RETHINKING ACTIVATION FUNCTIONS: REDUCING BIAS CAN IMPROVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Many widely used activation functions, including ReLU, GELU, Swish, and Sigmoid, produce predominantly positive outputs, resulting in a nonzero post-activation mean that we term *activation bias*. Despite its ubiquity, the effect of activation bias on learning dynamics remains poorly understood. Using controlled models, we show that activation bias accelerates initial loss reduction but is accompanied by overshooting, non-smooth trajectories, and a steeper loss landscape. This early advantage also coincides with the model's use of task-relevant information encoded in the activation bias component. In contrast, reducing activation bias slows initial learning but ultimately yields a lower final loss. Experiments on larger vision and language models exhibit a similar pattern: smaller activation bias is associated with better late-stage performance. Thus, this seemingly minor property can substantially redirect learning and warrants explicit consideration in future activation function design.

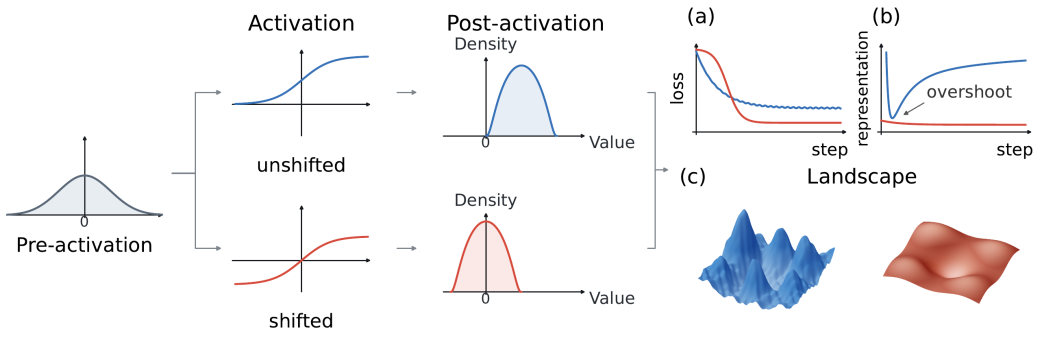


Figure 1: Overview of activation bias and its effects on learning. Blue denotes the unshifted activation, and red denotes the shifted activation. Reducing activation bias changes the post-activation distribution, model performance, hidden-state trajectory, and loss landscape.

1 INTRODUCTION

Many common activation functions, including ReLU, GELU, Swish, and Sigmoid, favor positive outputs while suppressing or excluding negative ones. These asymmetries typically yield a nonzero post-activation mean. For a pre-activation z, we define the *activation bias* of an activation function ϕ as the mean of its post-activation:

$$b_\phi \equiv \mathbb{E}[\phi(z)]. \tag{1}$$

These functions are widely used and perform well across diverse tasks. Their success, however, does not establish how the post-activation bias they induce affects learning dynamics.

Prior work connects activation design to trainability through signal propagation, dynamical isometry, critical initialization, and normalization (Clevert et al., 2016; Brock et al., 2021; Qi et al., 2020; Lesser & Chowdhury, 2026; Björck et al., 2018; Santurkar et al., 2018). These studies primarily address propagation stability or optimization conditioning, leaving open how activation bias affects

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.

learning over time and whether the bias built into common activations is optimal for learning. We therefore ask:

Q: How does activation bias shape learning dynamics?

(i) How does it alter training dynamics? (ii) Does the activation bias component carry useful information, and how does the model use that information during learning?

We first investigate the effect of activation bias on training dynamics in a controlled deep residual teacher–student model. For each activation function, we use a fixed post-activation shift to control the degree of activation bias. Both students learn from the same fixed teacher, with identical initial parameters, training data, and optimization settings; they differ only in whether the shift is applied.

Reducing activation bias slows initial loss reduction, but the shifted student eventually overtakes its unshifted counterpart and attains a lower final loss. Theory and experiments associate this reversal with smoother optimization dynamics, distinct parameter-update directions, and a flatter loss landscape, reflected in substantially smaller Hessian eigenvalues.

We then use a controlled image-reconstruction task to test whether the activation bias component carries information the model can exploit. Early in training, decoding this component alone recovers recognizable image content. Its output later departs from the full-representation reconstruction, suggesting that the model relies less on this information as training proceeds. This transient use reveals a non-smooth feature-learning trajectory and is largely suppressed when activation bias is reduced.

Finally, we test whether these findings persist in larger models and across a wider range of tasks. These models exhibit analogous non-smooth dynamics under the original activations, while reducing activation bias yields comparable or better late-stage performance. Thus, this seemingly small activation property can produce pronounced changes in learning across architectures and tasks.

Our contributions are the following:

- We use a controlled model to investigate activation bias and reveal its effect: it accelerates early learning but impedes later progress.
- We characterize how activation bias shapes optimization dynamics and how the model uses information carried by the activation bias component during learning.
- We observe similar learning dynamics and improved late-stage performance with reduced activation bias across larger models and a wider range of tasks.

To our knowledge, this is the first study to examine how activation bias shapes learning dynamics throughout training. Our results show that a small change in activation bias can redirect optimization, yield smoother learning trajectories, and improve late-stage performance. This sensitivity calls for treating activation bias as an explicit property in activation-function design rather than an incidental by-product.

2 Analysis of Training Dynamics

2.1 Controlled Teacher–Student Setup

To isolate the effect of activation bias from other factors, we adopt a simple controlled model. We omit token embeddings, attention, and the output head to isolate the activation function's role while retaining the pre-normalized MLP blocks and residual-stream architecture of modern large language models. For layers $\ell = 0, \ldots, L-1$, the model evolves as

$$\mathbf{x}^{(\ell+1)} = \mathbf{x}^{(\ell)} + \mathbf{W}_2^{(\ell)} \phi\Big(\mathbf{W}_1^{(\ell)} \,\mathrm{LN}\Big(\mathbf{x}^{(\ell)}\Big)\Big), \tag{2}$$

where $\mathbf{x}^{(\ell)}$ denotes the residual-stream representation, $\mathbf{W}_1^{(\ell)}$ and $\mathbf{W}_2^{(\ell)}$ are the up- and down-projection matrices, respectively, LN denotes pre-normalization, and ϕ is the nonlinear activation function.

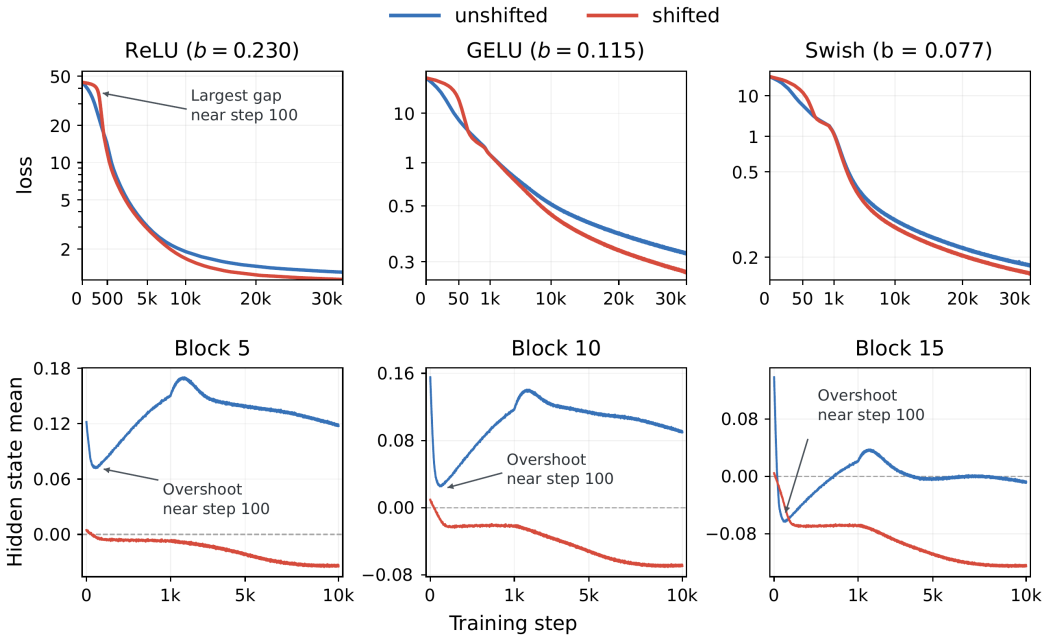


Figure 2: Activation bias accelerates early learning, but this advantage coincides with a pronounced hidden-state overshoot and reverses later in training. Top: The shifted model learns more slowly initially but ultimately reaches a lower loss across activation functions. Bottom: The unshifted model's hidden state overshoots and follows a non-smooth trajectory.

We control activation bias by subtracting the same fixed scalar b from the activation output in every layer:

$$\phi_{\text{shifted}}(z; b) = \phi(z) - b. \tag{3}$$

To avoid confounding activation bias with complex data structure that is difficult to isolate and characterize, we use a teacher–student framework. The teacher and both students share the architecture in Equation 2. A fixed teacher using the unshifted activation ϕ generates targets $y = f_{\text{T}}(\mathbf{x})$ for sampled inputs $\mathbf{x}$. We refer to the student using ϕ as the *unshifted student* and the student using ϕ_{shifted} as the *shifted student*. For each activation, we estimate b as its expected post-activation output at initialization. This shift keeps the post-activation mean near zero over an extended period of training, as detailed in Appendix A.

Table 1: Activation shifts b used for each activation function.

Activation functions	ReLU	GELU	Swish	Sigmoid
b	0.230	0.115	0.077	0.500

2.2 Early Learning Advantage Coincides with Hidden-State Overshoot

We first compare the two students through their loss curves. As shown in the top panel of Figure 2, the unshifted student learns faster initially, but the shifted student eventually overtakes it and converges to a lower loss. This reversal is notable because the unshifted student uses exactly the same activation as the teacher yet is ultimately outperformed by the shifted student.

Motivated by this loss reversal, we first track the residual-stream hidden state over training. We average its elements for each input and then average across inputs, yielding a scalar trajectory that tracks the hidden state's evolution during training. In the bottom panel of Figure 2, the unshifted student's hidden state overshoots around step 100, coinciding with its period of faster loss reduction. The shifted student's hidden state instead evolves smoothly. These distinct hidden-state trajectories motivate examining whether the two students also follow different paths in parameter space.

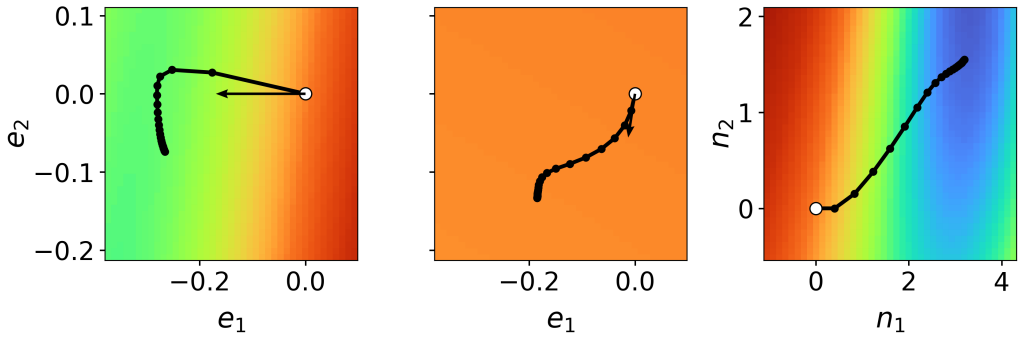


Figure 3: Activation bias redirects parameter updates, producing a distinct trajectory with a sharp, non-smooth turn. Left: the unshifted student on the initial-gradient plane. Center: the shifted student on the same plane. Right: the shifted student on the complementary plane. Colors closer to red indicate higher loss, while colors closer to blue indicate lower loss.

