dora B. Heras. Simpler: Efficient foundation model adaptation via similarity-guided layer pruning for earth observation. In *Eur. Conf. Comput. Vis.*, 2026.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Yohann Cabon, Lucas Stoffl, Leonid Antsfeld, Gabriela Csurka, Boris Chidlovskii, Jerome Revaud, and Vincent Leroy. Must3r: Multi-view network for stereo 3d reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2025.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 11618–11628, 2020.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. Streamlining redundant layers to compress large language models. In *Int. Conf. Learn. Represent.*, 2025a.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Int. Conf. Comput. Vis.*, 2021.

Xinrui Chen, Haoli Bai, Tao Yuan, Ruikang Liu, Kang Zhao, Xianzhi Yu, Lu Hou, Tian Guan, Yonghong He, and Chun Yuan. A simple linear patch revives layer-pruned large language models. In *Adv. Neural Inform. Process. Syst.*, 2025b.

Yutian Chen, Yuheng Qiu, Ruogu Li, Jay Patrikar, and Sebastian Scherer. Co-me: Confidence guided token merging for visual geometric transformers. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026.

Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *Adv. Neural Inform. Process. Syst.*, 2023.

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 *Int. Conf. Learn. Represent.*, 2021.

Weilun Feng, Haotong Qin, Mingqiang Wu, Chuanguang Yang, Yuqi Li, Xiangqi Li, Zhulin An, Libo Huang, Yulun Zhang, Michele Magno, et al. Quantized visual geometry grounded transformer. *arXiv preprint arXiv:2509.21302*, 2025.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers. In *Int. Conf. Learn. Represent.*, 2025.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022.

Shwai He, Guoheng Sun, Zheyu Shen, and Ang Li. What matters in transformers? not all attention is needed. *arXiv preprint arXiv:2406.15786*, 2024.

Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *Int. Conf. Comput. Vis.*, 2021.

Cristian Hinostroza, Rodrigo Toro Icarte, Christ Devia, Andres Carvallo, Eugenio Herrera-Berg, Denis Parra, and Jorge Silva. Rethinking layer relevance in large language models beyond cosine similarity. In *Int. Conf. Learn. Represent.*, 2026.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *Int. Conf. Learn. Represent.*, 2022.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. Phi-2: The surprising power of small language models. *Microsoft Research Blog*, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Minkyu Kim, Vincent-Daniel Yun, Youngrae Kim, Suin Cho, Woosang Lim, and Sunwoo Lee. Rethinking layer redundancy: Calibration matters more than search in llm depth pruning. *arXiv preprint arXiv:2604.24938*, 2026.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Int. Conf. Mach. Learn.*, 2019.

Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3d with mast3r, 2024.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Int. Conf. Comput. Vis.*, 2021.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. In *Adv. Neural Inform. Process. Syst.*, 2023.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect. In *Findings of ACL*, 2025.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. In *Adv. Neural Inform. Process. Syst.*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaa El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Zhizhen Pan, Hesong Wang, and Huan Wang. Qvggt: Post-training quantized visual geometry grounded transformer. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026.

Namuk Park and Songkuk Kim. How do vision transformers work? In *Int. Conf. Learn. Represent.*, 2022.

You Shen, Zhipeng Zhang, Yansong Qu, and Liujuan Cao. Fastvggt: Training-free acceleration of visual geometry transformer. *arXiv preprint arXiv:2509.02560*, 2025.

Dmitriy Shopkhoev, Ammar Ali, Magauiya Zhussip, Valentin Malykh, Stamatios Lefkimmiatis, Nikos Komodakis, and Sergey Zagoruyko. Replaceme: Network simplification via depth pruning and transformer block linearization. In *Adv. Neural Inform. Process. Syst.*, 2025.

Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2013.

Zhijian Shu, Cheng Lin, Tao Xie, Wei Yin, Ben Li, Zhiyuan Pu, Weize Li, Yao Yao, Xun Cao, Xiaoyang Guo, and Xiao-Xiao Long. Litevggt: Boosting vanilla vggt via geometry-aware cached token merging. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2020.

Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. In *Adv. Neural Inform. Process. Syst.*, 2025.

Xianbing Sun, Zhikai Zhu, Zhengyu Lou, Bo Yang, Jinyang Tang, Liqing Zhang, He Wang, and Jianfu Zhang. AVGGT: Rethinking global attention for accelerating VGGT. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026.

Zhenggang Tang, Yuchen Fan, Dilin Wang, Hongyu Xu, Rakesh Ranjan, Alexander Schwing, and Zhicheng Yan. Mv-dust3r+: Single-stage scene reconstruction from sparse views in 2 seconds. *arXiv preprint arXiv:2412.06974*, 2024.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Int. Conf. Mach. Learn.*, 2021.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Matthew Walmer, Saksham Suri, Kamal Gupta, and Abhinav Shrivastava. Teaching matters: Investigating the role of supervision in vision transformers. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2023.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2025.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026a.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 20697–20709, June 2024.

Weitian Wang, Lukas Meiner, Rai Shubham, Cecilia De La Parra, and Akash Kumar. Httm: Head-wise temporal token merging for faster vggt. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026b.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *Int. Conf. Learn. Represent.*, 2026c.

Peihao Xiang, Kaida Wu, and Ou Bai. Entropy reveals block importance in masked self-supervised vision transformers. *arXiv preprint arXiv:2602.03918*, 2026.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. SimMIM: A simple framework for masked image modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022.

Zhenda Xie, Zigang Geng, Jingcheng Hu, Zheng Zhang, Han Hu, and Yue Cao. Revealing the dark secrets of masked image modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2023.

Jianing Yang, Alexander Sax, Kevin J. Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3r: Towards 3d reconstruction of 1000+ images in one forward pass. In *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2025.

Yifei Yang, Zouying Cao, and Hai Zhao. Laco: Large language model pruning via layer collapse. *arXiv preprint arXiv:2402.11187*, 2024.

Jinhao You, Shuo Lyu, Zhuohang Lyu, Tanxuan Li, Zibo Zhao, Jiaxiang Hu, Kai Tang, and Yichen Guo. RegimeVGGT: Layer-wise spatially preserving redundancy removal for visual geometry grounded transformer. *arXiv preprint arXiv:2606.18439*, 2026.

Vincent-Daniel Yun, Junhyuk Jo, Sai Praneeth Karimireddy, and Sunwoo Lee. Ghosted layers: Unconstrained activation alignment for recovering layer-pruned llms. *arXiv preprint arXiv:2605.15491*, 2026.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arxiv:2410.03825*, 2024.

Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021.

A Related Work

A.1 3D Visual Geometry Transformers and Efficient Inference

DUSt3R (Wang et al., 2024) pioneered feed-forward 3D reconstruction by directly predicting dense pointmaps from image pairs, with subsequent works (Leroy et al., 2024; Yang et al., 2025; Cabon et al., 2025; Tang et al., 2024; Zhang et al., 2024) extending this paradigm to larger sets or dynamic scenes. VGGT (Wang et al., 2025) further advances this line of work by predicting camera parameters and dense geometry within a unified framework. Building on VGGT, π^3 (Wang et al., 2026c) removes the dependence on a reference view, while VGGT-Ω (Wang et al., 2026a) demonstrates predictable scaling with model capacity and data size. Their computational and memory requirements have motivated a range of efficiency methods. FastVGGT (Shen et al., 2025), Co-Me (Chen et al., 2026), HTTM (Wang et al., 2026b), and LiteVGGT (Shu et al., 2026) reduce computation through token pruning or merging. AVGGT and RegimeVGGT (Sun et al., 2026; You et al., 2026) exploit depth-dependent attention redundancy to reduce computation. QuantVGGT (Feng et al., 2025) and QVGGT (Pan et al., 2026) instead reduce numerical precision through post-training quantization. These approaches largely preserve parameter count; our work complements them by removing entire layers, reducing both parameter count and computation for inference and subsequent fine-tuning.

A.2 Layer Redundancy in Transformers

LLM studies often report greater redundancy in *middle-to-late* layers (Men et al., 2025; Gromov et al., 2025; Sun et al., 2025; He et al., 2024) across Llama (Touvron et al., 2023), Qwen (Bai et al., 2023), Mistral (Jiang et al., 2023), Phi-2 (Javaheripi et al., 2023), and DeepSeek (Bi et al., 2024). More recently, Kim et al. (2026) show that identified redundancy patterns can change substantially with the calibration objective in Llama and Qwen models, challenging the view of redundancy as an intrinsic property of a pretrained network. The ViT literature presents a more varied picture of cross-layer similarity. DeepViT (Zhou et al., 2021) and SIMPLER (Barreiro et al., 2026) report deeper-layer feature stabilization in supervised ViTs and in self-supervised ViT-MAE (He et al., 2022) models, respectively. However, broader analyses show that similarity patterns vary along several dimensions: architecture, across ViT (Dosovitskiy et al., 2021), PiT (Heo et al., 2021), and Swin (Liu et al., 2021), as analyzed by Park & Kim (2022); training strategy, across DeiT (Touvron et al., 2021), MoCo (Chen et al., 2021), and SimMIM (Xie et al., 2022) on a common ViT-B backbone, as examined by Xie et al. (2023); and model scale, within the MAE and BEiT (Bao et al., 2022) families, as studied by Walmer et al. (2023). Yet representational similarity alone does not establish functional redundancy, and whether these patterns extend to 3DVGTs remains unexplored.

