Threshold Differential Attention for Sink-Free, Ultra-Sparse, and Non-Dispersive Language Modeling

Xingyue Huang[1*†], **Xueying Ding**[2*†], **Mingxuan Ju**[3], **Yozen Liu**[3], **Neil Shah**[3], **Tong Zhao**[3]
[1]University of Oxford [2]Carnegie Mellon University [3]Snap Inc.

Abstract

Softmax attention struggles with long contexts due to structural limitations: the strict sum-to-one constraint forces attention sinks on irrelevant tokens, and probability mass disperses as sequence lengths increase. We tackle these problems with Threshold Differential Attention (TDA), a sink-free attention mechanism that achieves ultra-sparsity and improved robustness at longer sequence lengths without the computational overhead of projection methods or the performance degradation caused by noise accumulation of standard rectified attention. TDA applies row-wise extreme-value thresholding with a length-dependent gate, retaining only exceedances. Inspired by the differential transformer, TDA also subtracts an inhibitory view to enhance expressivity. Theoretically, we prove that TDA controls the expected number of spurious survivors per row to $O(1)$ and that consensus spurious matches across independent views vanish as context grows. Empirically, TDA produces $> 99\%$ exact zeros and eliminates attention sinks while maintaining competitive performance on standard and long-context benchmarks.

1 Introduction

The widespread success of large language models (LLMs) (Vaswani et al., 2017; Achiam et al., 2023; Touvron et al., 2023) has fueled growing interest in architectural innovations that can scale to increasingly long contexts (Beltagy et al., 2020; Zaheer et al., 2020; Dao et al., 2022; Xiao et al., 2023). In most modern architectures, Softmax serves as a core activation function: converting scaled dot products into normalized attention weights. During pretraining, Softmax is favored for its differentiable normalization with efficient vectorizable implementations (Vaswani et al., 2017; Joulin et al., 2017). However, despite its ubiquity, there has been a growing debate over its suitability for next-generation LLMs. Specifically, Softmax's strict sum-to-one constraint induces two major pathologies: (*i*) *attention sink*, where the model is forced to assign non-trivial probability mass to irrelevant tokens to satisfy normalization constraints; and (*ii*) *attention dispersion*, where probability mass is progressively diluted as sequence length grows, weakening the model's focus on salient tokens and making it inefficient for long-sequence modeling.

To mitigate the aforementioned limitations, prior work has explored *sparse attention mechanisms* that assign exact zero weights to alleviate attention dispersion (Veličković et al., 2025). Various methods (Martins and Fernández Astudillo, 2016; Peters et al., 2019; Vasylenko et al., 2025) replace exponential normalization with sparse projections to improve long-context behavior. However, these approaches typically rely on computationally expensive sorting or iterative projection that scale poorly. Moreover, as they still enforce sum-to-one normalization, they do not fundamentally resolve the attention sink phenomenon.

Alternatively, *rectified attention mechanisms* replace softmax with ReLU-style activations (e.g., ReLA) (Zhang et al., 2021; Shen et al., 2023), removing the sum-to-one constraint and naturally inducing sparsity without expensive sorting. While early proposals such as ReLA is efficient without enforcing attention sink, they often underperform dense Softmax attention, particularly in long-context regimes, due to limited expressivity and noise accumulation.

In this work, we revisit rectified attention and identify two core challenges to close the performance gap: increasing performance via enhancing expressive capacity, and mitigating long-context degradation by suppressing extreme-value noise. To enhance expressivity, we adopt the underlying view of differential Transformer by computing an *inhibitory* view to inhibit the noise, which enables

[*]Equal contribution.
[†]Work done as intern in Snap Inc.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 18073–18092

Method	Exact 0	Negative	No sum-to-1	Length-aware
Softmax	✗	✗	✗	✗
SSMax	✗	✗	✗	✓
Entmax	✓	✗	✗	✗
LSSAR	✗	✗	✗	✓
ReLA	✓	✗	✓	✗
Diff Softmax	✗	✓	✓	✗
TRA (ours)	✓	✗	✓	✓
TDA (ours)	✓	✓	✓	✓