2.3 Activation Bias Separates Parameter-Update Directions

To compare parameter trajectories, let $\boldsymbol{\theta}_0$ be the common initialization and let $\mathbf{g}_\mathrm{u}$ and $\mathbf{g}_\mathrm{s}$ be the students' initial gradients, where the subscripts u and s denote the unshifted and shifted students, respectively. We construct the *initial-gradient plane* with orthonormal basis

$$\mathbf{e}_1 = \frac{\mathbf{g}_\mathrm{u}}{\|\mathbf{g}_\mathrm{u}\|}, \qquad \mathbf{e}_2 = \frac{(\mathbf{I} - \mathbf{e}_1\mathbf{e}_1^\top)\mathbf{g}_\mathrm{s}}{\|(\mathbf{I} - \mathbf{e}_1\mathbf{e}_1^\top)\mathbf{g}_\mathrm{s}\|}, \tag{4}$$

where $\mathbf{e}_1$ follows the unshifted gradient and $\mathbf{e}_2$ is the orthogonal component of the shifted gradient.

To better understand the early optimization geometry, we visualize each student's loss landscape on the initial-gradient plane. At each coordinate (α, β), we reconstruct the parameters as $\boldsymbol{\theta}_0 + \alpha\mathbf{e}_1 + \beta\mathbf{e}_2$ and evaluate the corresponding student against the fixed teacher on the evaluation set. The resulting losses form a heat map of the projected landscape. Projecting training checkpoints onto the same plane overlays each student's trajectory. As shown in the left and center panels of Figure 3, the two trajectories initially point in nearly orthogonal directions. The unshifted student initially follows the direction of strongest descent in its projected landscape and then turns sharply, revealing a distinctly non-smooth trajectory. The shifted student's loss varies little on this plane, indicating that its early update develops largely outside these initial-gradient directions.

To visualize the shifted student's parameter movement outside this plane, we remove the parts of its parameter displacement along $\mathbf{e}_1$ and $\mathbf{e}_2$:

$$\mathbf{r}_t = (\mathbf{I} - \mathbf{e}_1\mathbf{e}_1^\top - \mathbf{e}_2\mathbf{e}_2^\top)(\boldsymbol{\theta}(t) - \boldsymbol{\theta}_0). \tag{5}$$

Here, $\mathbf{r}_t$ is the component of the shifted student's parameter displacement orthogonal to the initial-gradient plane. To obtain two orthogonal directions from the shifted student's parameter displacements outside the initial-gradient plane, we apply Gram–Schmidt orthogonalization to these displacements and use the first two resulting basis vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ as its axes. We then visualize the loss and trajectories on this plane as above. The right panel of Figure 3 reveals clear loss variation along the shifted student's smooth trajectory without a comparable turn.

Together, these views show that activation bias induces qualitatively distinct parameter-update directions. The two students initially move in nearly orthogonal directions, and their subsequent updates remain largely separated: the unshifted trajectory turns within the initial-gradient plane, whereas the shifted trajectory develops primarily in complementary directions and evolves smoothly. Further theoretical analysis and experiments on update directions are provided in Appendix B.

2.4 Activation Bias Steepens the Loss Landscape

Although the projected landscapes suggest differences in curvature, each shows only a two-dimensional slice. We therefore quantify local curvature in the full parameter space using the

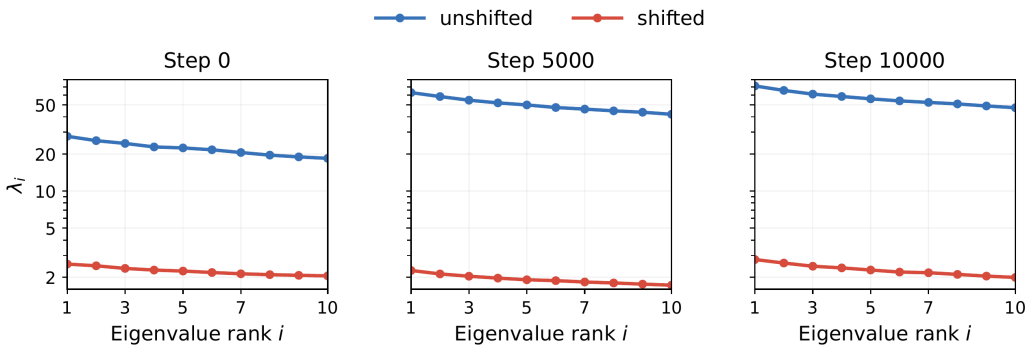


Figure 4: Reducing activation bias substantially lowers Hessian eigenvalues throughout training, yielding a flatter local loss landscape.

leading eigenvalues of the Hessian $\mathbf{H} = \nabla^2 \mathcal{L}$, whose largest eigenvalue is commonly termed sharpness. Figure 4 shows that the shifted student has substantially smaller eigenvalues throughout training. Thus, in this controlled model, activation bias is associated with a markedly steeper local loss landscape, whereas reducing the bias yields a flatter one.

To examine how activation bias affects loss landscape curvature, we consider a simplified case with a post-activation vector $\mathbf{a}(x)$ and a trainable down-projection $\mathbf{W}_2$ with squared loss

$$\mathcal{L}_b(\mathbf{W}_2) = \frac{1}{2}\mathbb{E}_x \left\| \mathbf{W}_2\big(\mathbf{a}(x) - b\mathbf{1}\big) - \mathbf{y}(x) \right\|^2. \tag{6}$$

Suppose $\mathbb{E}[\mathbf{a}] = \mu\mathbf{1}$, and let $\Sigma = \mathrm{Cov}(\mathbf{a})$. The Hessian $H(b)$ with respect to $\mathbf{W}_2$ then satisfies

$$H(b) = \left[\Sigma + (\mu - b)^2 \mathbf{1}\mathbf{1}^\top\right] \otimes I \succeq \Sigma \otimes I, \tag{7}$$

where I is the identity matrix and $\otimes$ denotes the Kronecker product. Since $(\mu - b)^2 \mathbf{1}\mathbf{1}^\top \otimes I$ is positive semidefinite, the largest eigenvalue satisfies $\lambda_{\max}(H(b)) \geq \lambda_{\max}(H(\mu))$. Thus, $b = \mu$ minimizes Hessian sharpness in this setting. A full proof and an extension to a more general multi-layer model are provided in Appendix B.

Taken together, these measurements show that activation bias gives the unshifted student a clear early learning advantage, but this advantage coincides with an overshooting hidden-state trajectory and a markedly steeper loss landscape. Reducing the bias slows initial learning, changes the parameter-update directions, and produces smoother dynamics with stronger long-term performance. Additional ablations are provided in Appendix C.

3 Analysis of Activation Bias Component Information

Section 2 shows that activation bias yields a transient early advantage yet worse long-run optimization. We next ask whether the associated activation bias component carries task-relevant information that the model uses.

3.1 Image Reconstruction Setup

Image reconstruction makes information carried by the activation bias component directly observable in decoded images. We therefore choose this task and use a simplified VGG-style autoencoder,

$$\mathbf{x} \xrightarrow{\text{encoder}} \mathbf{h} \xrightarrow{\text{decoder}} \widehat{\mathbf{x}},$$

where $\mathbf{x}$ is an input image, $\mathbf{h}$ is its latent representation, and $\widehat{\mathbf{x}}$ is the reconstructed image. We use part of the VGG-19 architecture (Simonyan & Zisserman, 2015) as the encoder, followed by a convolutional decoder. The latent representation is the encoder's final post-activation output, with 512 channels at each spatial location. The detailed architecture is provided in Appendix D. We compare

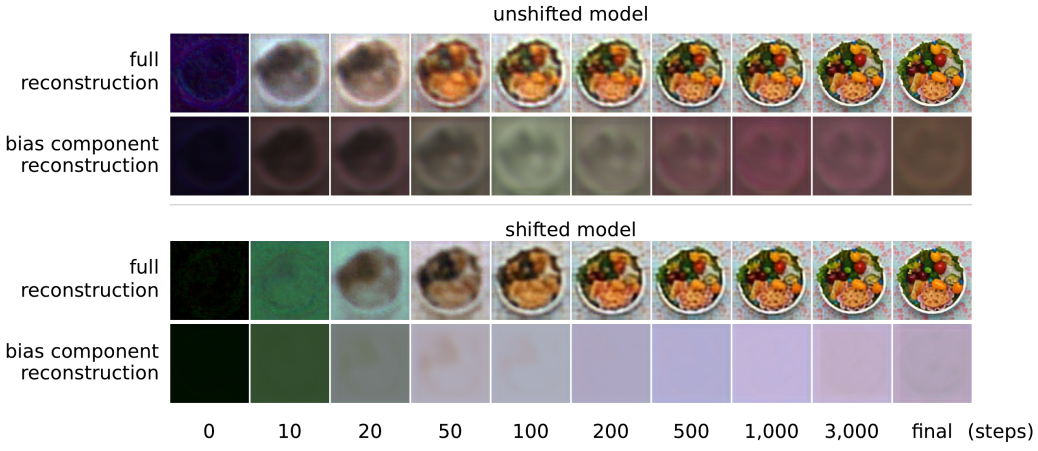


Figure 5: The unshifted model recovers recognizable image content from the activation bias component early in training but relies less on it later; the shifted model recovers little content from this component.

otherwise identical models using the unshifted and shifted activations. Consistent with the controlled teacher–student model in Section 2, the unshifted model learns faster initially, but the shifted model slightly surpasses it later in training, as shown in Appendix D.

Throughout training, both decoders receive the full representation $\mathbf{h}$. For post-hoc analysis at selected checkpoints, let $\mathbf{h}_p \in \mathbb{R}^m$ denote the hidden state at spatial location p, where $m = 512$ is the number of channels. For every location p, we decompose it as

$$\mathbf{h}_p = \underbrace{\left(\frac{1}{m}\mathbf{1}^\top \mathbf{h}_p\right)\mathbf{1}}_{\mathbf{h}_{p,\text{bias}}} + \underbrace{(\mathbf{h}_p - \mathbf{h}_{p,\text{bias}})}_{\mathbf{h}_{p,\text{dev}}}. \tag{8}$$

3.2 Visualization and Analysis of Activation-Bias-Component Information

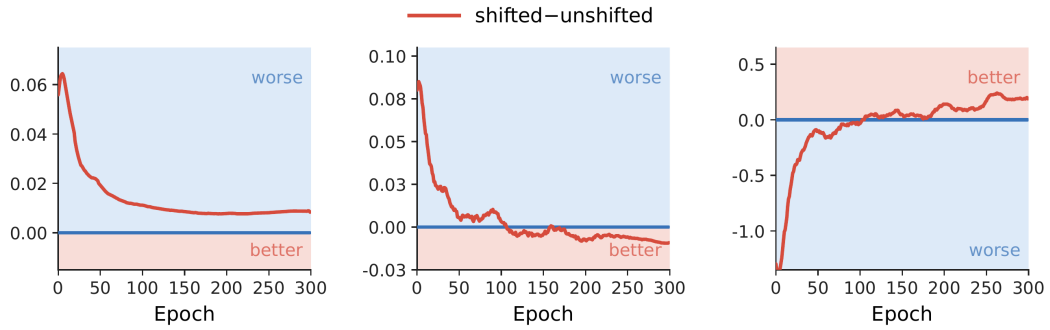
For each selected checkpoint, we pass the two components separately through the decoder. Figure 5 shows the resulting component-wise reconstructions. For the unshifted model, reconstruction from $\mathbf{h}_{\text{bias}}$ alone yields a recognizable, though incomplete, image that retains coarse structure, color, and texture, implying that the model uses some information encoded in the activation bias component. By contrast, the corresponding reconstruction for the shifted model contains little recoverable structure, suggesting that this model makes less use of the component.