A.3 Layer Pruning and Recovery in Transformers

Cosine-based representation similarity commonly guides layer pruning (Men et al., 2025; Gromov et al., 2025; Chen et al., 2025a; Muralidharan et al., 2024). LaCo (Yang et al., 2024) and ReplaceMe (Shopkhoev et al., 2025) compare cosine-based criteria with alternatives including CKA (Kornblith et al., 2019), KL divergence, L_2 distance, and CCA, favoring cosine in their respective settings. SIMPLER (Barreiro et al., 2026) uses CKA to locate representational stabilization and prunes subsequent layers before downstream adaptation. Beyond activation similarity, Gardener (Xiang et al., 2026) uses pretrained-weight entropy for data-free one-shot pruning. However, more recent work (Hinostroza et al., 2026) reports weak correlations between cosine similarity and pruning-induced degradation, while Kim et al. (2026) find that different search strategies often converge to similar solutions under a fixed calibration objective. Removing transformer layers can disrupt downstream representations and degrade performance. Natural solutions use additional training: Minitron (Muralidharan et al., 2024) employs knowledge distillation, while LLM-Pruner (Ma et al., 2023) and UIDL (Gromov et al., 2025) use LoRA (Hu et al., 2022) and QLoRA (Dettmers et al., 2023), respectively. Another line of work performs local recovery using calibration activations from the intact model, avoiding end-to-end retraining under the original task objective. LLM-Streamline (Chen et al., 2025a) trains a small replacement module to approximate the removed layers, while LinearPatch (Chen et al., 2025b) corrects activation-magnitude mismatch through Hadamard transformations and channel-wise scaling. ReplaceMe (Shopkhoev et al., 2025) absorbs a linear transformation into the preceding block's FFN output projection, whereas Ghost (Yun et al., 2026) directly maps the full boundary hidden state to the activation produced after each removed layer.

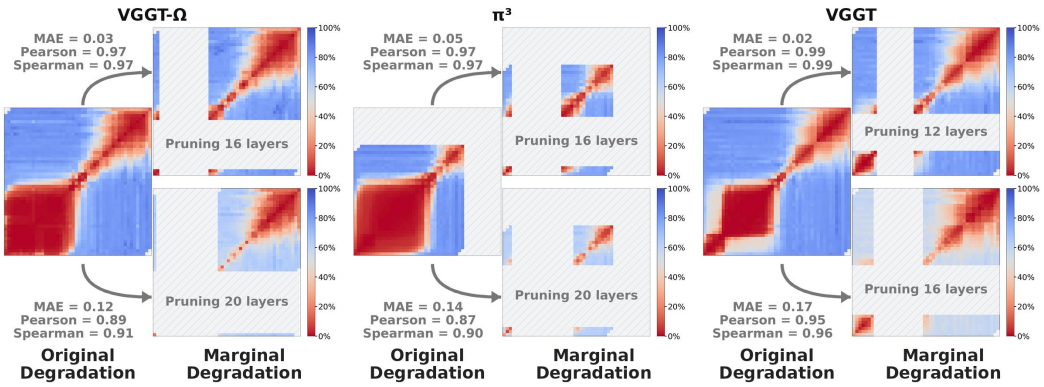

Figure 9: **Scope of approximate additivity.** Original degradation $D_{\mathcal{M}}(B)$ is compared with marginal degradation $D_{\mathcal{M}}(B \mid A)$ using the same normalization constants. Upper: shorter deletions within the early redundancy region largely preserve the original landscape. Lower: longer deletions reaching its boundary weaken the later redundancy region and reduce agreement.

B Scope of Approximate Additivity

Section 4.1 introduces the additive approximation and demonstrates its effectiveness for pruning selection. Here, we further investigate when it remains accurate and when it begins to break down. For each model, we select two early intervals A: a shorter interval lying well within the dominant redundancy region and a longer interval reaching its boundary. For each choice, we evaluate the marginal degradation $D_{\mathcal{M}}(B \mid A)$ for intervals B separated from A, using the calibration sets of all four datasets and the original normalization constants.

Fig. 9 compares the original and marginal degradation landscapes. For shorter deletions lying within the dominant region (upper row), the later low-degradation region remains largely unchanged, with MAE of 0.02–0.05 and Pearson and Spearman correlations of at least 0.97. For longer deletions reaching the early region's boundary (lower row), parts of the later region become less redundant. MAE increases to 0.12–0.17, while Pearson and Spearman correlations fall to as low as 0.87 and 0.90, respectively. These results indicate that sufficiently extensive early deletions alter the effect of subsequent removals, weakening the additive approximation.

These results suggest avoiding deletions that exhaust the dominant early redundancy region when using the additive approximation. Such extensive removals are usually unnecessary at moderate compression levels. At larger budgets, two-interval pruning can instead allocate additional removals to the later redundancy region. Fig. 10 illustrates this behavior: single-interval pruning must progressively extend the early deletion as the budget grows, whereas two-interval pruning keeps the early interval relatively stable and places additional removals in the later region. This both avoids the boundary regime where additivity becomes less accurate and helps explain the advantage of two-interval pruning at larger budgets.

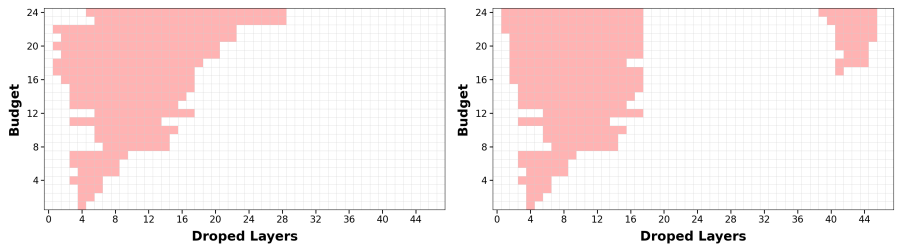

Figure 10: **Selected pruning intervals across budgets on VGGT-Ω.** Left: single-interval pruning progressively extends the early deletion as the budget grows. Right: guided by approximate additivity, our two-interval selection keeps the early interval relatively stable and allocates additional removals to the later redundancy region, avoiding excessive pruning near the early-region boundary.

C Pruning Implementation Details

Common protocol. We replace selected aggregator layers by identity operations and evaluate the resulting network. Every method uses the same ordered layer catalog and is capped at 24 removed layers. Reported curves evaluate budgets $b \in \{2, 4, \ldots, 24\}$. The similarity criteria within our framework use the token-region selection described in Appendix D. External baselines retain their original scoring and selection designs. For methods originally defined on LLM or ViT layers, we map their scoring and ranking rules to the individual aggregator layers of each backbone. Other adaptations are stated below.

Measured-degradation and similarity-based selectors. The measured-degradation selector evaluates all single-interval removals and computes their normalized degradation $D_{\mathcal{M}}$. The single-interval variant selects the interval with minimum normalized degradation at each budget.

The CKA, cosine, inverse-MSE, and CCA selectors use the same search procedure, replacing measured normalized degradation with the corresponding degradation proxy $\widetilde{D}_q([i,j]) = 1 - s_q(i,j)$. For selection allowing up to two intervals, a candidate contains either one interval or two non-overlapping intervals separated by at least one retained layer. The two-interval score is the sum of the constituent single-interval scores. For similarity-based selection, this gives

$$\widetilde{D}_q\big([i,j] \cup [k,l]\big) = 2 - s_q(i,j) - s_q(k,l). \tag{4}$$

ShortGPT Men et al. (2025). Following the original paper and official implementation, we use one-shot block-influence ranking. For layer ℓ, we rank its influence using

$$\mathrm{BI}_\ell = 1 - S_{\cos}(H_{\ell-1}, H_\ell). \tag{5}$$

Because $S_{\cos}$ is remapped to $[0,1]$, this score is a positive rescaling of the influence defined using unremapped cosine similarity and preserves the ranking. Layers are sorted once by ascending influence, equivalently descending input–output cosine similarity, with the lower layer index breaking ties. A budget-b configuration removes the first b layers in this ordering. ShortGPT does not constrain the number or location of the resulting intervals and does not recompute influence after a layer has been removed.

Data-free Gardener Xiang et al. (2026). For every aggregator layer, we concatenate all recursively named rank-two parameters, covering the attention projections and the two MLP linear layers. Following the released implementation, we compute exact-value weight-number entropy rather than histogram-bin entropy:

$$E_\ell = -\sum_v p_{\ell,v} \log p_{\ell,v}, \tag{6}$$

where $p_{\ell,v}$ is the frequency of the exact floating-point value v in layer ℓ. Layers are ranked once by ascending entropy, ties are resolved by layer index, and each pruning configuration is a prefix of this ranking. Entropy is not recomputed after pruning.

ReplaceMe Shopkhoev et al. (2025). We reproduce the released angular-distance scan. For an interval of length r starting at layer i, its score is $d_{\mathrm{ang}}(H_{i-1}, H_{i+r-1})$. Candidates are generated in start-index order, stably sorted by increasing distance, and greedily accepted if they do not overlap a previously selected interval. The single variant selects one interval. The double variant selects two equal-length intervals and therefore supports only even total pruning budgets. Unlike the separated-interval search in our framework, this baseline allows adjacent intervals, whose union forms a single contiguous removal. The mean of the selected distances is recorded for diagnostics but is not used as a joint optimization objective. Distances are not recomputed after the first interval is selected.

