A LASTING BIAS OF INITIALIZATION: HOW INITIALIZATION SHAPES GENERALIZATION IN VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Initialization is typically designed to stabilize optimization, yet its effects can persist throughout training. Recent work for example showed that warming up Vision Transformers (ViTs) on procedural text, such as Dyck languages, surprisingly improves downstream image classification despite seeing no images during warm-up. We investigate which changes to initialization led to this. We find that much of the improvement can be recovered by matching a small set of layerwise parameter and activation statistics, including effective weight scales, LayerNorm statistics, low attention entropy and negative MLP pre-activations in early-to-middle layers, and large residual write ratios in late layers. We extract a total of 108 of these scalar statistics across the layers of procedurally warmed-up reference models and show that imposing them on a randomly initialized ViT is sufficient to match or exceed the improvements from procedural warm-up. Models trained from procedural and our modified initializations also show persistently lower intermediate-layer prediction accuracy and memorize fewer corrupted labels. These observations are consistent with an implicit regularizing effect. Together, our results show that a few initialization statistics can reproduce much of the benefit of procedural warm-up and provide evidence that initialization can impose a lasting inductive bias on learning and generalization.

1 INTRODUCTION

Deep neural networks are heavily overparameterized, so many parameter configurations can achieve similarly low training loss while generalizing very differently, as illustrated by phenomena such as grokking (Power et al., 2022). The training objective and data alone therefore do not determine the solution found: architecture, optimization, hyperparameters, and other details of the training procedure all bias which solutions are reached. Initialization is one such factor, but is often considered as a means to ensure stable and reliable optimization (Glorot & Bengio, 2010; He et al., 2015).

Yet initialization can influence learning well beyond the first few optimization steps. Properties of initial parameters can persist through training and affect generalization (Zhang et al., 2020; Ramasinghe et al., 2023), and carefully choosing parameter statistics or structural properties at initialization can improve performance (Raghu et al., 2019; Trockman & Kolter, 2023). More surprisingly, Shinnick et al. (2026) showed that Vision Transformers (ViTs; Dosovitskiy et al., 2021) benefit from a *procedural warm-up*: before seeing any images, the transformer is trained on symbolic sequences generated by formal languages such as Dyck languages. Although warm-up contains no images and is unrelated to the image classification task, it still improves downstream classification accuracy by ca. 2 pp (Table 1). What, then, does ViT learn during procedural warm-up that transfers to vision? Is it an "abstract computational prior," or does the warm-up induce some specific structural bias?

To answer this, we compare procedurally warmed-up and randomly initialized models both at initialization and throughout downstream training. We first observe a persistent difference in intermediate-layer class predictability. Compared with random initialization, procedurally warmed-up models have substantially lower logit lens accuracy in early and middle layers, with accurate class predictions emerging only in the final layers (Figures 1a and 1b). Procedurally warmed-up models also

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.

memorize corrupted labels substantially less readily (Figure 1c), suggesting a regularizing effect. However, how does procedural warm-up achieve this?

We trace much of the benefits of procedural warm-up to a small set of initialization properties (108 scalars). Procedural warm-up changes the effective weight scales (Figure 2d) and late layers write much strongly to the residual stream than early and middle layers (Figure 2a). It also produces significantly sharper attention patterns than random initialization (Figure 2b) and suppresses MLPs (Figure 2c) in early-to-middle layers. To test whether these properties are sufficient to recover the performance gains of procedural warm-up, we apply them to a randomly initialized model. We show that they are sufficient to match, or even exceed, the performance gains from procedural warm-up (Table 2 and Figures 3 and 11). Thus, the benefit of procedural warm-up need not come from transferring task-specific symbolic representations. Instead, procedural warm-up appears to be one way of constructing an initialization that imposes a persistent bias on how learning is distributed across network depth and shaped over the course of training, with a lasting impact on generalization.

Our contributions are as follows:

- We show that procedural warm-up induces persistent changes in the learning dynamics of ViTs: class information becomes decodable later in the network (Figures 1a and 1b), and the models are less prone to memorizing corrupted labels (Figure 1c).
- We identify a small set of initialization properties (totaling 108 scalars): mostly larger effective weight scales, sharper attention and suppressed MLPs in early-to-middle layers, and substantially larger residual writes in late layers (Section 3.1).
- We test these properties by applying them to randomly initialized models. Combining the early-layer and late-layer interventions recovers most, and in some settings all, of the generalization gains from procedural warm-up (Sections 3.2 to 3.4).
- These results show that initialization can act as a lasting inductive bias, shaping where and how learning occurs across network depth and favoring solutions that generalize better.

We will release the code upon acceptance.

2 What drives the performance gains from procedural warm-up?

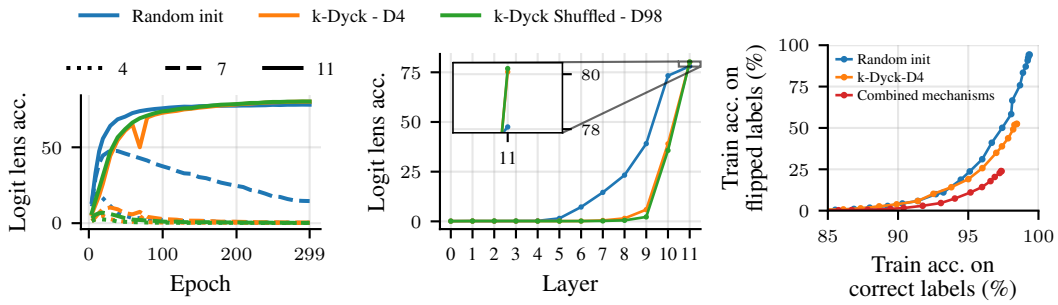
Recently, Shinnick et al. (2026) showed that Vision Transformers (ViTs) can be warmed-up by training them on procedural text data generated from formal languages, such as Dyck languages, to improve final performance. Below, we first review the procedural warm-up approach and the improvements it yields. We then compare procedurally warmed-up and randomly initialized models during and after training, finding premature learning in early-to-middle layers and regularizing as potential explanations for the performance gap. This motivates our broader hypothesis that initializations can provide regularizing inductive biases, which we will support in the following section.

2.1 Procedural warm-up

The core idea of procedural warm-up is to train a ViT on non-visual symbolic sequences generated from formal languages before beginning downstream vision training. As warm-up data, Shinnick et al. (2026) considered regular (WW), context-free (k-Dyck), and context-sensitive (k-Dyck Shuffle) languages. To process these symbolic sequences with a ViT, we replace the visual patch embedding with a frozen random token embedding whose token vectors are mutually orthogonal, keep the positional encoding frozen, and train the model using a masked-token prediction objective. This encourages the model to learn to adapt its attention and MLP layers. After warm-up, the token embedding and token prediction head are discarded, while the visual patch embedding and classification head are newly initialized. The warmed-up transformer parameters (i.e., LayerNorm parameters, linear weights, and biases from the attention and MLP layers) are then used to initialize standard image classification training. Further procedural warm-up and standard image classification training details are provided in Appendix A.

Table 1: **Procedural warm-up (PW) improves classification accuracy (%).** Appendix B provides results for other variants.

	Random	PW (k-Dyck D4)
ViT-S / ImageNet-100	84.37 ± 0.11	86.09 ± 0.55
ViT-B / ImageNet-1k	78.08 ± 0.17	80.08 ± 0.12



(a) Logit lens during training. (b) Logit lens after training. (c) Random-label memorization.

Figure 1: **Procedural warm-up delays class decodability and reduces memorization.** We compare random initialization, k-Dyck D4, and k-Dyck Shuffled D98 for ViT-B on ImageNet-1k. a: Logit lens accuracy at layers 4, 7, and 11 over training. Under random initialization, accuracy at layers 4 and 7 increases early and later declines, whereas it remains low throughout training after procedural warm-up. b: Logit lens accuracy across depth at the end of training. Under random initialization, accuracy increases gradually from layer 6 onward; after procedural warm-up, it remains low until the final layers and then increases sharply, reaching higher final-layer accuracy. c: Trainings with 10% randomly flipped labels. Each point corresponds to a checkpoint during training. At matched accuracy on correctly labeled training examples (x-axis), procedurally warmed-up models memorize substantially fewer corrupted labels than randomly initialized models (y-axis).

Surprisingly, this procedural warm-up improves image classification accuracies of ViT compared to randomly initialized models; see Table 1 for replicated results for ViT-S on ImageNet-100 and ViT-B on ImageNet-1k. This result is, however, not very intuitive: the procedural data contains neither visual nor semantic information and does not attempt to mimic the structures learned by models trained on natural images. What, then, does procedural warm-up change in the transformer that makes subsequent vision training more effective?

In what follows, we considered the following k-Dyck with the following choices: depth (4 vs. 98, controlling the maximum nesting level), truncation (whether sequences are cut mid-structure, breaking closure), and shuffling (whether any active open bracket can be closed rather than strictly following the chronological order, making the language context-sensitive). Examples for these variants are provided in Appendix A. For ViT-S we considered all of the variants. For ViT-B we focused on k-Dyck D4 and k-Dyck Shuffled D98 due to computational costs.

2.2 Delayed class decodability and less memorization

To better understand how procedural warm-up changes downstream image classification, we first tracked when class information becomes (directly) decodable during training. For this, we used logit lens (nostalgebraist, 2020), which applies ViT-B's final LayerNorm and classification head to the `cls` tokens of intermediate layers. Figure 1a shows that classes are decodable from earlier layers for the randomly initialized models compared to procedurally warmed-up ones (similar for ViT-S, see Figure 4a). After 30 epochs, for example, the layer 7 logit lens reaches 49.1% accuracy under random initialization, compared with 10.3% for k-Dyck D4 and 6.0% for k-Dyck Shuffled D98. This difference persists throughout training (Figure 1b, Figure 4b for ViT-S results). By the end of training, the decodability of the class gradually increases with depth under random initialization, whereas under procedural warm-up the classes become decodable only in the very last layers of the network. Additional results are provided in Appendix C.

Part of this difference is due to alignment of intermediate representation with the classification head rather than the absence of class information. When we applied TunedLens (Belrose et al., 2023, logit lens with trainable affine transformation), the gap to random initialization in layers 8–10 shrinks from 22–38 to 7–13 pp, but it does not vanish. Thus, procedural warm-up appears to both delay linearly decodable class information and the alignment with the final class readout.

We next tested whether enhancing class decodability at intermediate layers is harmful to generalization. For this, we trained procedurally warmed-up models with an auxiliary classification loss $\mathcal{L}_{\text{aux}}$, computed by applying the model's final LayerNorm and classification head to the layer 7 `cls`

token. For the first 50 epochs, we optimized $\mathcal{L} = \mathcal{L}_{\text{final}} + 0.3\mathcal{L}_{\text{aux}}$ and then continued with only the standard classification loss $\mathcal{L}_{\text{final}}$. As designed, layer 7 logit lens accuracy at epoch 50 increases to 41.4% and 37.7% for k-Dyck D4 and k-Dyck Shuffled D98, respectively. This also initially improves accuracies over the procedurally warmed-up models trained with $\mathcal{L}_{\text{final}}$ alone (+1.49 pp and +0.67 pp at epoch 100). By the end of training, however, this reverses (-1.58 pp and -0.91 pp). Thus, earlier intermediate decodability seems to help early training but ultimately hurts generalization.

To complement this finding, we restricted the classifier head to the summed attention and MLP outputs of the last three layers, so that earlier layers affect the prediction only indirectly (details and full results in Appendix D). This reproduces the late logit-lens profile of procedural warm-up (Figure 7b) and improves accuracy by 1.3 pp over random initialization. This further supports that intermediate decodability could act as a residual shortcut that helps early but hurts performance the longer we train.

Above observations are reminiscent of regularization: fitting the training objective more readily early on does not necessarily lead to better final generalization. We therefore tested whether procedural warm-up also affects memorization. Following Zhang et al. (2017), we randomly flipped 10% of the training labels. Randomly initialized models readily memorize these corrupted labels, whereas procedurally warmed-up models memorize substantially fewer of them at matched training accuracy on the correctly labeled examples (Figure 1c). This suggests that the effect is not simply due to slower optimization, but is consistent with a regularizing effect.