The role of the unshifted model's activation bias component evolves non-monotonically. Its isolated reconstruction becomes increasingly similar to the full reconstruction, reaching its closest agreement early in training. It subsequently diverges from the full reconstruction. This rise-and-fall pattern mirrors the early learning advantage and hidden-state overshoot in Section 2.2.

Together, these visualizations show that activation bias can change how the model uses image information during training. The unshifted model's transient use of the activation bias component provides a feature-learning counterpart to its early learning advantage, whereas the shifted model makes little use of this component.

4 Scale-Up Experiments

We next examine whether our findings extend to larger, more complex architectures and diverse tasks. We consider supervised image classification, diffusion-based image generation, and language modeling. Unlike in the controlled model, subtracting the same fixed scalar in every layer is insufficient because activation bias differs across layers and evolves rapidly during training. We



(a) Training-loss residual (b) Test-loss residual (c) Top-1 accuracy residual

Figure 6: For the ViT image-classification task, the shifted model retains a higher training loss but ultimately achieves lower test loss and higher test accuracy. The loss residuals are $L_{\text{shifted}} - L_{\text{unshifted}}$, and the accuracy residual is $\text{Acc}_{\text{shifted}} - \text{Acc}_{\text{unshifted}}$.

therefore use *adaptive shifting* as the bias-reduced condition. This layerwise intervention subtracts the current post-activation mean from every neuron in each layer:

$$\widetilde{\phi}^{(\ell)}\left(\mathbf{z}^{(\ell)}\right) = \phi\left(\mathbf{z}^{(\ell)}\right) - \frac{1}{N_\ell}\sum_{i=1}^{N_\ell}\phi\left(z_i^{(\ell)}\right)\mathbf{1}, \tag{9}$$

where $\mathbf{z}^{(\ell)} \in \mathbb{R}^{N_\ell}$ is the pre-activation vector in layer ℓ, N_ℓ is the number of neurons, and i indexes neurons. The shifted output has zero mean across neurons by construction.

4.1 Vision Transformer Image Classification

We first consider Vision Transformers for supervised image classification. Following the ImageNet-1K training protocol of Zhu et al. (2025), we train ViT-Base (Dosovitskiy et al., 2021) for 300 epochs. We hold all other training choices fixed across conditions, varying only whether adaptive shifting is applied. Representative ViT-Base results are shown in Figure 6. The model with adaptive shifting initially trails the unshifted baseline by a substantial margin, but the ordering reverses during training: by the end, it attains both lower test loss and higher top-1 classification accuracy, despite retaining a higher training loss. This result is consistent with our controlled-model observation that reduced activation bias is associated with lower Hessian sharpness and a flatter local landscape, which is often linked to better generalization and less overfitting.

Figure 7(a) tracks the residual-stream hidden-state trajectory using the statistic defined in Section 2.2. The unshifted model exhibits a pronounced early overshoot during the period in which it holds a substantial performance advantage. Under this larger architecture and more complex task, both trajectories are noisy, but the adaptively shifted model exhibits a substantially smaller overshoot. This contrast echoes the pattern observed in the controlled model. Further examples of overshoot trajectories in scale-up language models are provided in Appendix E.2. Together, these results show that reducing activation bias remains beneficial for image classification at larger scale.

4.2 Diffusion Transformer Image Generation

We next consider image generation with Diffusion Transformers (DiTs) (Peebles & Xie, 2023). Following the official `facebookresearch/DiT` implementation, we train DiT-B/2 on ImageNet-256 for 400,000 optimization steps. We hold all other training choices fixed and vary only whether adaptive shifting is applied. As shown in Figure 7(b), adaptive shifting achieves comparable or lower training loss throughout training. Additional results in Appendix E show that the shifted model achieves lower FID at every evaluated checkpoint. These results demonstrate that reducing activation bias can also benefit larger-scale image generation.

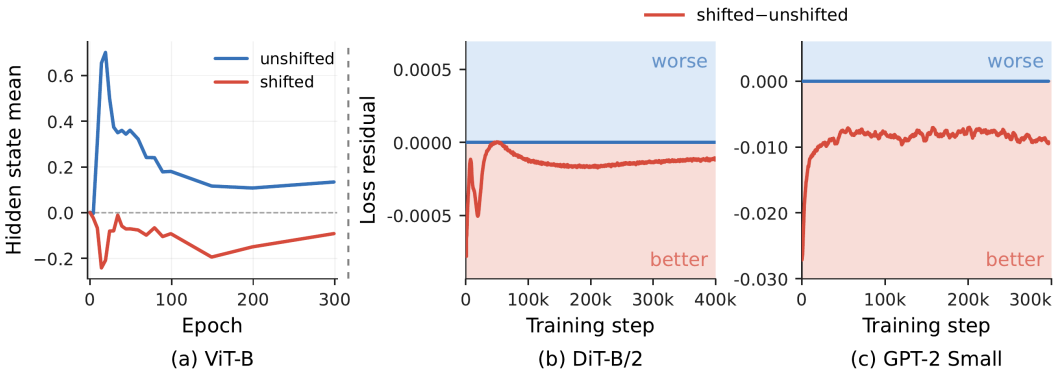


Figure 7: (a) Reducing activation bias suppresses hidden-state overshoot in ViT. (b) and (c): The shifted model performs better in both DiT image generation and GPT-2 language pretraining. The loss residual is $L_{\text{shifted}} - L_{\text{unshifted}}$.

4.3 GPT-2 Pretraining

We then move from vision to language modeling. Following Karpathy's training setup (Karpathy, 2022), we train GPT-2 Small (Radford et al., 2019) from scratch on the full OpenWebText corpus (Gokaslan et al., 2020) for 300,000 optimization steps. As above, we hold all other training choices fixed and vary only whether adaptive shifting is applied. Figure 7(c) reports the results. Adaptive shifting consistently improves the performance, stably lowering loss by about 0.01 in absolute terms and perplexity by 1% relative to the unshifted baseline. These gains show that the effect of reducing activation bias extends to language modeling.

The larger-scale models and tasks considered here are widely adopted and have been extensively optimized, leaving relatively little headroom for improvement from a single change to the activation. Even so, adaptive shifting yields small but consistent gains. Together with the observed changes in learning dynamics, these gains indicate that activation bias has a non-negligible effect in practical models. Additional scale-up results are provided in Appendix E.

5 Discussions

The actual effect of activation bias may vary in different settings. The extent of early acceleration, hidden-state overshoot, and final performance differences may depend on data and task structure, model scale, architectural components such as attention, and weight initialization. For example, during GPT-2 Small pretraining, the unshifted model does not exhibit the initial performance advantage seen in the controlled model, potentially due to the small initialization scale used by this architecture. Nevertheless, across the settings studied here, smaller activation bias consistently yields better late-stage performance. How these factors interact with activation bias merits further investigation.

We use fixed or adaptive shifts to isolate a learning-dynamical consequence of activation bias. Although reducing the bias improves performance in our experiments, these results neither establish that zero post-activation bias is universally optimal nor guarantee that either intervention will benefit every architecture, task, or training regime. Performance can also depend on other desirable activation properties, including selectivity, signal scale, and smoothness. The methods used here to control activation bias therefore serve as diagnostic interventions instead of general prescriptions for activation design.

Our results identify a set of connected signatures of activation bias: an early–late learning pattern, hidden-state overshoot and non-smooth dynamics, distinct parameter-update directions, a steeper loss landscape, and transient use of information in the activation bias component. We do not yet have a unified mathematical theory or a complete causal account of how these effects arise. Moreover, our larger-model experiments do not systematically test whether all the phenomena and theoretical

results from the controlled setting extend to more complex architectures. Addressing these gaps could provide more principled guidance for the design of new activation functions.

Activation functions have many properties beyond bias that we do not study here. Post-activation variance is one example: the variance most conducive to learning may depend on architecture, task, scale, and training regime, yet its effects on learning dynamics and final performance remain unclear. Future work could isolate variance and other properties of activation functions under controlled conditions, track their effects throughout training, and test whether the resulting patterns transfer across models and tasks. Such analyses would provide a broader foundation for activation-function design.

6 Related Work

Activation-function design spans nonlinearities with markedly different input–output profiles. Early work used sigmoid and tanh (LeCun et al., 1998); later proposals introduced Softplus (Dugas et al., 2000), ReLU (Nair & Hinton, 2010), GELU (Hendrycks & Gimpel, 2016), and Swish (Ramachandran et al., 2017). Leaky ReLU (Maas et al., 2013), PReLU (He et al., 2015), and Mish (Misra, 2020) further broadened this design space. More recent work has explored gated activations such as SwiGLU (Shazeer, 2020), learnable activations such as ACON (Ma et al., 2021), and alternatives such as SMU (Biswas et al., 2022) and StarReLU (Yu et al., 2024). These activation functions have proved effective across tasks, yet many of them produce predominantly positive outputs and thus introduce activation bias. While prior work has focused on designing and evaluating the functions themselves, the effect of this bias on learning remains less understood. Our work complements these studies by isolating activation bias and examining how it shapes learning dynamics.

Previous work on activation functions and their statistics has primarily sought to improve trainability. To stabilize intermediate signals, normalization methods center or rescale them (Ioffe & Szegedy, 2015; Ba et al., 2016; Arpit et al., 2016; Björck et al., 2018); mean-only batch normalization retains centering alone, whereas RMSNorm retains rescaling alone (Salimans & Kingma, 2016; Zhang & Sennrich, 2019). Santurkar et al. (2018) showed that batch normalization also smooths the loss landscape. To sustain signal and gradient flow through depth, mean-field and dynamical-isometry studies analyze activation choices (Schoenholz et al., 2017; Pennington et al., 2017; Hayou et al., 2019; Lesser & Chowdhury, 2026), and Qi et al. (2020) designed a shifted ReLU to improve isometry. To counter shifts associated with nonzero activation outputs, Raiko et al. (2012) centered neuron outputs and slopes to improve gradient conditioning, Nøkland & Eidnes (2018) introduced bipolar activations to facilitate deep training, and Kutsuna (2019) constrained weights to reduce shifts in downstream pre-activations. These studies alter normalization, activation shape, or weights, whereas we isolate activation bias as a single factor and investigate its effects on learning dynamics.

Some studies have examined nonzero activation bias. Glorot & Bengio (2010) linked the positive mean of sigmoid activations to upper-layer saturation at initialization. Clevert et al. (2016) showed that ELUs can reduce bias shift, accelerate learning, and improve classification accuracy; Klambauer et al. (2017) designed SELUs to stabilize activation means and variances across depth; and Brock et al. (2021) controlled the growth of positive channel means in unnormalized ResNets. These studies establish consequences of activation bias in restricted settings. We examine its effects throughout training and show that reducing it can improve model performance across different architectures and tasks.

