CoupledNorm: Efficient Normalization via Shared RMS Statistics

Martin Loretz [1] **Sepp Hochreiter** [1 2]

Abstract

Normalization layers are treated as computationally inexpensive, yet they still introduce non-negligible latency during autoregressive decoding. We propose CoupledNorm, a simple modification to Pre-Norm Transformer blocks that removes one of the two RMS calculations per layer. CoupledNorm shares the per-token RMS statistics between the Attention and MLP sub-layers while retaining separate learned affine parameters. In GPT-2-scale pretraining, CoupledNorm matches the training loss and yields similar downstream performance. For pretrained 0.6B–8B models, we introduce CoupledNorm post hoc via distillation with small downstream degradation. By fusing the remaining pre-MLP normalization operations into existing kernels, CoupledNorm achieves an end-to-end decoding speedup of up to 2%. These results suggest that shared second-order statistics are sufficient for effective normalization, challenging the need for independent normalization per sub-layer while improving efficiency.

1. Introduction

Normalization layers are crucial for stabilizing the training of large language models (Vaswani et al., 2017). Although typically regarded as computationally inexpensive, they still account for a non-negligible fraction of total inference latency during autoregressive decoding.

For models using the standard Pre-Norm configuration (Xiong et al., 2020), normalization is applied before the Attention and MLP sub-layers. However, due to the residual connections bridging these components, the hidden states at these sequential positions are not independent quantities. Consequently, when applying RMSNorm (Zhang & Sennrich, 2019), the independently calculated root mean square (RMS) statistics at these two positions result in very similar values. The residual stream changes directionally through attention, and its second-order scale changes predictably enough that a fresh RMS calculation before the MLP is often redundant. Motivated by this observation, we propose CoupledNorm, a streamlined normalization scheme that computes the RMS scaling factor once prior to the Attention block and reuses this exact statistic to normalize the subsequent MLP block.

In summary, we make the following contributions:

- We introduce CoupledNorm, a minimal architectural change that removes one RMS calculation per block.
- We demonstrate that CoupledNorm can be integrated during pretraining with nearly identical dynamics or introduced post hoc via distillation.
- We accelerate inference by up to 2% across various models by fusing scaling operations into existing kernels, thereby eliminating one kernel launch per block.

The source code of a reference implementation is available at `https://github.com/NX-AI/coupled-norm`.

2. Background

Language Model Normalization Modern architectures employ the Pre-Norm configuration to maintain robustness by preserving an unhindered identity path for gradient propagation along the residual stream. While LayerNorm (Ba et al., 2016) was the original standard, the field has transitioned to RMSNorm (Zhang & Sennrich, 2019) to reduce computational overhead by demonstrating that mean-centering is not strictly necessary for variance stabilization.

For the purpose of our analysis, we conceptualize normalization as a two-step process. In the first step, we compute the root mean square (RMS) of the input vector $x \in \mathbb{R}^n$:

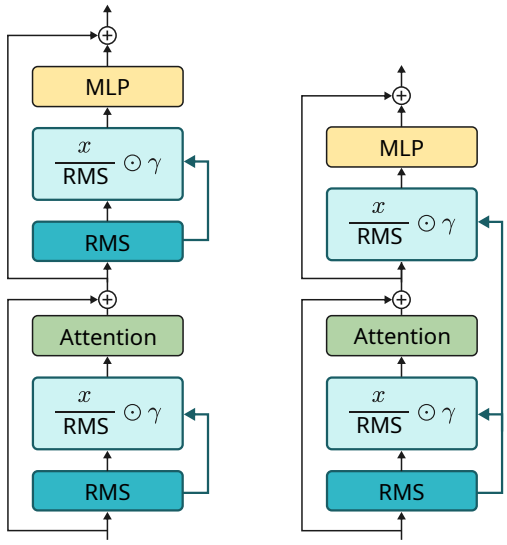
$$\mathrm{RMS}(x) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2 + \epsilon} \tag{1}$$

In the second step, we apply this scaling factor and the learned element-wise weights $\gamma \in \mathbb{R}^n$:

$$y = \frac{x}{\mathrm{RMS}(x)} \odot \gamma \tag{2}$$

[1]NXAI, Austria [2] Johannes Kepler University Linz, Austria. Correspondence to: Martin Loretz <martin.loretz@nx-ai.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.



