Differential Attention at Small Scale: A Paired, Cross-Stack Reproduction with No Generalization Benefit

Guy J. Grigsby
guy@aeryx.ai

November 21, 2026

Abstract

The Differential Transformer reports consistent gains over standard attention at 3B parameters and 1T tokens. We test whether the mechanism helps at the scale a single laptop can reach. We reimplement differential attention in MLX on Apple Silicon with custom Metal kernels, verify it against the official PyTorch reference at 10^{-7} and replicate the training run in PyTorch on NVIDIA CUDA to rule out framework artifacts. Every comparison is paired. Both variants start from byte-identical shared weights and see identical data order, so a single-run delta is meaningful. At 30M parameters and 100M tokens, four rerun seeds put the paired val-loss delta at -0.016 $\pm$ 0.023 nats with a sign flip across seeds, and the originally recorded single-seed win sits inside that band. A single-seed win of the size we first observed is seed noise. At 162M parameters and 2.0B tokens, differential attention wins train loss by -0.111 nats yet loses held-out validation by +0.035 nats, with a late-run overfitting signature. Binning held-out loss by token position shows vanilla uniformly ahead across the whole 2048-token window, so the long-context advantage the mechanism is built for does not appear. These results sit three orders of magnitude below the paper's setting and refute nothing about it. They show no benefit in the small-scale, short-context regime. All code, kernels, training logs and checkpoints are released.

1 Introduction

Differential attention computes two softmax attention maps per head and subtracts one from the other, scaled by a learned λ. The subtraction cancels common-mode attention noise. The original paper [6] reports consistent wins at 3B parameters and 1T tokens, with the largest gains on long-context and retrieval tasks. Training at that scale requires a cluster. Most practitioners who consider adopting the mechanism will first try it far smaller.

This paper asks a narrow question. Does differential attention help at the scale one machine can reach? The question needs care, because at small scale the noise floor is high and a single training run can show a spurious win. We control for this three ways. Every comparison is paired, with both variants starting from byte-identical shared weights and consuming identical data order. The implementation is verified against the official reference at 10^{-7}. The headline delta is replicated cross-stack, in a different framework on a different vendor's silicon.

The answer is no. A paired seed band at 30M parameters shows the delta is seed noise. At 162M parameters and 2.0B tokens differential attention memorizes the training stream harder and generalizes slightly worse, and its held-out loss shows no improvement at any token position. The mechanism's long-context pitch does not show up at short context and small scale. None

of this contradicts the paper's results at 3,000× the token budget. It bounds where the benefit begins, and the bound is above what a laptop reaches.

2 Method

2.1 Implementation

We implement a pre-norm decoder-only transformer in MLX [1] with rotary position embeddings [4], SwiGLU [3], RMSNorm [7] and tied embeddings, with standard multi-head attention [5] as the baseline. The differential variant follows the paper's canonical form. Diff heads pair adjacent attention heads with an interleaved split, λ vectors are stored fp32 with the paper's depth schedule and a per-head RMSNorm applies after the subtraction. The forward path uses the linearity identity $(A_1 - \lambda A_2)V = A_1V - \lambda A_2V$, so no $T \times T$ attention map is materialized.

The forward path runs on custom Metal kernels written through `mx.fast.metal_kernel`. A row-wise softmax kernel and a causal scaled-dot-product-attention kernel with separate query-key and value widths, so the doubled-V diff head runs in one dispatch. Autograd hooks delegate backward to MLX. Kernel outputs match MLX's own operators inside the bf16 ULP-noise band.

2.2 Verification

Correctness rests on three independent checks. First, the model output matches the vendored official PyTorch reference at 3.6×10^{-7} max absolute difference on the CPU stream. This check caught two real convention bugs during development. One was an interleaved head-pair split the internal oracle missed because the oracle shared the bug. The other was a RoPE convention mismatch. Second, the custom kernels are validated against MLX's built-in operators across the full shape matrix, forward and backward. Third, the entire Stage 0 experiment is replicated in PyTorch on an NVIDIA RTX 3070 Ti. Same algorithm, different framework, different silicon. The paired delta tracks across stacks, with the same crossover step and the same order of magnitude, so the observed deltas are not framework artifacts.

2.3 Paired initialization

Both variants are built from one seed and share every weight that has a counterpart, byte-identical. Data order is identical between variants. The paired delta from a single seed then reflects the architectural difference plus run noise, not initialization luck. The seed band in Section 4.1 measures how much run noise remains.

