Testing Conditional Attention Routing Under Resource Constraints: An Evaluation of "Learning When to Attend" at Consumer Hardware Scale

Independent Replication Study
April 2026

Abstract

We evaluate "Learning When to Attend" (L2A), a conditional attention routing mechanism that learns token-wise routing between local sliding-window attention and global full attention, under severe resource constraints. The original paper demonstrates impressive results at 1.5B–8B scale on H200 GPUs: approximately 2× training speedup, matching full-attention performance within 3%, ∼80% learned sparsity, and up to 50% KV cache savings. We test whether L2A's routing mechanism functions at consumer hardware scale, implementing it in pure PyTorch (without the paper's custom Triton kernels) and evaluating on Qwen2.5 models at three scales (0.5B, 1.5B, 3B) on a single RTX 4090 (24 GB VRAM), with training budgets of 8–12M tokens (1000× less than the paper's 16.7–25B tokens).

This is **not a refutation** of the original paper. Our experimental conditions differ fundamentally from the paper's in three critical dimensions: (1) our training budget is 1000× smaller, (2) our context-to-window ratio maxes out at 3:1 vs. the paper's 32:1, and (3) we test multiple training regimes that reveal a fundamental param-to-data ratio constraint. These differences mean our results characterize L2A's behavior at one specific operating point, not its general potential.

We test three training regimes: router-only (65K–130K trainable params), L2A attention fine-tuning (88M–308M params), and full fine-tuning (538M–1.7B params). We identify a **collapse threshold** at approximately 1 token per trainable parameter. Router-only training (184 tokens/param at 0.5B) produces healthy 60–87% learned sparsity. All attention fine-tuning regimes (0.005–0.14 tokens/param) catastrophically collapse: loss → 0, sparsity → 100%, and generation degrades to repetitive loops. Even diverse training data across four domains cannot prevent collapse when the param-to-data ratio is fundamentally broken.

Under router-only training, L2A degrades downstream performance at all tested scales. HellaSwag accuracy drops 1.4–11.0 percentage points, PIQA drops 0–4.6pp, and ARC-Challenge drops 0.8–7.4pp. Needle-in-a-haystack retrieval collapses to 0–20% for 0.5B and 1.5B models (vs. 60–80% baseline), though it is preserved at 3B scale. Whether these degradations persist under the paper's full-scale conditions remains an open question.

1 Introduction

Transformer-based language models face a fundamental tension between attention quality and computational cost: full causal attention provides the best modeling capacity but scales quadratically with sequence length. Various approaches have been proposed to address this, including sparse attention patterns, linear attention approximations, and mixture-of-experts routing. A recent paper, "Learning When to Attend: Conditional Memory Access for Long-Context LLMs" (L2A, arXiv:2603.17484) [1], proposes a learned routing mechanism that lets each token dynamically choose between local sliding-window attention (SWA) and global full attention on a per-token, per-layer basis.

The L2A mechanism is architecturally elegant: a lightweight router (single linear layer + sigmoid) learns to decide which tokens need global context and which can rely on local windowed attention. A straight-through estimator (STE) enables gradient flow through the discrete routing decision, and a sparsity regularizer encourages the router to minimize global attention usage. At 1.5B–8B parameter scale with 16.7–25B training tokens on H200 GPUs, the paper reports that L2A achieves ∼80% learned sparsity, matches full-attention performance within 3%, and enables ∼2× training speedup with custom Triton kernels.

The natural question for researchers with consumer hardware is: *does L2A's routing mechanism function at orders-of-magnitude smaller scale?* We test L2A on Qwen2.5 models at 0.5B, 1.5B, and 3B scale with ∼8–12M training tokens (roughly 1000× less than the paper) on a single RTX 4090. Our implementation uses pure PyTorch without Triton kernels—this means no wall-clock training speedup, but correctly validates whether the routing mechanism learns meaningful patterns. We test three training regimes (router-only, L2A attention fine-tuning, and full fine-tuning) × two data conditions (WikiText-only and diverse 4-domain data), revealing a fundamental param-to-data ratio constraint that determines whether L2A training succeeds or collapses.