(a) Standard Block *(b)* CoupledNorm Block

Figure 1. Architectural comparison of a standard Pre-Norm block (a) and the proposed CoupledNorm block (b). The RMSNorm operation is decomposed into the calculation of the RMS scaling factor and its element-wise application.

3. Related Work

Normalization-Free Transformers Recent research has challenged the necessity of explicit variance stabilization (Zhu et al., 2025; Chen et al., 2025). For instance, Zhu et al. (2025) introduced Dynamic Tanh (DyT), an element-wise operation that replaces traditional normalization by emulating its scaling behavior without calculating activation statistics. Other approaches aim to phase out normalization layers entirely during the training process (Kanavalau et al., 2026; Baroni et al., 2025), typically through the use of an auxiliary loss. These findings suggest that strict, per-layer variance tracking is not essential for stability. CoupledNorm builds upon this intuition by retaining the empirical benefits of RMS scaling while reducing its computational frequency.

Parallel Attention and MLP Structural optimizations, such as those introduced in GPT-J (Wang & Komatsuzaki, 2022), Falcon (Almazrouei et al., 2023), and PaLM (Chowdhery et al., 2023), employ a parallel architecture where the Attention and MLP sub-layers share the same normalized input. While this configuration requires only a single normalization pass per block, it inherently reduces the computational depth. CoupledNorm achieves comparable efficiency gains while maintaining the full sequential depth of the model. Instead of parallelizing the sub-layers, we share the normalization statistics across the sequential boundary, thereby eliminating one reduction operation per block.

4. CoupledNorm

4.1. Motivation and Design

A standard Pre-Norm block applies normalization independently before the Attention and MLP sub-layers, requiring two separate RMS calculations per layer (Figure 1a). However, because these sub-layers are bridged by a residual connection, the hidden state preceding the MLP remains similar to the hidden state preceding the Attention layer. Consequently, the RMS statistics at these two sequential positions are correlated. CoupledNorm exploits this by reusing the Attention sub-layer's scaling factor for the subsequent MLP, effectively coupling the two normalization operations (Figure 1b).

4.2. Mathematical Formulation

Instead of calculating the RMS statistic twice per block, CoupledNorm computes it strictly once prior to the Attention sub-layer and reuses it for the subsequent operation. Formally, given an input vector $x \in \mathbb{R}^n$, we first compute the shared RMS scaling factor:

$$\sigma_{\text{rms}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2 + \epsilon} \tag{3}$$

This factor is used to normalize the input for the Attention sub-layer, alongside an independently learned affine parameter $\gamma_{\text{attn}} \in \mathbb{R}^n$:

$$\hat{x}_{\text{attn}} = \frac{x}{\sigma_{\text{rms}}} \odot \gamma_{\text{attn}} \tag{4}$$

After the Multi-Head Attention (MHA) operation, we obtain the intermediate state h by adding the Attention output to the residual stream:

$$h = x + \text{MHA}(\hat{x}_{\text{attn}}) \tag{5}$$

Crucially, CoupledNorm normalizes h for the subsequent MLP block by reusing the previously computed σ_{rms} alongside a second independent affine parameter $\gamma_{\text{mlp}} \in \mathbb{R}^n$:

$$\hat{x}_{\text{mlp}} = \frac{h}{\sigma_{\text{rms}}} \odot \gamma_{\text{mlp}} \tag{6}$$