7 Conclusion

We study how activation bias shapes training dynamics and the use of task-relevant information. In the controlled setting, activation bias confers an early learning advantage but is associated with hidden-state overshoot, non-smooth trajectories, a steeper loss landscape and worse late-stage performance. Our reconstruction analysis further shows that the learned model transiently uses information in the bias component. Across larger and more diverse settings, adaptive shifting retains a late-stage performance advantage. These results motivate a more complete causal account of activation bias and its explicit consideration in future activation-function design.

AI USE STATEMENT

We used generative AI tools to brainstorm ideas, formulate candidate mathematical claims, draft code following explicit author instructions, and polish the writing. These tools were not used to create synthetic datasets, perform data analysis, or write mathematical proofs. The authors critically reviewed all AI-assisted ideas and text, independently inspected and verified generated code, and conducted the data analysis and proofs themselves. The authors take full responsibility for the final content of this paper.

REFERENCES

Devansh Arpit, Yingbo Zhou, Bhargava U. Kota, and Venu Govindaraju. Normalization propagation: A parametric technique for removing internal covariate shift in deep networks. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1168–1176, 2016.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430, 2023. URL https://proceedings.mlr.press/v202/biderman23a.html.

Koushik Biswas, Sandeep Kumar, Shilpak Banerjee, and Ashish Kumar Pandey. Smooth maximum unit: Smooth activation function for deep networks using smoothing maximum technique. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/papers/Biswas_Smooth_Maximum_Unit_Smooth_Activation_Function_for_Deep_Networks_Using_CVPR_2022_paper.pdf.

Nils Björck, Carla P. Gomes, Bart Selman, and Kilian Q. Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Andrew Brock, Soham De, and Samuel L. Smith. Characterizing signal propagation to close the performance gap in unnormalized ResNets. In *International Conference on Learning Representations*, 2021.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). In *International Conference on Learning Representations*, 2016.

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 *International Conference on Learning Representations*, 2021.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. In *Advances in Neural Information Processing Systems*, volume 13, 2000.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, 2010.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. OpenWebText. Zenodo dataset, 2020. URL https://zenodo.org/records/3834942.

Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. On the impact of the activation function on deep neural networks training. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2672–2680, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, 2015.

Andrej Karpathy. nanoGPT: The simplest, fastest repository for training and fine-tuning medium-sized GPTs. GitHub repository, 2022. URL https://github.com/karpathy/nanoGPT.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Takuro Kutsuna. Linearly constrained weights: Reducing activation shift for faster training of neural networks. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2019)*, pp. 266–282, 2019. doi: 10.1007/978-3-030-46147-8_16.

Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*. Springer, 1998.

Omri Lesser and Debanjan Chowdhury. Competing nonlinearities, criticality, and order-to-chaos transition in deep networks. *arXiv preprint arXiv:2605.05294*, 2026.

Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, et al. LLM360: Towards fully transparent open-source LLMs, 2023. URL https://arxiv.org/abs/2312.06550.

Ningning Ma, Xiangyu Zhang, Ming Liu, and Jian Sun. Activate or not: Learning customized activation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8032–8042, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Ma_Activate_or_Not_Learning_Customized_Activation_CVPR_2021_paper.html.

Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

Diganta Misra. Mish: A self regularized non-monotonic activation function. In *Proceedings of the British Machine Vision Conference*, 2020.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 807–814, 2010.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, Stefano Ermon, Christopher Ré, and Stephen Baccus. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1872. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/86ab6927ee4ae9bde4247793c46797c7-Abstract.html.

Arild Nøkland and Lars Hiller Eidnes. Shifting mean activation towards zero with bipolar activation functions. In *International Conference on Learning Representations Workshop*, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Jeffrey Pennington, Samuel S. Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: Theory and practice. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Haozhi Qi, Chong You, Xiaolong Wang, Yi Ma, and Jitendra Malik. Deep isometric learning for visual recognition. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Tapani Raiko, Harri Valpola, and Yann LeCun. Deep learning made easier by linear transformations in perceptrons. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pp. 924–932, 2012.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Samuel S. Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. In *International Conference on Learning Representations*, 2017.

Noam Shazeer. GLU variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Team OLMo, Pete Walsh, Luca Soldaini, et al. 2 OLMo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. MetaFormer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):896–912, 2024. doi: 10.1109/TPAMI.2023.3329173.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Jiachen Zhu, Xinlei Chen, Kaiming He, Yann LeCun, and Zhuang Liu. Transformers without normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14901–14911, 2025.

A ESTIMATING b FOR DIFFERENT ACTIVATION FUNCTIONS

This appendix examines fixed, layer-shared activation shifts in the controlled model. We first derive the shift b_{theory} that brings the post-activation mean near zero at initialization and show that the empirical means remain close to zero during training. We then sweep b and measure the final loss. Across ReLU, GELU, Swish, Tanh, and sigmoid, the theory-predicted shifts achieve near-optimal final loss.

A.1 THEORY-PREDICTED ACTIVATION MEANS

For a pre-norm MLP block, let $\mathbf{x} \in \mathbb{R}^d$ be its input and $\mathbf{u} = \mathrm{LN}(\mathbf{x})$. Let $\mathbf{w}_i$ be the ith row of the up-projection linear layer. The pre-activation of neuron i is

$$Z_i = \mathbf{w}_i^\top \mathbf{u}. \tag{10}$$

The weights are initialized independently of $\mathbf{u}$ as

$$w_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{U}\left[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right]. \tag{11}$$

Using $\|\mathbf{u}\|_2^2 \approx d$ from LayerNorm gives

$$\mathbb{E}_{\mathbf{w}_i,\mathbf{x}}[Z_i] = 0, \qquad \mathbb{E}_{\mathbf{w}_i,\mathbf{x}}[Z_i^2] = \frac{1}{3d}\mathbb{E}_{\mathbf{x}}\|\mathbf{u}\|_2^2 \approx \frac{1}{3}. \tag{12}$$

We approximate the pre-activations by

$$Z \sim \mathcal{N}\left(0, \frac{1}{3}\right). \tag{13}$$

The shifted activation is

$$\phi_{\text{shifted}}(Z; b) = \phi(Z) - b. \tag{14}$$

Thus,

$$b_{\text{theory}}(\phi) := \mathbb{E}[\phi(Z)]. \tag{15}$$

ReLU. For $\phi(z) = \max\{0, z\}$,

$$\begin{aligned} b_{\text{theory}}(\text{ReLU}) &= \int_{-\infty}^{\infty} \max\{0, z\} \frac{1}{\sqrt{2\pi/3}} \exp\left(-\frac{z^2}{2/3}\right) \mathrm{d}z \\ &= \sqrt{\frac{3}{2\pi}} \int_0^{\infty} z \exp\left(-\frac{3z^2}{2}\right) \mathrm{d}z \\ &= \frac{1}{\sqrt{6\pi}} \\ &\approx 0.230. \end{aligned} \tag{16}$$

GELU. We use the exact GELU, $\phi(z) = z\Phi(z)$, where Φ and φ are the standard Gaussian cumulative distribution and density functions.

$$\begin{aligned} b_{\text{theory}}(\text{GELU}) &= \int_{-\infty}^{\infty} z\Phi(z) \frac{1}{\sqrt{2\pi/3}} \exp\left(-\frac{z^2}{2/3}\right) \mathrm{d}z \\ &= \frac{1}{\sqrt{6\pi}} \int_{-\infty}^{\infty} \varphi(z) \exp\left(-\frac{3z^2}{2}\right) \mathrm{d}z \\ &= \frac{1}{\sqrt{24\pi}} \approx 0.115. \end{aligned} \tag{17}$$

Swish. For $\phi(z) = z/(1 + e^{-z})$,

$$\begin{aligned} b_{\text{theory}}(\text{Swish}) &= \int_{-\infty}^{\infty} \frac{z}{1 + e^{-z}} \frac{1}{\sqrt{2\pi/3}} \exp\left(-\frac{z^2}{2/3}\right) \mathrm{d}z \\ &\approx 0.077. \end{aligned} \tag{18}$$

Tanh. Since tanh is odd,

$$b_{\text{theory}}(\tanh) = \mathbb{E}[\tanh(Z)] = 0. \tag{19}$$

Sigmoid. For $\phi(z) = 1/(1+e^{-z})$, we have $\phi(z) + \phi(-z) = 1$. Since Z and $-Z$ have the same distribution,

$$\begin{aligned} b_{\text{theory}}(\text{sigmoid}) &= \mathbb{E}[\phi(Z)] \\ &= \frac{1}{2}\mathbb{E}[\phi(Z) + \phi(-Z)] \\ &= \frac{1}{2} = 0.500. \end{aligned} \tag{20}$$

Figure 8 shows that the predicted shifts keep the post-activation means near zero during training.

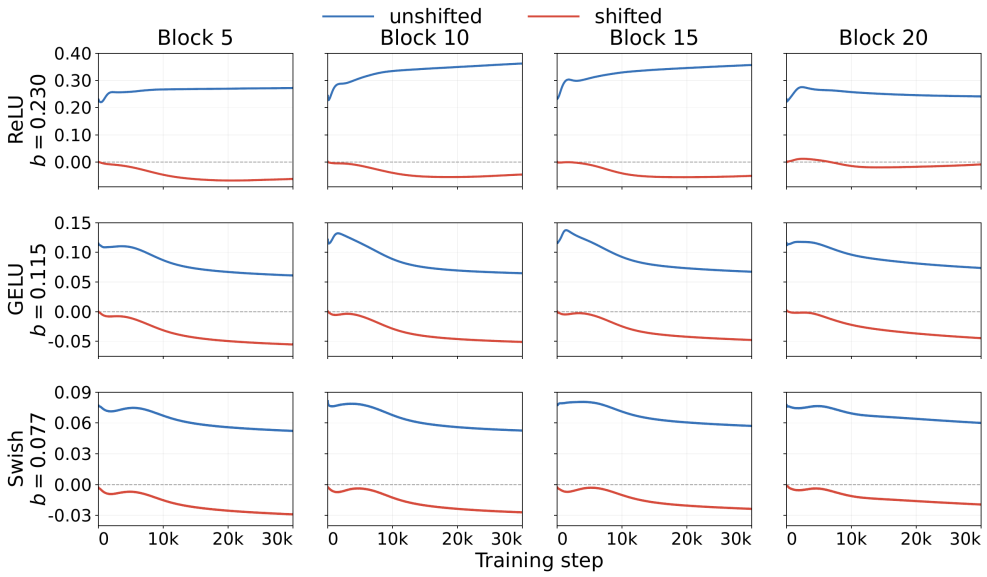


Figure 8: The post-activation means of the shifted models remain near zero throughout training.

A.2 Final loss as a function of the activation shift

Using the same teacher–student training procedure, we sweep the shift b for each activation over multiple random seeds. Within each seed, all compared shifts use the same teacher, student initialization, and sequence of training batches. Figure 9 plots final loss against b. Across the tested activations, the theory-predicted shifts achieve near-optimal final loss.

B Theoretical analysis of optimization landscape

This appendix analyzes how activation bias affects loss landscape curvature and changes the direction of parameter updates early in training.