Taken together, above results suggest that procedural warm-up induces a *lasting regularizing bias*. Compared with random initialization, it delays class decodability and alignment with the final readout, and reduces random-label memorization. This may bias learning away from solutions that fit the classification objective early and directly, and toward representations whose predictive computation is deferred to later layers. So, which properties of procedural warm-up produce this bias?

3 How does procedural warm-up achieve the performance gains?

3.1 How procedural warm-up changes the initial weights

To identify which initialization properties explain the differences in learning dynamics and final performance, we compared randomly initialized and procedurally warmed-up models. Matching a small set of parameter and activation statistics recovers most or all of the performance gains (Sections 3.2 and 3.3), and combining the interventions can exceed these gains (Section 3.4).

Large residual writes in late layers. The first property we identified is that procedural warm-up increases the relative contribution of late attention and MLP sublayers to the residual stream. To quantify this, we measure the norm of each sublayer's output relative to the residual stream it is added to. Concretely, let X_ℓ denote the residual stream entering layer ℓ, and let $\widetilde{X}_\ell = X_\ell + \mathrm{Attn}_\ell(X_\ell)$ denote the stream after its attention sublayer. Here, Attn_ℓ and MLP_ℓ include their respective preceding LayerNorms. We define the residual write ratios as

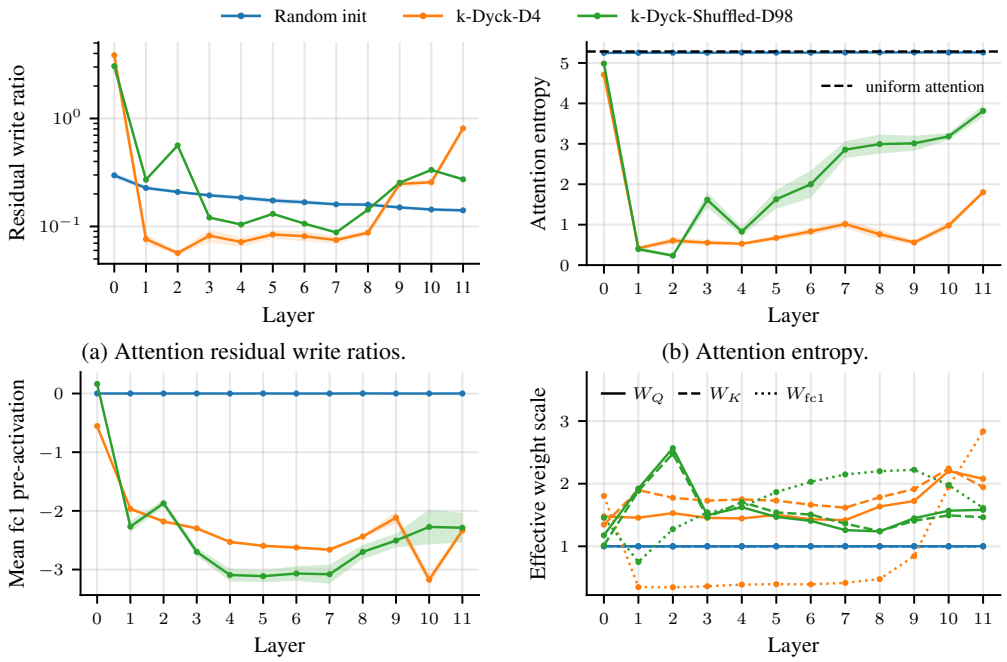
$$\rho_\ell^{\mathrm{Attn}} = \mathbb{E}_{I,t}\left[\frac{\left\|[\mathrm{Attn}_\ell(X_\ell)]_t\right\|_2}{\left\|[X_\ell]_t\right\|_2}\right], \qquad \rho_\ell^{\mathrm{MLP}} = \mathbb{E}_{I,t}\left[\frac{\left\|[\mathrm{MLP}_\ell(\widetilde{X}_\ell)]_t\right\|_2}{\left\|[\widetilde{X}_\ell]_t\right\|_2}\right], \tag{1}$$

where the expectation is over images I and token positions t, and each norm is taken over the token's feature dimension. We omit the dependence of X_ℓ on I for readability. Figure 2a shows that, compared with random initialization, procedural warm-up yields smaller attention residual write ratios in most early and middle layers (except for the very first layer) and larger ratios in the late layers. MLP residual write ratios show a similar U-shaped pattern (Figure 12).

Low attention entropy. We also found that early and middle layers have lower attention entropy after procedural warm-up. We measure attention entropy (in nats, uniform attention over N tokens is $\log N$) as

$$H_\ell = \mathbb{E}_{I,h,q}\left[-\sum_{k=1}^{N} a_{\ell,h,q,k}(I)\log a_{\ell,h,q,k}(I)\right], \tag{2}$$

where $a_{\ell,h,q,k}(I)$ is the attention score from query q to key k in head h of layer ℓ for image I. We average entropy over images, heads, and query tokens. Figure 2b shows that random initializa-



(a) Attention residual write ratios.

(b) Attention entropy.

(c) Mean `fc1` preactivation.

(d) Effective weight scales.

Figure 2: **Procedural warm-ups induce distinct parameter and activation statistics.** We compare a randomly initialized ViT-B with models warmed-up on k-Dyck D4 or shuffled k-Dyck D98. All quantities are measured at initialization. a: Attention residual write ratios (Equation 1) are lower across most early and middle layers and increase toward the final layers. Results for MLP residual write ratios and ViT-S are provided in Appendix I. b: Warmed-up models produce sharper attention with lower entropy. Under random initialization, attention is nearly uniform. c: Mean `fc1` preactivations are strongly negative through most layers after procedural warm-up, compared with approximately zero under random initialization. d: Both warm-ups increase the effective query and key weight scales (Equation 3). In most early and middle layers, effective MLP input projection scales decrease under k-Dyck D4 but increase under shuffled k-Dyck D98.

tion produces nearly uniform attention throughout the model, whereas procedural warm-up yields substantially lower entropy beyond the first layer. Thus, concentrating attention mass on fewer keys.

Negative MLP pre-activations. Procedural warm-up also shifts the mean of MLP pre-activations, i.e., the outputs of the input projection `fc1` before GELU, toward negative values. We compute this mean over images, tokens, and MLP hidden units. Figure 2c shows that random initialization yields mean pre-activations near zero across all layers. Under both procedural warm-ups, the mean pre-activations are mostly negative across all layers.

Effective weight scales and LayerNorm statistics. Finally, procedural warm-up changes the scales of the input projections, and the gains and biases of their preceding LayerNorms. In a pre-norm transformer, the linear part of an input projection acting on a standardized token is $W\,\mathrm{diag}(\gamma)$. Let $\gamma_{\mathrm{Attn},\ell}$ and $\gamma_{\mathrm{MLP},\ell}$ denote the gain vectors of the LayerNorms preceding the attention and MLP sublayers, respectively. We then define the effective weight scales for $m \in \{Q, K\}$ and `fc1` as

$$e^{\mathrm{Attn}}_{\ell,m} = \frac{\mathrm{RMS}(W_{\ell,m}\,\mathrm{diag}(\gamma_{\mathrm{Attn},\ell}))}{\sigma_0}, \quad e^{\mathrm{MLP}}_{\ell} = \frac{\mathrm{RMS}(W_{\ell,\mathrm{fc1}}\,\mathrm{diag}(\gamma_{\mathrm{MLP},\ell}))}{\sigma_0}, \tag{3}$$

where RMS denotes the root mean square of the matrix entries and $\sigma_0 = 0.02$ is the standard deviation of the random weight initialization. Thus, $e = 1$ corresponds to the random-initialization scale. Both procedural warm-ups increase the effective query and key weight scales. In most early and middle layers, the effective MLP input projection scale decreases under k-Dyck D4 but increases under shuffled k-Dyck Shuffled D98.

The effective scales combine projection weights and LayerNorm gains, but do not characterize the LayerNorm parameters separately or capture the additive contribution of the biases. We thus also

Table 2: **Late-layer residual write ratios account for the benefit of transferring procedural weights to those layers.** ImageNet-1k accuracy (%) for ViT-B. PW denotes procedural warm-up; models with PW in late layers only use random initialization in earlier layers. We intervene on layers 9–11 for k-Dyck D4 and 8–11 for shuffled k-Dyck D98. Other late-layer choices, ViT-S, and additional procedural variants yield similar results (Appendix H).

	PW	Late layers only		
		PW	PW w/ Random init residual write ratios	Random init w/ PW residual write ratios
k-Dyck - D4	80.08	78.89	77.37	79.99
k-Dyck Shuffled - D98	80.20	77.88	76.63	78.60

Table 3: **Residual write ratios (by scaling weights) mainly drive the improvements.** Matching procedural write ratios yields most of the accuracy improvement. Changing the relative update scale has only a smaller effect. Experiment done in the ViT-B in the k-Dyck D4 setting.

Relative update scale	Residual write ratios (layers 9–11)	
	random	procedural
Baseline	78.08 ($\|W\|\times 1$, lr$\times 1$, wd$\times 1$)	80.46 ($\|W\|\times a$, lr$\times a$, wd$/a$)
Reduced	78.72 ($\|W\|\times 1$, lr$/a$, wd$\times a$)	79.99 ($\|W\|\times a$, lr$\times 1$, wd$\times 1$)

record the mean and standard deviation of the LayerNorm gains and biases across channels in each layer. In early and middle layers, both procedural warm-ups reduce the mean LayerNorm gains below their random initialization value of 1 and introduce variation in gains and biases across channels.

3.2 Large residual writes of late layers are sufficient and necessary

Matching residual write ratios. To test whether the large residual writes in the later layers are necessary and sufficient for the improvement, we constructed a hybrid model with randomly initialized early layers (e.g., 0–8) and late layers (e.g., 9–11) copied from the procedurally warmed-up model (illustrated in Appendix E). In Appendix F, we verified that this hybrid model trains normally and achieves similar improvements to initializing all layers with procedurally warmed-up weights.

We measured the attention and MLP residual write ratios (Equation 1) in the later layers of the hybrid model and a fully randomly initialized model sharing its early-layer initialization. To test necessity, we rescaled the hybrid model to match the random model's ratios. For attention layer ℓ, we set $s = \hat{\rho}_\ell^{\text{Attn}}/\rho_\ell^{\text{Attn}}$, where ρ_ℓ^{Attn} is the current ratio and $\hat{\rho}_\ell^{\text{Attn}}$ the target from the corresponding random-model layer, and multiply both W_V and W_O by $\sqrt{s}$. Because the attention output is linear in each projection (ignoring biases), this scales its residual write by s. For MLPs, we multiply W_{fc2} simply by s. We rescaled sublayers in forward order because each change affects subsequent residual write ratios. To test sufficiency, we instead rescaled the same matrices in the random model to match the hybrid model's ratios. The procedure is illustrated in Appendix E and pseudcode is provided in Appendix G.

Are larger residual writes necessary and sufficient? Table 2 (third column, *PW w/ Random init residual write ratios*) shows that accuracy decreases when the residual write ratios are rescaled to match those of a randomly initialized model. This indicates that retaining the larger residual writes is necessary. Because degrading an initialization is easy, testing for sufficiency provides stronger evidence. Table 2 (fourth column, *Random init w/ PW residual write ratios*) shows that matching the residual write ratios of the hybrid model significantly improves accuracy. While this does not fully bridge the gap to the fully procedurally warmed-up model, it exceeds the performance of the hybrid model. Figure 14 shows that these model variants mimic the logit lens and write ratio profiles of procedural warm-up.