SIMPLER Barreiro et al. (2026). Let $\mathbf{S}$ denote its CKA similarity matrix over layer outputs. For a cutoff retaining layers $1, \ldots, c$, define the retained and pruned submatrices as

$$\mathbf{S}_{\mathrm{TL}} = \mathbf{S}_{1:c,\,1:c}, \qquad \mathbf{S}_{\mathrm{BR}} = \mathbf{S}_{c+1:L,\,c+1:L}, \tag{7}$$

where the index ranges are inclusive. We compute the mean absolute difference between consecutive rows, $\delta(\cdot)$, and record the cutoff score

$$g(c) = \delta(\mathbf{S}_{\mathrm{TL}}) - \delta(\mathbf{S}_{\mathrm{BR}}). \tag{8}$$

For a fixed budget b, suffix pruning uniquely determines the removed interval as $[L-b+1, L]$. We therefore report the full suffix-pruning curve rather than selecting a different-shaped interval using the cutoff score.

D Similarity Implementation and Visualization

Activation collection. Following the main text, we index the L aggregator layers from 1 to L in execution order, treating each complete frame or global transformer block as a separate pruning unit. Let A_0 denote the aggregator input and A_ℓ the output of layer ℓ. For a scene, $A_\ell \in \mathbb{R}^{B\times S\times P\times C}$, where B is the batch size, S the number of frames, P the number of tokens per frame, and C the channel dimension. We run the intact model once per calibration scene and retain the aggregator input and the output of every pruning unit. Similarities are computed on the GPU in FP32 after flattening the batch, frame, and token axes:

$$H_\ell = \mathrm{flat}(A_\ell) \in \mathbb{R}^{N\times C}, \qquad N = BSP. \tag{9}$$

Thus, H_0 denotes the flattened aggregator input, consistently with the main text. No token subsampling is used in the reported runs.

Token regions and aggregation. For VGGT and VGGT-Ω, we compute similarity matrices over three token regions: special tokens (camera and register tokens), patch tokens, and all tokens jointly. Matrices are computed independently per scene and then averaged equally across scenes within each dataset and across calibration datasets. For each similarity measure q, we consider the all-token matrix $\mathbf{S}_q^{\mathrm{all}}$ and the equally weighted special–patch matrix

$$\mathbf{S}_q^{\mathrm{s-p}} = \frac{1}{2}\left(\mathbf{S}_q^{\mathrm{special}} + \mathbf{S}_q^{\mathrm{patch}}\right). \tag{10}$$

For each measure, pruning variant, and budget, we derive one pruning configuration from each matrix. We evaluate both configurations on the calibration data and retain the one with lower measured normalized degradation, averaged over all seven metrics and the calibration datasets. The selected configuration is frozen for subsequent evaluation.

This selection applies to CKA, cosine similarity, CCA, and inverse MSE within our framework. External pruning baselines retain their original scoring and selection procedures. Since π^3 contains only patch tokens, it bypasses token-region splitting, averaging, and selection. Computing the similarity proxies requires one intact-model forward pass per calibration input. Token-region selection additionally evaluates the two resulting pruned models on the calibration data.

Similarity measures. For two flattened activation matrices $X, Y \in \mathbb{R}^{N\times C}$, we use the following implementations. Cosine similarity is the mean token-wise cosine, remapped from $[-1,1]$ to $[0,1]$:

$$S_{\cos}(X,Y) = \frac{1}{2}\left(1 + \frac{1}{N}\sum_{n=1}^{N}\frac{x_n^\top y_n}{(\|x_n\|_2+\epsilon)(\|y_n\|_2+\epsilon)}\right), \qquad \epsilon = 10^{-8}. \tag{11}$$

The angular distance used by ReplaceMe is

$$d_{\mathrm{ang}}(X,Y) = \frac{1}{N}\sum_{n=1}^{N}\frac{\arccos(\mathrm{clip}(\cos(x_n,y_n),-1,1))}{\pi}. \tag{12}$$

For linear CKA, we center each feature dimension, $\bar{X} = X - \mathbf{1}\mu_X^\top$ and $\bar{Y} = Y - \mathbf{1}\mu_Y^\top$, and compute

$$S_{\mathrm{CKA}}(X,Y) = \frac{\|\bar{X}^\top\bar{Y}\|_F^2}{\max\left(\sqrt{\|\bar{X}^\top\bar{X}\|_F^2\|\bar{Y}^\top\bar{Y}\|_F^2}, \epsilon\right)}. \tag{13}$$

Inverse MSE is

$$S_{\mathrm{invMSE}}(X,Y) = \left(1 + \frac{\|X-Y\|_F^2}{NC}\right)^{-1}. \tag{14}$$

Our CCA score is an SVCCA-style implementation: each centered activation is first projected onto at most 64 principal components, covariance matrices are regularized by $10^{-5}\mathbf{I}$, and the score is the mean canonical correlation, clipped to $[0,1]$.

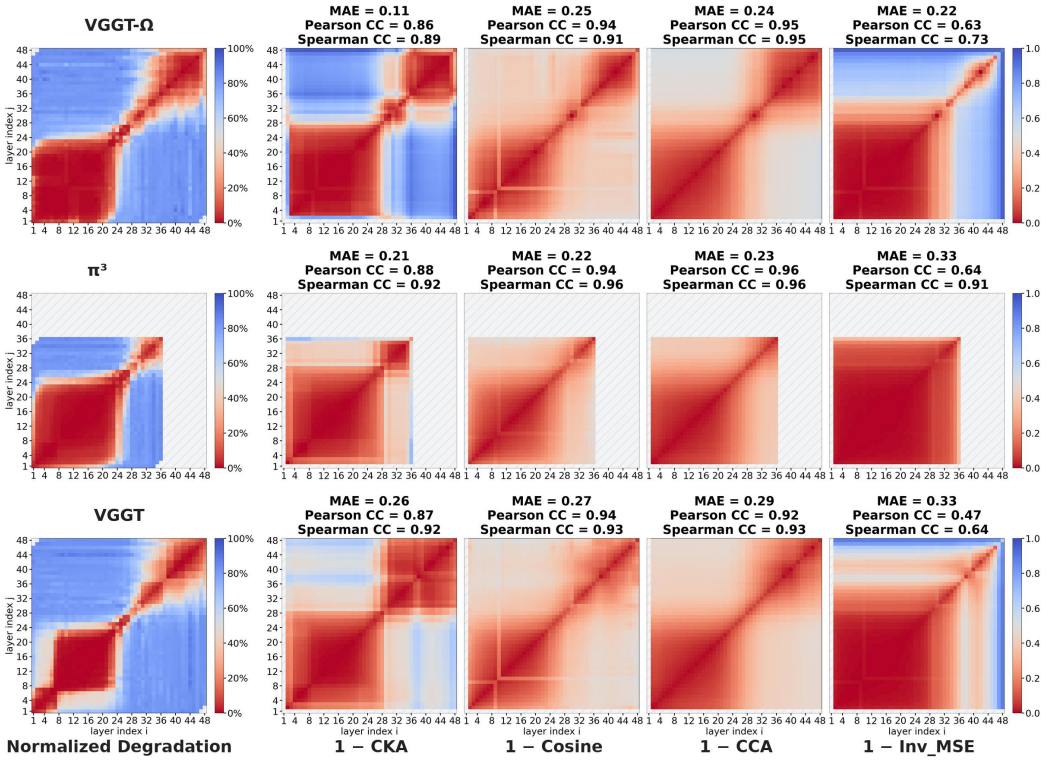

Figure 11: **Representation similarity versus measured normalized degradation.**

From activation similarity to an interval score. The uppercase notation $S_q(X, Y)$ denotes a similarity measure evaluated on two activation matrices, whereas $s_q(i, j)$ denotes the similarity score associated with pruning interval $[i, j]$. All interval indices are inclusive. Deleting $[i, j]$ bypasses layers $i, \ldots, j$, whose input and output representations are H_{i-1} and H_j, respectively. We define

$$s_q(i, j) = S_q(H_{i-1}, H_j), \qquad 1 \le i \le j \le L, \tag{15}$$

where $q \in \{\cos, \mathrm{CKA}, \mathrm{invMSE}, \mathrm{CCA}\}$. As in the main text, the corresponding degradation proxy is

$$\widetilde{D}_q([i, j]) = 1 - s_q(i, j). \tag{16}$$

Pairwise matrices are saved per scene and after aggregation, allowing the similarity-based methods to reuse the same cached activations and ordered layer catalog.

Similarity visualization. Fig. 11 compares these proxies $\widetilde{D}_q$ with directly measured normalized degradation D. All four measures capture the broad early and narrow later redundancy regions, although they differ from D in region boundaries, numerical values, trends, and rankings. CKA, cosine similarity, and CCA show strong agreement with measured degradation, while their relative performance varies across backbones and criteria. These results suggest representation similarity as a proxy for normalized degradation.