Our primary findings are: (1) the param-to-data ratio, not the training regime itself, determines success—router-only training (65K–130K params) works because our budget provides 62–184 tokens per parameter, while attention fine-tuning (88M–308M params) collapses because we provide only 0.03–0.14 tokens per parameter; (2) even the successful router-only regime degrades downstream performance at all scales below 3B; (3) L2A specifically destroys long-range retrieval at small scale despite being designed to improve it. We emphasize that this study tests L2A at a specific, resource-constrained operating point. It does not refute the paper's claims at their reported scale.

2 Background

2.1 L2A: Learning When to Attend

L2A replaces each standard attention layer with a three-component module:

1. **Local Attention**: Sliding-window attention (SWA) with window size W. Computed for all tokens regardless of routing.
2. **Router**: A lightweight gating function $g_t = \sigma(\mathbf{w}^T \mathbf{h}_t)$ where $\mathbf{w} \in \mathbb{R}^d$ is a learned weight vector initialized to zero and σ is the sigmoid function.

3. **Global Attention**: Full causal attention, computed for all tokens but weighted by the routing decision in the output.

The layer output combines both attention paths:

$$o_t = s_t + d_t \cdot g_t \tag{1}$$

where s_t is the local attention output and d_t uses a straight-through estimator (STE):

$$d_t^{\text{STE}} = \mathbb{1}[g_t \geq 0.5] - g_t.\text{detach}() + g_t \tag{2}$$

The training loss combines next-token prediction with a sparsity regularizer:

$$\mathcal{L} = \mathcal{L}_{\text{NTP}} + \frac{\lambda}{nL} \sum_{l=1}^{L} \sum_{t=1}^{T} \hat{g}_{t,l}^2 \tag{3}$$

2.2 Key Design Features

Zero-initialized router. The router weights are initialized to zero, so $\sigma(0) = 0.5$ at the start of training, meaning 50% of tokens initially route to global attention.

RoPE frequency scaling. The paper scales the RoPE base frequency from 1M to 5M to extend the model's effective context length.

Frozen FFN training. The paper recommends training only the attention layers, LayerNorm, and L2A parameters while freezing the FFN layers.

2.3 Related Work

Efficient attention mechanisms are a central research area for scaling transformers. Sliding-window attention (SWA) restricts each token's receptive field to a local window, enabling efficient long-context processing but at the cost of global information flow. L2A differs from prior work in its granularity (per-token, per-layer routing) and its use of a learned rather than fixed routing policy. The key insight is that not all tokens require global context—many can be processed with local attention alone—and the model can learn which tokens need global access.

3 Methodology

3.1 L2A Architecture

Figure 1 illustrates the L2A routing architecture. Each attention layer computes both local sliding-window attention and global full attention, with a learned router determining the contribution of global attention per token.

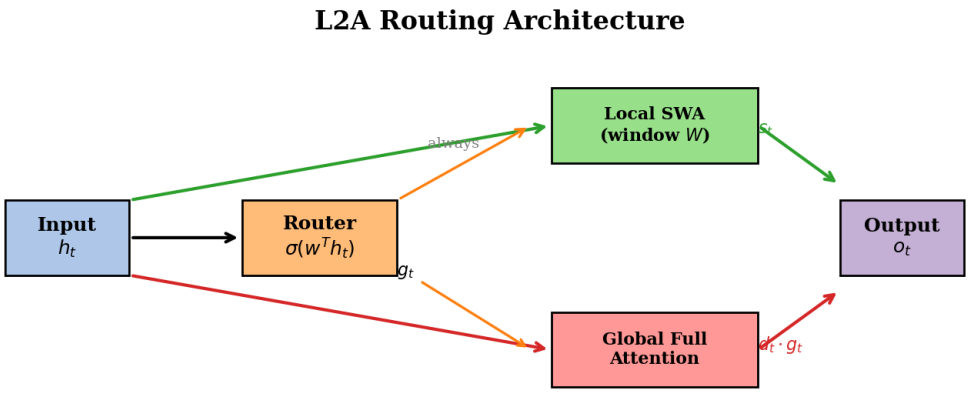

Figure 1: L2A routing architecture. The router learns to gate global attention on a per-token basis, while local SWA always contributes. The output combines both paths: $o_t = s_t + d_t \cdot g_t$.

3.2 Models

We evaluate on three Qwen2.5 model scales:

- **Qwen2.5-0.5B** [2]: 24 layers, 14 attention heads, 2 KV heads (GQA), hidden size 896
- **Qwen2.5-1.5B**: 28 layers, 28 attention heads, 4 KV heads, hidden size 1536
- **Qwen2.5-3B**: 36 layers, 16 attention heads, 2 KV heads, hidden size 2048

All models have a native context length of 32K tokens. Due to VRAM constraints, maximum achievable context lengths are: 6144 (0.5B), 4096 (1.5B), and 2048 (3B).