Larger residual writes or smaller update scales? Scaling weights to match residual write ratios also changes their scale relative to subsequent optimizer updates. Let a denote the multiplier applied to an individual weight matrix: $a = \sqrt{s}$ for W_V and W_O, and $a = s$ for W_{fc2}, using each sublayer's corresponding residual write multiplier s. In AdamW, adaptive normalization makes the

magnitude of the update approximately insensitive to an overall rescaling of the gradient. If this normalized update remains comparable after $W \to aW$, its size relative to the weight norm decreases approximately as $1/a$. Thus, upscaled weights effectively evolve more slowly in relative terms unless the learning rate is adjusted accordingly. To disentangle these effects, we independently varied the residual write ratios via weight scaling and the nominal relative update scale in ViT-B for the k-Dyck D4 setting. For weights scaled by a, we multiplied their learning rate lr by a to restore the baseline relative update scale. Conversely, for unscaled weights, we divided their learning rate by the corresponding a to obtain the reduced relative update scale.[1]

Table 3 shows that residual write ratios mainly drive the improvement. At a fixed relative update scale, matching procedural write ratios improves accuracy by 2.38 pp at the baseline scale and 1.27 pp at the reduced scale. Reducing the relative update scale has a smaller, inconsistent effect: accuracy increases by 0.64 pp for unscaled weights but falls by 0.47 pp for scaled weights.

3.3 Intervening on early layers is sufficient

Late-layer scaling alone does not consistently recover the full benefit of procedural warm-up. Together with the distinct logit lens dynamics in early and middle layers (Section 2.2), this motivates testing whether matching their initialization properties can reproduce the benefits of transferring key properties of the procedural weights to these layers.

Effective weight scales and LayerNorm statistics. First, we sample the entries of the LayerNorm gain vectors and bias vectors of layers 0–7 from Gaussians whose means and standard deviations are taken across channels of the corresponding vectors in the respective procedurally warmed-up model. We then rescale each randomly initialized query, key, and `fc1` weight matrix W^0 by the ratio of its target and current effective scales $W = \frac{e_{\text{target}}}{e_{\text{current}}} W^0$. Both scales are defined as in Equation 3: e_{target} is computed from the corresponding procedurally warmed-up model, while e_{current} is computed from W^0 using the newly sampled LayerNorm gains.

Attention entropy. We match the attention entropy of the procedurally warmed-up model in layers $\ell = 1, \ldots, 7$. Let c_ℓ denote the normalized mean of the attention inputs after LayerNorm, estimated over tokens and a few images. We add rank-one components to the query and key weight matrices:

$$W'_{\ell,Q} = s_{\ell,Q} W_{\ell,Q} + \lambda_\ell P_\ell c_\ell^\top, \qquad W'_{\ell,K} = s_{\ell,K} W_{\ell,K} + \lambda_\ell P_\ell r_\ell^\top, \tag{4}$$

where P_ℓ concatenates one random unit direction per head, r_ℓ is a random unit direction in the input space, $\lambda_\ell \geq 0$ controls the intervention strength, and $s_{\ell,Q}, s_{\ell,K}$ are renormalization factors (c.f., Appendix K.2 for details). The query component responds to the shared direction c_ℓ, encouraging queries from different tokens to align within each head. The key component varies across tokens along the same per-head directions, according to each token's projection onto r_ℓ. This thereby encourages different queries to favor common keys, concentrating attention mass (see Figure 16 for a visual).

MLP pre-activations. In the same layers, we use a similar construction to match the mean MLP pre-activations. Let c_ℓ^{MLP} denote the normalized mean of that layer's MLP inputs after LayerNorm, estimated over tokens and the same few images. We modify the MLP input projections as follows

$$W'_{\ell,\text{fc1}} = s_{\ell,\text{fc1}} W_{\ell,\text{fc1}} - \frac{\lambda_\ell^{\text{MLP}}}{\sqrt{n}} \mathbf{1}_n (c_\ell^{\text{MLP}})^\top, \tag{5}$$

where n is the number of MLP hidden units and $\lambda_\ell^{\text{MLP}} \geq 0$ controls the intervention strength. The subtracted component gives every hidden unit the same input-dependent shift, negative on average because it opposes the mean input direction. Moving pre-activations sufficiently far into the negative regime suppresses both GELU outputs and derivatives. Section K.3 provides further details.

Calibration and weight scale preservation. We calibrate the two intervention strengths by bisection to match the reference model's mean attention entropy H_ℓ and mean MLP pre-activation, respectively (Appendix K.2 and Section K.3). Layers are processed in depth order, with attention calibrated before the MLP, so each intervention accounts for preceding modifications. For both

[1] When changing the learning rate, we scaled weight decay wd inversely to keep AdamW's per-step decay factor $1 - \mathrm{lr} \cdot \mathrm{wd}$ fixed across rows within each column of Table 3.

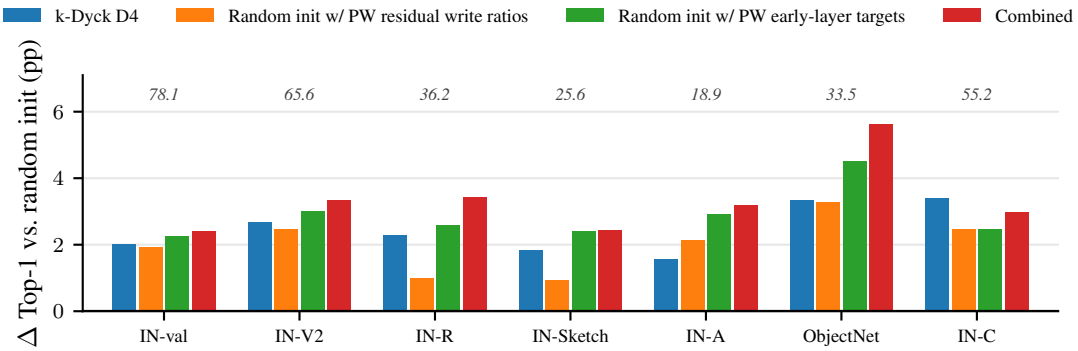


Figure 3: **Combining the early- and late-layer interventions improves accuracy across standard classification benchmarks.** We compare the gains over random initialization from procedural warm-up (k-Dyck D4), the early-layer intervention, the late-layer intervention, and their combination on ImageNet (IN)-val, IN-V2 (Recht et al., 2019), IN-R (Hendrycks et al., 2021a), IN-Sketch (Wang et al., 2019), IN-A (Hendrycks et al., 2021b), ObjectNet (Barbu et al., 2019), and IN-C (Hendrycks & Dietterich, 2019). The models match the learning dynamics (delayed class decoding and U-shaped write ratio pattern) we observed for procedurally warmed-up models; see Appendix J.

interventions, the rescaling factors s preserve the effective weight scales established above (see Section K.1 for details).

Results. Figure 3 shows that when matching these properties of procedural warm-up in a randomly initialized model, the resulting model (green) achieves comparable ImageNet validation accuracy to the corresponding procedurally warmed-up model (blue). Furthermore, the early-layer matched model achieves similarly strong generalization – it matches or outperforms the procedural model on 5 out of the 6 tested distribution shifts, except ImageNet-C (Hendrycks & Dietterich, 2019)) where it lacks behind by -0.94 pp. In comparison, matching only the later-layer residual write ratios (orange) underperforms matching the early-layer properties on all 7 datasets with especially large gaps on ImageNet-R (Hendrycks et al., 2021a) and ImageNet-Sketch (Wang et al., 2019).

3.4 Do they compose?

The previous sections showed that each intervention can recover much, and in some cases nearly all, of the accuracy gains from procedural warm-up. We next asked whether combining them could yield even larger improvements. Starting from a randomly initialized ViT-B, we applied the early-layer interventions from Section 3.3 to layers 0–7, left layer 8 unchanged, and matched the procedural residual write ratios in the final layers 9–11. All target statistics were taken from the k-Dyck D4 setting.

Figure 3 shows that our composed variant indeed improves classification accuracies across common image classification tasks. Note that these performance improvements come from only 108 scalar initialization statistics (ignoring the parameters of the random initialization and procedural warm-up weights as reference point): 24 effective weight scales, 64 means and standard deviations of LN gains and biases, 7 attention-entropy targets, 7 mean fc1 pre-activation targets, and 6 late-layer residual write ratios. For comparison, procedural warm-up needs all ca. 85M parameters of ViT-B, although using a hybrid model can reduce this number by up to roughly one order of magnitude. Note that this combined variant also matches the logit lens and write ratio profiles of procedural warm-up (Figure 15). Finally, we find that it surprisingly seems also have a larger regularization effect, as fewer corrupted labels are memorized (Figure 1c).

4 Related work

Evolution of pretraining of vision models. A recurring theme in (visual) representation learning has been to reduce the reliance on human supervision and hand-crafted inductive biases. Early methods used engineered features such as SIFT (Lowe, 2004) or HOG (Dalal & Triggs, 2005). Later learning representation replaced them enabled by large annotated datasets such as ImageNet

(Krizhevsky et al., 2012; Deng et al., 2009). Subsequent work reduced the need for curated labels through image-text supervision (Radford et al., 2021; Jia et al., 2021) and self-supervised learning (Dosovitskiy et al., 2014; Chen et al., 2020; Caron et al., 2021).

These approaches still all rely on large collections of natural images. This motivated a line of work that pretrained vision models on synthetic or procedurally generated data such as fractals (Kataoka et al., 2020; Nakamura et al., 2023; 2024), contours (Kataoka et al., 2022), and shader images (Baradad Jurjo et al., 2021; Rodríguez-Muñoz et al., 2025). Recently, Shinnick et al. (2026) proposed procedural warm-up in which a ViT is pretrained on non-visual symbolic sequences from formal languages such as Dyck languages.

We investigated how procedural warm-up improves vision performance. We found little evidence for a transferred computational prior; instead, procedural warm-up reshapes layerwise parameter and activation statistics, inducing a lasting regularizing inductive bias during subsequent vision training.

Initialization of (vision) models. Most initialization schemes are designed to stabilize optimization and signal propagation through principled scaling or structural constraints (Glorot & Bengio, 2010; He et al., 2015; Zhang et al., 2019; Huang et al., 2020; Zhao et al., 2022). Other work injects data- or architecture-derived biases, for example by mimicking trained attention patterns Trockman & Kolter (2023), imposing convolutional structure on attention (Giri, 2025; Zheng et al., 2025), or transferring layerwise weight statistics (Raghu et al., 2019). Other work also has indicated that initialization can affect generalization (Zhang et al., 2020; Ramasinghe et al., 2023).

Our work focuses on how initialization shapes generalization through persistent, depth-dependent learning dynamics, and we showed that imposing a small set of such properties at initialization can reproduce the benefits of procedural warm-up.

5 Discussion

Why does the first layer behave differently from other early-to-middle layers? It may seem surprising that the first layer writes substantially more to the residual stream than subsequent layers. One possible explanation is the positional encoding, which in ViT is added only at the input. The first layer is therefore the first opportunity to transform this positional signal and integrate it with the patch representations, which may require a larger residual write than in subsequent early layers.

Why do our interventions sometimes end up with larger gains? One possible explanation is that procedural warm-up retains task-specific structure from the warm-up objective that is not fully beneficial for subsequent vision training. Our interventions reproduce only the initialization properties associated with the downstream gains while otherwise preserving random weights, which may leave the model more adaptable to the new training task.

Limitations. Our goal is to explain which properties of the initialization account for the benefits of procedural warm-up (Shinnick et al., 2026), rather than to propose a new general-purpose initialization method. We identify a small set of structural properties, measure them from procedurally warmed-up models, and show that reproducing them in randomly initialized models recovers much of the performance gain. However, our interventions still rely on target statistics extracted from warmed-up models and therefore do not yet constitute an off-the-shelf initialization scheme. Preliminary results suggest that for example the late-layer residual write target could be set by hand (Appendix L). Following Shinnick et al. (2026), our experiments focus on image classification. It is unclear how broadly our identified mechanisms (or procedural warm-up itself) generalize to other domains. Finally, while the identified properties are sufficient to recover much of the gain, they may not provide a minimal explanation. We present preliminary ablations of the early-layer interventions in Appendix M, but leave a more systematic analysis of required mechanisms to future work.