3 Experimental setup

Two scales, both on FineWeb-Edu [2] tokenized with `cl100k_base` at context length 2048. Stage 0 trains 30M parameters for 100M tokens. Stage 1 trains 162M parameters for 2.0B tokens with effective batch 32, peak learning rate 4×10^{-4}, and 1000-step warmup, in bf16 mixed precision. Training runs on one Apple M5 Max with MLX. The cross-stack replication runs on an NVIDIA RTX 3070 Ti in PyTorch. Held-out validation is a disjoint document split. The evaluation metric is val loss in nats, measured on the same tokens for both variants of a pair.

seed	vanilla	diff	δ
0 (recorded)	4.720	4.700	-0.020
1	4.724	4.702	-0.022
2	4.717	4.695	-0.022
3	4.710	4.727	+0.017
4	4.725	4.687	-0.038

Table 1: Stage 0 (30M params, 100M tokens) paired val loss by seed. Seed 0 is the original run on the first data build, recorded before the band was run. Seeds 1 to 4 rerun the full paired experiment on a rebuilt tokenization of the same corpus. Generated from committed run logs.

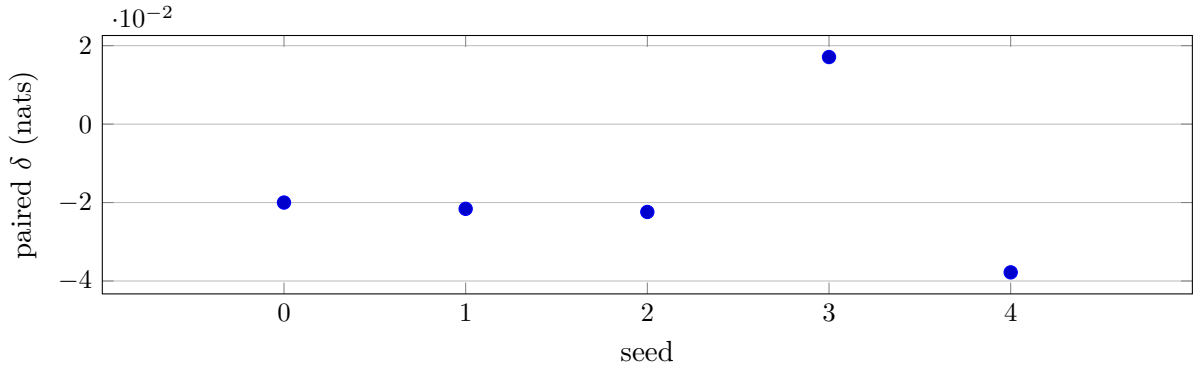


Figure 1: The paired delta by seed. Negative favors differential attention. The recorded single-seed win (seed 0) sits inside the band of the four reruns, which includes a sign flip. Generated from the same evidence as Table 1.

Hardware constraints are part of the study. Appendix A reports the throughput, memory, thermal and power-delivery behavior of sustained training on a laptop, including two throttling modes that do not announce themselves.

4 Results

4.1 Stage 0: the paired delta is seed noise

Our first Stage 0 run showed diff ahead by -0.020 nats on held-out loss, matching the paper's direction. Rerunning the paired experiment at four more seeds dissolves the result. Table 1 and Figure 1 show the band. The mean delta across the four reruns is -0.016 nats with sample standard deviation 0.023, spanning -0.038 to +0.017, and one seed flips sign. A single-seed delta of ± 0.02 nats at this scale carries no information about the architecture. This is the paper's central methodological point. Small-scale architecture comparisons need either paired seeds in a band or an effect larger than the band, and ours was neither.

4.2 Stage 1: train-loss win, held-out loss

At 162M parameters and 2.0B tokens the two loss signals disagree, and the disagreement is the finding. Differential attention finishes -0.111 nats ahead on train loss and +0.035 nats behind

metric	diff	vanilla	δ
final train loss (last 1000-step mean)	3.041	3.153	-0.111
held-out val loss at step 30000	3.362	3.326	+0.035

Table 2: Stage 1 (162M params, 2.0B tokens) endpoints, seed 0, paired init, identical data order. Positive δ favors vanilla. Generated from the released training logs.