Table 1: **Feature comparison of attention mechanisms. Exact 0**: can output exact zeros. **Negative**: can output signed weights. **No sum-to-1**: does not enforce row normalization. **Length-aware**: explicitly depends on sequence length.

negative attention weights, thereby increasing expressive capacity. To tackle long-context limitations, we derived from extreme-value theory and introduced a thresholding mechanism that progressively filters noise as the context length grows. We term this approach Threshold Differential Attention (**TDA**). Our contribution includes:

- We propose TDA, a drop-in, *non-softmax* attention mechanism. By applying length-dependent thresholding with an inhibitory differential view, TDA yields a sink-free, ultra-sparse attention that is robust to long-context modeling.
- We provide a theoretical analysis under sub-Gaussian assumptions. While spurious attention scores in dense models grow with context length, we prove that TDA effectively controls this noise: the expected number of spurious survivors per row remains $O(1)$, and consensus spurious matches across independent views vanish.
- Empirically, we show that TDA achieves $>$ 99% sparsity while maintaining competitive performance on both standard and long-context QA benchmarks, and eliminates attention sinks across layers and heads.
- We provide a fused Triton kernel implementation[1] and show that the resulting TRA kernel is competitive with FlashAttention-2 under BF16, achieving consistent speedups at long contexts.

2 Related Works

Attention Sink. The attention sink phenomenon, wherein models allocate excessive probability mass to initial tokens, is a structural byproduct of the softmax sum-to-one requirement. Barbero et al. (2025) provide a formal analysis connecting attention sinks to the prevention of representational collapse in deep Transformers. Similarly, Gu et al. (2025) show that sinks emerge as learned key biases that store excess mass without influencing value computation, and that replacing softmax with sigmoid-style attention can prevent their emergence. To mitigate this, Softpick (Zuhri et al., 2025) uses a non-sum-to-one kernel to bypass sinks by design. Recently, Qiu et al. (2025) proposed *Gated Attention*, which applies a head-specific sigmoid gate after SDPA; they report that the induced sparsity mitigates attention sinks (and "massive activations") and improves long-context extrapolation.

Sparse Attention for Attention Dispersion. Standard attention mechanisms compute a dense probability distribution that tends toward uniformity as sequence length increases, an effect known as *attention dispersion* (Veličković et al., 2025). To counter this, several methods were proposed to produce exact-zero attention weights. Martins and Fernández Astudillo (2016) and the α-entmax family (Peters et al., 2019) replace the softmax exponential with a transformation that projects logits onto a sparse support. Recent variants like ASEntmax (Vasylenko et al., 2025) and AdaSplash (Gonçalves et al., 2025) further refine this by adaptively calibrating sparsity or utilizing GPU-efficient kernels to improve length generalization.

Orthogonal to projection methods, rectified attention models such as ReLA (Zhang et al., 2021) show high performance without the normalization constraints of the exponential kernel. ReLUFormer (Shen et al., 2023) replaces Softmax with ReLU-style activations inside attention and showcases improved performance on long-sequence settings. Zuhri et al. (2025) advance this by removing the sum-to-one constraint, employing a non-probabilistic, rectified activation that yields sparse outputs without the computational overhead of iterative projection. Furthermore, Sliced ReLU Attention (Boufadène and Vialard, 2025) demonstrates that applying ReLU to projected scores maintains the expressive power of standard Transformers while enabling quasi-linear efficiency via sorting.

Differential Attention. Differential Attention (Ye et al., 2025) introduces an inhibitory view that enables explicit noise cancellation, yielding a *signed* attention signal, encoding both positive and negative interactions. Dex (Kong et al., 2025) further analyzes the interaction between this machanism and pretrained self-attention, proposing modifications that retain pretrained capabilities while

[1] https://github.com/snap-research/TDA.git

unlocking the benefits of differential inhibition.