Conclusion. Procedural warm-up leaves a lasting regularizing bias on how ViTs learn. It reshapes initialization in ways that delay class predictability to later layers and reduces memorization. A small set of initialization properties (sharper attention, suppressed MLPs, altered input weight effective scales, stronger late-layer residual writes) capture much of the regularizing effect and we showed that they are sufficient to reproduce similar gains or even improve. More broadly, our results show that initialization can act as a lasting inductive bias that shapes optimization and generalization throughout training.

AI USE STATEMENT

The original research idea and initial hypotheses (extract few parameter and activation statistics from procedurally warmed-up models, regularizing effect) were developed by the authors without AI assistance after reading Shinnick et al. (2026). Generative AI tools were subsequently used for code generation and editing, feedback on experimental design, support of analysis and interpretation of results, refinement of conceptual explanations, assistance in deriving some mathematical details, additional literature research, and writing (paragraph-level). Beyond that, AI assistance also suggested the "larger residual writes vs. smaller update scales" experiment (Table 3), helped identify the early-layer components analyzed in Section 3.1, and mainly designed the rank-one interventions in Section 3.3. AI-assisted work was reviewed by the authors. AI-generated or modified code was tested for correctness; code underlying important or technically complex experiments, including the effective-scale measurements and the MLP and attention interventions, was manually reviewed by two authors. AI-assisted ideas, mathematical derivations, and interpretations were checked against relevant literature, empirical results, and the authors' own reasoning. We take responsibility for the final content of this work, including all text, claims, code, and other artifacts produced with the aid of generative AI.

REFERENCES

Manel Baradad Jurjo, Jonas Wulff, Tongzhou Wang, Phillip Isola, and Antonio Torralba. Learning to See by Looking at Noise. *NeurIPS*, 2021.

Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *NeurIPS*, 2019.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting Latent Predictions from Transformers with the Tuned Lens. *arXiv preprint arXiv:2303.08112*, 2023.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *ICCV*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, 2020.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

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

Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative Unsupervised Feature Learning with Convolutional Neural Networks. *NeurIPS*, 2014.

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 *ICLR*, 2021.

Adithya Giri. IBiT: Utilizing Inductive Biases to Create a More Data Efficient Attention Mechanism. *arXiv preprint arXiv:2509.22719*, 2025.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *ICCV*, 2015.

Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *ICLR*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. In *ICCV*, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural Adversarial Examples. In *CVPR*, 2021b.

Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. Improving Transformer Optimization Through Better Initialization. In *ICML*, 2020.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In *ICML*, 2021.

Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. Pre-training without Natural Images . In *ACCV*, 2020.

Hirokatsu Kataoka, Ryo Hayamizu, Ryosuke Yamada, Kodai Nakashima, Sora Takashima, Xinyu Zhang, Edgar Josafat Martinez-Noriega, Nakamasa Inoue, and Rio Yokota. Replacing Labeled Real-image Datasets with Auto-generated Contours. In *CVPR*, 2022.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *NeurIPS*, 2012.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *ICLR*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019.

David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004.

Ryo Nakamura, Hirokatsu Kataoka, Sora Takashima, Edgar Josafat Martinez Noriega, Rio Yokota, and Nakamasa Inoue. Pre-training Vision Transformers with Very Limited Synthesized Images. In *ICCV*, 2023.

Ryo Nakamura, Ryu Tadokoro, Ryosuke Yamada, Yuki M Asano, Iro Laina, Christian Rupprecht, Nakamasa Inoue, Rio Yokota, and Hirokatsu Kataoka. Scaling Backwards: Minimal Synthetic Pre-Training? In *ECCV*, 2024.

nostalgebraist. Interpreting GPT: The Logit Lens. *LessWrong*, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021.

Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding Transfer Learning for Medical Imaging. *NeurIPS*, 2019.

Sameera Ramasinghe, Lachlan Ewen Macdonald, Moshiur Farazi, Hemanth Saratchandran, and Simon Lucey. How much does Initialization Affect Generalization? In *ICML*, 2023.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet Classifiers Generalize to ImageNet? In *ICML*, 2019.

Adrián Rodríguez-Muñoz, Manel Baradad, Phillip Isola, and Antonio Torralba. Separating Knowledge and Perception with Procedural Data. In *ICML*, 2025.

Zachary Shinnick, Liangze Jiang, Hemanth Saratchandran, Damien Teney, and Anton van den Hengel. Can You Learn to See Without Images? Procedural Warm-Up for Vision Transformers. In *CVPR*, 2026.

Asher Trockman and J Zico Kolter. Mimetic Initialization of Self-Attention Layers. In *ICML*, 2023.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning Robust Global Representations by Penalizing Local Predictive Power. *NeurIPS*, 2019.

Zhiqiu Xu, Yanjie Chen, Kirill Vishniakov, Yida Yin, Zhiqiang Shen, Trevor Darrell, Lingjie Liu, and Zhuang Liu. Initializing Models with Larger Ones. In *ICLR*, 2024.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. Residual Learning Without Normalization via Better Initialization. In *ICLR*, 2019.

Yaoyu Zhang, Zhi-Qin John Xu, Tao Luo, and Zheng Ma. A type of generalization error induced by initialization in deep neural networks. In *MSML*, 2020.

Jiawei Zhao, Florian Tobias Schaefer, and Anima Anandkumar. ZerO Initialization: Initializing Neural Networks with only Zeros and Ones. *TMLR*, 2022.

Jianqiao Zheng, Xueqian Li, Hemanth Saratchandran, and Simon Lucey. Structured Initialization for Vision Transformers. In *NeurIPS*, 2025.

Table 4: **Training hyperparameters for the procedural pretraining stage.**

Setting	Value
Batch size	256
Training steps	15,000
Mask ratio	0.5 (close-only)
Optimizer	AdamW
Learning rate	$2e^{-3}$
Weight decay	0.05
Betas	(0.9, 0.999)
LR schedule	Cosine decay
Warmup steps	1,000

Table 5: **Training hyperparameters for the image training stage.**

Setting	Value
Optimizer	AdamW
LR schedule	Cosine decay
Warmup	Linear (50 epochs)
Base learning rate	$2e^{-3}$
Epochs	300
Batch size	512 (4096 for ViT-B / ImageNet-1k)
Augmentation	RandAugment, Mixup, CutMix, label smoothing

A Experimental setup

Training setup The pipeline uses a two-stage training setup, procedural pretraining followed by image classification training, applied to two ViT backbones (ViT-S and ViT-B), both with patch size 16 on 224×224 images. In procedural pretraining, the ViT (without a class token) is trained on masked k-Dyck bracket sequences (k=64) generated on-the-fly, with patch and positional embeddings frozen. Only the transformer blocks and classification head are updated. Training uses AdamW (Loshchilov & Hutter, 2019) with linear warmup and cosine annealing (Loshchilov & Hutter, 2017) for a fixed number of steps (Table 4). After pretraining, the transformer block weights are transferred to initialize the image-training model, while the classification head, class token, positional embeddings, and patch embeddings are randomly initialized.

For image training, the transferred blocks are fine-tuned with no frozen components, using the optimizer, augmentation, LR schedule, and warmup/epoch counts from Shinnick et al. (2026) (following the ViT training recipe of Xu et al. (2024)), summarized in Table 5. ImageNet-100 is used for ViT-S and ImageNet-1k (Deng et al. (2009)) for ViT-B. ViT-S configurations are each run over 3 seeds, while ViT-Base runs are mostly limited to a single seed due to higher computational costs.

Procedural data Table 6 presents one example for each of the four possible configurations of k-Dyck with shuffling and truncation at a given depth.

B Performance across k-Dyck variants and model size

We trained ViT-S for image classification on ImageNet-100 following procedural warm-up with eight k-Dyck variants. These variants span combinations of depth (4, 98), truncation (truncated vs. not truncated), and shuffling (shuffled vs. unshuffled). ViT-B is trained using the two top-performing variants from ViT-S: k-Dyck D4 and k-Dyck Shuffled D98. The grammar and structure of the procedural data influence the performance of the downstream model, but across the eight procedural k-Dyck variants tested, the exact relationship is complex and cannot easily be attributed to any single property of the procedural data.

Table 7 provides the performance of ViT-S or ViT-B after random initialization or different procedural warm-up variants.

Table 6: **The four truncation/shuffling variants of k-Dyck at a given depth D.** Cell entries give the reference name by which each variant is denoted elsewhere in the text, together with an example sequence for $k = 3$ bracket types, sequence length $L = 8$, and depth $D = 2$.

	not shuffled	**shuffled**
not truncated	*k-Dyck* { [ ] ( ) } ( )	*k-Dyck Shuffled* { [} ( ] ( ))
truncated	*k-Dyck Truncated* { [ ] ( ) } ([	*k-Dyck Truncated Shuffled* { [} ( ] ( ) [

Table 7: **Performance of procedural warm-up across k-Dyck variants.** Performance varies substantially across procedural data variants.

Initialization	Accuracy
ViT–S	
Random initialization	84.37±0.11
k-Dyck - D4	86.09 ± 0.55
k-Dyck - D98	85.61±0.48
k-Dyck Truncated - D4	85.60±0.54
k-Dyck Truncated - D98	85.85±0.59
k-Dyck Shuffled - D4	84.31±0.64
k-Dyck Shuffled - D98	86.37±0.39
k-Dyck Truncated Shuffled - D4	85.93±0.35
k-Dyck Truncated Shuffled - D98	83.95±0.71
ViT–B	
Random initialization	78.08±0.17
k-Dyck - D4	80.08±0.12
k-Dyck Shuffled - D98	80.20

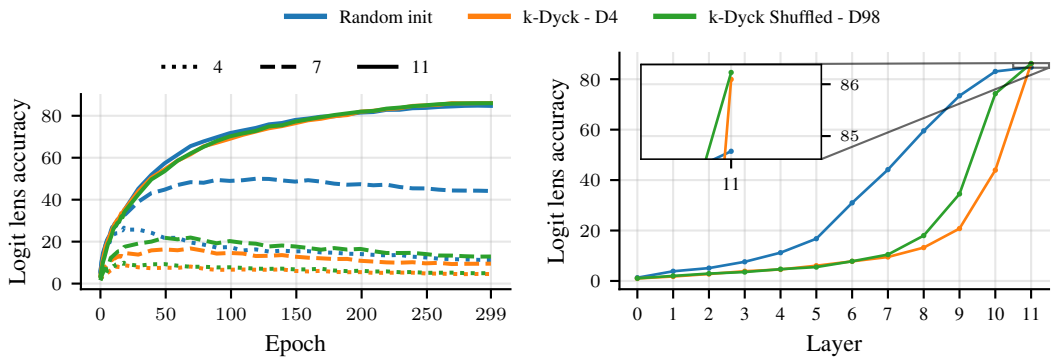
C Additional logit lens results

Figure 4 shows a similar delayed class decodability for ViT-S, as we found for ViT-B (Figures 1a and 1b). Figure 5 correlates final accuracies with the logit lens results for different type of hybrid models for ViT-B (see Appendix E for details). Figure 6 correlates final accuracies and logit lens results of different procedural warm-ups for ViT-S. For both figures, we surprisingly find a relatively consistent pattern between final accuracies and logit lens, i.e., the better the final accuracy, the later classes are decodable from intermediate `cls` tokens. We leave further investigation of this for future work.

D Restricting classifier readouts to late layers

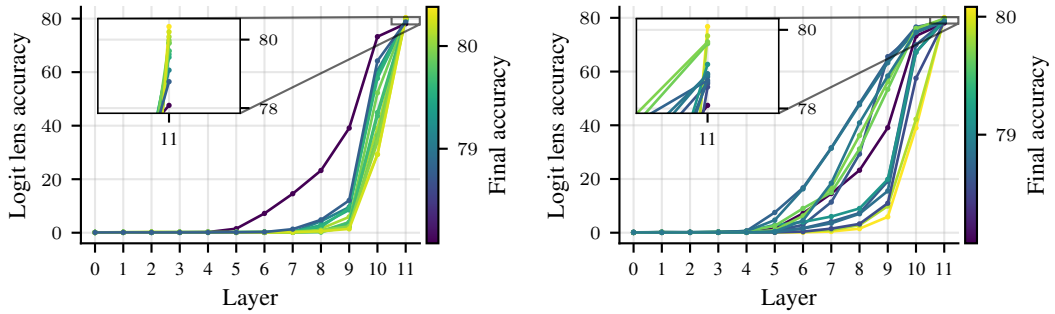
For the readout restriction experiment, we trained a randomly initialized ViT-B on ImageNet-1k with the same recipe as the random initialization baseline (Appendix A). The only change is that the classifier head (final LayerNorm and linear layer, applied to the class token) receives the sum of the attention and MLP outputs of layers 9–11 instead of the full residual stream. All layers still read and write the full residual stream, so earlier layers can only influence the prediction through/mediated by layers 9–11.