B.1 Activation mean and landscape sharpness

B.1.1 A single trainable down-projection

We start with the simplest setting, where $a(x) \in \mathbb{R}^m$ is fixed and only a down-projection layer $B \in \mathbb{R}^{d_{\text{out}} \times m}$ is trained. Define

$$z_b(x) := a(x) - b\mathbf{1}, \qquad f_B(x) = Bz_b(x). \tag{21}$$

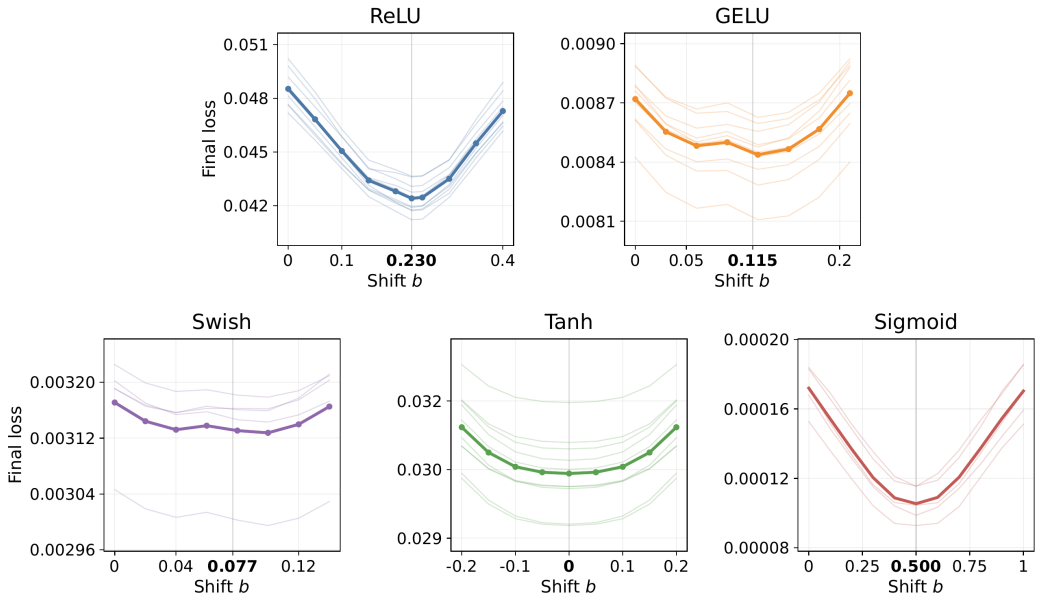


Figure 9: Across all five activations, our theory's predicted shifts achieve near-optimal final loss. Thin curves show individual seeds, and thick curves show their mean.

For target $y(x)$, define

$$\mathcal{L}_b(B) = \frac{1}{2}\mathbb{E}_x \left\| Bz_b(x) - y(x) \right\|^2, \tag{22}$$

$$M_b := \mathbb{E}_x \left[z_b(x) z_b(x)^\top \right]. \tag{23}$$

Proposition B.1 (Exact down-projection sharpness)**.** *The Hessian with respect to the vectorized matrix B is*

$$H_B(b) = M_b \otimes I_{d_{\text{out}}}, \qquad \lambda_B(b) := \lambda_{\max}(H_B(b)) = \lambda_{\max}(M_b). \tag{24}$$

Here $\otimes$ denotes the Kronecker product. Suppose

$$\mathbb{E}[a] = \mu \mathbf{1}, \tag{25}$$

so all neurons have the same mean activation μ. Then $b = \mu$ globally minimizes $\lambda_B(b)$ over all scalar shifts.

Proof. For any perturbation ΔB,

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2} \mathcal{L}_b(B + t\Delta B)\Big|_{t=0} = \mathbb{E}\|\Delta B z_b\|^2 = \mathrm{tr}(\Delta B M_b \Delta B^\top), \tag{26}$$

which gives equation 24. Let $\Sigma = \mathrm{Cov}(a)$. Then

$$M_b = \Sigma + (\mu - b)^2 \mathbf{1}\mathbf{1}^\top \succeq \Sigma = M_\mu. \tag{27}$$

$$\lambda_B(b) = \lambda_{\max}(M_b) \geq \lambda_{\max}(M_\mu) = \lambda_B(\mu). \qquad \square$$

When neuron-wise activation means differ, write

$$\mathbb{E}[a] = \bar{\mu}\mathbf{1} + \varepsilon, \qquad \bar{\mu} := \frac{1}{m}\mathbf{1}^\top \mathbb{E}[a], \qquad \varepsilon \perp \mathbf{1}.$$

Since $M_{\bar{\mu}} = \Sigma + \varepsilon\varepsilon^\top$ and $M_b \succeq \Sigma$ for every b,

$$0 \leq \lambda_{\max}(M_{\bar{\mu}}) - \min_b \lambda_{\max}(M_b) \leq \|\varepsilon\|^2. \tag{28}$$

Thus shifting by the average activation mean is within $\|\varepsilon\|^2$ of the optimal scalar shift in largest-eigenvalue curvature. The gap is small whenever $\|\varepsilon\|$ is small.

B.1.2 JOINTLY TRAINING THE UP- AND DOWN-PROJECTIONS

We then allow both the up-projection A and the down-projection B to be trained. At a fixed parameter state, let

$$a(x) = \phi(Ax), \qquad z_b(x) = a(x) - b\mathbf{1}, \qquad f_{A,B}(x) = Bz_b(x), \tag{29}$$

and assume $\mathbb{E}[a] = \mu\mathbf{1}$. Let $D_x = \mathrm{diag}(\phi'(Ax))$. To first order,

$$\Delta f(x) = BD_x\Delta Ax + \Delta Bz_b(x). \tag{30}$$

Let $J = [J_A\ J_B]$ be the output Jacobian with respect to $\theta = (\mathrm{vec}(A), \mathrm{vec}(B))$. Under column-wise vectorization, equation 30 gives

$$J_A(x) = x^\top \otimes (BD_x), \qquad J_B(x) = z_b(x)^\top \otimes I_{d_{\text{out}}}. \tag{31}$$

For squared loss, the Gauss–Newton (GN) matrix is

$$G_b = \mathbb{E}[J(x)^\top J(x)] = \begin{pmatrix} G_{AA} & G_{AB}(b) \\ G_{BA}(b) & G_{BB}(b) \end{pmatrix}, \tag{32}$$

where

$$G_{AA} = \mathbb{E}\left[(xx^\top) \otimes (D_xB^\top BD_x)\right], \tag{33}$$

$$G_{AB}(b) = \mathbb{E}\left[(xz_b(x)^\top) \otimes (D_xB^\top)\right], \qquad G_{BA}(b) = G_{AB}(b)^\top, \tag{34}$$

$$G_{BB}(b) = \mathbb{E}\left[z_b(x)z_b(x)^\top\right] \otimes I_{d_{\text{out}}} = M_b \otimes I_{d_{\text{out}}}. \tag{35}$$

Thus G_{AA} is independent of b, while the mixed blocks can depend on b. Using $\lambda_B(b)$ from Proposition B.1, define

$$\lambda_A := \lambda_{\max}(G_{AA}), \qquad \lambda_G(b) := \lambda_{\max}(G_b). \tag{36}$$

Proposition B.2 (Full-GN bound and factor-two guarantee)**.** *For every shift b,*

$$\max\{\lambda_A, \lambda_B(b)\} \le \lambda_G(b) \le \lambda_A + \lambda_B(b). \tag{37}$$

Let $b^* \in \arg\min_b \lambda_G(b)$. Consequently,

$$\lambda_G(b^*) \le \lambda_G(\mu) \le 2\lambda_G(b^*). \tag{38}$$

Proof. Restricting a parameter direction to the A or B block gives the lower bound. For any perturbation satisfying $\|\Delta A\|_F^2 + \|\Delta B\|_F^2 = 1$, the triangle inequality and Cauchy–Schwarz give

$$\mathbb{E}\|J_A\,\mathrm{vec}(\Delta A) + J_B\,\mathrm{vec}(\Delta B)\|^2 \le \left(\sqrt{\lambda_A}\|\Delta A\|_F + \sqrt{\lambda_B(b)}\|\Delta B\|_F\right)^2 \tag{39}$$

$$\le \lambda_A + \lambda_B(b), \tag{40}$$

which proves equation 37. Since $b = \mu$ minimizes $\lambda_B(b)$,

$$\lambda_G(b^*) \le \lambda_G(\mu) \le \lambda_A + \lambda_B(\mu) \le \lambda_A + \lambda_B(b^*) \le 2\lambda_G(b^*). \tag{41}$$

□

We test these bounds in the one-block model across ten seeds. At each training checkpoint, we hold A and B fixed, compute the average activation mean $\bar{\mu} = \mathbb{E}_{x,i}[a_i(x)]$, and sweep b to compare $\lambda_G(b)$ with the bounds in equation 37. The minimum over the tested shifts estimates $\lambda_G(b^*)$. In Figure 10, the bounds closely track $\lambda_G(b)$ across shifts (left), while $\lambda_G(\bar{\mu})$ remains close to the empirical minimum throughout training (right).

Thus, removing the activation bias yields nearly optimal Gauss–Newton sharpness in this model, with a factor-two upper bound relative to the optimum.

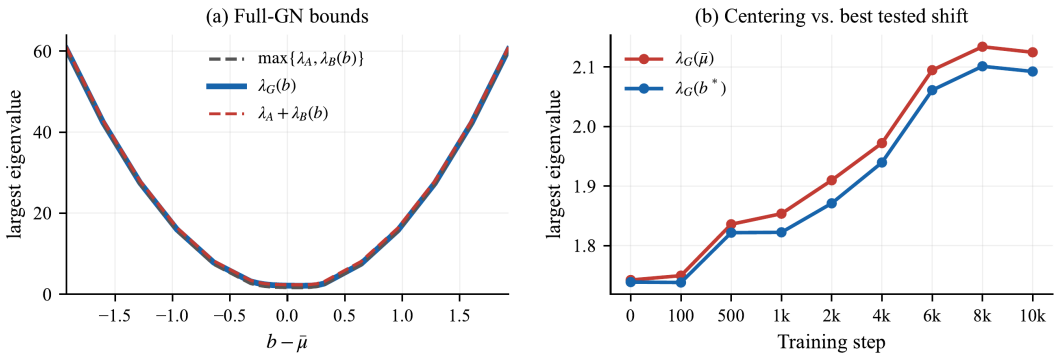


Figure 10: Empirical evaluation of Proposition B.2. Left: The theoretical lower and upper bounds closely track the measured leading Gauss–Newton eigenvalue $\lambda_G(b)$. Right: The value at the average-mean shift, $\lambda_G(\bar{\mu})$, remains close to the empirical minimum, $\lambda_G(b^*)$, throughout training.