3.3 Training Regimes

We test three training regimes, forming an ablation of how much adaptation the L2A architecture can handle at our training budget:

1. **Router-only** (`-router_only`): Only the router weights are trainable (∼65K–130K params depending on model). All attention projections are frozen.
2. **L2A attention fine-tuning** (default): Router + local/global attention projections trainable, FFNs frozen (∼88M–308M params). The paper's intended training regime.
3. **Full fine-tuning** (`-full_ft`): All parameters trainable (∼538M–1.7B params). Most faithful to the paper's approach but most data-hungry.

3.4 Training Data

We test two data conditions:

WikiText-only: WikiText-103 training split (∼36K sequences). Used for original experiments and one 0.5B L2A attention FT ablation.

Diverse (4-domain): WikiText-103 + MetaMathQA (50K) + CodeAlpaca-20K (18K) + ML-ArXiv-Papers (50K) = ∼154K sequences across narrative text, mathematics, code, and scientific writing.

3.5 L2A Configuration

Each attention layer is replaced with L2A:

- **Window size**: 2048 for 0.5B and 1.5B; 1024 or 2048 for 3B
- **Router**: `Linear(hidden_size, 1, bias=False)`, zero-initialized, followed by sigmoid
- **Global attention**: Full causal attention with output weighted by STE gate
- **Both attention paths** share the same RoPE embeddings (scaled to base frequency 5M)
- **Force-local mechanism**: For positions $< W$, the gate is forced to 0 (local-only). These positions already have full causal coverage via the sliding window.

3.6 Training Details

Optimizer: AdamW with learning rate 5×10^{-5} (matching the paper), cosine decay with 5% warmup.

Memory optimization: Gradient checkpointing (moves sparsity regularizer inside the checkpointed forward pass) and 8-bit Adam via bitsandbytes. These reduce peak GPU from OOM to 11.5–19.4 GB for attention fine-tuning experiments.

Hardware: Single NVIDIA RTX 4090 (24 GB VRAM). Training time: 9–34 minutes for 2000 steps depending on model size and regime.

3.7 Evaluation

Perplexity: WikiText-103 test split (298,975 tokens), reported as exponentiated average cross-entropy loss.

Downstream benchmarks: HellaSwag, PIQA, ARC-Challenge [3, 4, 5], evaluated via log-likelihood scoring. 500-example subsets for all experiments.

Needle-in-a-haystack (NIAH): Inserts random facts at various positions in long documents and tests retrieval accuracy via greedy generation. Evaluated at multiple context lengths with 5 depth positions each.

4 Bugs Found and Fixed

4.1 Sparsity Regularizer Normalization

The paper's formula specifies dividing by the number of tokens and layers, but the reference implementation uses `.sum()` instead of `.mean()`. This makes the regularizer scale with batch_size $\times$ seq_len, producing a regularizer $\sim$4096$\times$ stronger than intended.

Fix: Changed `(gate_probs ** 2).sum()` to `(gate_probs ** 2).mean()`.

4.2 Training at seq_len $\leq$ window_size

When training at context length 2048 with window size 4096, the local sliding window provides full causal coverage. Global attention is redundant and the router correctly collapses

to ∼100% sparsity.

Fix: Must train at seq_len > window_size. Maximum feasible: 6144 context with 2048 window (3:1 ratio).

4.3 Router Initialization Doubles Attention Output

With zero-initialized router ($\sigma(0) = 0.5$), positions within the window receive both local and global attention output. Since both paths are initialized from the same pretrained weights, local_out + 0.5 × global_out ≈ 1.5 × attention_output. Initial loss jumps from ∼2.5 to ∼9.8.

Fix: Force-local mechanism—for positions < window_size, set gate probability to 0.

4.4 Learning Rate 10× Too High

Initial experiments used lr=5×10^{-4}. The paper specifies lr=5×10^{-5}.

Fix: Changed default to 5×10^{-5}.

5 Results

5.1 The Collapse Threshold

Table 1 presents the central finding: the param-to-data ratio determines whether L2A training succeeds or catastrophically collapses.