Figure 7 shows that this reproduces the characteristic logit lens profile of procedurally warmed-up models while also improving final validation performance by 1.3 pp.



(a) Logit lens accuracy during training. (b) Logit lens accuracy after training.

Figure 4: **ViT-S with procedural warm-ups resemble the layer-wise logit lens of ViT-B after training.** Random initialization, k-Dyck D4, k-Dyck Shuffled D98 in ViT-S for ImageNet-100. a: Procedural warm-up maintains low logit lens accuracy in the early and middle layers. The accuracies of layers 4, 7 of the randomly initialized model rises early and declines as training progresses. b: Logit lens accuracy at the end of training. Accuracy under procedural warm-up increases steeply only in the late layers while randomly initialized model shows a gradual increase.



(a) ViT-B
Procedural warm-up / Random initialization

(b) ViT-B
Random initialization / Procedural warm-up

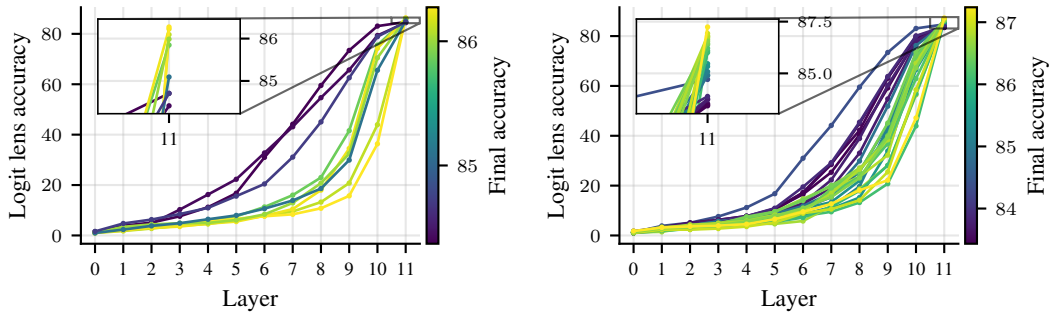
Figure 5: **ViT-B: Logit lens curves correlate with accuracy when procedural weights are introduced from the early layers.** a: Runs with higher final accuracy have high logit lens accuracy only in later layers, showing an ordering of layer-wise prediction decodability with final performance. b This correlation is not distinct when procedural weights are introduced from late layers onward. (Configurations from Figure 9b)

E STITCHED, HYBRID MODELS

Figure 8 illustrates how stitched together the randomly initialized (row 1) and procedurally warmed-up models (row 2). For the hybrid model (row 3), we used the weights from random initialization, here, for the first nine layers and procedurally warmed-up weights for the remaining three layers. To test sufficiency of residual write ratios (row 4, Equation 1), we match ratios of the randomly initialized model (row 1) to the hybrid model (row 3). To test necessity (row 5), we do the opposite: we match ratios of the hybrid model (row 3) to the ratios measured on the randomly initialized model (row 1). Details of the matching procedure are provided in Appendix G.

F RESULTS FOR STITCHED, HYBRID MODELS

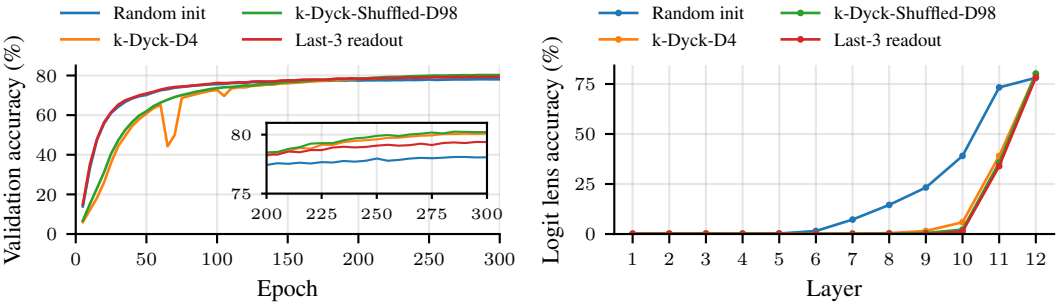
For scaling the late layer residual write ratios, we need a reference model from which to obtain the target ratios. One option would be to use the fully procedurally warmed-up model. However, this would introduce a mismatch: if we only transfer the late-layer ratios to a randomly initialized model, the residual stream entering those layers is generated by random rather than procedurally



(a) ViT-S
Eight k-Dyck variants

(b) ViT-S
All hybrid models from Figure 9a for k-Dyck - D4

Figure 6: **ViT-S: Logit lens curves for high performing models are observed to be lower, with high accuracy jumps in the late layers.** This pattern is similar to that observed in ViT-B with models that are procedurally warmed-up. The accuracy of models increase sharply from layer 9 onwards.



(a) Validation accuracy over training.

(b) Logit lens profiles after training.

Figure 7: **Effect of restricting classifier head readouts in a randomly initialized model to the last three layers.** The restricted model (red) closely follows the logit lens profiles of the procedurally warmed-up models (green and orange in b) while tracking the early training validation accuracies of the random initialization (blue) but ultimately ending about 1.3 pp higher at the end of training (a).

warmed-up early layers. From the perspective of the fully warmed-up model, the preceding layers have also been replaced, which can substantially change the scale of the residual stream and make its late-layer write ratios a poor reference.

Thus, we constructed hybrid initializations by combining procedurally warmed-up and randomly initialized layers (see Figure 8 for schematics). Specifically, for a split point ℓ, we initialized layers $0, \ldots, \ell$ from random initialization and layers $\ell + 1, \ldots, L$ from procedural warm-up. We also considered the reverse, i.e., first procedural warm-up and afterward random weights. We then trained this hybrid models using the standard image training procedure (Appendix A).

Figure 9 shows that these hybrid initializations achieve performance comparable to, and in some cases even better than, the fully procedurally warmed-up initialization.

G Sequential per-layer rescaling of residual write ratios

Algorithm 1 provides pseudocode for the sequential rescaling procedure of residual write ratios. Specifically, we scale the weights $W \subseteq \{W_v, W_{proj}, W_{fc2}\}$ of a base model M to match the residual write ratios of a target model M^* over specific layers. When we have $n > 1$ weight matrices per block, we scale each weight matrix by the n-th root of the write ratio (lines 11 & 19) so that we distribute scaling evenly. Because upstream modification affect downstream activations (particularly, the residual stream), we re-measure write ratios of the current sublayer in M before matching it to the target ratio of M^* at that sublayer.

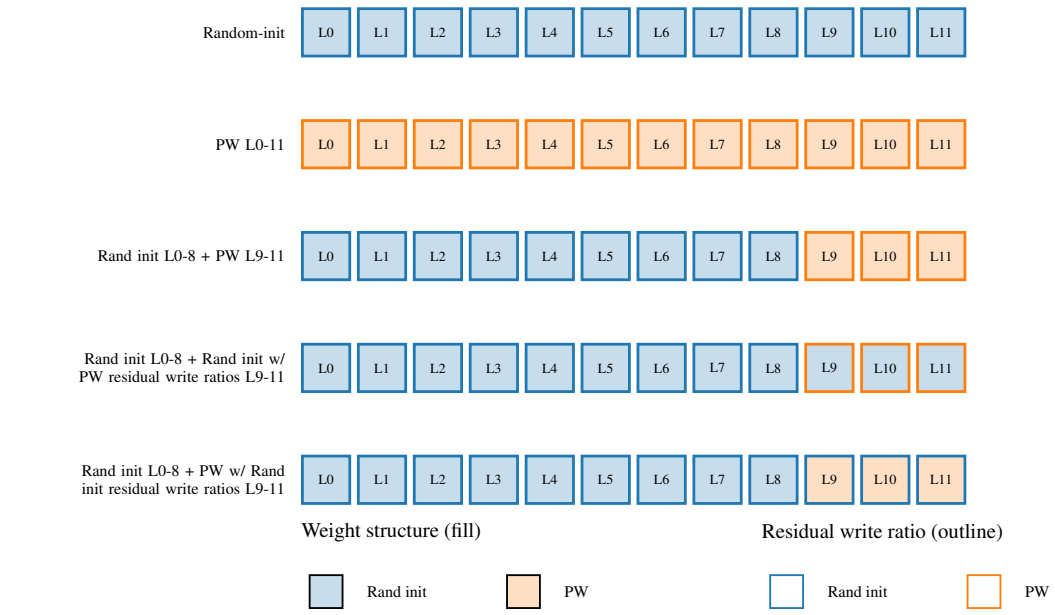
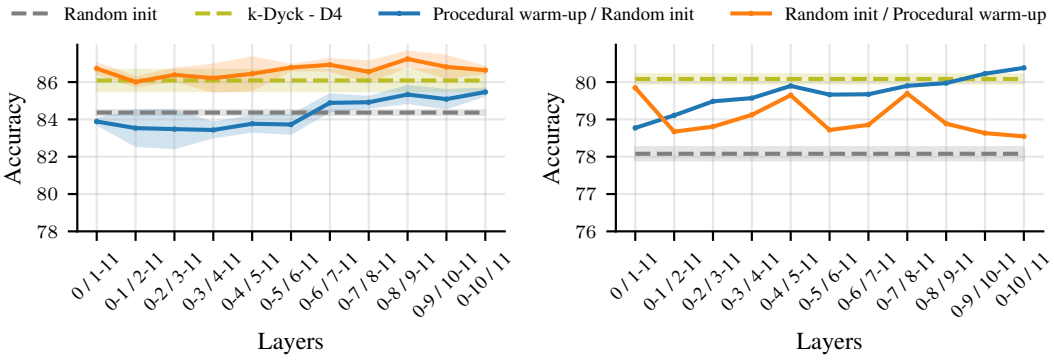


Figure 8: **Illustration of hybrid models and how we tested for sufficiency and necessity.** Each row is a weight configuration: randomly initialized model (row 1), procedurally warmed-up model (row 2), hybrid model (row 3), model to test sufficiency (row 4), and model to test necessity (row 5).



(a) ViT-S on ImageNet-100.

(b) ViT-B on ImageNet-1k.

Figure 9: **Late layers yield stable performance gains in ViT-S, but early layers remain crucial for ViT-B.** Layer ablation on ViT-S and ViT-B across model depth. a: In ViT-S, late layers drive performance gains, early layers alone are insufficient. b: Early layers of ViT-B provide stable performance gains across model depth, whereas late-layer benefits are inconsistent.

The sufficiency experiments used a randomly initialized M and a procedurally warmed-up M^*, while the necessity experiment reverses these roles. Both configurations of the ViT-S experiments in Section 3.2 use late layers 9–11 and weight matrices $W \subseteq \{W_v, W_{proj}, W_{fc2}\}$. ViT-B experiments in Table 2 use the same weight matrices with layers 9–11 for k-Dyck D4 and layers 8–11 for k-Dyck Shuffled D98.

Why this choice of weight parameters? We limit this scaling to only few weight matrices because some weight matrices, e.g., query or key weight matrices, not only affect the output scale but also the sharpness of attention. Similarly, in the MLP block, we scale only the second linear layer (`fc2`).

