Peri-LN: Revisiting Normalization Layer in the Transformer Architecture

Jeonghoon Kim[1,2] Byeongchan Lee[2] Cheonbok Park[1,2] Yeontaek Oh[1] Beomjun Kim[2] Taehwan Yoo[1]
Seongjin Shin[1] Dongyoon Han[3] Jinwoo Shin[†,2] Kang Min Yoo[†,1]
[1]NAVER Cloud [2]Korea Advanced Institute of Science and Technology (KAIST) [3]NAVER AI Lab

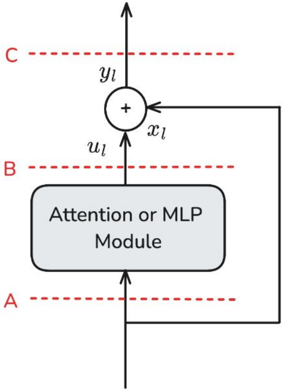
In this study, we :

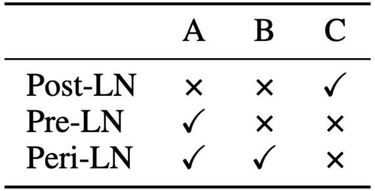
1. Present an in-depth analysis of Post-LN and Pre-LN in large-scale Transformers, examining **how variance and gradient properties evolve beyond initialization**.
2. Investigate **Peri-LN** to understand how normalizing both the inputs and outputs of each module moderates hidden-state behavior during forward and backward propagation, providing a systematic perspective on this underexplored alternative.
3. Provide quantitative evidence on **how large activation influences** training stability, benchmark performance, and model behaviors.

Motivation

- **Massive activations remain poorly understood.** Persistent high-magnitude activations in Pre-LN can act as residual biases that distort attention and limit generalization; their root causes and long-term effects are unclear.
- **Theory predicts depth-dependent instability.** Signal-propagation analyses link exploding/vanishing gradients to LN placement, depth, and residual pathways, but empirical validation at scale is limited.
- **Underexplored alternative: Peri-LN.** While Post- and Pre-LN dominate practice, why Gemma2 & 3, and Olmo2 use different architecture?
- **Goal of this study.** Provide a rigorous, large-scale empirical and theoretical comparison of Post-, Pre-, and Peri-LN to clarify when each placement best balances training stability, computational cost, and downstream performance.

Transformer Architecture Through the Placement of Layer Normalization





	A	B	C
Post-LN	×	×	✓
Pre-LN	✓	×	×
Peri-LN	✓	✓	×

Figure 2. Placement of normalization in Transformer sub-layer.

$$\mathrm{Var}(x_{l+1}) \;\approx\; \mathrm{Var}(x_l) + \beta_0, \tag{4}$$

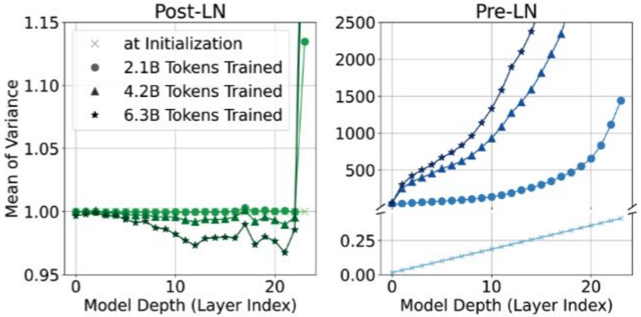
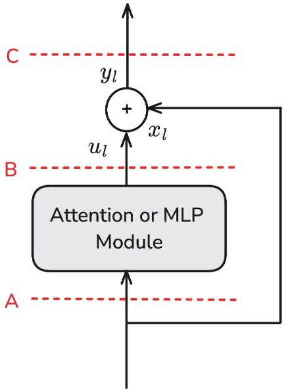


Figure 1. Illustration of hidden-state variance across different model depths and training iterations. From the initialization stage up to the point where 6.3 billion tokens were trained, we observed the variance growth of hidden states for Pre-LN and Post-LN architectures. The analysis was conducted using a 1.5B-parameter model, and consistent trends were observed across models of different sizes. Detailed settings and more results are in Section 4.4.2.