E Recovery Implementation Details

Baseline provenance and adaptation. The recovery baselines are ported from their original papers and released implementations. We preserve their objectives, operator parameterizations, and default hyperparameters unless an adaptation is explicitly described below. The batch, frame, and token axes of the collected activations are flattened into sample rows. Activation sources, fitting order, and deployment differ across methods and are specified individually below. Closed-form normal-equation statistics are accumulated in FP64. Fitted linear operators are stored and applied in FP32, and their outputs are cast back to the residual-stream dtype.

LinearPatch Chen et al. (2025b). Following the original formulation and its released reference implementation, LinearPatch uses a normalized Sylvester Hadamard basis Q. Our width-compatibility adaptation for non-power-of-two channel dimensions constructs the next larger Hadamard matrix, crops its leading $C \times C$ submatrix, and orthogonalizes it by QR decomposition.

For a single contiguous removal $[i, j]$, we collect $X = H_{i-1}$ and $Y = H_j$ from the intact model and fit one operator across the interval. For removals comprising multiple contiguous components, the implementation used in our experiments fits one operator per removed layer ℓ, using $X = H_{\ell-1}$ and $Y = H_\ell$ from the intact model. In each case, $X, Y \in \mathbb{R}^{N \times C}$. Channel scales and the resulting transform are

$$s_c = \frac{\sum_n |(YQ)_{nc}|}{\sum_n |(XQ)_{nc}| + 10^{-8}}, \qquad W = Q \,\mathrm{diag}(s) Q^\top. \tag{17}$$

Inference applies XW at the corresponding replacement location.

Ghost Yun et al. (2026). Following the official implementation, the baseline fits one matrix independently for each removed layer ℓ. Its source and target activations are $X = H_{\ell-1}$ and $Y = H_\ell$, collected from the intact model. It fits an identity-centered ridge regression to the boundary residual:

$$(X^\top X + \lambda \mathbf{I})\Delta = X^\top (Y - X), \qquad W = \mathbf{I} + \Delta, \tag{18}$$

with $\lambda = 10^{-6}$. Each removed layer is replaced by its fitted linear transformation. All matrices are fitted independently from the intact-model activation cache and applied in model order.

ReplaceMe-LS and ReplaceMe-Cosine Shopkhoev et al. (2025) ReplaceMe modifies the MLP residual of the source layer rather than adding an external boundary module. For a removed interval $[i, j]$, the source is layer $i - 1$ and the target is the output of layer j. Let H be the source layer output, R its MLP residual, and Y the target output. Replacing R by RA gives $H - R + RA$, so the desired transformed residual is $T = Y + R - H$. ReplaceMe-LS solves

$$(R^\top R + \alpha \mathbf{I})A = R^\top T, \tag{19}$$

with the default $\alpha = 0$. Each interval is fitted independently using activations from the intact model.

ReplaceMe-Cosine initializes $A = \mathbf{I}$ and minimizes $1 - \mathrm{mean}_n \cos((RA)_n, T_n)$ using Adam for 10 epochs, a row batch size of 1,024, learning rate 10^{-4}, and seed 0. For multiple intervals, it follows the released front-to-back pipeline: H, R, and Y for a later interval are collected from the current model after preceding replacements and pruning have been installed.

For both variants, the fitted transform is fused into the source MLP's second linear layer, including the appropriate LayerScale change of basis when present, and therefore introduces no runtime hook. The choices $\alpha = 0$, 10 epochs, row batch size 1,024, and learning rate 10^{-4} are retained from the released implementation; the seed is fixed for reproducibility.

LLM-Streamline FFN Chen et al. (2025a). We retain the released replacement-network architecture and optimization settings while adapting its training target to an activation-boundary objective for the VGGT-family backbones. For each maximal contiguous removed interval $[i, j]$, we collect $X = H_{i-1}$ and $Y = H_j$ from the intact model. Each interval receives a replacement network

$$f(X) = \mathrm{ReLU}(XW_1 + b_1)W_2 + b_2, \qquad C \to 4C \to C, \tag{20}$$

where $W_1 \in \mathbb{R}^{C \times 4C}$, $W_2 \in \mathbb{R}^{4C \times C}$, and biases are broadcast across sample rows. The network is trained to minimize $\mathrm{MSE}(f(X), Y)$.

We use AdamW for one epoch with row batch size 16,384, learning rate 2×10^{-4}, minimum learning rate 5×10^{-6}, weight decay 10^{-3}, betas $(0.9, 0.95)$, seed 0, and a 3% linear warmup followed by cosine decay. Replacement networks are fitted independently from the intact-model activation cache and applied in model order at inference.

F Proof of the Shared-Mapping Reconstruction Penalty

We prove equation 2 for a fixed interval and calibration context. For each token type $r \in \{s, p\}$, let $L_r(W) = \mathbb{E}\|x_r W - y_r\|_2^2$, $\Sigma_r = \mathbb{E}[x_r^\top x_r] \succ 0$, and $T_r = \arg\min_W L_r(W)$. We assume finite second moments and positive weights π_s, π_p summing to one. Define $Q_r = \pi_r \Sigma_r$ and $\Delta_T = T_s - T_p$.

Quadratic decomposition. The least-squares optimum satisfies the normal equation $\Sigma_r T_r = \mathbb{E}[x_r^\top y_r]$. Expanding the loss around T_r therefore eliminates the cross term and gives

$$L_r(W) = L_r(T_r) + \operatorname{tr}\big[(W - T_r)^\top \Sigma_r (W - T_r)\big]. \tag{21}$$

Consequently, the excess loss of the optimal shared mapping over separate mappings is

$$\mathcal{B}_{\text{role}} = \min_W \sum_{r\in\{s,p\}} \operatorname{tr}\big[(W - T_r)^\top Q_r (W - T_r)\big]. \tag{22}$$

Optimal shared mapping. Let $S = Q_s + Q_p$. Differentiating the quadratic objective yields $W^* = S^{-1}(Q_s T_s + Q_p T_p)$. Writing $Z = W - T_p$, the same objective becomes

$$\begin{aligned}&\operatorname{tr}\big[(Z - \Delta_T)^\top Q_s (Z - \Delta_T) + Z^\top Q_p Z\big] \\ &\quad = \operatorname{tr}\big[(Z - S^{-1}Q_s\Delta_T)^\top S (Z - S^{-1}Q_s\Delta_T)\big] + \operatorname{tr}\big[\Delta_T^\top (Q_s - Q_s S^{-1} Q_s)\Delta_T\big].\end{aligned} \tag{23}$$

The first term vanishes at the optimum. Moreover, $Q_s - Q_s S^{-1} Q_s = Q_s S^{-1} Q_p$, whose inverse is $Q_p^{-1} S Q_s^{-1} = Q_p^{-1} + Q_s^{-1}$. Thus,

$$\mathcal{B}_{\text{role}} = \operatorname{tr}\big[\Delta_T^\top (Q_s^{-1} + Q_p^{-1})^{-1} \Delta_T\big], \tag{24}$$

which proves equation 2. Since $(Q_s^{-1} + Q_p^{-1})^{-1}$ is positive definite, this penalty is zero if and only if $T_s = T_p$.

Token imbalance and balanced fitting. For the illustrative case $\Sigma_s = \Sigma_p = \Sigma$, the shared optimum reduces to $W^* = \pi_s T_s + \pi_p T_p$. Defining $D_T = \|\Sigma^{1/2}(T_s - T_p)\|_F^2$, we obtain

$$\mathcal{B}_{\text{role}} = \pi_s \pi_p D_T, \qquad L_s(W^*) - L_s(T_s) = \pi_p^2 D_T, \qquad L_p(W^*) - L_p(T_p) = \pi_s^2 D_T. \tag{25}$$

When π_s is small, the pooled excess loss can be small even though the special-token excess remains close to D_T. Balanced weights remove this asymmetry but still incur a shared-mapping penalty of $D_T/4$. Thus, reweighting alone need not resolve disagreement between the optimal mappings.

Scope of the analysis. The result concerns unregularized population reconstruction loss within the linear mapping class; the deleted computation itself need not be linear. Our implementation instead fits identity-centered ridge mappings from finite calibration samples. Separate mappings remove the shared-mapping constraint but may incur greater estimation error, particularly for the substantially less numerous special tokens. The analysis therefore motivates token-specific fitting and an adaptive safeguard, but does not derive our selection criterion or threshold, nor guarantee improved downstream geometric accuracy.

G Calibration and runtime cost

All experiments are conducted on a single NVIDIA RTX 6000 Ada GPU with 48 GB memory. The main calibration cost comes from pruning selection. Without approximation, exhaustive two-interval search requires $O(L^4)$ model evaluations. Our approximate additivity reduces this to evaluating only the $\binom{L+1}{2} = O(L^2)$ single intervals, i.e., 1,176 pruned configurations per scene for the 48-layer VGGT and VGGT-Ω, and 666 for the 36-layer π^3. For 100 calibration scenes, the measured wall-clock times are approximately 17 hours for VGGT, 11 hours for VGGT-Ω, and 6 hours for π^3. CKA-based selection further reduces the calibration cost. Including cached-feature similarity computation and the additional evaluations used for token-region selection, calibration on 100 scenes takes less than one hour for each backbone. These wall-clock times are rough measurements under our current implementation and hardware setup. They depend on implementation details, such as the degree of overlap between GPU inference and CPU-side evaluation, as well as dataset characteristics such as image resolution and preprocessing cost, and therefore need not scale directly with the number of model evaluations. Post-pruning recovery adds little additional cost: the recovery mappings are obtained by closed-form least squares without iterative training, with fitting taking only a few seconds. At inference, each recovery transformation adds only a matrix multiplication at a pruning boundary, whose overhead is negligible relative to the retained transformer computation.