Algorithm 1 Sequential per-layer scaling to match residual attention and MLP -write ratios

Require: model M whose weights are to be rescaled; target model M^* whose residual write ratios are to be matched; selected layers L, ordered from earliest to latest; weight matrices W

$W_{\text{Attn}} \leftarrow \{W_v,\ W_{\text{proj}}\};\ W_{\text{MLP}} \leftarrow \{W_{\text{fc2}}\}$

$r^*_{\text{Attn}} \leftarrow \{\};\ r^*_{\text{MLP}} \leftarrow \{\}$

for $l \in L$ **do** ▷ record residual write ratios of target model M^*

  $r^*_{\text{Attn}} \leftarrow r^*_{\text{Attn}} \cup \frac{||\text{Attn}_l(x_l)||_2}{||x_l||_2};\ r^*_{\text{MLP}} \leftarrow r^*_{\text{MLP}} \cup \frac{||\text{MLP}_l(\tilde{x}_l)||_2}{||\tilde{x}_l||_2}$ ▷ Equation 1

end for

for $l \in L$ **do** ▷ earliest layer is scaled first

  $r \leftarrow \frac{||\text{Attn}_l(x_l)||_2}{||x_l||_2}$ ▷ measured against M's current initialization

  if $|W_{\text{Attn}}| > 0$ **then**

    $s_{\text{Attn}} \leftarrow \left(\frac{r^*_{\text{Attn}}[l]}{r}\right)^{1/|W_{\text{Attn}}|}$

    for $W \in W_{\text{Attn}}$ **do** ▷ scale attention weight matrices W_{Attn} of block l in M

      $W_l \leftarrow s_{\text{Attn}} \cdot W_l$

    end for

  end if

  $r \leftarrow \frac{||\text{MLP}_l(\tilde{x}_l)||_2}{||\tilde{x}_l||_2}$ ▷ re-measured because preceding attention block might have changed

  if $|W_{\text{MLP}}| > 0$ **then**

    $s_{\text{MLP}} \leftarrow \left(\frac{r^*_{\text{mlp}}[l]}{r}\right)^{1/|W_{\text{MLP}}|}$

    for $W \in W_{\text{MLP}}$ **do** ▷ scale MLP weight matrices W_{MLP} of block l in M

      $W_l \leftarrow s_{\text{MLP}} \cdot W_l$

    end for

  end if

end for

return M

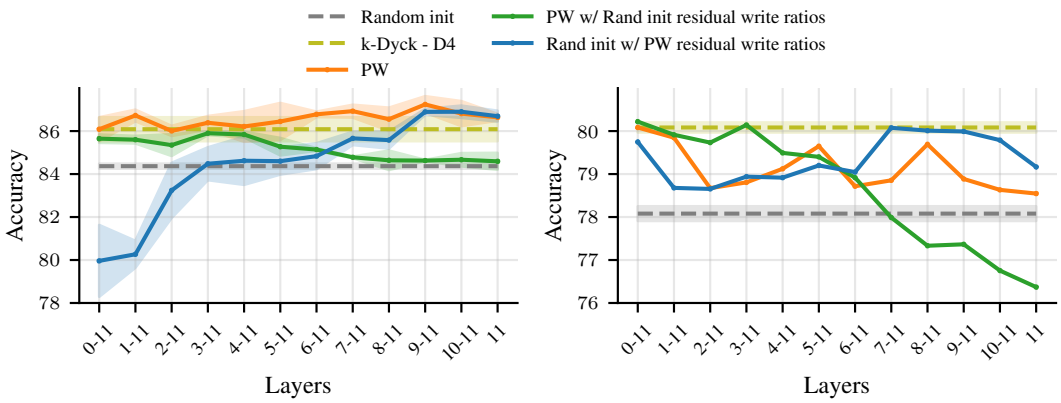
H Additional late layer scaling results

We extend the sufficiency and necessity experiments of Section 3.2 across model depth for k-Dyck-D4 on ViT-S and ViT-B. Figure 10 shows results for varying split points of the hybrid model (Appendix E). In general, matching write ratios of the randomly initialized model to the procedural variants at late layers (blue line) achieves accuracies matching or exceeding full procedural warm-up. Including more and more earlier layers, however, degrades performance; particularly, for ViT-S. The opposite happens when we match write ratios from the randomly initialized model for the hybrid models (green): performance degrades when late layers are scaled. In contrast, scaling early layers is more stable.

We repeated the experiments for ViT-S on ImageNet-100 also across a range of formal languages. Specifically, we varied the grammar type (context–free vs. context-sensitive), maximum k-Dyck nesting depth (4 vs. 98), and whether the generated symbolic sequence was truncated to the fixed token budget (truncated or not). Figure 11 shows that the result is consistent across these variations.

I Additional results of residual write ratios at initialization and after training

Figure 12 shows the attention and MLP write ratios of ViT–B at initialization and after training for a randomly initialized model and two procedural warm-ups k-Dyck D4 and k-Dyck Shuffled D98. Under procedural warm-up, attention and MLP write ratios decrease in early layers and increase in late layers. Figure 13 shows a similar profile for procedural warm-ups in ViT-S, with a clear increase in the late-layer residual write ratios after training relative to a randomly initialized model.



(a) ViT–S on ImageNet-100 (b) ViT–B on ImageNet-1k

Figure 10: **Matching late-layer residual write ratios improves random initialization performance.** Scaling the attention value, attention output, and MLP output weight matrices in late layers to match the residual write ratios of a hybrid procedurally warmed-up model improves model performance. The needed number of scaled late layers varies depending on the model and procedural variant.

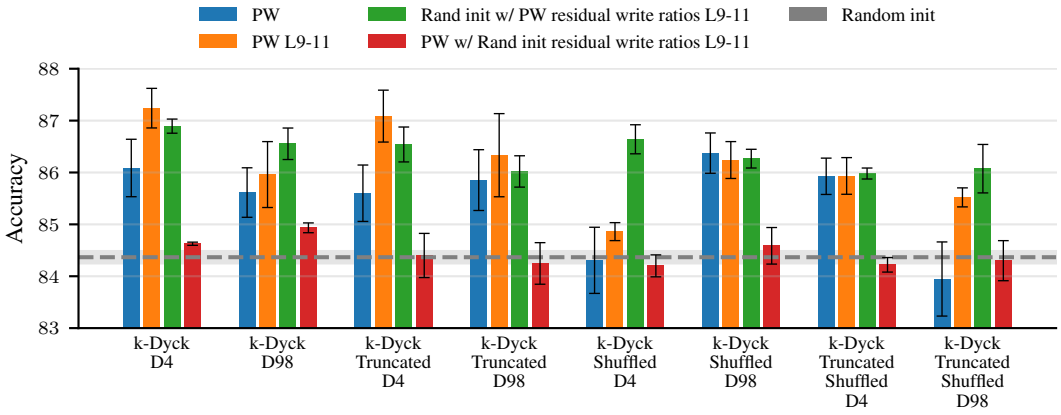
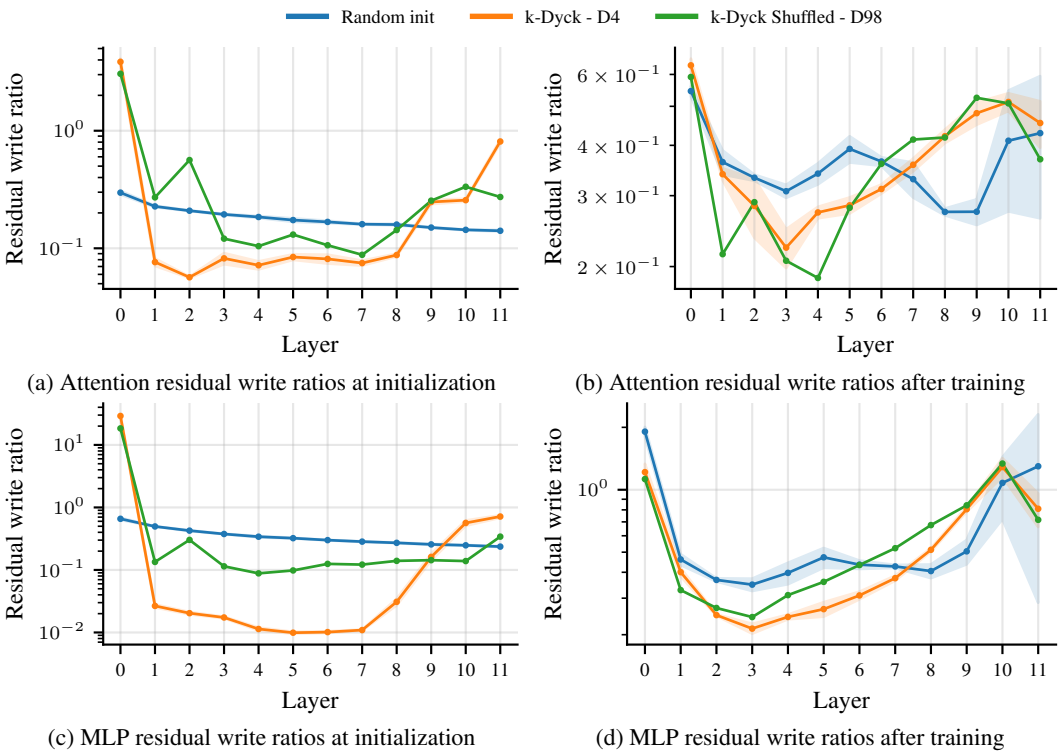


Figure 11: **Late-layer residual write ratios are necessary and sufficient to match or exceed procedural performance gains in ViT–S.** The residual write ratios from PW layers 9–11 alone can replicate procedural performance, while removing them reverts the model to near randomly initialized levels. Layers 0–8 remain randomly initialized across the orange, green, and red configurations.

J Effect of our late and early interventions on logit lens and residual write ratios

Large residual write match the logit lens and residual write ratios results of procedural warm-up. Table 2 shows that ViT-B's accuracy increases when the late layers are scaled to match the residual write ratios of the hybrid procedurally warmed-up model. Figure 14 shows the logit lens curves and residual write ratios for both models (*Random init w/ PW residual write ratios*, k-Dyck D4: 9–11; k-Dyck Shuffled D98: 8–11). They match the results we have seen for the procedurally warmed-up model (Figures 1a, 1b and 12).

Early and middle layer interventions match logit lens and residual write ratio results of procedural warm-up. Figure 15 shows that when we match the early parameter or activation targets (effective weight scales, LayerNorm gains and biases, attention entropy, MLP pre-activations) they also match the logit lens and residual write ratio patterns we found for procedurally warmed-up ViT-B models (Figures 1a, 1b and 12).



(a) Attention residual write ratios at initialization

(b) Attention residual write ratios after training

(c) MLP residual write ratios at initialization

(d) MLP residual write ratios after training

Figure 12: **Compared to randomly initialized models, procedurally warmed-up models show decreased write ratios in early layers and increased ratios in later layers after training.** Attention and MLP residual write ratios of the procedural warm-ups start from a distinct layer-wise profile. They evolve differently from a randomly initialized model during training.

Both combined also follow the patterns of procedural warm-up. Figure 15 also shows that combining both matches the patterns found for procedurally warmed-up ViT-B models.

K Matching early-layer attention entropy and MLP pre-activation targets

To match attention entropy and MLP pre-activations, we record them and then sequentially update the query/key and `fc1` weight matrices from earlier to later layers. Below, we describe the procedures in detail.