Transformer Architecture Through the Placement of Layer Normalization



	A	B	C
Post-LN	×	×	✓
Pre-LN	✓	×	×
Peri-LN	✓	✓	×

Figure 2. Placement of normalization in Transformer sub-layer.

1. *(Optional) Initial Embedding Normalization:*

$$y_o = \mathrm{Norm}(x_o),$$

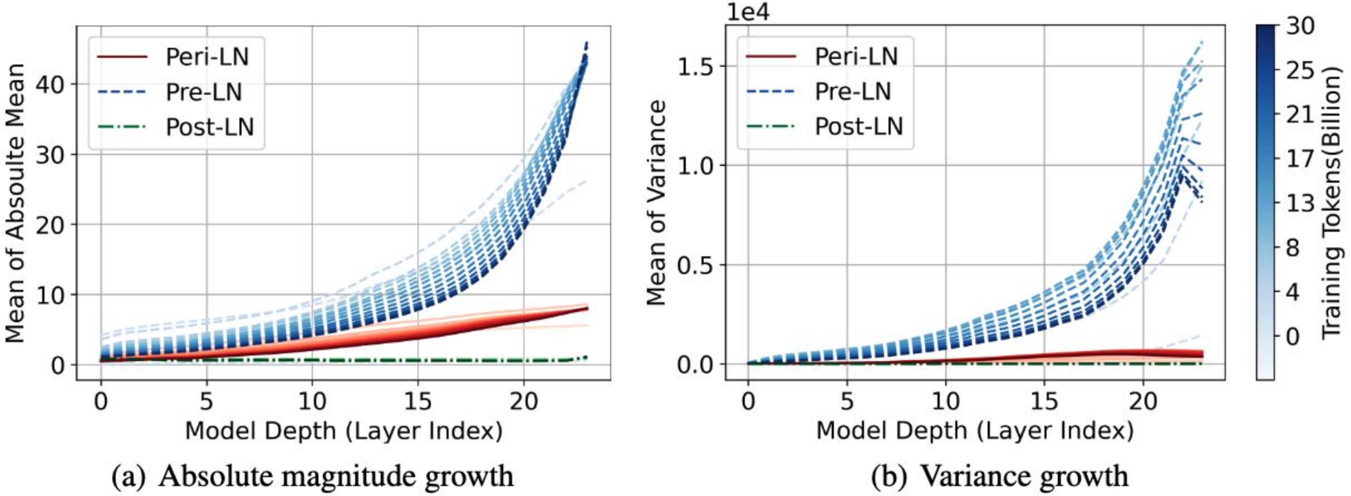
2. *Input- & Output-Normalization per Layer:*

$$y_l = x_l + \mathrm{Norm}\Big(\mathrm{Module}\big(\mathrm{Norm}(x_l)\big)\Big), \qquad (3)$$