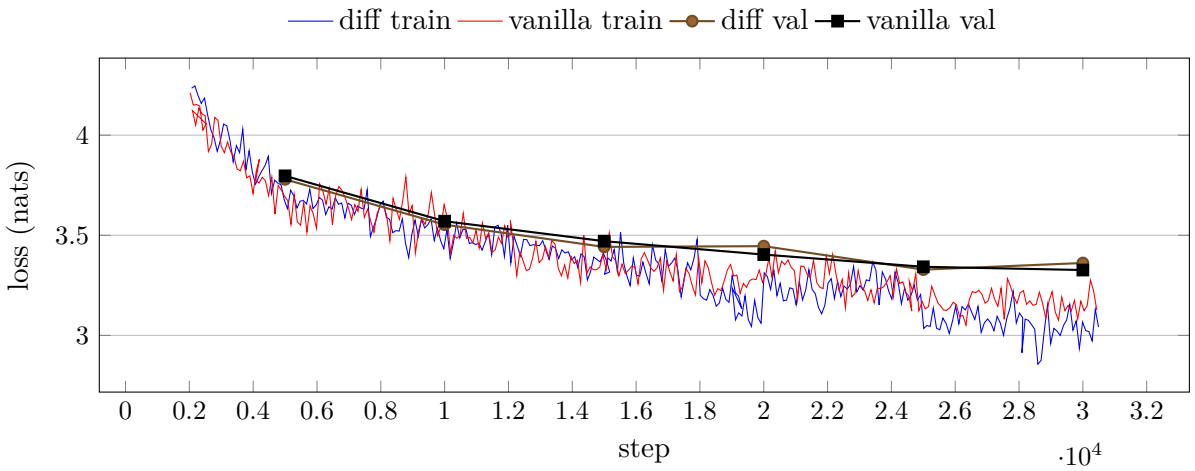


Figure 2: Stage 1 train curves (thin, downsampled, from step 2000) and held-out val points (marked). Diff holds the lower train loss throughout while vanilla takes the lower val loss from mid-run on. Generated from the released training logs.

on held-out validation (Table 2, Figure 2). The late run gives it away. Over the final 5000 steps diff's val loss rises from 3.328 to 3.362 nats while vanilla's falls from 3.343 to 3.326. Diff is memorizing the training stream harder, not learning a better function.

4.3 No positional structure in the delta

Differential attention's pitch is that cancelling common-mode attention noise matters most for long or noisy context, so its advantage should grow with token position. We bin held-out negative log-likelihood by position across the 2048-token window on both final checkpoints (Table 3, Figure 3). Vanilla is ahead in every bin. The delta is flat, spanning +0.035 to +0.045 nats with +0.036 in the last bin, and the overall gap of +0.038 nats matches the training-time val gap measured on a different token budget. No widening with position, no crossover at long range. The mechanism's designed advantage does not appear in this regime.

position	diff	vanilla	δ
0–255	3.574	3.529	+0.045
256–511	3.380	3.341	+0.039
512–767	3.339	3.304	+0.035
768–1023	3.335	3.296	+0.039
1024–1279	3.323	3.287	+0.036
1280–1535	3.324	3.288	+0.035
1536–1791	3.333	3.297	+0.035
1792–2047	3.337	3.301	+0.036
overall	3.368	3.330	+0.038

Table 3: Held-out NLL by token position, both Stage 1 final checkpoints. Positive δ favors vanilla. Generated from the committed per-position evaluation.

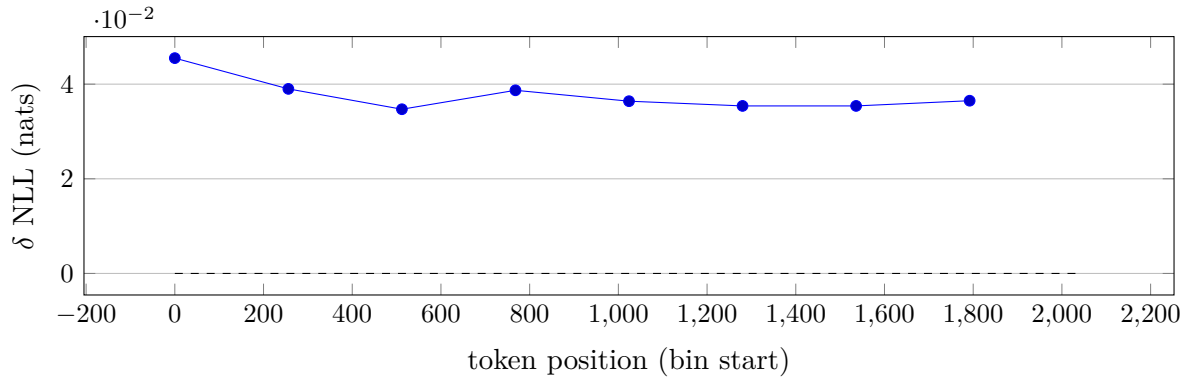


Figure 3: The positional delta is flat. If common-mode cancellation helped long context here, the curve would fall with position. Generated from the same evidence as Table 3.