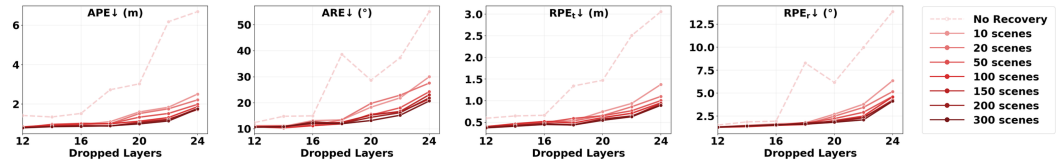

Figure 12: **Effect of calibration-set size on post-pruning recovery.**

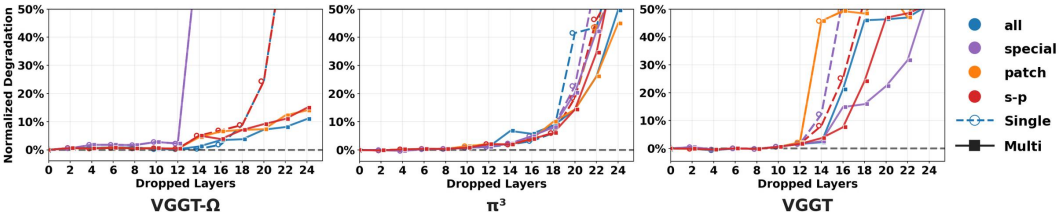

Figure 13: **Effect of token-region handling on similarity-based pruning.** Using special or patch token type alone generally performs poorly. Neither the all-token nor the special–patch mean variant consistently dominates, motivating selection based on measured calibration degradation.

H Ablation Study

Calibration-set Size. We examine post-pruning recovery using equal numbers of ScanNet and nuScenes calibration scenes, evaluating on their held-out splits and the unseen 7Scenes and Waymo datasets. Fig. 12 shows that even 10 scenes substantially reduce pruning-induced errors. Increasing the size from 10 to 100 scenes generally improves all seven metrics, especially at higher pruning budgets. Beyond 100 scenes, trends across adjacent budgets become smoother, but accuracy gains are limited. We therefore use 100 scenes for the recovery benchmark to balance cost and accuracy.

Token-region Selection. We compare four similarity-based pruning variants for VGGT and VGGT-Ω: special tokens only, patch tokens only, all tokens jointly, and the mean of separately computed special-token and patch-token similarities. Fig. 13 shows that using either token type alone generally performs poorly, while neither the all-token nor the special–patch mean variant consistently outperforms the other. Given the low computational cost of similarity-based scoring, we generate pruning configurations from both latter variants and select the one with lower measured degradation on the calibration data. The selected configuration is then frozen for evaluation. Implementation details are provided in Appendix D.

I Long-sequence Generalization

We further evaluate whether pruning configurations and recovery transformations calibrated on 30-frame sequences generalize to longer inputs. We use the same compressed models as in Table 1, calibrated only on 30-frame sequences, and apply them unchanged to longer inputs. Table 2 reports results on held-out scenes using 30, 100, and 200 frames. Across all three backbones, the compressed models retain comparable accuracy while preserving parameter, latency, and memory savings, demonstrating generalization beyond the calibration sequence length.

J Raw Per-Metric Results

This section presents the raw per-metric counterparts to the figures in the main paper, without normalization or cross-metric averaging. These results support the conclusions drawn from the normalized, aggregated results in the main text.

Table 2: **Generalization across scenes and sequence lengths.** Pruning configurations and recovery transformations are calibrated using only 30-frame sequences and applied unchanged to held-out test scenes with 30, 100, and 200 frames. The results demonstrate generalization to both unseen scenes and longer sequences. Parameter counts include only the aggregator, whereas latency and peak allocated GPU memory are measured over the complete forward pass.

			Trajectory Errors				Geometric Errors			System Efficiency		
Method	Frames	Layers	APE ↓ (m)	ARE ↓ (°)	RPE_t ↓ (m)	RPE_r ↓ (°)	Acc. ↓ (m)	Comp. ↓ (m)	CD ↓ (m)	Params ↓ (M)	Latency ↓ (s)	Memory ↓ (GiB)
VGGT-Ω	30	48	0.31	9.99	0.18	0.93	1.15	1.08	1.11	605	1.98	8.2
+ **Ours**	30	**30**	0.33	9.07	0.19	1.00	1.13	1.09	1.11	378$_{\downarrow 37.5\%}$	1.58$_{\downarrow 20.4\%}$	6.9$_{\downarrow 15.9\%}$
VGGT-Ω	100	48	0.31	9.97	0.14	0.52	1.13	1.07	1.10	605	6.58	11.3
+ **Ours**	100	**30**	0.32	8.51	0.14	0.54	1.12	1.08	1.10	378$_{\downarrow 37.5\%}$	4.87$_{\downarrow 26.1\%}$	10.0$_{\downarrow 11.3\%}$
VGGT-Ω	200	48	0.31	10.02	0.14	0.45	1.13	1.07	1.10	605	12.16	13.2
+ **Ours**	200	**30**	0.32	8.54	0.14	0.47	1.12	1.08	1.10	378$_{\downarrow 37.5\%}$	8.67$_{\downarrow 28.7\%}$	11.9$_{\downarrow 9.6\%}$
π^3	30	36	0.36	10.59	0.21	0.96	1.25	1.12	1.18	454	1.16	6.5
+ **Ours**	30	**20**	0.37	10.92	0.22	1.13	1.27	1.14	1.20	252$_{\downarrow 44.5\%}$	0.80$_{\downarrow 30.9\%}$	5.3$_{\downarrow 18.5\%}$
π^3	100	36	0.35	9.99	0.16	0.54	1.23	1.10	1.17	454	4.70	8.0
+ **Ours**	100	**20**	0.35	10.93	0.15	0.58	1.26	1.13	1.19	252$_{\downarrow 44.5\%}$	2.99$_{\downarrow 36.4\%}$	6.9$_{\downarrow 14.1\%}$
π^3	200	36	0.34	10.01	0.16	0.49	1.23	1.10	1.17	454	9.40	9.1
+ **Ours**	200	**20**	0.35	10.95	0.15	0.51	1.26	1.13	1.19	252$_{\downarrow 44.5\%}$	5.78$_{\downarrow 38.5\%}$	7.9$_{\downarrow 12.5\%}$
VGGT	30	48	0.90	9.90	0.52	1.24	1.13	1.12	1.12	605	1.78	8.8
+ **Ours**	30	**38**	1.12	12.05	0.48	1.38	1.12	1.15	1.13	479$_{\downarrow 20.8\%}$	1.48$_{\downarrow 14.2\%}$	8.2$_{\downarrow 6.8\%}$
VGGT	100	48	0.86	10.03	0.40	0.70	1.12	1.10	1.11	605	6.68	10.9
+ **Ours**	100	**38**	1.08	12.25	0.36	0.74	1.12	1.14	1.13	479$_{\downarrow 20.8\%}$	5.62$_{\downarrow 15.8\%}$	10.2$_{\downarrow 6.4\%}$
VGGT	200	48	0.86	10.04	0.40	0.62	1.12	1.10	1.11	605	13.21	12.5
+ **Ours**	200	38	1.08	12.26	0.36	0.65	1.12	1.14	1.13	479$_{\downarrow 20.8\%}$	10.85$_{\downarrow 17.8\%}$	11.9$_{\downarrow 5.5\%}$