3. *Final Embedding Normalization:*

$$y_L = \mathrm{Norm}(x_L),$$

Growth of Hidden State

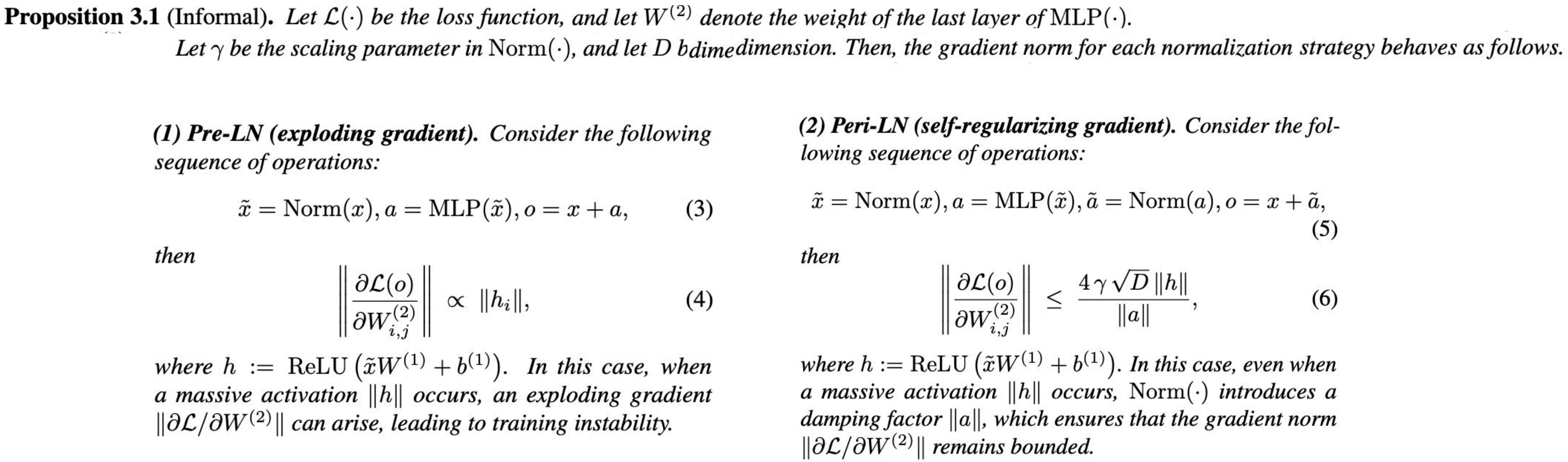


(a) Absolute magnitude growth

(b) Variance growth

Figure 5. This figure shows the forward growth patterns of hidden states for different architectures, highlighting the structural impact of normalization placement. Each model contains 1.5 billion parameters (excluding the embedding size). We confirmed that the observed trend remains consistent across all model sizes.

Transformer Architecture Through the Placement of Layer Normalization



Proposition 3.1 (Informal)**.** *Let $\mathcal{L}(\cdot)$ be the loss function, and let $W^{(2)}$ denote the weight of the last layer of* $\mathrm{MLP}(\cdot)$*. Let γ be the scaling parameter in* $\mathrm{Norm}(\cdot)$*, and let D bdimedimension. Then, the gradient norm for each normalization strategy behaves as follows.*

(1) Pre-LN (exploding gradient). *Consider the following sequence of operations:*

$$\tilde{x} = \mathrm{Norm}(x), a = \mathrm{MLP}(\tilde{x}), o = x + a, \tag{3}$$

then

$$\left\| \frac{\partial \mathcal{L}(o)}{\partial W^{(2)}_{i,j}} \right\| \propto \|h_i\|, \tag{4}$$

where $h := \mathrm{ReLU}\left(\tilde{x}W^{(1)} + b^{(1)}\right)$. In this case, when a massive activation $\|h\|$ occurs, an exploding gradient $\|\partial\mathcal{L}/\partial W^{(2)}\|$ can arise, leading to training instability.

(2) Peri-LN (self-regularizing gradient). *Consider the following sequence of operations:*