5 Limitations

Scale. This study sits three orders of magnitude below the original paper's 3B-parameter, 1T-token setting. Nothing here refutes the paper's results in its own regime. The claim is only that the benefit is absent in ours.

Context and evals. The window is 2048 tokens and the metric is language model loss. The paper's strongest results are long-context and retrieval evals we do not run. The position-binned analysis is a proxy for the long-context claim, not a replacement.

Seeds. The seed band is measured at Stage 0, where a run costs an hour. Stage 1 remains a single seed. Its noise floor should be lower with twenty times the tokens, and the overfitting signature is a mechanism rather than a point estimate, but the Stage 1 delta does not carry a measured band.

Data build. The four band seeds ran on a rebuilt tokenization with a smaller held-out split than the original build. The band is internally consistent. The recorded seed 0 delta crosses builds and carries that caveat.

One machine, author-run. Training, evaluation and analysis are all ours, on hardware whose thermal and power behavior varied across runs (Appendix A). Loss comparisons are unaffected. Wall clock and throughput numbers are descriptive, not controlled.

6 Conclusion

At laptop scale, differential attention gave us nothing. A paired seed band shows our initial small-scale win was seed noise, the scaled-up run trades generalization for memorization, and the positional profile shows no trace of the long-context advantage the mechanism is designed for. The negative is bounded and honest. The mechanism's reported gains live at a scale this study does not reach, and anyone evaluating it below that scale should expect what we measured, a delta indistinguishable from noise. The methodological contribution stands apart from the negative. Byte-identical paired initialization, a reference cross-check strict enough to catch convention bugs and a cross-stack replication make single-machine architecture comparisons meaningful. Without them, our seed 0 result would have been reported as a win.

7 Availability

Code (MIT), Metal kernels, the PyTorch cross-stack reference, training logs, evaluation scripts and both Stage 1 checkpoints are at `https://github.com/guygrigsby/diff-mlx` and `https://huggingface.co/guygrigsby/diff-mlx`.

A Training on Apple Silicon

Numbers a reader planning similar work will want. Sustained bf16 training of the 162M model on an M5 Max 14 inch runs about 14k tokens per second at context 2048, roughly 5 to 10 percent of nominal bf16 peak. The workload is dispatch-bound, not compute-bound, which is what the custom kernels target.

Per-token cost is flat in micro-batch until unified memory runs out, then falls off a cliff. A micro-batch of 32 at Stage 1 exceeded the 128 GB ceiling and swap-thrashed to 950 tokens per

second. Micro-batch 8 with gradient accumulation restored 14k. When a run is slow, check swap before checking math.

Two throttles do not announce themselves. Under the stock fan curve a closed laptop thermal-throttles within ten minutes of sustained training. Forced fans and cross-flow hold throughput flat for days. Separately, mid-run throughput dropped about ten percent while the chip sat cool at 73°C, with the battery slowly draining under a reported charging state. The cause was power delivery. The machine was charging through a Thunderbolt dock that also drove a monitor, and switching to the direct first-party brick restored clocks and throughput. The 14-inch machine negotiates at most 100 W regardless of the adapter (its fast-charge spec is 96 W and it does not use the higher EPR profile), so sustained training plus charging leaves little headroom, and a dock delivering less than its rated upstream power pushes the shortfall onto the battery and the GPU clocks. A low temperature does not rule out throttling. Confirm the negotiated adapter wattage before benchmarking, and state cooling and power conditions with any published throughput number.

References

[1] Awni Hannun, Jagrit Digani, Angelos Katharopoulos, and Ronan Collobert. MLX: Efficient and flexible machine learning on apple silicon, 2023.

[2] Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems 37 (NeurIPS), Datasets and Benchmarks Track*, 2024. arXiv:2406.17557.

[3] Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[4] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. arXiv:2104.09864.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, 2017. arXiv:1706.03762.

[6] Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.05258.

[7] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, 2019. arXiv:1910.07467.