Finally, the block completes its forward pass by adding the MLP output to the intermediate residual stream:

$$x_{\text{out}} = h + \text{MLP}(\hat{x}_{\text{mlp}}) \tag{7}$$

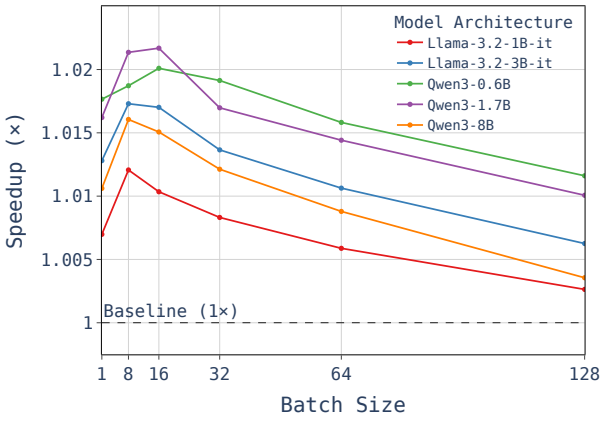


Figure 2. Relative decoding speedup of CoupledNorm over the baseline across various batch sizes. Speedup is defined as the ratio of CoupledNorm generation throughput to that of the baseline.

4.3. Hardware Efficiency

For autoregressive decoding, particularly at small batch sizes, inference latency is significantly impacted by GPU kernel launch overheads rather than by matrix multiplications. By omitting the secondary RMS calculation, CoupledNorm effectively eliminates one kernel launch per Transformer block. This is achieved by fusing the remaining pre-MLP normalization operations into adjacent kernels: the residual addition is integrated into the bias term of the Attention output projection with an `addmm` operation, while the affine weights γ_{mlp} are statically folded into the MLP up-projection weights. Furthermore, since scalar division is a linear operation, it can be moved past the up-projection and executed within the subsequent SiLU kernel, where the σ_{rms} scaling is applied immediately prior to the activation. By eliminating a fixed latency cost per layer regardless of batch size, this fusion strategy provides a relative speedup that is most apparent for smaller batches, yielding a total latency reduction that scales proportionally with the depth of the network.

In contrast, model throughput during pretraining is typically bounded by memory bandwidth and kernel synchronization overheads. Standard RMSNorm requires a reduction operation across the hidden dimension in the forward pass, as well as a complex secondary reduction during the backward pass to compute the variance gradient. These reductions require cross-thread synchronizations that are inherently slower than simple element-wise operations. CoupledNorm bypasses these bottlenecks by treating the reused σ_{rms} as a scalar constant for the MLP block. This architectural adjustment reduces the secondary normalization to a purely element-wise operation, completely eliminating the need for a variance gradient reduction and directly translating to accelerated training throughput.

5. Experiments

5.1. Performance

To evaluate our approach, we begin by analyzing its impact on decoding performance. Specifically, we evaluate CoupledNorm against the original Qwen 3 (Yang et al., 2025) and Llama 3.2 (Grattafiori et al., 2024) checkpoints, serving as our baselines. Both configurations were evaluated on an NVIDIA H100 GPU across a variety of batch sizes. For this evaluation, each model was tasked with generating a 128-token response to the prompt, "Who was Alan Turing?".

Figure 2 illustrates the relative speedups, defined as the ratio of the CoupledNorm model's generation throughput to that of the baseline. Our results demonstrate that our method achieves a 1–2% speedup across all evaluated models at smaller batch sizes. Furthermore, relative speedups scale with model depth due to cumulative kernel launch reductions, yielding smaller gains for the 16-layer Llama 1B compared to the other models (28+ layers). Because CoupledNorm eliminates a constant overhead, this speedup naturally diminishes at larger batch sizes where matrix multiplications dominate overall compute latency.

5.2. Pretraining

To validate the pretraining stability of our approach, we trained a GPT-2 model (Radford et al., 2019) from scratch on a 10-billion-token subset of the FineWeb-Edu dataset (Lozhkov et al., 2024). The training was conducted using the NanoGPT framework (Karpathy, 2023) with default hyperparameters, where all standard Transformer blocks were replaced with our proposed CoupledNorm architecture. We subsequently evaluated downstream performance on the HellaSwag benchmark (Zellers et al., 2019).