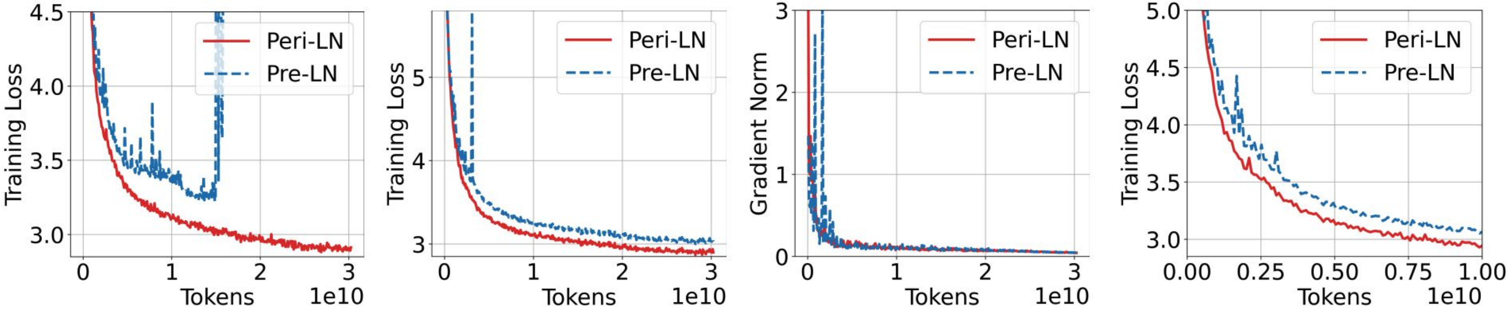
$$\tilde{x} = \mathrm{Norm}(x), a = \mathrm{MLP}(\tilde{x}), \tilde{a} = \mathrm{Norm}(a), o = x + \tilde{a}, \tag{5}$$

then

$$\left\| \frac{\partial \mathcal{L}(o)}{\partial W^{(2)}_{i,j}} \right\| \leq \frac{4\,\gamma\,\sqrt{D}\,\|h\|}{\|a\|}, \tag{6}$$

where $h := \mathrm{ReLU}\left(\tilde{x}W^{(1)} + b^{(1)}\right)$. In this case, even when a massive activation $\|h\|$ occurs, $\mathrm{Norm}(\cdot)$ *introduces a damping factor $\|a\|$, which ensures that the gradient norm $\|\partial\mathcal{L}/\partial W^{(2)}\|$ remains bounded.*

Early Stage Instability in Pre-Training



(a) Divergence at seed 2 (b) Loss spike at seed 3 (c) Gradient spikes at seed 5 (d) Loss spikes at seed 5

Figure 4. Common case of early stage instability in pre-training. In most of our experiments across different random seeds, the Pre-LN architecture exhibited early-stage instability. Although we initially suspected that a high learning rate might be the root cause, lowering it did not substantially mitigate these issues. By contrast, under the same settings, Peri-LN displayed stable training curves.

We posit that the instability of Pre-LN arises from three factors:

1. the hidden state variance exhibits a sudden surge from initialization through the early stages of optimization, deviating from the linear trend predicted by Eq.(4)

$$\mathrm{Var}(x_{l+1}) \approx \mathrm{Var}(x_l) + \beta_0, \tag{4}$$

2. the exponential growth of hidden state variance across both depth and training steps
3. the instability caused by the massive activations (Proposition 3.1).