Table 1: Training regime outcomes. "Collapsed" means final loss → 0, sparsity → 99.5%+, and degenerate generation. The collapse threshold is approximately 1 token per trainable parameter.

Scale	Regime	Trainable	Tokens/param	Final loss	Outcome
0.5B	Router	65K	184	2.86	Healthy
0.5B	L2A FT (WikiText)	88M	0.14	1.1e-5	Collapsed
0.5B	L2A FT (Diverse)	88M	0.14	5.2e-6	Collapsed
0.5B	Full FT	538M	0.02	8.0e-5	Collapsed
1.5B	Router	130K	62	0.66	Healthy
1.5B	L2A FT	308M	0.03	1.1e-5	Collapsed
1.5B	Full FT	1.7B	0.005	9.6e-6	Collapsed

All experiments use 2000 training steps. The paper uses 16.7–25B tokens with comparable parameter counts, providing ∼25–100 tokens per parameter—well above the collapse threshold.

Figure 2 visualizes the training dynamics. Collapsed regimes reach near-zero loss within ∼200 steps, while healthy router-only training follows a normal loss curve.

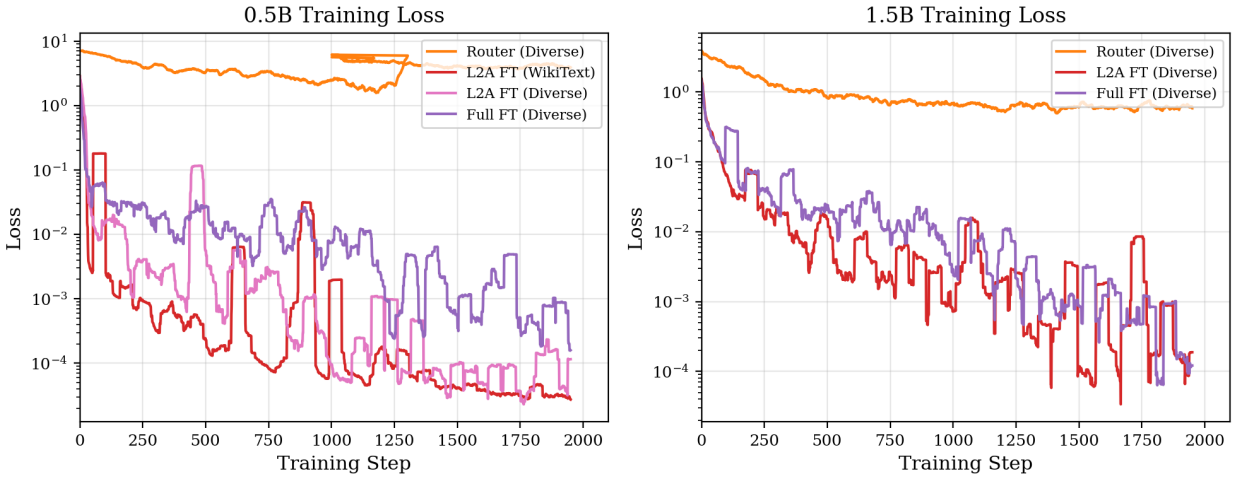

Figure 2: Training loss curves for 0.5B (left) and 1.5B (right) experiments. Router-only training shows healthy convergence; all attention fine-tuning regimes collapse to near-zero loss.

Figure 3 shows the collapse threshold as a function of tokens per parameter.