Attention Scaling for Length Generalization. Complementary to changing the attention activation, long-context performance can be improved via length-dependent scaling. Nakanishi (2025) propose SSMax to adjust attention scaling with sequence length to mitigate softmax flattening , and recent theory formalizes a critical scaling regime for long-context transformers (Chen et al., 2025). Relatedly, LSSAR (Gao and Spratling, 2025) replaces the exponential nonlinearity with a Softplus transform and introduces a length-dependent scaling factor for better extrapolations. We summarize representative methods in Table 1.

3 Preliminaries

Notation. Let $T \in \mathbb{N}$ denote the sequence length and $\mathcal{V}$ be the vocabulary with size $|\mathcal{V}|$. A token sequence is $\boldsymbol{x}_{1:T}$, where $\boldsymbol{x}_i \in \{1, \ldots, |\mathcal{V}|\}$. The model (embedding) dimension is d_{model}, the number of layers is L, the number of attention heads is H, and the per-head dimension is d. $\mathbf{0}$ denotes an all-zeros matrix of the appropriate shape. We write $\langle \cdot, \cdot \rangle$ for the Euclidean inner product.

Transformer architecture. A Transformer stacks L identical blocks, each consisting of multi-head self-attention (MHSA) and a position-wise feed-forward network (FFN), wrapped with residual connections and normalization (Vaswani et al., 2017; He et al., 2016; Ba et al., 2016). The input to the first layer consists of token embeddings summed with positional encodings. While earlier architectures used absolute position vectors (Vaswani et al., 2017; Devlin et al., 2019), modern LLMs typically incorporate relative or rotary position information directly into the attention mechanism (Shaw et al., 2018; Su et al., 2021). Each layer $\ell \in \{1, \ldots, L\}$ computes

$$\widehat{\boldsymbol{H}}^{(\ell)} = \mathrm{Norm}\Big(\boldsymbol{H}^{(\ell-1)} + \mathrm{MHSA}\big(\boldsymbol{H}^{(\ell-1)}\big)\Big),$$
$$\boldsymbol{H}^{(\ell)} = \mathrm{Norm}\Big(\widehat{\boldsymbol{H}}^{(\ell)} + \mathrm{FFN}\big(\widehat{\boldsymbol{H}}^{(\ell)}\big)\Big),$$

where causal masking (decoder-only LMs) restricts each position to attend to its prefix. Norm denotes LayerNorm (Ba et al., 2016). MHSA denotes Multi-head self-attention, and FFN is a Multi-layer Perceptron.

Attention Mechanisms. Let $\boldsymbol{x}_{1:T}$ be a length-T token sequence with hidden states $\boldsymbol{X} \in \mathbb{R}^{T \times d_{\text{model}}}$. For a single attention head with head dimension d, define projections

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_Q, \qquad \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_K, \qquad \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_V,$$

where $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{T \times d}$ and $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V \in \mathbb{R}^{d_{\text{model}} \times d}$. Scores are

$$\boldsymbol{S} = \frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}} \in \mathbb{R}^{T \times T}, \qquad \boldsymbol{S}_{ij} = \frac{\langle \boldsymbol{q}_i, \boldsymbol{k}_j \rangle}{\sqrt{d}}.$$

With causal masking, query i attends to $\mathcal{S}(i) = \{1, \ldots, i\}$ by setting $\boldsymbol{S}_{ij} = -\infty$ for $j \notin \mathcal{S}(i)$.

Standard (softmax) Attention. Softmax attention uses row-stochastic weights (Vaswani et al., 2017):

$$\boldsymbol{A}_{ij} = \frac{\exp(\boldsymbol{S}_{ij})}{\sum_{t \in \mathcal{S}(i)} \exp(\boldsymbol{S}_{it})}, \qquad \boldsymbol{O} = \boldsymbol{A}\boldsymbol{V}.$$

Rectified Linear Attention (ReLA). ReLA replaces softmax with a rectifier and removes the sum-to-one constraint (Zhang et al., 2021):

$$\boldsymbol{A} = \max(\boldsymbol{S}, \mathbf{0}), \qquad \boldsymbol{O} = \mathrm{Norm}(\boldsymbol{A}\boldsymbol{V}).$$

where Norm is LayerNorm (Ba et al., 2016).