To ensure a robust evaluation, both the baseline and CoupledNorm configurations were trained across eight distinct random seeds (8–15). As detailed in Table 1, the CoupledNorm architecture increased training throughput by 0.54% while converging to the same final loss as the baseline. This training acceleration is attributed primarily to reduced computational work, specifically the elimination of the secondary variance gradient reduction during the backward pass.

Table 1. GPT-2 pretraining performance for Baseline and CoupledNorm, averaged over 8 runs (seeds 8–15).

Model	Train TP (kT/s) ↑	Train Loss ↓	HellaSwag ↑
Baseline	443.0 (± 3.3)	3.096 (± 0.0012)	30.51 (± 0.0014)
CoupledNorm	445.4 (± 3.4)	3.096 (± 0.0014)	30.32 (± 0.0019)

5.3. Distillation

To adapt pretrained models to our proposed architecture, we employed architecture distillation (Hinton et al., 2015; Sanh et al., 2019), initializing the student model with the teacher's weights. We distilled various Qwen 3 (Yang et al., 2025) and Llama 3.2 (Grattafiori et al., 2024) models ranging from 0.6B to 8B parameters. To facilitate rapid adaptation while preserving model expressivity, we utilized DoRA (Liu et al., 2024) with a rank $r = 1024$ and $\alpha = 2048$. The distillation process used the stem subset of the Nemotron Post-Training Dataset v2 (Nathawani et al., 2025) and relied exclusively on Kullback-Leibler (KL) divergence to align the student's output distribution with that of the teacher.

To mitigate the distribution shift introduced by the architectural modification, we apply a fixed adjustment factor α. We define this scaling factor for the MLP block as $\sigma'_{\text{mlp}} = \alpha \cdot \sigma_{\text{rms}}$, where σ_{rms} is the statistic computed at the preceding Attention layer. This factor is derived from the average increase in RMS between these two sub-layers of the pretrained model, as illustrated in Section A, yielding $\alpha = 1.1$ for Llama and $\alpha = 1.09$ for Qwen. This calibration introduces no additional computational overhead, as it is absorbed into the affine weights γ_{mlp} during inference. Given the sharp increase in RMS values observed within the initial layers (see Figure 3), we preserve the standard RMSNorm architecture for the first layer in Qwen and the first four layers in Llama.

The distilled models were evaluated in a 5-shot setting on the MMLU (Hendrycks et al., 2020) and GPQA (Rein et al., 2024) benchmarks (Table 2). While the distilled variants exhibit a slight degradation in performance relative to their baseline architectures, we anticipate that this gap could be further minimized by adopting a more comprehensive distillation strategy.

Table 2. Five-shot downstream evaluation of the baseline and distilled CoupledNorm models.

MODEL	ARCHITECTURE	MMLU ↑	GPQA ↑
LLAMA 1B	BASELINE	46.0	28.8
	COUPLEDNORM	45.9	29.0
LLAMA 3B	BASELINE	60.6	33.5
	COUPLEDNORM	60.3	31.9
QWEN 0.6B	BASELINE	47.4	27.9
	COUPLEDNORM	47.3	26.6
QWEN 1.7B	BASELINE	60.2	29.9
	COUPLEDNORM	59.9	29.7
QWEN 8B	BASELINE	74.9	38.4
	COUPLEDNORM	74.6	38.6

6. Conclusion

In this work, we introduced CoupledNorm, an efficient normalization strategy that reuses the RMS statistics from the Attention layer to normalize the subsequent MLP layer. We demonstrated empirically that applying CoupledNorm during the pretraining of a GPT-2 model maintains competitive downstream performance while increasing training throughput by 0.54%. By scaling this approach to larger models via architecture distillation, we achieved inference speedups of up to 2% across various models.

Future work will primarily focus on pretraining larger models to evaluate whether training stability is maintained at scale. Additionally, we intend to extend the sharing of normalization statistics across multiple blocks, ultimately developing an architecture that computes explicit normalizations only every few layers.