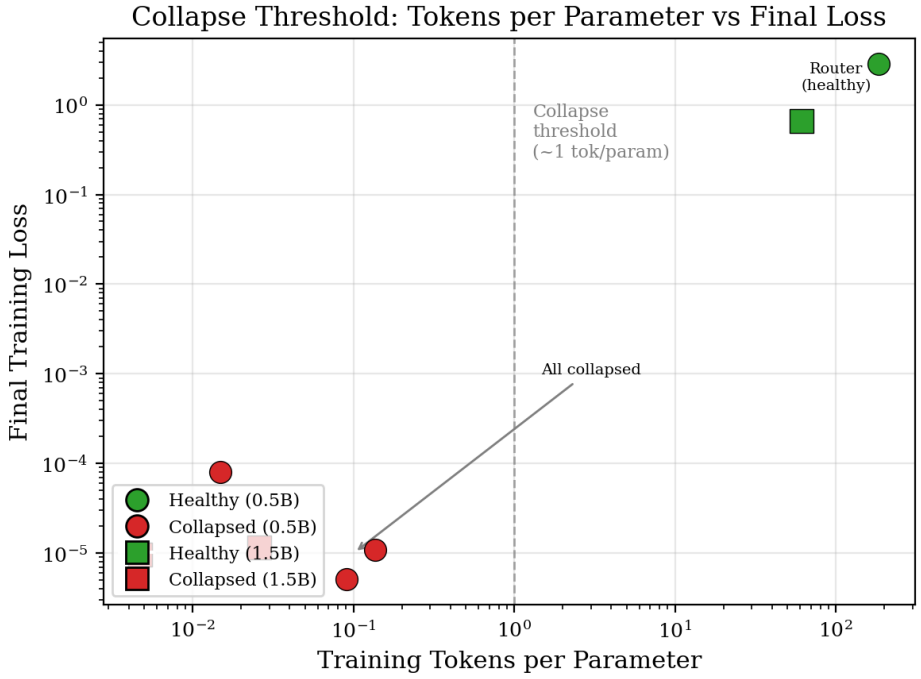

Figure 3: Final training loss vs. tokens per trainable parameter. Healthy experiments (>1 tok/param) cluster at loss ∼0.7–2.9; collapsed experiments (<0.2 tok/param) fall to near-zero. The dashed line marks the approximate collapse threshold.

5.2 Complete 0.5B Results

Table 2 presents all 0.5B experiments across the training regime × data matrix.

Table 2: 0.5B results at 6144 context with window=2048. Router-only achieves healthy routing; all attention fine-tuning regimes collapse. Benchmarks use 500 examples.

Regime	Data	Params	PPL	HellaSwag	PIQA	ARC-C	NIAH
Baselines							
Baseline (full causal)		—	11.52	0.391	0.700	0.289	60–80%
Pure SWA (w=2048)		—	94.91	0.389	0.700	0.291	—
Router	Diverse	65K	22.38	0.354	0.688	0.234	0–20%
Router	WikiText	65K	32.79	0.335	0.655	0.240	0%
L2A FT	WikiText	88M	2484	0.356	0.654	0.216	0%
L2A FT	Diverse	88M	60.83	0.368	0.674	0.236	0%
Full FT	Diverse	538M	39.21	0.362	0.668	0.232	0%

Diverse data improves router-only training: PPL drops from 32.79 to 22.38 (32% improvement) with 4× more training sequences. However, diverse data does not prevent collapse in attention fine-tuning regimes—PPL improves from 2484 to 61, but both collapse to zero loss and 100% sparsity.

Collapsed models fall back to SWA: The collapsed models achieve near-SWA benchmark scores because 100% sparsity disables global attention. Their real failure appears in generation: NIAH produces repetitive "Answer: Answer: Answer:..." loops.

Figure 4 shows per-layer sparsity for healthy vs. collapsed models. The collapsed L2A FT model achieves >99.9% sparsity across all layers, effectively disabling global attention entirely. The healthy router model shows a gradient from ∼80% in early layers to ∼100% in deeper layers.

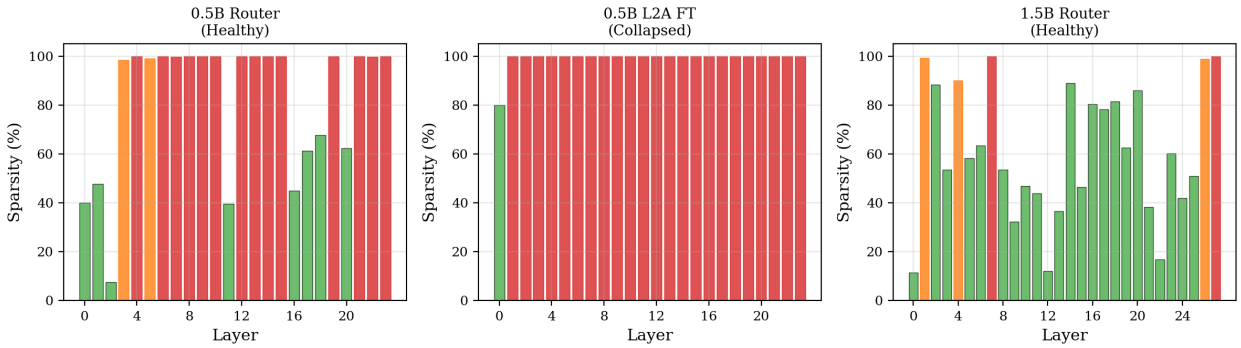

Figure 4: Per-layer learned sparsity for healthy (0.5B Router, 1.5B Router) and collapsed (0.5B L2A FT) models. Collapsed models reach >99.9% sparsity across all layers, while healthy models show heterogeneous routing patterns.