Evaluations of Post-LN, Pre-LN, and Peri-LN Transformers

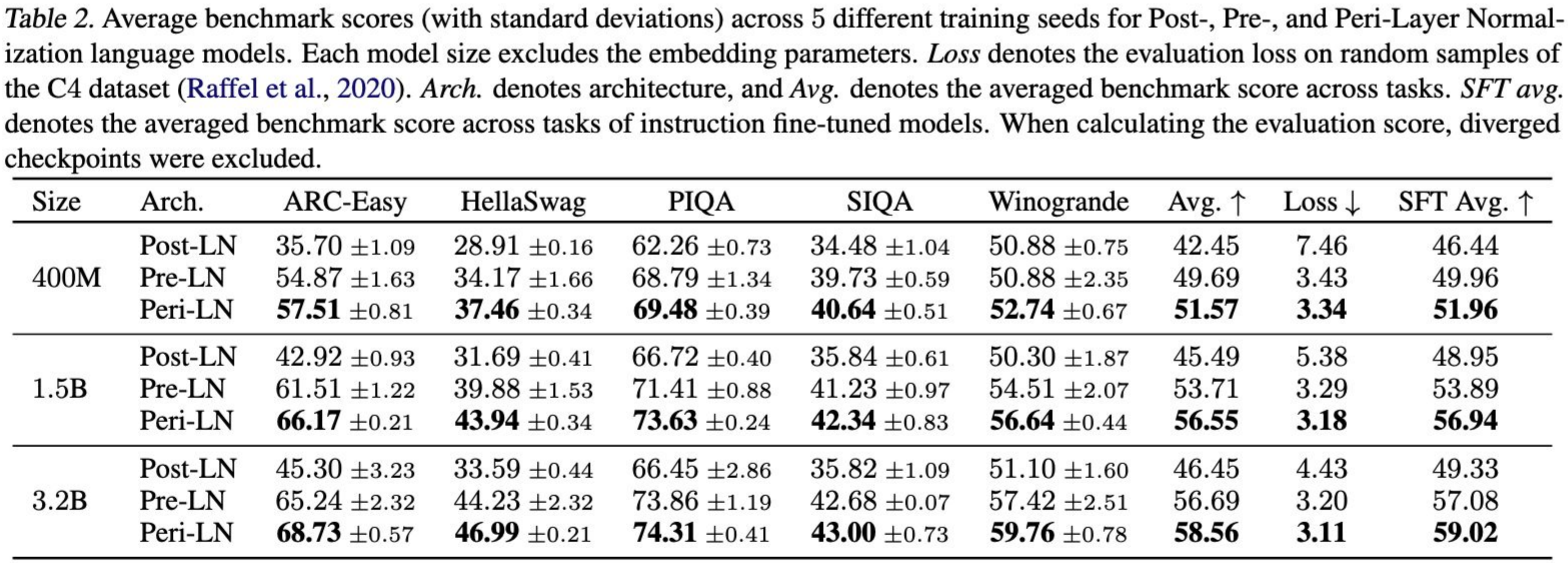


Table 2. Average benchmark scores (with standard deviations) across 5 different training seeds for Post-, Pre-, and Peri-Layer Normalization language models. Each model size excludes the embedding parameters. *Loss* denotes the evaluation loss on random samples of the C4 dataset (Raffel et al., 2020). *Arch.* denotes architecture, and *Avg.* denotes the averaged benchmark score across tasks. *SFT avg.* denotes the averaged benchmark score across tasks of instruction fine-tuned models. When calculating the evaluation score, diverged checkpoints were excluded.

Size	Arch.	ARC-Easy	HellaSwag	PIQA	SIQA	Winogrande	Avg. ↑	Loss ↓	SFT Avg. ↑
400M	Post-LN	35.70 ±1.09	28.91 ±0.16	62.26 ±0.73	34.48 ±1.04	50.88 ±0.75	42.45	7.46	46.44
	Pre-LN	54.87 ±1.63	34.17 ±1.66	68.79 ±1.34	39.73 ±0.59	50.88 ±2.35	49.69	3.43	49.96
	Peri-LN	**57.51** ±0.81	**37.46** ±0.34	**69.48** ±0.39	**40.64** ±0.51	**52.74** ±0.67	**51.57**	**3.34**	**51.96**
1.5B	Post-LN	42.92 ±0.93	31.69 ±0.41	66.72 ±0.40	35.84 ±0.61	50.30 ±1.87	45.49	5.38	48.95
	Pre-LN	61.51 ±1.22	39.88 ±1.53	71.41 ±0.88	41.23 ±0.97	54.51 ±2.07	53.71	3.29	53.89
	Peri-LN	**66.17** ±0.21	**43.94** ±0.34	**73.63** ±0.24	**42.34** ±0.83	**56.64** ±0.44	**56.55**	**3.18**	**56.94**
3.2B	Post-LN	45.30 ±3.23	33.59 ±0.44	66.45 ±2.86	35.82 ±1.09	51.10 ±1.60	46.45	4.43	49.33
	Pre-LN	65.24 ±2.32	44.23 ±2.32	73.86 ±1.19	42.68 ±0.07	57.42 ±2.51	56.69	3.20	57.08
	Peri-LN	**68.73** ±0.57	**46.99** ±0.21	**74.31** ±0.41	**43.00** ±0.73	**59.76** ±0.78	**58.56**	**3.11**	**59.02**

Q & A

jeonghoon.samuel@gmail.com