B.1.3 From Gauss–Newton to the true Hessian

Let $r(x) = f_\theta(x) - y(x)$ and $g(t) = f_{\theta+t\Delta\theta}(x)$. For squared loss,

$$\Delta\theta^\top H \Delta\theta = \underbrace{\mathbb{E}\|g'(0)\|^2}_{\Delta\theta^\top G \Delta\theta} + \mathbb{E}\langle r(x), g''(0)\rangle. \tag{42}$$

For ReLU, consider a local parameter region in which the activation gates $D_x = \mathrm{diag}(\mathbf{1}_{Ax>0})$ remain fixed for each input x. Then

$$f_{A,B}(x) = B(D_x A x - b\mathbf{1}), \tag{43}$$

and along a joint direction $(\Delta A, \Delta B)$,

$$g''(0) = 2\Delta B D_x \Delta A x. \tag{44}$$

The diagonal blocks satisfy $H_{AA} = G_{AA}$ and $H_{BB} = G_{BB}$. For a unit joint direction,

$$|2\mathbb{E}\langle r, \Delta B D_x \Delta A x\rangle| \le \mathbb{E}[\|r\|\,\|x\|] =: R_b, \tag{45}$$

so

$$|\lambda_{\max}(H_b) - \lambda_{\max}(G_b)| \le \|H_b - G_b\|_2 \le R_b. \tag{46}$$

In particular,

$$\mathcal{L}_b = 0 \quad \Longrightarrow \quad H_b = G_b.$$

Figure 11 shows that the largest eigenvalues of the Hessian and Gauss–Newton matrix at $b = \bar{\mu}$ remain close for most of training.

B.1.4 Multiple down-projections with fixed activations

We finally consider multiple trainable down-projections with fixed activations:

$$f(x) = \sum_{\ell=1}^{L} B_\ell\,(a_\ell(x) - b_\ell\mathbf{1})\,, \qquad \mathbb{E}[a_\ell] = \mu_\ell\mathbf{1}. \tag{47}$$

Define

$$\Sigma_{\text{stack}} := \mathrm{Cov}\left(\begin{pmatrix} a_1(x) \\ \vdots \\ a_L(x) \end{pmatrix}\right), \qquad v_{\mathbf{b}} := \begin{pmatrix} (\mu_1 - b_1)\mathbf{1} \\ \vdots \\ (\mu_L - b_L)\mathbf{1} \end{pmatrix}. \tag{48}$$

Then

$$H_{\text{joint}}(\mathbf{b}) = G_{\text{joint}}(\mathbf{b}) = \left(\Sigma_{\text{stack}} + v_{\mathbf{b}} v_{\mathbf{b}}^\top\right) \otimes I_{d_{\text{out}}}. \tag{49}$$

Thus, setting $b_\ell = \mu_\ell$ for every layer minimizes the joint sharpness.

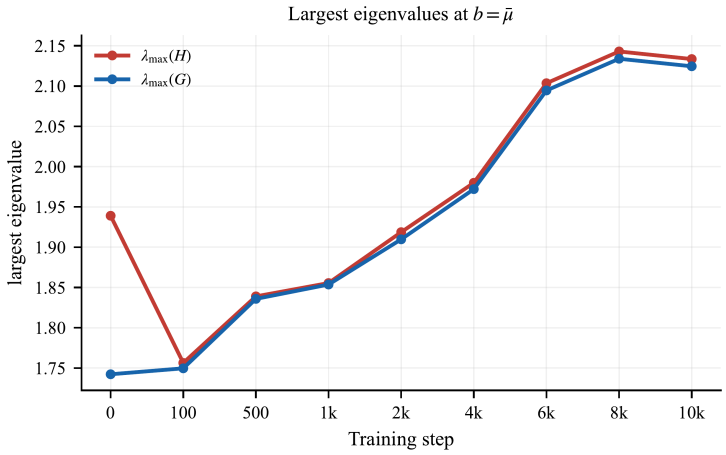


Figure 11: After 100 updates, the largest eigenvalues of the exact Hessian and its Gauss–Newton approximation remain close at $b = \bar{\mu}$.

B.2 Activation mean and parameter gradients

We next study the different update directions of unshifted and shifted models during the early stage of training. We analyze a simplified setting in which only a down-projection layer is trainable. Suppose

$$\mathbb{E}[a(x)] = \mu\mathbf{1}, \qquad \mathrm{Cov}(a(x)) = qI, \qquad y(x) = B_\star a(x). \tag{50}$$

The unshifted model uses $a(x)$, while the shifted model uses

$$z_\mu(x) = a(x) - \mu\mathbf{1}. \tag{51}$$

Both models start from B_{init} and use gradient descent with learning rate η. Let $B_{k-1}^{(0)}$ and $B_{k-1}^{(\mu)}$ denote their parameters before the kth update, and define

$$g_{0,k} := \nabla_B \mathcal{L}_0\left(B_{k-1}^{(0)}\right), \qquad g_{\mu,k} := \nabla_B \mathcal{L}_\mu\left(B_{k-1}^{(\mu)}\right). \tag{52}$$

Let

$$e = \frac{\mathbf{1}}{\sqrt{m}}. \tag{53}$$

For the unshifted and shifted models,

$$M_0 = qI + m\mu^2 ee^\top, \qquad M_\mu = qI. \tag{54}$$

Along e, the eigenvalue of M_0 is $\kappa = (q + m\mu^2)/q$ times that of M_μ. The loss gradients satisfy

$$\nabla_B \mathcal{L}_0(B) = (B - B_\star)M_0, \qquad \nabla_B \mathcal{L}_\mu(B) = (B - B_\star)M_\mu. \tag{55}$$

Let

$$\Delta = B_{\text{init}} - B_\star = \Delta_\parallel + \Delta_\perp, \tag{56}$$

where

$$\Delta_\parallel = \Delta ee^\top, \qquad \Delta_\perp = \Delta(I - ee^\top). \tag{57}$$

The gradients used for the kth update satisfy

$$\begin{aligned} g_{0,k} &= q(1-\eta q)^{k-1}\left(\Delta_\perp + t_k\Delta_\parallel\right), \\ g_{\mu,k} &= q(1-\eta q)^{k-1}\Delta, \end{aligned} \tag{58}$$

where

$$t_k = \kappa\left(\frac{1-\eta\kappa q}{1-\eta q}\right)^{k-1}. \tag{59}$$

Suppose

$$\|\Delta_{\|}\|_F^2 = \frac{1}{m}\|\Delta\|_F^2. \tag{60}$$

Then the angle ψ_k between the gradients satisfies

$$\cos\psi_k = \frac{\mathrm{tr}(g_{0,k}g_{\mu,k}^\top)}{\|g_{0,k}\|_F\|g_{\mu,k}\|_F} = \frac{m-1+t_k}{\sqrt{m\,(m-1+t_k^2)}}. \tag{61}$$

Using the ReLU calculation in Appendix A,

$$\mu = \frac{1}{\sqrt{6\pi}}, \qquad q = \mathbb{E}[\mathrm{ReLU}(Z)^2] - \mu^2 = \frac{1}{6} - \mu^2 = \frac{\pi-1}{6\pi}. \tag{62}$$

Table 2 compares our one-layer theory prediction with full-model gradient angles measured during Adam training.

Table 2: Angles between the unshifted and shifted parameter gradients. The theoretical and measured angles are close early in training.

Step k	Theory angle	Full-model angle
1	77.45°	76.25°
5	77.44°	75.01°
20	77.40°	69.22°
100	77.22°	46.58°

Both the one-layer calculation and the full-model measurements show that the gradients of unshifted and shifted models are nearly orthogonal at the beginning of training.

C Ablation studies

This section tests whether the effect of activation bias is specific to the model configuration. We first examine a joint sweep over model size, then study how the optimal shifting changes with model size, and finally vary other configurations of the controlled model.

C.1 Joint sweep over dimension, width, and depth

We first compare shifted and unshifted ReLU models across a joint sweep of residual-stream dimension d, MLP hidden width m, and number of blocks L. Figure 12 shows the results for $L = 10$ and $L = 20$. The improvement from reducing activation bias is generally larger at greater depth and width.

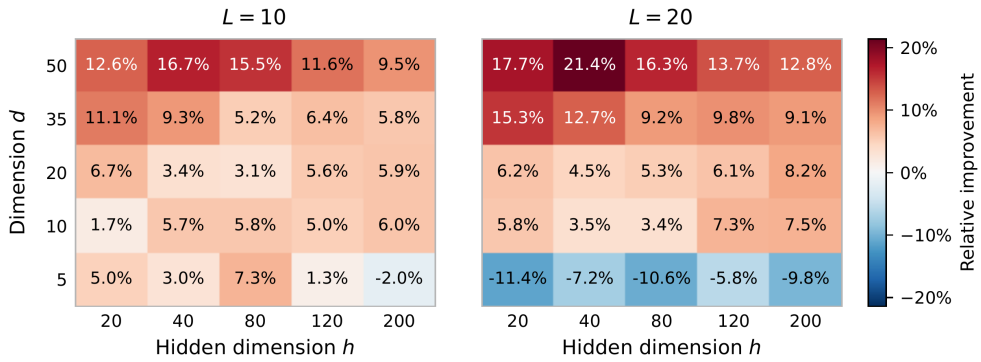


Figure 12: Shifting tends to yield greater improvements as model width and depth increase. For ReLU, colors show the relative improvement $(\mathcal{L}_{\text{unshifted}} - \mathcal{L}_{\text{shifted}})/\mathcal{L}_{\text{unshifted}}$, with red indicating improvement and blue indicating deterioration.

C.2 Seed-level optimal shifts across model sizes

We next compare three representative controlled ReLU models: a small model $(d, m, L) = (5, 20, 2)$, a medium model $(20, 80, 10)$, and a large model $(50, 200, 20)$. We use 100 paired seeds for each model. For each seed, the empirically optimal shift is the value of b with the lowest final loss. Figure 13 shows the distribution of these shifts across seeds for each model size. As model size increases, the optimal shifts concentrate near the theory-predicted value, suggesting that reducing post-activation bias becomes more consistently beneficial across seeds.

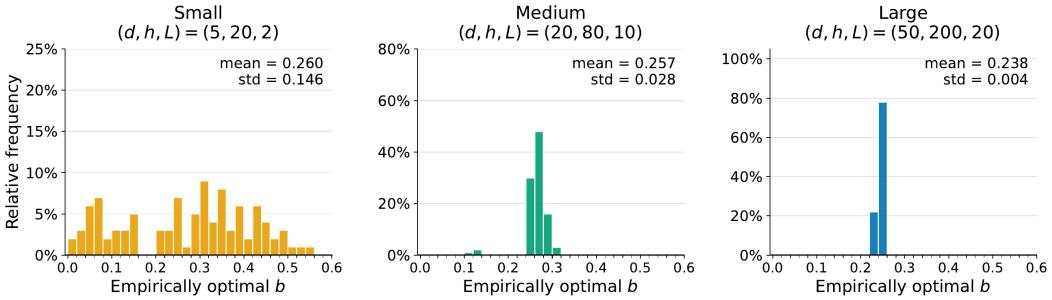


Figure 13: As model size increases, the empirically optimal shifts across seeds become more concentrated near the theory-predicted shift.

C.3 Robustness to architectural changes and cross-entropy loss

We finally test whether reducing activation bias remains effective under three changes to the controlled model. First, we replace pre-norm blocks with post-norm blocks, moving LayerNorm from the block input to the residual-sum output. This tests whether a fixed shift remains effective when the distributions entering the activations can vary more across blocks. Second, we enable trainable biases in the teacher's and students' linear projections to test whether learned affine offsets can compensate for activation bias. Third, we replace mean-squared error with soft-target cross-entropy to test a different training objective. All three variants use ReLU, with all other settings unchanged.

Figure 14 shows that the shifted model achieves lower loss than the unshifted model in all three variants. Thus, its advantage is not limited to pre-norm blocks, bias-free projections, or mean-squared error.