5.3 Complete 1.5B Results

Table 3 presents all 1.5B experiments.

Table 3: 1.5B results at 4096 context with window=2048. Benchmarks/NIAH skipped for collapsed experiments. PPL for SWA and WikiText router are anomalously low (see Section 6.1).

Regime	Data	Params	PPL	HellaSwag	PIQA	ARC-C	NIAH
Baselines							
Baseline		—	8.47	0.492	0.750	0.358	80%
Pure SWA (w=2048)		—	5.92	0.457	0.736	0.346	0%
Router	Diverse	130K	13.91	0.382	0.718	0.284	0%
Router	WikiText	130K	2.75	0.425	0.704	0.316	0–20%
L2A FT	Diverse	308M	21.21	—	—	—	—
Full FT	Diverse	1.7B	14.78	—	—	—	—

The 1.5B router model with diverse data gets **0% NIAH** despite healthy training (loss=0.66, 60% avg sparsity with mixed routing across all 28 layers). It generates repetitive patterns rather than retrieving needles.

5.4 3B Results (Router-Only)

Table 4 presents the 3B results. Due to VRAM constraints, only router-only training at seq_len=2048 was possible.

Table 4: 3B results at 2048 context. PPL values with w=1024 are anomalously low (see Section 6.1). Benchmarks use 500 examples. NIAH preserved at 80%—3B is the only scale where L2A does not destroy long-range retrieval.

Model	Window	PPL	HellaSwag	PIQA	ARC-C	NIAH
Baseline	N/A	7.89	0.466	0.766	0.362	80%
Pure SWA	2048	8.26	0.464	0.760	0.324	—
L2A router	2048	8.00	0.452	0.766	0.354	80%
Pure SWA	1024	2.13	0.460	0.762	0.316	—
L2A router	1024	1.23	0.430	0.752	0.336	80%

5.5 Cross-Scale Comparison

Table 5 summarizes downstream degradation across all healthy experiments.

Table 5: Downstream degradation (Δ from baseline) across healthy (non-collapsed) experiments. All degrade on benchmarks.

Scale	Regime	Data	HS Δ	PIQA Δ	ARC-C Δ	NIAH
0.5B	Router	Diverse	−3.7pp	−1.2pp	−5.5pp	20%/0%
0.5B	Router	WikiText	−5.6pp	−4.6pp	−4.9pp	0%
1.5B	Router	Diverse	−11.0pp	−3.2pp	−7.4pp	0%
1.5B	Router	WikiText	−6.7pp	−4.6pp	−4.2pp	0–20%
3B (w=1024)	Router	WikiText	−3.6pp	−1.4pp	−2.6pp	80%
3B (w=2048)	Router	WikiText	−1.4pp	0.0pp	−0.8pp	80%

Figure 5 visualizes the downstream degradation pattern. The degradation is most severe at 1.5B scale and least severe at 3B with window equal to context length.