Differential (softmax) Attention. Differential attention constructs two softmax maps (t_1, t_2), and subtracts them (Ye et al., 2025; Kong et al., 2025):

$$\boldsymbol{A}^{(t)} = \mathrm{softmax}\left(\frac{\boldsymbol{Q}^{(t)}(\boldsymbol{K}^{(t)})^\top}{\sqrt{d}}\right), \quad t \in \{1, 2\},$$
$$\boldsymbol{A}^{\Delta} = \boldsymbol{A}^{(1)} - \lambda\, \boldsymbol{A}^{(2)}, \qquad \boldsymbol{O} = \boldsymbol{A}^{\Delta}\boldsymbol{V},$$

where λ is a learnable (often layer-dependent) scalar. Note $\boldsymbol{A}^{\Delta}$ may be signed.

Attention Dispersion. We adapt the notion of *attention dispersion* from Vasylenko et al. (2025) for unnormalized and signed attention weights. Let $\boldsymbol{a}_i \in \mathbb{R}^i$ be the vector of attention weights for a query at position i (where $\boldsymbol{a}_{ij} = 0$ if masked). We define the *effective entropy* $H(\boldsymbol{a}_i)$ as the Shannon entropy of the ℓ_1-normalized absolute weights:

$$\hat{p}_{ij} = \frac{|\boldsymbol{a}_{ij}|}{\|\boldsymbol{a}_i\|_1 + \epsilon}, \qquad H(\boldsymbol{a}_i) = -\sum_{j=1}^{i} \hat{p}_{ij} \log \hat{p}_{ij}.$$

The attention mechanism is said to be *dispersive* if the effective entropy grows at the same rate as the maximum possible entropy (i.e., the uniform distribution):

$$\lim_{i \to \infty} \frac{\mathbb{E}[H(\boldsymbol{a}_i)]}{\log i} = 1.$$

Conversely, the mechanism is *non-dispersive* if this ratio approaches 0.

Attention Sink. Attention sinks describe abnormally large attention allocated to a fixed *position* (often the first token) (Xiao et al., 2023; Gu et al., 2025). Importantly, some attention mechanisms we consider produce weights that are not probabilities (e.g., ReLA is unnormalized, differential attention can be signed). To compare positional dominance across such mechanisms, for layer ℓ and head h, we define a *generalized sink ratio* by first ℓ_1-normalizing absolute weights per query following Gu et al. (2025):

$$\tilde{\mathbf{A}}_k^{\ell,h} = \frac{1}{T-k+1}\sum_{i=k}^{T}\frac{|\mathbf{A}_{i,k}^{\ell,h}|}{\sum_{t\in\mathcal{S}(i)}|\mathbf{A}_{i,t}^{\ell,h}|}$$

and then reporting the times-uniform ratio $\mathrm{gSinkRatio}^{\ell,h}(k) := \tilde{\mathbf{A}}_k^{\ell,h}/\tilde{\mathbf{A}}_k^{\mathrm{unif}}$, where

$$\tilde{\mathbf{A}}_k^{\mathrm{unif}} := \frac{1}{T-k+1}\sum_{i=k}^{T}\frac{1}{|\mathcal{S}(i)|}.$$

We write $\mathrm{gSinkRatio}(k)$ as the average (over layers and heads) of the ratio between the total attention mass assigned to key position k.

4 Methodology

We first introduce Threshold Rectified Attention (TRA), which scales the rectification threshold with context length to suppress extreme-value noise. We then extend it to Threshold Differential Attention (TDA), which subtracts an inhibitory thresholded view to further cancel spurious matches noise.

4.1 Threshold Rectified Attention (TRA)

Rectified attention replaces Softmax with a simple rectifier, producing *un-normalized* and often *sparse* weights: it assigns exact zeros and avoids the sum-to-one constraint that underlies attention sinks (Zhang et al., 2021; Shen et al., 2023). However, plain ReLA-style attention often underperforms Softmax attention in long-context regime (Zuhri et al., 2025). We attribute this to *noise accumulation*: as the context grows, unrelated (query, key) pairs produce larger *maximum* dot-products by chance (extreme values), and a *fixed* rectifier threshold eventually fails to inhibit such spurious pairs, polluting the value aggregation.