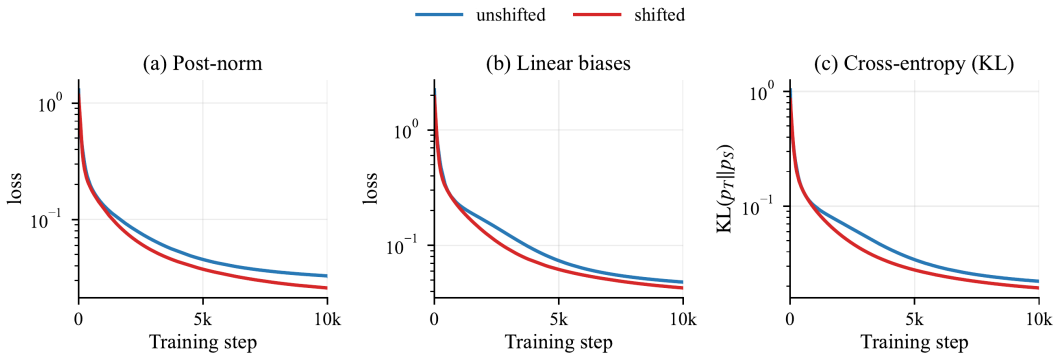


Figure 14: The shifted model retains its loss advantage in more general settings.

C.4 Residual-stream trajectories across all blocks

Figure 15 shows the complete hidden-state mean trajectories across all 20 blocks for the ReLU experiment in Figure 2. The unshifted model exhibits a pronounced transient overshoot across most of the blocks, whereas shifting the activation substantially reduces this overshoot.

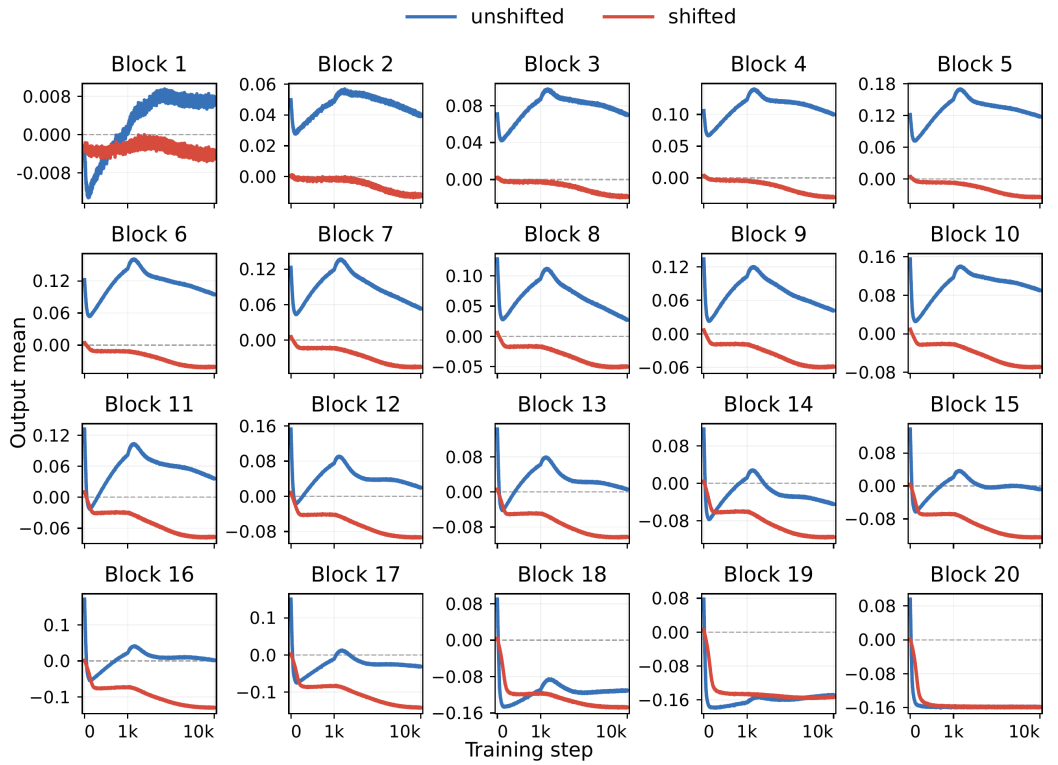


Figure 15: Reducing activation bias suppresses hidden-state overshoot and yields smoother hidden-state mean trajectories.

D ADDITIONAL VGG-STYLE AUTOENCODER EXPERIMENTS

This appendix provides the architecture and additional experimental results for the simplified VGG-style autoencoder introduced in Section 3.1. Figure 16 shows the architecture.

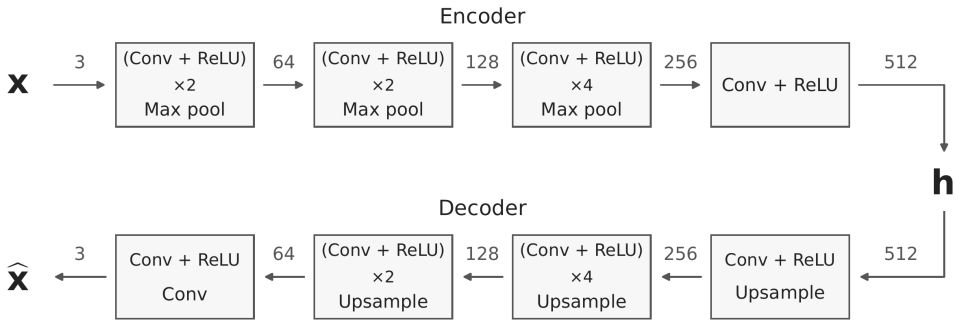


Figure 16: Architecture of the autoencoder. Numbers on arrows denote channel counts. The shifted model replaces each ReLU with $\mathrm{ReLU}(z) - b$.

We also track the reconstruction loss throughout training. Figure 17 shows that the shifted model initially has a higher loss than the unshifted model but achieves a lower loss later.

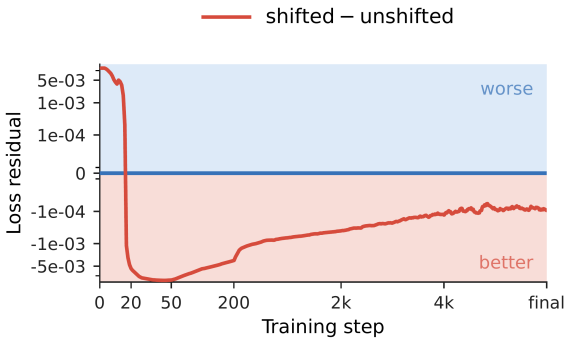


Figure 17: The shifted model initially has a higher reconstruction loss but achieves a lower loss later in training. The plotted residual is $L_{\text{shifted}} - L_{\text{unshifted}}$.

Figure 18 provides additional examples of the component-wise reconstructions analyzed in Section 3.2. Across these images, the unshifted model's activation bias component retains recognizable image structure, whereas the corresponding reconstructions from the shifted model contain little discernible content.

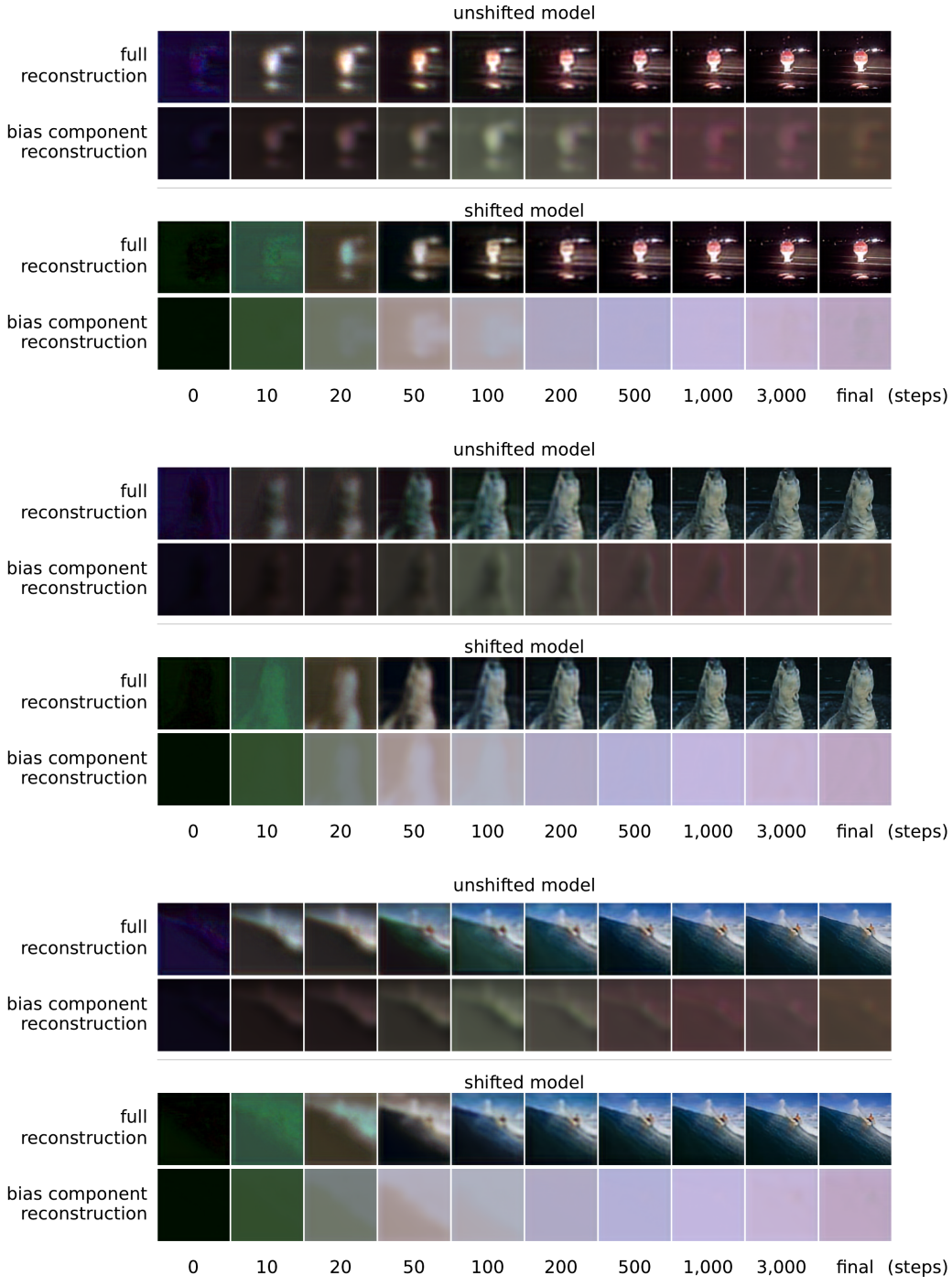


Figure 18: Across different images, reconstructions from the unshifted model's activation bias component first approach and then diverge from the full reconstructions, whereas those from the shifted model contain little recognizable structure.

E ADDITIONAL SCALE-UP EXPERIMENTS

This appendix provides additional scale-up results on performance, hidden-state dynamics, and activation distributions.