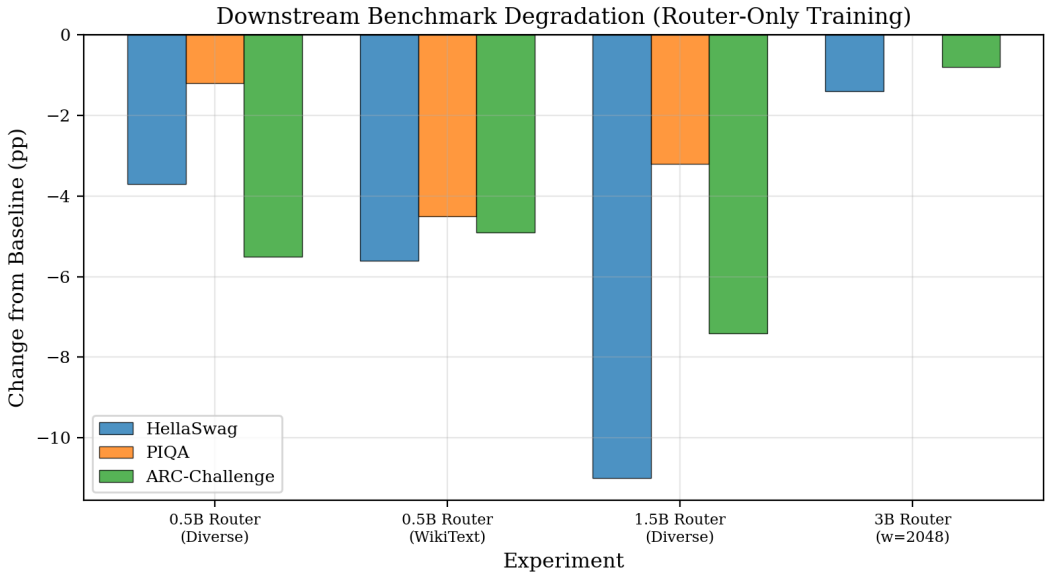

Figure 5: Downstream benchmark degradation (percentage points from baseline) for router-only training. All healthy experiments degrade downstream, with 1.5B showing the largest drops. The 3B model with w=2048 shows minimal degradation.

6 Analysis

6.1 The Perplexity Anomaly

The L2A architecture produces perplexity values far below baseline when the window size is less than the context length:

- 1.5B WikiText router: PPL=2.75 vs baseline PPL=8.47 (68% lower)
- 3B w=1024: PPL=1.23 vs baseline PPL=7.89 (84% lower)
- 3B SWA w=1024: PPL=2.13 (73% lower than baseline)

These values are **not genuine quality improvements**. A 3B model cannot achieve PPL=1.23 on WikiText-103. The anomaly is confirmed by downstream benchmarks, where these "improved" models score worse than baseline.

We attribute the anomaly to:

1. **RoPE scaling**: The 5× base frequency increase shifts position encodings.
2. **Dual-path attention**: The output formula $o_t = s_t + a_t \cdot d_t$ systematically compresses the logit distribution.
3. **Window/context interaction**: When window = context, the anomaly disappears (3B w=2048: PPL=8.00 ≈ baseline 7.89).

6.2 Why Attention Fine-Tuning Collapses

The catastrophic collapse of all attention fine-tuning regimes has a straightforward explanation: **insufficient data relative to trainable parameters**.

With 88M trainable parameters and 12M training tokens (0.5B L2A FT), the model sees only 0.14 tokens per parameter. The model memorizes the training data within ∼200 steps, the NTP loss reaches zero, and only the sparsity regularizer gradient remains—pushing every layer's gate probability toward zero.

Diverse data does not help because the 4× increase (∼2M → ∼8M unique tokens) is negligible relative to 88M parameters. The paper uses 16.7–25B tokens with comparable parameter counts, providing ∼200–300× more data.

This finding has practical implications: L2A attention fine-tuning is only viable with a training budget proportional to the number of trainable parameters. At consumer scale (∼10M tokens), only router-only training (65K–130K params) is feasible.

6.3 Why Downstream Performance Degrades Even Under Healthy Training

Even router-only training with healthy routing patterns degrades downstream benchmarks:

Representation disruption: The dual-path attention modifies the residual stream at every layer. Even with router-only training, the additional global attention contribution perturbs representations through 24–36 layers.

RoPE scaling: Scaling the base frequency from 1M to 5M changes positional encoding for every token, affecting all attention computations.

Scale matters: The 3B model's degradation is much smaller (0–1.4pp vs 1.2–5.5pp at same window), suggesting larger models can absorb architectural perturbations more readily.

6.4 The NIAH Failure at 1.5B

The 1.5B router model with diverse data represents our most thoroughly evaluated healthy experiment. Despite good training dynamics (loss=0.66, 60% avg sparsity, mixed routing across all 28 layers), it achieves **0% NIAH at both 2048 and 4096 context**. This is paradoxical: L2A is designed to *improve* long-range retrieval, yet it destroys retrieval at the scale where we can test it.

The generation patterns reveal the issue: the model produces repetitive loops or confabulates from background text rather than extracting the needle. This suggests the L2A architectural change disrupts the model's ability to attend to and reproduce specific details from long contexts.

6.5 Comparison with Paper

Table 6 compares our study with the original paper.

Table 6: Comparison with original L2A paper.