K.1 Renormalization

Both interventions combine an existing weight matrix W by adding a rank-one C:

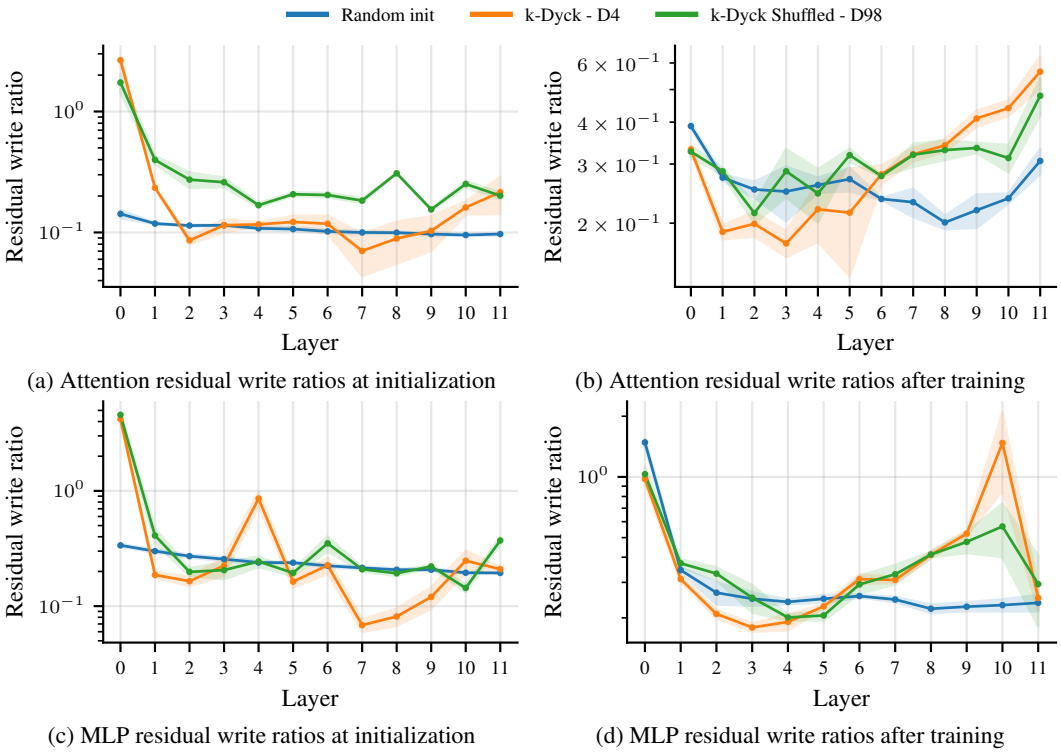
$$W' = s(\lambda)W + \lambda C, \qquad \lambda \geq 0. \tag{6}$$

Here, W already has the effective scale that we set (see Section 3.3). To preserve the effective weight scale, we therefore choose $s(\lambda) > 0$ such that

$$\left\|(s(\lambda)W + \lambda C)G\right\|_F^2 = \|WG\|_F^2 \quad , \tag{7}$$

where $G = \mathrm{diag}(\gamma)$ is the preceding LayerNorm gain. This preserves the effective weight scale because the Frobenius norm and elementwise RMS differ only by a constant factor for matrices of the same dimensions. Expanding the constraint gives a scalar quadratic in $s(\lambda)$, for which we take the positive root. To ensure that this solution exists and remains strictly positive, we restrict search to

$$0 \leq \lambda \leq 0.98 \, \frac{\|WG\|_F}{\|CG\|_F} \quad . \tag{8}$$



(a) Attention residual write ratios at initialization

(b) Attention residual write ratios after training

(c) MLP residual write ratios at initialization

(d) MLP residual write ratios after training

Figure 13: **ViT-S with procedural warm-ups show larger late-layer residual write profiles similar to ViT-B after training.** Random initialization, k-Dyck D4, k-Dyck Shuffled D98 in ViT-S for ImageNet-100. a,c: Unlike ViT-B, the early layers lack a distinct low-write pattern. b,d: The late layers maintain larger write ratios than the randomly initialized model, suggesting the critical role of late layers in ViT-S.

The factor 0.98 provides a small safety margin for numerical stability at the boundary. LayerNorm gains and biases are kept fixed.

K.2 Matching attention entropy

Let $Y_\ell = \mathrm{LN}_{\mathrm{Attn},\ell}(X_\ell)$ be the LayerNorm outputs, and let

$$c_\ell = \frac{\mathbb{E}_{I,t}[[Y_\ell]_t]}{\|\mathbb{E}_{I,t}[[Y_\ell]_t]\|_2} \tag{9}$$

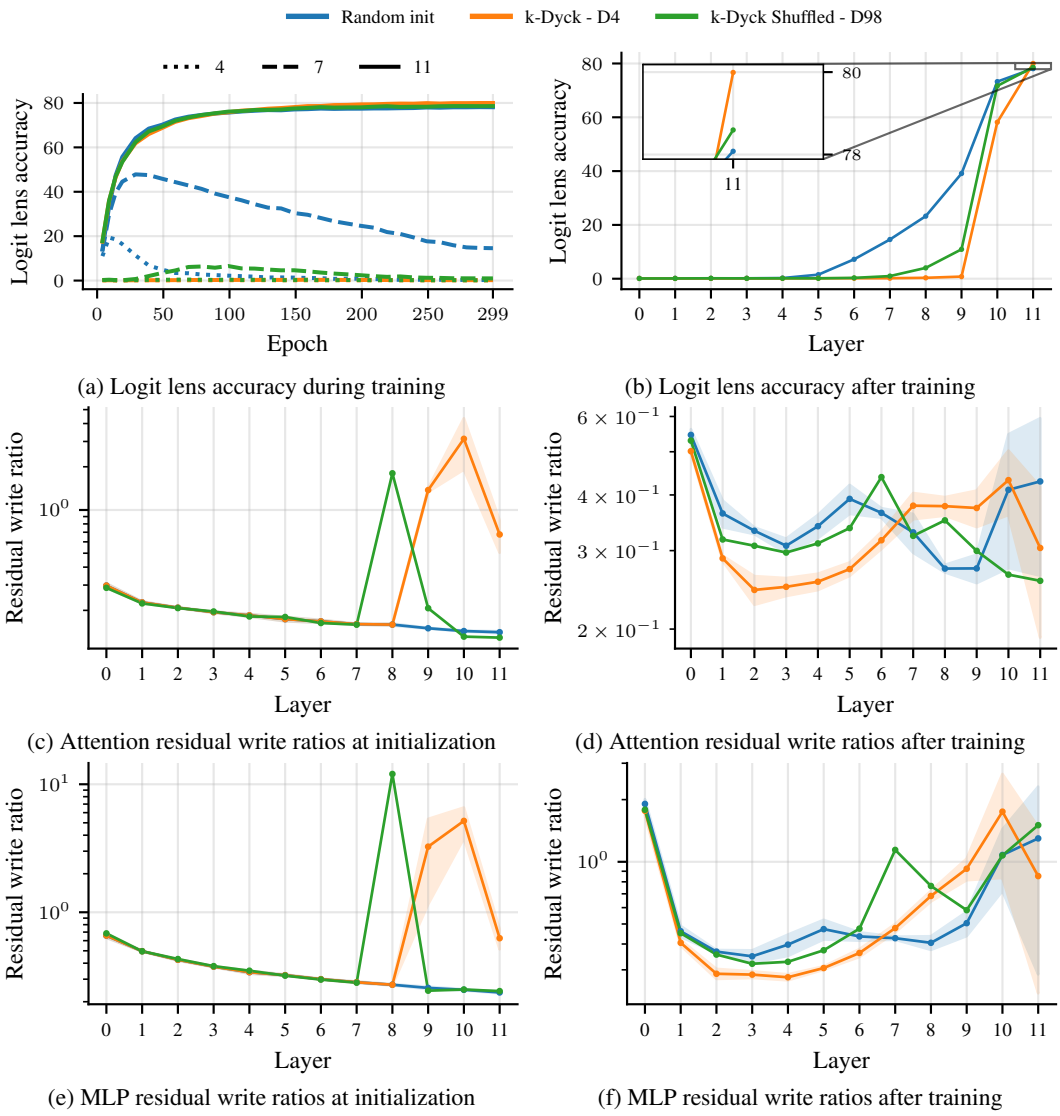
be their normalized mean direction, where we take expectation over images I and all tokens t. For each head h, we sample a random unit vector $p_{\ell,h} \in \mathbb{R}^{d_{\mathrm{head}}}$ and concatenate these vectors into $P_\ell \in \mathbb{R}^d$, where $d = d_{\mathrm{head}} \times n_{\mathrm{heads}}$ is the model width. We also sample a random unit vector $r_\ell \in \mathbb{R}^d$.

We then modify the query and key projections as follows:

$$\begin{aligned} W'_{\ell,Q} &= s_{\ell,Q} W_{\ell,Q} + \lambda_\ell P_\ell c_\ell^\top, \\ W'_{\ell,K} &= s_{\ell,K} W_{\ell,K} + \lambda_\ell P_\ell r_\ell^\top. \end{aligned} \tag{10}$$

Both modifications share λ_ℓ, but have separate rescaling factors to preserve their respective effective scales (see Section K.1 for details). We therefore restrict λ_ℓ using the smaller of the two corresponding upper bounds from Equation 8.

This construction encourages different queries to attend to the same keys (see Figure 16 for an example). To see why, let $y_{\ell,t} = [Y_\ell]_t$. In head h, the added query component for token t_1 is $\lambda_\ell p_{\ell,h}(c_\ell^\top y_{\ell,t_1})$ and key component for token t_2 is $\lambda_\ell p_{\ell,h}(r_\ell^\top y_{\ell,t_2})$. Because the query and key



(a) Logit lens accuracy during training

(b) Logit lens accuracy after training

(c) Attention residual write ratios at initialization

(d) Attention residual write ratios after training

(e) MLP residual write ratios at initialization

(f) MLP residual write ratios after training

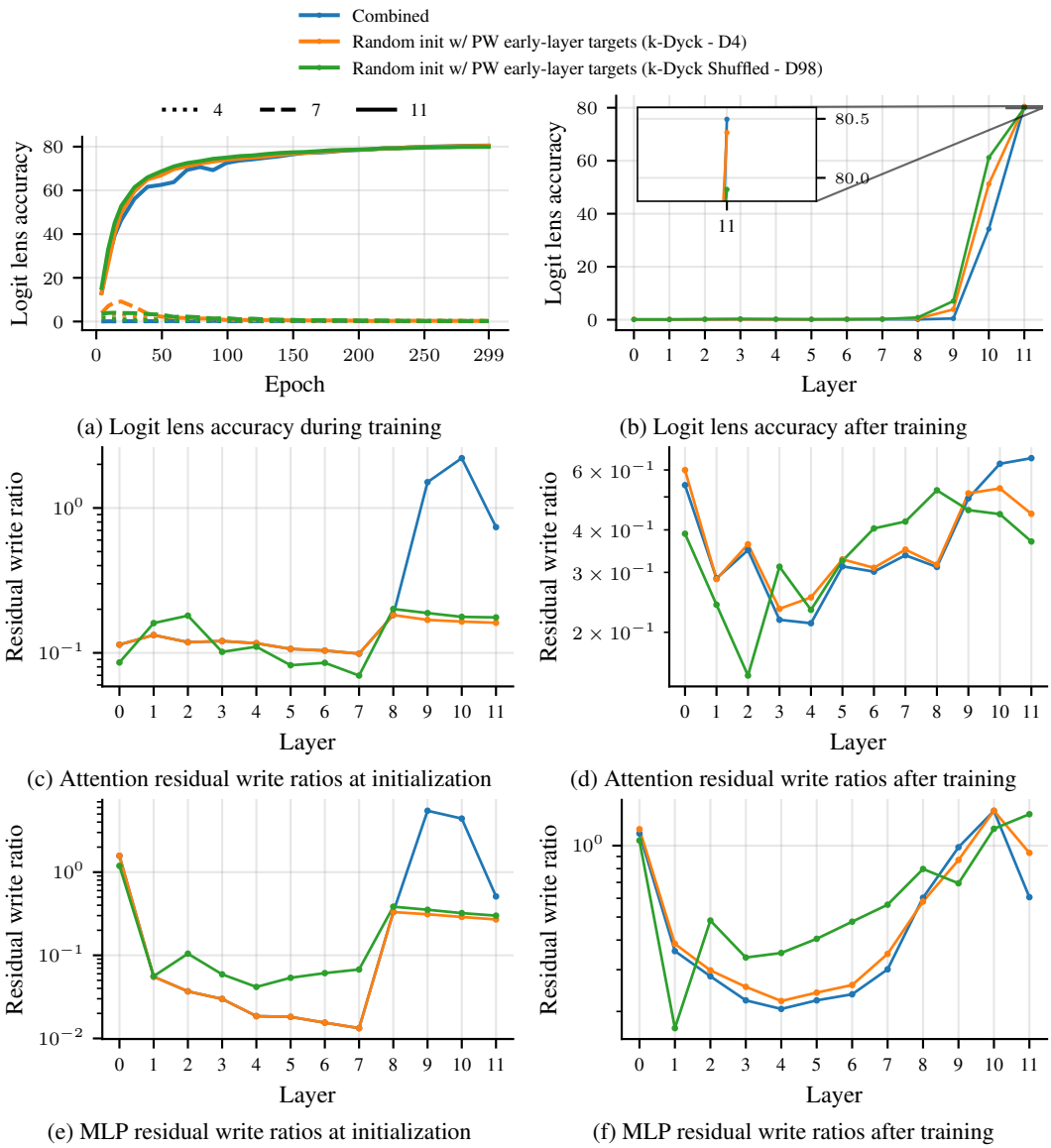
Figure 14: **Modifying a randomly initialized model's late layers to match procedural warm-up increases performance (Table 2) over random initialization and also matches the logit lens and residual write ratio dynamics observed in Section 2.2 and Figure 12.**

components use the same direction $p_{\ell,h}$, their dot product contributes

$$\frac{\lambda_\ell^2}{\sqrt{d_{\text{head}}}}(c_\ell^\top y_{\ell,t_1})(r_\ell^\top y_{\ell,t_2}) \tag{11}$$

to the attention logit from query t_1 to key t_2. For queries with $c_\ell^\top y_{\ell,t_1} > 0$, this term induces the same ranking of keys, determined by $r_\ell^\top y_{\ell,t_2}$. With large λ_ℓ, attention therefore tends to concentrate on common keys.