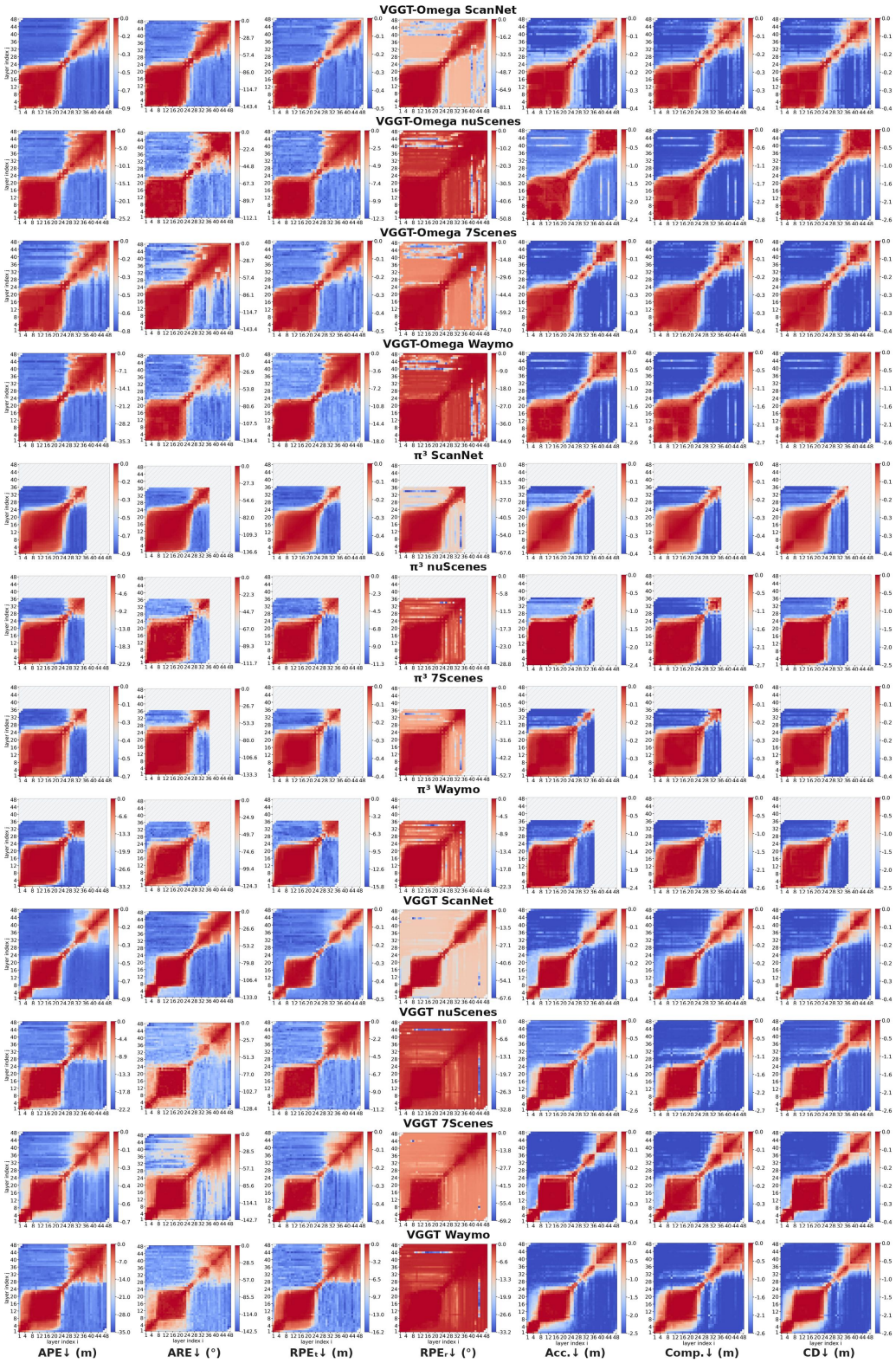

Figure 14: **A shared redundancy structure across the VGGT family.** Two separated low-degradation regions consistently appear across dataset domains and evaluation metrics, with the dominant region occurring early in the aggregator. Scores are mirrored across the diagonal. This is the unnormalized, per-metric counterpart to Fig. 2.

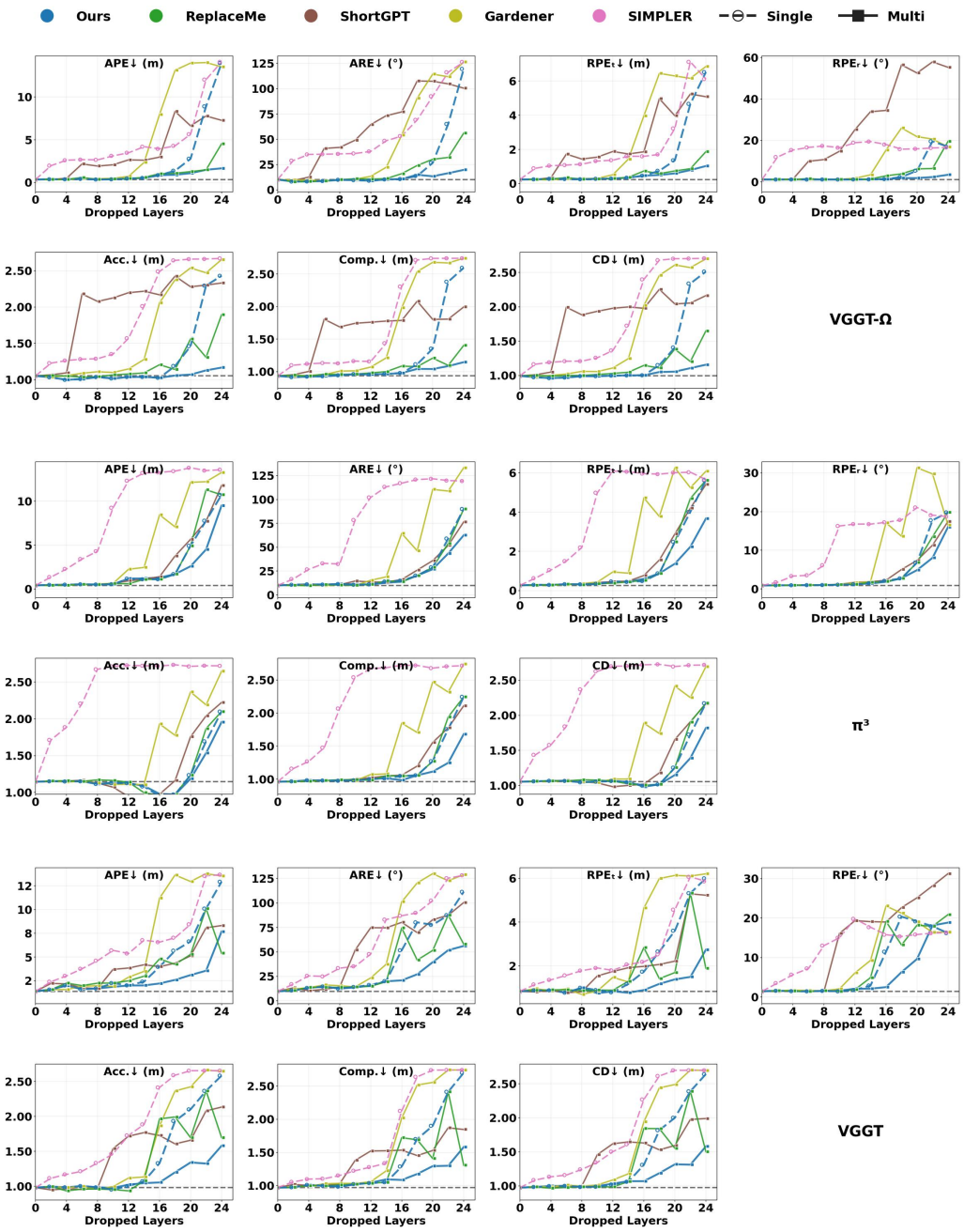

Figure 15: **Pruning quality and same-dataset generalization.** For each method and budget, one pruning configuration is selected based only on the combined 100-scene calibration set from ScanNet and nuScenes and applied unchanged to the held-out evaluation sets for same-dataset generalization. "Single" restricts removal to one contiguous interval, whereas "Multi" allows multiple intervals according to each method's search space; our method allows up to two separated intervals. This is the unnormalized, per-metric counterpart to Fig. 4.