E.1 PERFORMANCE ON ADDITIONAL SCALE-UP MODELS

We test whether the gains reported in Section 4 extend across additional model sizes, tasks, and metrics. We evaluate ViT-L on ImageNet image classification, OLMo-2-1B (Team OLMo et al., 2025) on language pretraining, and HyenaDNA (Nguyen et al., 2023) on DNA base-sequence prediction. We also evaluate DiT-B/2 on image generation using FID. Figure 19 reports the results. With adaptive shifting, ViT-L achieves lower test loss and higher accuracy despite a higher training loss; DiT achieves lower FID at every evaluated checkpoint, while OLMo-2-1B and HyenaDNA achieve better performance. Thus, the benefit extends beyond the models and metrics highlighted in the main text.

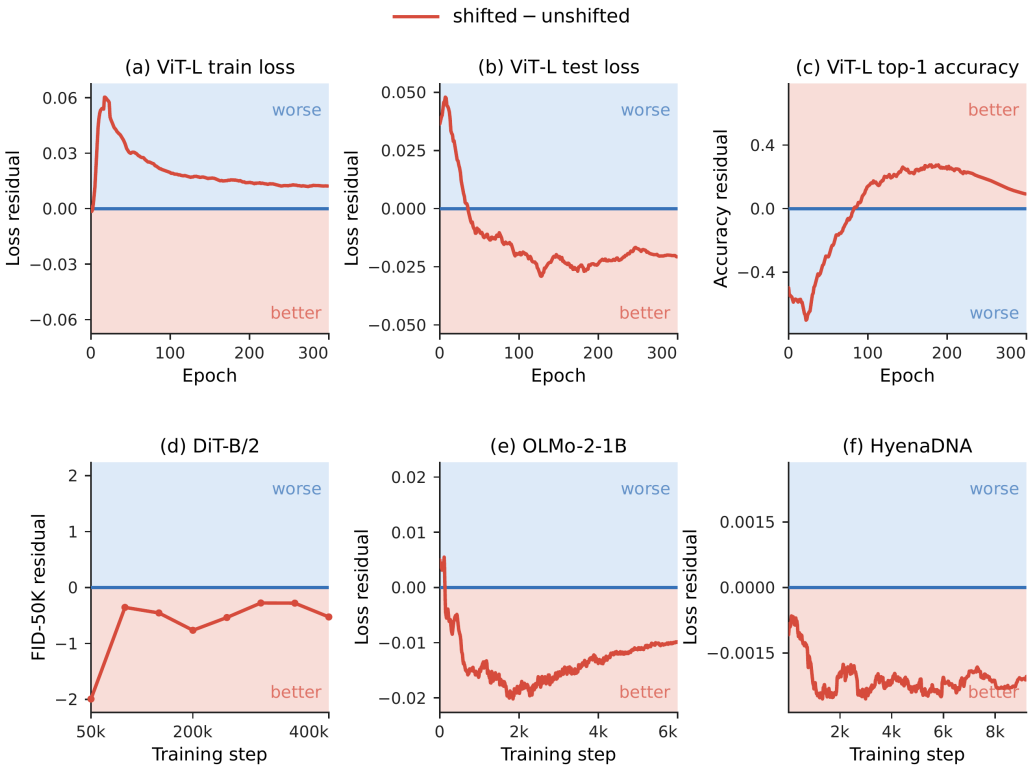


Figure 19: The performance improvements from shifting extend to additional scale-up models and tasks. Residuals are computed as $\Delta M = M_{\text{shifted}} - M_{\text{unshifted}}$, where M denotes loss, FID, or accuracy.

E.2 FURTHER EXAMPLES OF HIDDEN-STATE OVERSHOOT

We examine whether the hidden-state overshoot observed in Section 4.1 also appears in other models. We evaluate publicly released training checkpoints from Pythia-410M (Biderman et al., 2023), OLMo-2-1B (Team OLMo et al., 2025), and Amber-7B (Liu et al., 2023) on a fixed set of inputs and compute hidden-state means as in Section 2.2. Figure 20 presents representative trajectories from these models. Across them, representative blocks exhibit transient overshoots that subsequently recede or reverse, although their timing and direction vary.

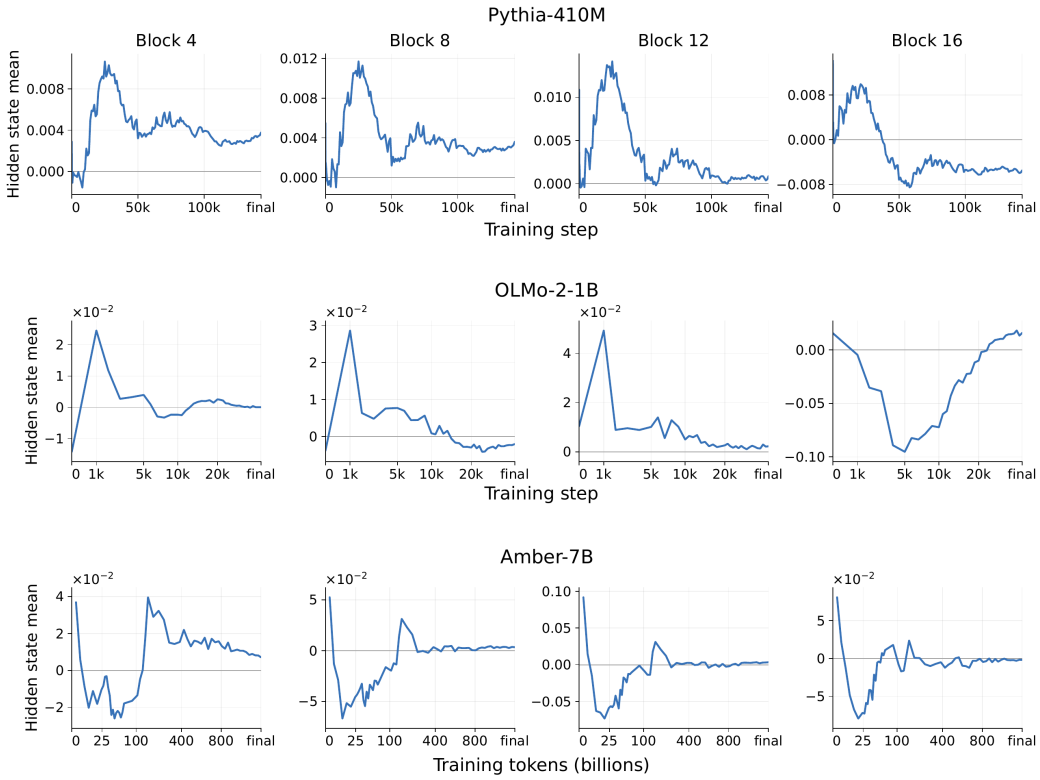


Figure 20: Hidden-state mean trajectories exhibit overshoot across different scale-up models.

E.3 Activation-Magnitude Distributions in OLMo-2-1B

Our analysis so far has focused on activation bias, but the mean alone cannot show how shifting affects the broader distribution of activation magnitudes. We therefore compare the absolute-value distributions of post-activations and residual-stream states. Figure 21 shows that, with shifting, the post-activation distributions change relatively little over training. In the unshifted model, they develop an increasingly pronounced concentration near zero in deeper blocks. The residual-stream distributions, however, remain broadly similar across the two models and more closely resemble the shifted model's post-activation distributions. This contrast suggests that the unshifted model must map a more sharply concentrated post-activation distribution to a comparatively smooth residual-stream distribution. Such a larger distributional adjustment may make learning less direct, offering a possible explanation for the improved dynamics observed when activation bias is reduced.

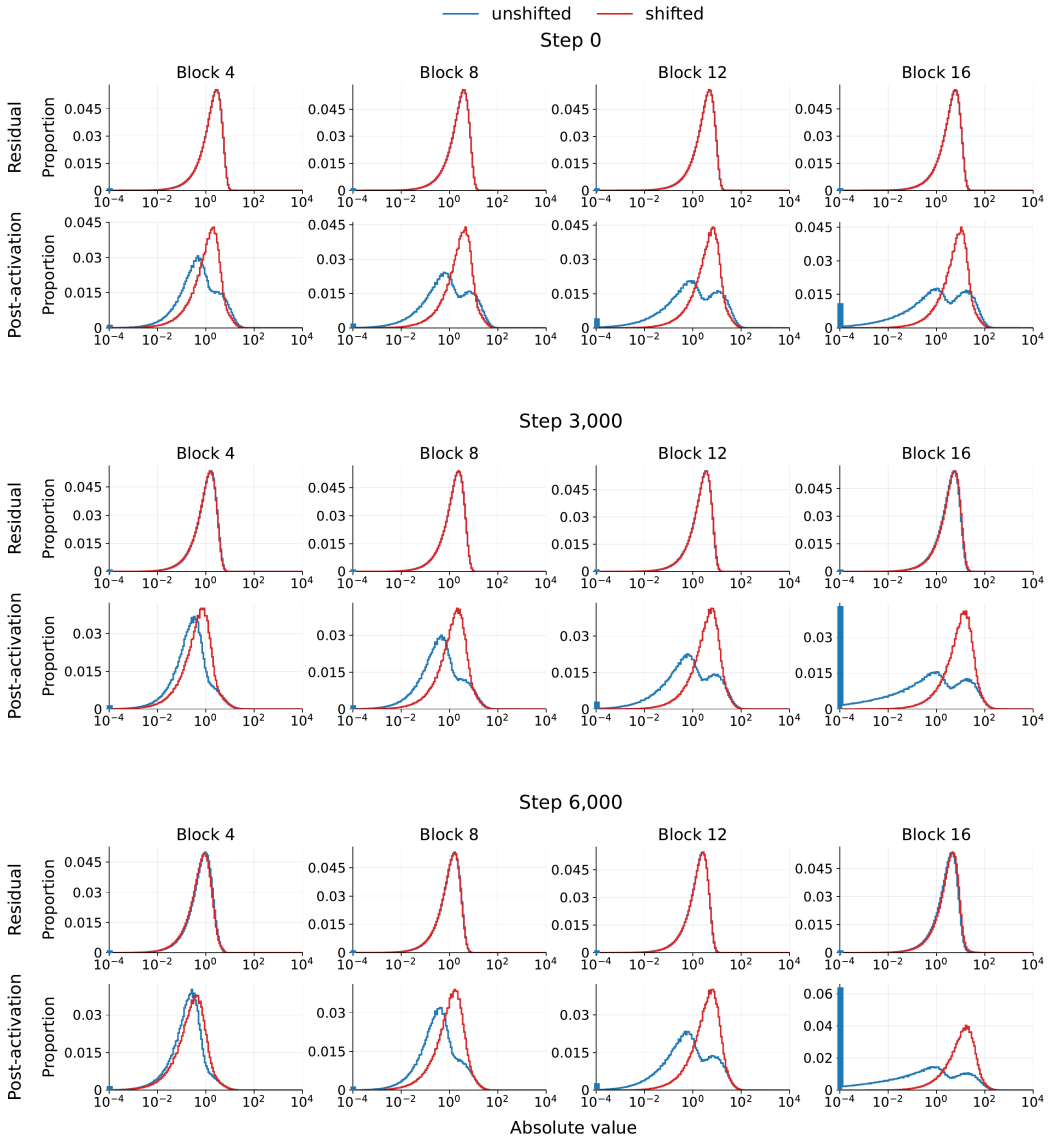


Figure 21: In OLMo-2-1B, the shifted model maintains more stable post-activation magnitude distributions over training, with less accumulation of near-zero values. Moreover, its post-activation and residual-stream magnitude distributions closely resemble each other, whereas those of the unshifted model differ substantially.