Finally, we choose λ_ℓ by bisection so that the mean attention entropy H_ℓ (Equation 2) matches that of the corresponding procedurally warmed-up model. Attention entropy is evaluated using the fully modified projections, including the original contributions and their cross terms with the added components.



(a) Logit lens accuracy during training

(b) Logit lens accuracy after training

(c) Attention residual write ratios at initialization

(d) Attention residual write ratios after training

(e) MLP residual write ratios at initialization

(f) MLP residual write ratios after training

Figure 15: **Matching early targets alone is sufficient.** Random init w/ PW early-layer targets (k-Dyck-D4), Random init w/ PW early-layer targets (k-Dyck Shuffled-D98), or combined with the late layer residual write ratio scaling (Combined) is sufficient to reproduce performance gains of procedural warm-up (Figure 3) and also matches the logit lens and residual write ratio dynamics observed in Section 2.2 and Figure 12.

K.3 Matching MLP pre-activations

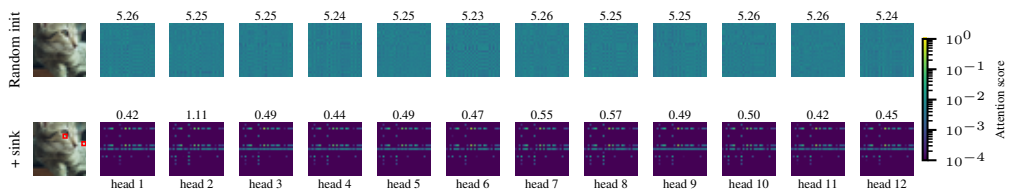
Let $\widetilde{X}_\ell = X_\ell + \mathrm{Attn}_\ell(X_\ell)$. We define the normalized MLP inputs and their mean direction as

$$Y_\ell^{\mathrm{MLP}} = \mathrm{LN}_{\mathrm{MLP},\ell}(\widetilde{X}_\ell), \qquad \bar{y}_\ell^{\mathrm{MLP}} = \mathbb{E}_{I,t}\left[[Y_\ell^{\mathrm{MLP}}]_t\right], \qquad c_\ell^{\mathrm{MLP}} = \frac{\bar{y}_\ell^{\mathrm{MLP}}}{\|\bar{y}_\ell^{\mathrm{MLP}}\|_2} \quad . \tag{12}$$

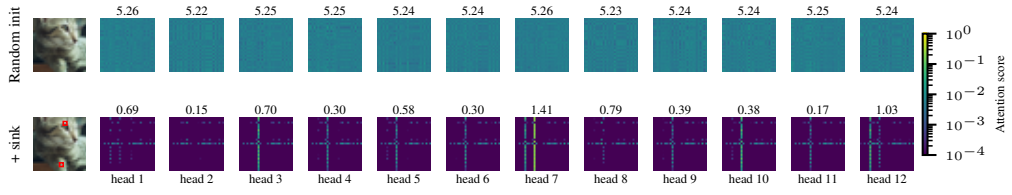
As for the attention entropy matching, the mean is taken over images and all token positions.

To shift the MLP pre-activations downward, we modify the first MLP projection as follows:

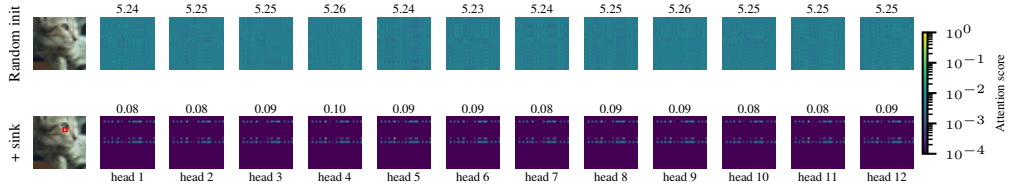
$$W'_{\ell,\mathrm{fc1}} = s_{\ell,\mathrm{fc1}} W_{\ell,\mathrm{fc1}} - \frac{\lambda_\ell^{\mathrm{MLP}}}{\sqrt{d_{\mathrm{MLP}}}} \mathbf{1}_{d_{\mathrm{MLP}}} (c_\ell^{\mathrm{MLP}})^\top, \qquad \lambda_\ell^{\mathrm{MLP}} \geq 0, \tag{13}$$



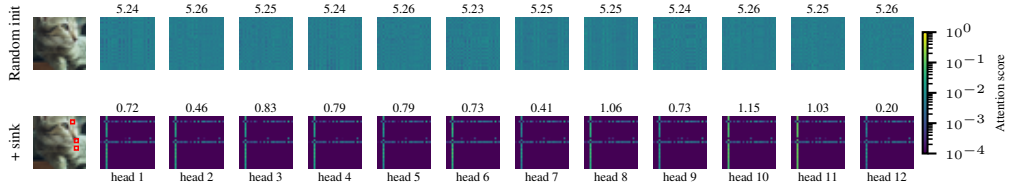
(a) Layer 1.



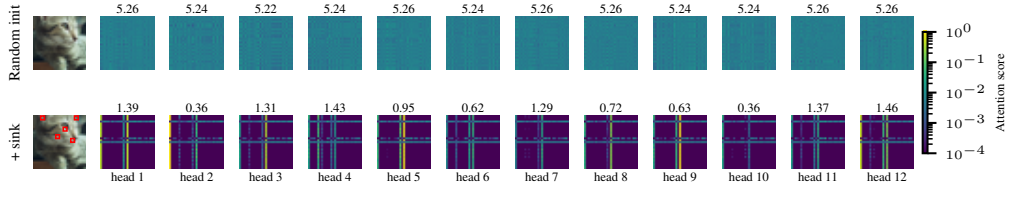
(b) Layer 2.



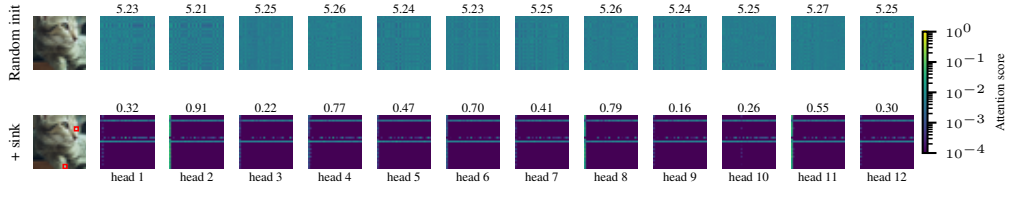
(c) Layer 3.



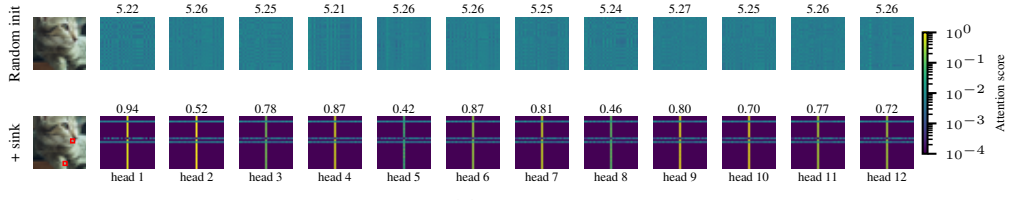
(d) Layer 4.



(e) Layer 5.



(f) Layer 6.



(g) Layer 7.

Figure 16: **The attention sink at initialization.** Attention before and after applying the procedure. Numbers above the panels are each head's attention entropy in nats (uniform attention: $\ln 197 = 5.28$). Red boxes mark the patches that the heads attend to most.

Table 8: **Late-layer residual write ratios can also be manually configured.** Final ImageNet top-1 (%, epoch 300; improvement over random in brackets) of a random ViT-B whose blocks 9–11 are modified to increase residual writes. (Section 3.2). In each of these blocks we scale W_v, W_{proj} and W_{fc2} until every sub-layer writes the target ratio. The target procedural ratios are taken from blocks 9–11 of the k-Dyck D4 checkpoint, where blocks 0–8 remain randomly initialized.

	Write ratio (blocks 9 / 10 / 11)		
Blocks 9–11	attention	MLP	top-1
Random	–	–	78.08
Procedural's ratios	1.38 / 2.10 / 0.78	4.76 / 4.46 / 0.53	79.99 (+1.91)
Set 1.4 manually	1.4 / 1.4 / 1.4	1.4 / 1.4 / 1.4	79.69 (+1.61)

Table 9: **Which components of the early-block initialisation are needed.** Final ImageNet top-1 (%, epoch 300) of ViT-B when blocks 0–7 get only the ticked components. *Effective weight scales/LN*: the checkpoint's per-tensor effective scales and LayerNorm statistics. *Attention entropy*: a rank-one attention sink calibrated to the checkpoint's attention entropy. *MLP pre-activations*: a rank-one term calibrated to the checkpoint's mean MLP pre-activation.

Component (blocks 0–7)			k-Dyck-D4	
Effective weight scales/LN	attention entropy	MLP pre-activations	top-1	Δ random
✗	✗	✗	78.08	–
✓	✗	✗	79.90	+1.82
✗	✓	✗	79.06	+0.98
✗	✗	✓	78.38	+0.30
✓	✓	✗	79.78	+1.70
✓	✗	✓	79.91	+1.83
✗	✓	✓	79.53	+1.45
✓	✓	✓	80.36	+2.28
Procedurally warmed-up model			80.08	+2.00

where d_{MLP} is the hidden MLP dimension. The factor $1/\sqrt{d_{\mathrm{MLP}}}$ normalizes the rank-one component to unit Frobenius norm, so that $\lambda_\ell^{\mathrm{MLP}}$ controls its strength independently of the d_{MLP}. We choose $s_{\ell,\mathrm{fc1}}$ using the procedure from Section K.1 with $G = \mathrm{diag}(\gamma_{\mathrm{MLP},\ell})$, thereby preserving the effective `fc1` weight scale e_ℓ^{MLP}.

L Late-layer scaling

Table 8 shows that we do not necessarily need the actual residual write ratios but could instead set an arbitrary scalar (here, 1.4) manually. We leave further investigation into how best to set this, including the other target statistics from the early and middle layers, for future work.

M Ablation of components in the earlier layer interventions

Table 9 removes the three components from the early-to-middle layer interventions (Section 3.3). It shows that MLP pre-activations alone are insufficient (0.3 pp better than random). However, combined with either the effective weight scales/LN or sparser attention it is more effective. In contrast, attention entropy and, in particular, the effective weight scales/LayerNorm statistics alone lead to larger improvements. We leave further investigation to future work.