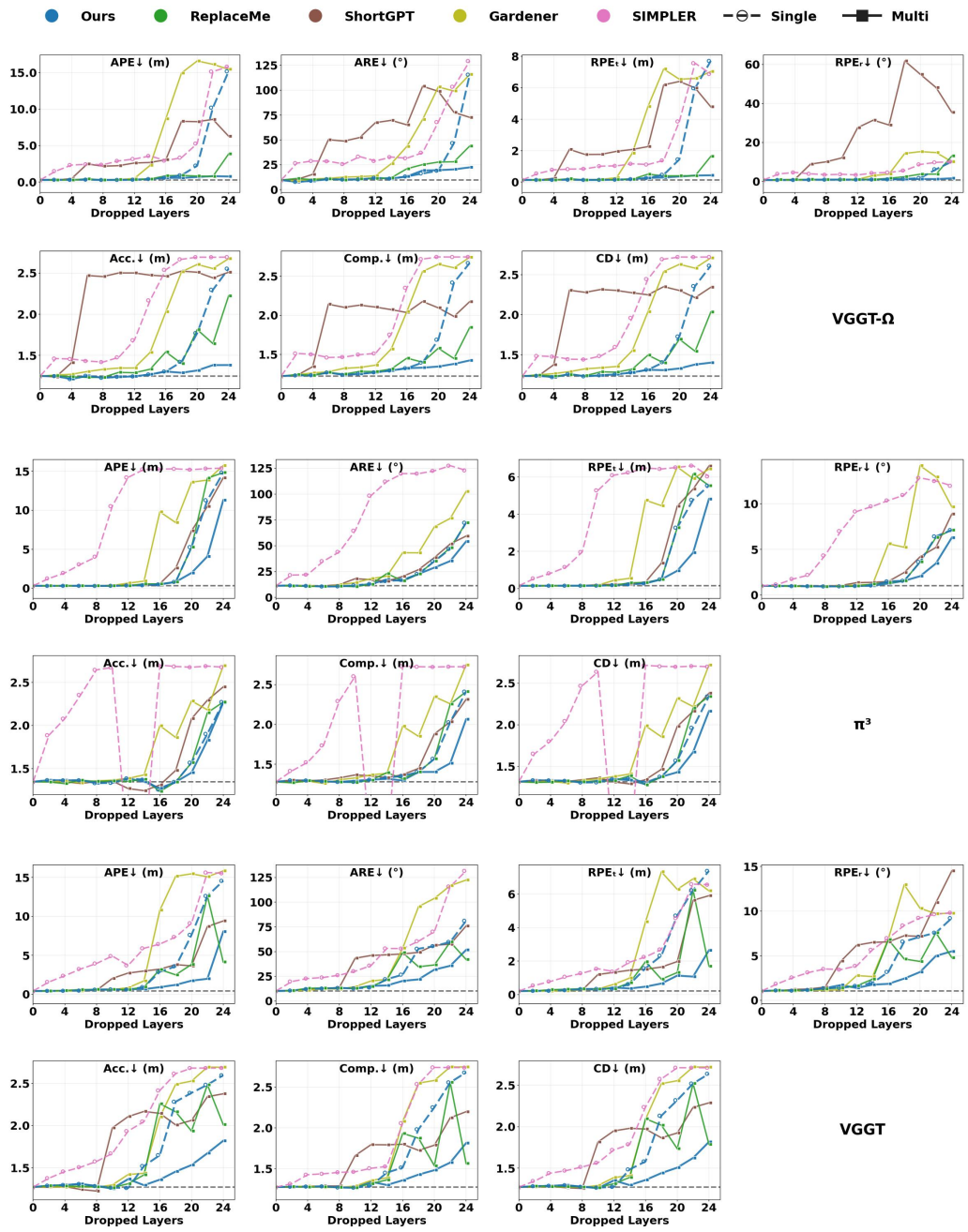

Figure 16: **Pruning quality and cross-dataset generalization.** For each method and budget, one pruning configuration is selected based only on the combined 100-scene calibration set from ScanNet and nuScenes and applied unchanged to the unseen datasets for cross-dataset generalization. "Single" restricts removal to one contiguous interval, whereas "Multi" allows multiple intervals according to each method's search space; our method allows up to two separated intervals. This is the unnormalized, per-metric counterpart to Fig. 4.

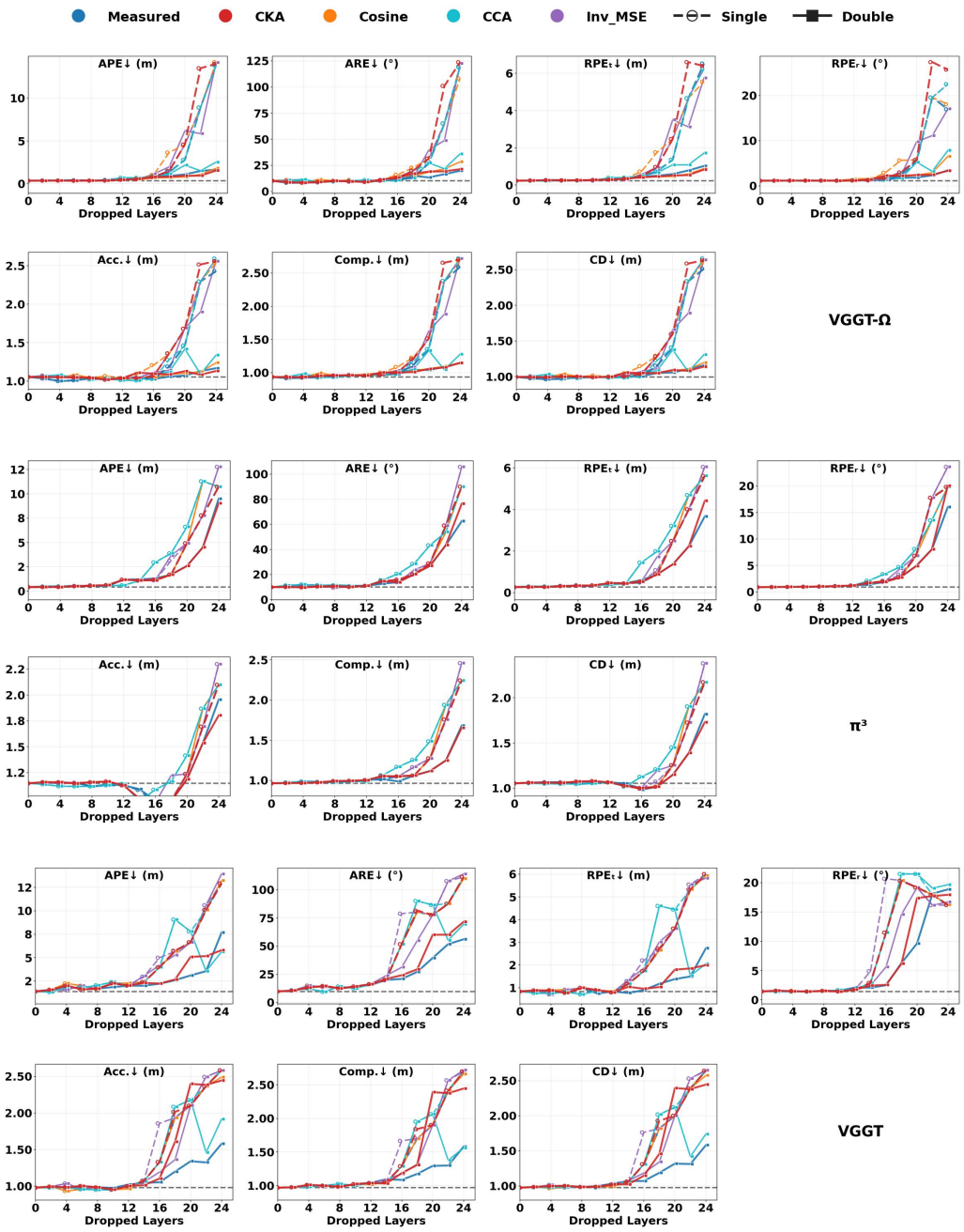

Figure 17: **Pruning quality and same-dataset generalization with representation-based proxies.** For each method and budget, one pruning configuration is selected based only on the combined 100-scene calibration set from ScanNet and nuScenes and applied unchanged to the held-out evaluation sets for same-dataset generalization. "Single" restricts removal to one contiguous interval, whereas "Double" allows up to two separated intervals. This is the unnormalized, per-metric counterpart to Fig. 6.

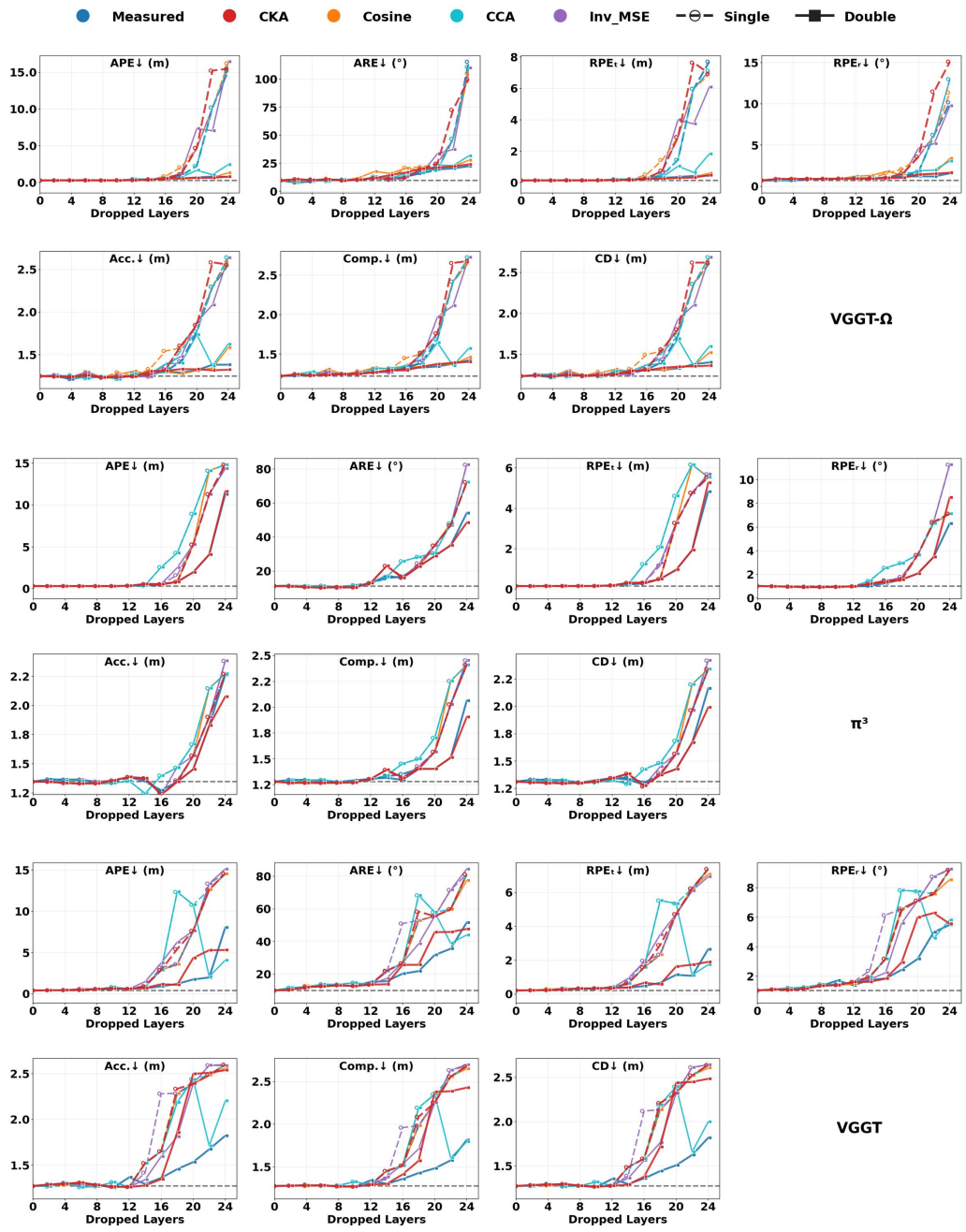

Figure 18: **Pruning quality and cross-dataset generalization with representation-based proxies.** For each method and budget, one pruning configuration is selected based only on the combined 100-scene calibration set from ScanNet and nuScenes and applied unchanged to the unseen datasets for cross-dataset generalization. "Single" restricts removal to one contiguous interval, whereas "Double" allows up to two separated intervals. This is the unnormalized, per-metric counterpart to Fig. 6.