Broader Impact This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Goffinet, É., Hesslow, D., Launay, J., Malartic, Q., et al. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*, 2023.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Baroni, L., Khara, G., Schaeffer, J., Subkhankulov, M., and Heimersheim, S. Transformers don't need layernorm at inference time: Scaling layernorm removal to gpt-2 xl and the implications for mechanistic interpretability. *arXiv preprint arXiv:2507.02559*, 2025.

Chen, M., Lu, T., Zhu, J., Sun, M., and Liu, Z. Stronger normalization-free transformers. *arXiv preprint arXiv:2512.10938*, 2025.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *Journal of machine learning research*, 24 (240):1–113, 2023.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring mas-

sive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Kanavalau, A., Alonso, C. A., and Lall, S. Gated removal of normalization in transformers enables stable training and efficient inference. *arXiv preprint arXiv:2602.10408*, 2026.

Karpathy, A. nanoGPT, January 2023. URL https://github.com/karpathy/nanoGPT.

Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

Lozhkov, A., Ben Allal, L., von Werra, L., and Wolf, T. Fineweb-edu: the finest collection of educational content, 2024. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu.

Nathawani, D., Ding, S., Lavrukhin, V., Gitman, I., Majumdar, S., Bakhturina, E., Ginsburg, B., and Polak Scowcroft, J. Nemotron-Post-Training-Dataset-v2, August 2025. URL https://huggingface.co/datasets/nvidia/Nemotron-Post-Training-Dataset-v2.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*, 2024.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, B. and Komatsuzaki, A. Gpt-j-6b: A 6 billion parameter autoregressive language model. 2021. *URL https://github. com/kingoflolz/mesh-transformer-jax*, pp. 8, 2022.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. On layer normalization in the transformer architecture. In *International conference on machine learning*, pp. 10524–10533. PMLR, 2020.

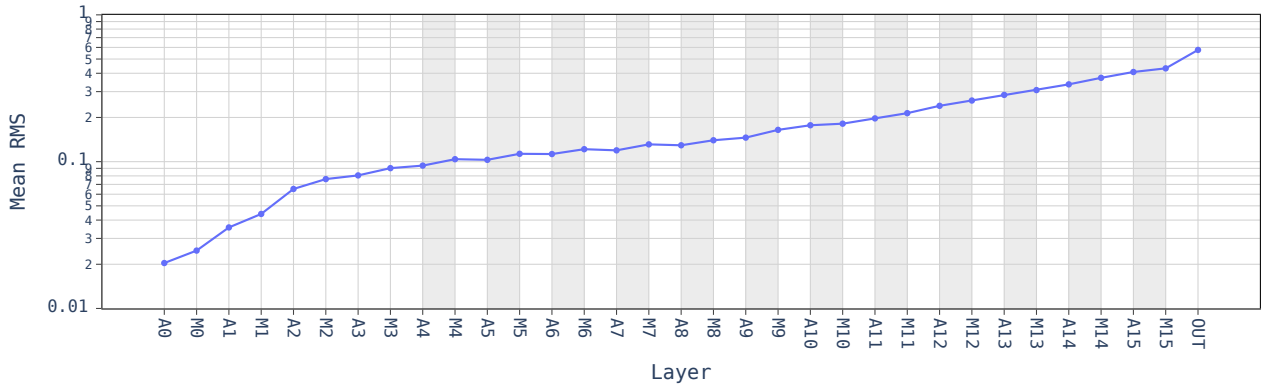
Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