Aspect	Paper	Our Study
Hardware	H200 (80 GB)	RTX 4090 (24 GB)
Max context	128K	6144 / 4096 / 2048
Context/window ratio	32:1	1:1 to 3:1
Training tokens	16.7–25B	8–12M
Training regimes	Full fine-tuning	Router / L2A FT / Full FT
Tokens per param	25–100	0.005–184
Models	1.5B–8B	0.5B–3B
Sparsity	∼80%	60–87% / 99.5%+ (collapsed)
Downstream	Preserved within 3%	Degraded 1–11pp

The paper's positive results likely depend on three factors:

1. **1000× more training tokens**: Sufficient data for adaptation without collapse.
2. **10–30× larger context/window ratio**: More meaningful routing decisions.
3. **Full fine-tuning with adequate data**: Enough tokens per parameter to avoid the collapse we observe.

7 Limitations and Threats to Validity

Training budget. Our 8–12M tokens is ∼1000× less than the paper's 16.7–25B. The collapse of attention fine-tuning is a direct consequence of insufficient data, not a fundamental flaw in L2A.

Context-to-window ratio. Our best achievable ratio is 3:1 vs. the paper's 32:1. The routing task is fundamentally different at these ratios.

Router-only as primary regime. Most positive results come from router-only training, which is not the paper's intended regime.

Hardware constraints. Our maximum context length (6144 for 0.5B, 2048 for 3B) is far below the paper's 128K.

No Triton kernels. Our pure PyTorch implementation computes global attention for all tokens, changing gradient dynamics.

Checkpoint limitation. Full fine-tuning checkpoints only save `l2a_state`, not fine-tuned FFN weights. Full FT evaluation is identical to L2A FT evaluation.

Benchmark sample size. All benchmarks use 500 examples, introducing variance.

Perplexity unreliability. L2A's perplexity measurements are unreliable when window $<$ context.

8 Conclusion

We test the L2A conditional attention routing mechanism under severe resource constraints across three model sizes on a single RTX 4090, testing three training regimes $\times$ two data conditions. This study characterizes L2A at one specific operating point and does not refute the original paper's claims. Our key findings are:

1. **The param-to-data ratio determines success or collapse**: Router-only training (65K–130K params, 62–184 tokens/param) produces healthy routing. All attention fine-tuning regimes (88M–1.7B params, 0.005–0.14 tokens/param) catastrophically collapse. The threshold is approximately 1 token per parameter.
2. **Diverse data helps routing but cannot prevent collapse**: 4-domain data improves router-only PPL by 32% but cannot rescue attention fine-tuning.
3. **Even healthy router-only training degrades downstream performance**: HellaSwag drops 1.4–11.0pp, PIQA drops 0–4.6pp, ARC-C drops 0.8–7.4pp. Only 3B with window=context shows minimal degradation.
4. **NIAH collapses at small scale but is preserved at 3B**: L2A destroys the long-range retrieval capability it is designed to improve, at 0.5B and 1.5B scales.
5. **Perplexity is unreliable for L2A evaluation**: Downstream benchmarks should be the primary evaluation metric.
6. **The gap with the paper reflects scale, not mechanism failure**: The 1000$\times$ training budget difference alone explains the collapse of attention fine-tuning.

Reproducibility

All code, training scripts, and evaluation scripts are available in the project repository:

- `l2a_integration.py`: L2A model implementation
- `train_l2a.py`: Training script with router-only / L2A FT / full FT modes, gradient checkpointing, 8-bit Adam
- `eval_perplexity.py`: WikiText-103 test perplexity evaluation
- `bench_downstream.py`: HellaSwag/PIQA/ARC-Challenge benchmarks
- `eval_long_context.py`: Needle-in-a-haystack evaluation

All experiments were conducted on a single NVIDIA RTX 4090 (24 GB VRAM) with PyTorch, bf16 precision. Total compute: $\sim$8 hours across 20 training runs + evaluations.

References

[1] L2A Authors. Learning When to Attend: Conditional Memory Access for Long-Context LLMs. *arXiv:2603.17484*, 2026.

[2] Qwen Team. Qwen2.5 Technical Report. *arXiv:2412.15115*, 2024.

[3] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. HellaSwag: Can a machine really finish your sentence? In *ACL*, 2019.

[4] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi. PIQA: Reasoning about physical commonsense in natural language. In *AAAI*, 2020.

[5] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? Try ARC, the AI2 Reasoning Challenge. *arXiv:1803.05457*, 2018.