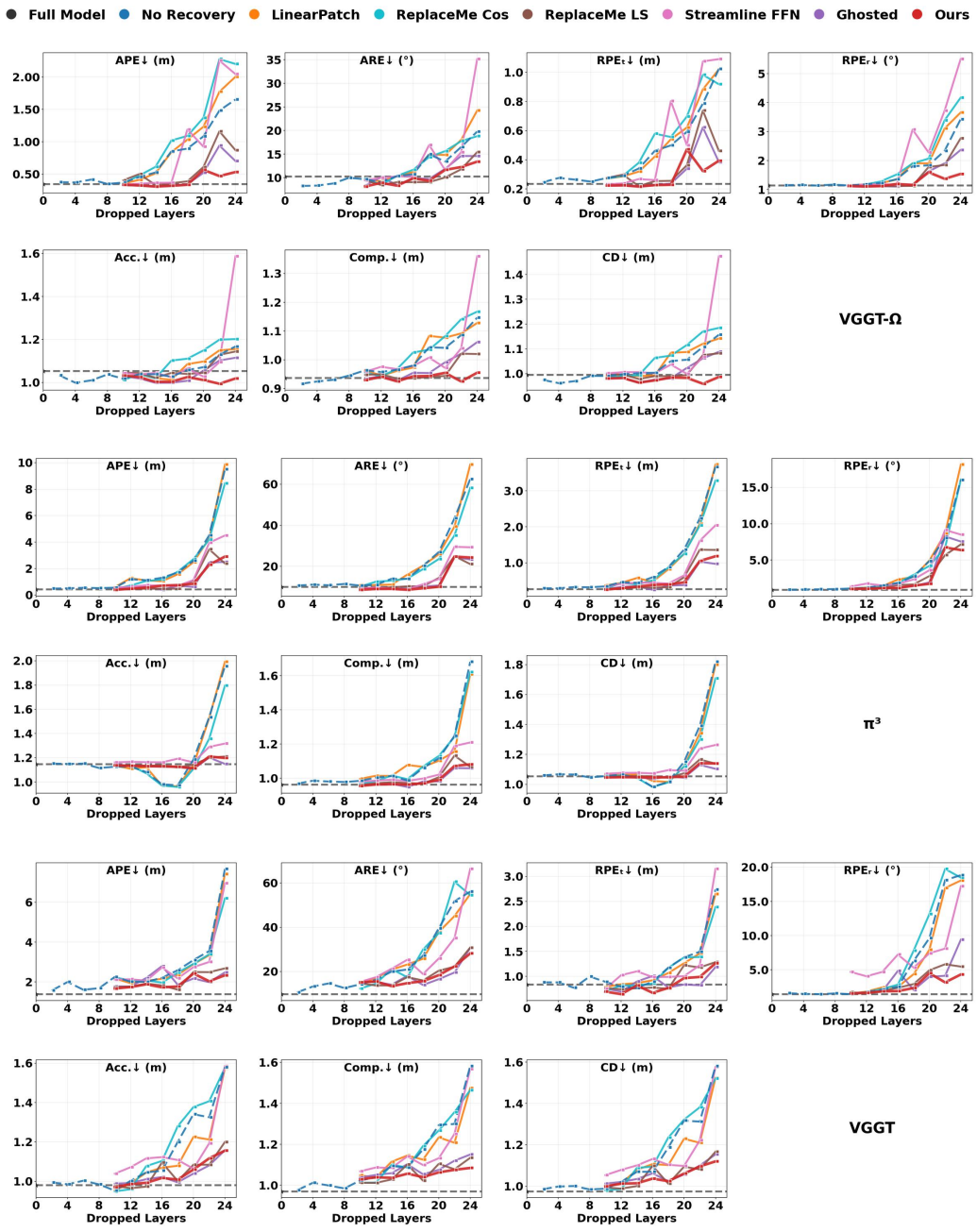

Figure 19: **Calibration-based post-pruning recovery benchmark.** Recovery transformations are fitted on the combined 100-scene calibration set from ScanNet and nuScenes and evaluated on their corresponding held-out evaluation sets. This is the unnormalized, per-metric counterpart to Fig. 7.

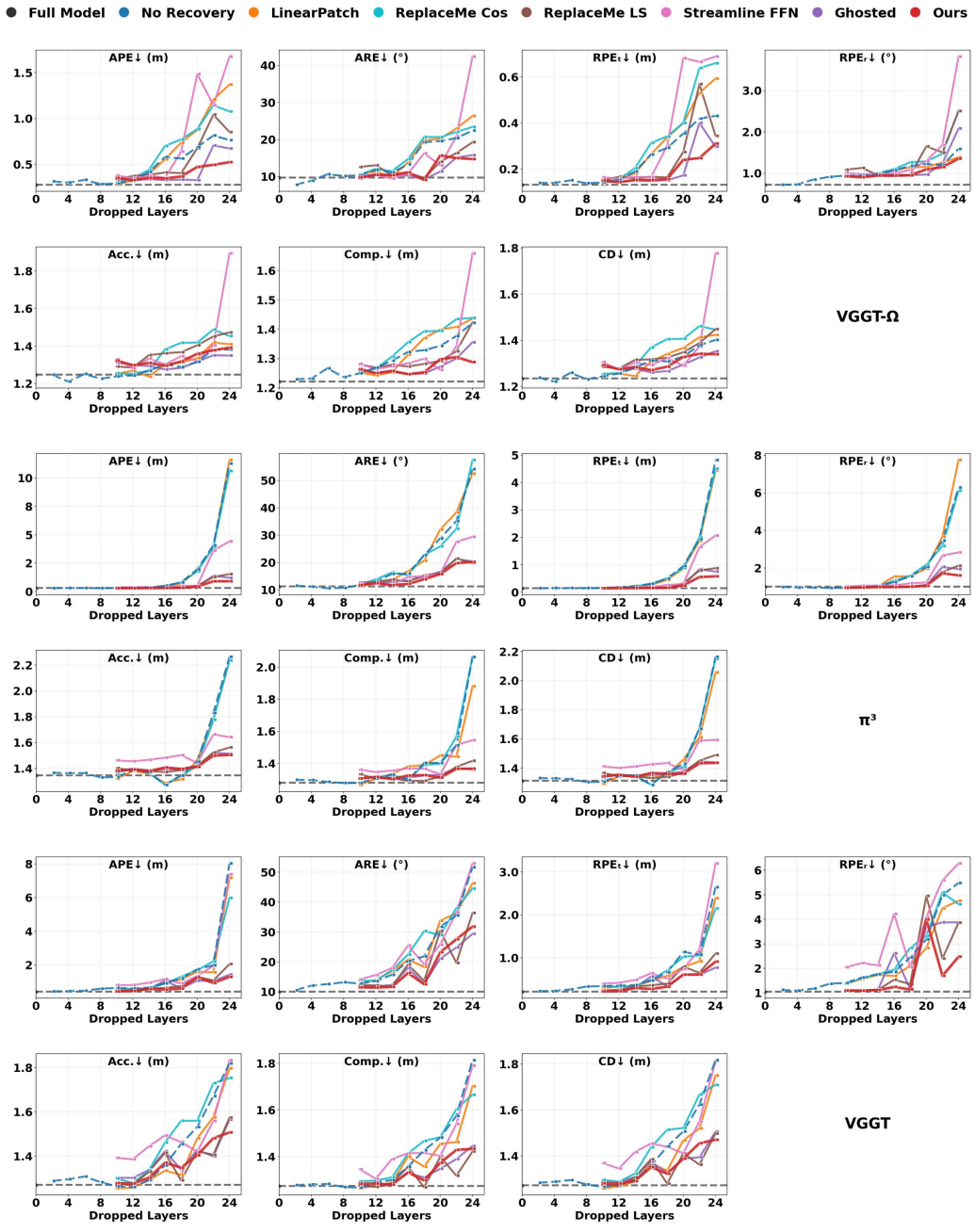

Figure 20: **Calibration-based post-pruning recovery benchmark.** Recovery transformations are fitted on the combined 100-scene calibration set from ScanNet and nuScenes and evaluated on the unseen 7Scenes and Waymo datasets. This is the unnormalized, per-metric counterpart to Fig. 7.