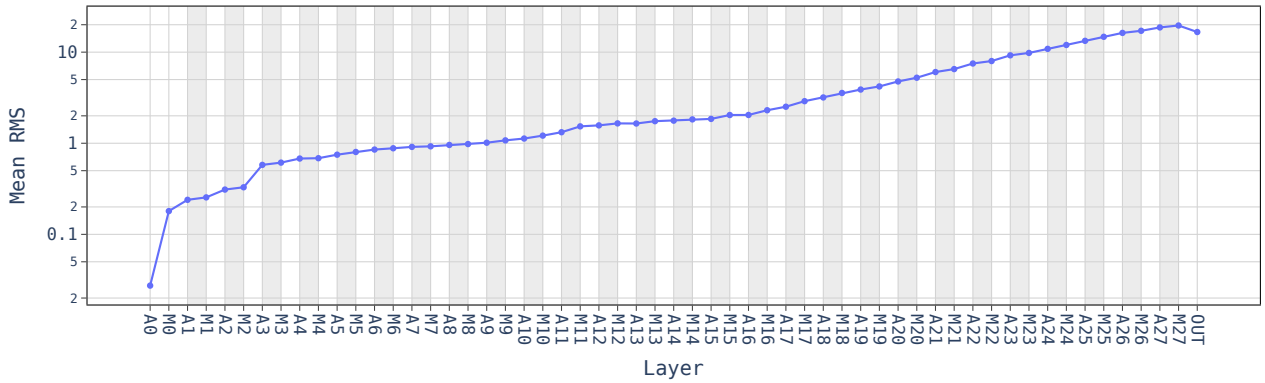
Zhang, B. and Sennrich, R. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Zhu, J., Chen, X., He, K., LeCun, Y., and Liu, Z. Transformers without normalization. In *Proceedings of the computer vision and pattern recognition conference*, pp. 14901–14911, 2025.

A. Layer-wise RMS Statistics



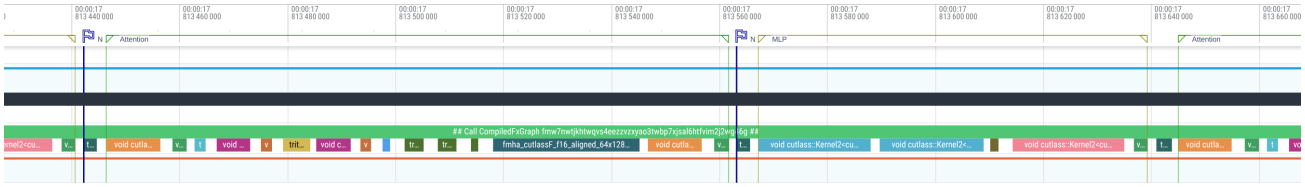
(a) Llama 3.2 1B



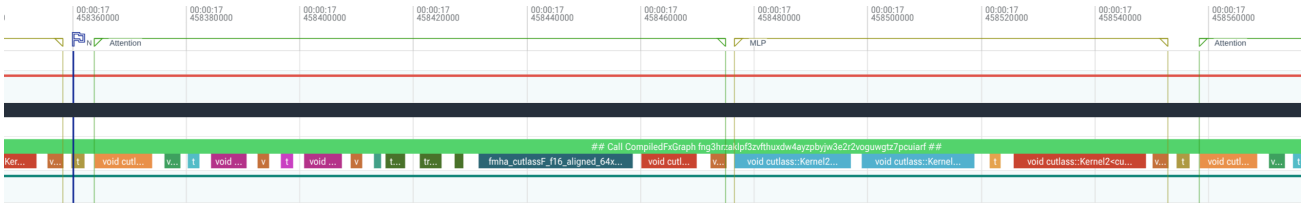
(b) Qwen 3 0.6B

Figure 3. Mean RMS values of hidden states per layer (log scale), measured prior to Attention (A) and MLP (M) sub-layers. Shaded regions indicate where CoupledNorm shares statistics. While RMS values increase sharply in early layers (first four for Llama, first for Qwen), the relative difference stabilizes to approximately 10% across the remainder of the network.

B. Profiler Traces



(a) Baseline



(b) CoupledNorm

Figure 4. Profiler traces comparing the baseline (a) and CoupledNorm (b) architectures for the Qwen 1.7B model at a batch size of 8, with all normalization kernels marked by blue flags. The baseline trace shows an additional normalization kernel compared to the CoupledNorm trace. Eliminating this kernel launch overhead is the primary source of the observed inference speedups. In CoupledNorm, the remaining scaling operations are efficiently fused into adjacent kernels: residual addition occurs via an `addmm` in the attention output projection (red), affine weights are statically folded into the MLP up-projection (blue), and RMS scaling is applied within the subsequent SiLU kernel (yellow).