Assumption 4.1 (Sub-Gaussian noise per row)**.** Fix a query position i with visible set $\mathcal{S}(i) = \{1,\dots,i\}$. For any *noise* key $j \in \mathcal{N}(i)$, the similarity $\boldsymbol{s}_{ij} = \langle\tilde{\boldsymbol{q}}_i, \tilde{\boldsymbol{k}}_j\rangle$ is mean-zero and σ^2/d-sub-Gaussian: $\mathbb{E}[\exp(t\boldsymbol{s}_{ij})] \le \exp\left(\frac{\sigma^2}{2d}t^2\right), \forall t \in \mathbb{R}$.

Assumption 4.2 (Bounded relevant survivors)**.** For each row, the number of relevant keys exceeding threshold is uniformly bounded by a constant r.

Definition. Let $\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{v}_j \in \mathbb{R}^d$ denote the per-head query, key, and value vectors. We normalize queries and keys, $\tilde{\boldsymbol{q}}_i := \frac{\boldsymbol{q}_i}{\|\boldsymbol{q}_i\|_2}$ and $\tilde{\boldsymbol{k}}_j := \frac{\boldsymbol{k}_j}{\|\boldsymbol{k}_j\|_2}$, and compute scores as $\boldsymbol{s}_{ij} := \langle\tilde{\boldsymbol{q}}_i, \tilde{\boldsymbol{k}}_j\rangle$. TRA applies a *length-dependent* threshold τ_i:

$$\tau_i := \beta\sqrt{\frac{2\log\left(\frac{i+1}{\kappa}\right)}{d}}, \qquad \kappa > 0. \tag{1}$$

$$\boldsymbol{a}_{ij} := \left(\boldsymbol{s}_{ij} - \tau_i\right)_+^p \quad \text{if } j \le i,$$

$$\boldsymbol{o}_i := \mathrm{Norm}\left(\sum_{j=1}^{i}\boldsymbol{a}_{ij}\boldsymbol{v}_j\right).$$

Here $(x)_+ = \max(x, 0)$, $\beta > 0$ is a learnable scalar (that controls the threshold size and overall sparsity), $p \ge 1$ is the power, and Norm is RMSNorm. The *row-wise gate* τ_i increases with causal context size $|\mathcal{S}(i)| = i$, ensuring the rectifier remains selective as i grows. This scaling follows the extreme-value behavior of sub-Gaussian noise: Vershynin (2018) gives the bound $\mathbb{P}(\max_{j\le i} X_j > \tau) \le i\exp(-\tau^2/(2\sigma^2))$; applying this to dot-products with variance proxy $\mathcal{O}(1/d)$ yields the threshold scale $\tau \asymp \sqrt{2\log i/d}$.

Properties. Let $\mathcal{N}(i) \subseteq \mathcal{S}(i)$ denote the set of noise keys (irrelevant to the query) at row i. A key is a *spurious survivor* if $j \le i$ and $j \in \mathcal{N}(i)$ yet $\boldsymbol{s}_{ij} > \tau_i$. The total number of spurious survivors is

$$S_i := \sum_{j\in\mathcal{N}(i)}\mathbf{1}(\boldsymbol{s}_{ij} > \tau_i),$$

where $\mathbf{1}$ is the indicator function. τ_i tracks the extreme-value scale of spurious dot-products, and hyperparameter κ to control the *expected number of spurious survivors per row*. Hence, TRA ensures the tail exceedance remain controlled, yielding stable sparsity and reducing long-context corruption from chance matches.

Theorem 4.3 (TRA keeps $O(1)$ spurious survivors per row)**.** *Under Assumption 4.1, fix any $\kappa > 0$, then for all $i \ge 1$, $\mathbb{E}[S_i] \le \kappa$.*

Under the sub-Gaussian noise model, a length-dependent threshold with $\beta \ge \sigma$ keeps the expected number of spurious survivors per query row bounded as context length grows (Proof in Appendix A.1) and prevents noise accumulation

from dominating long-context aggregation. Hence, TRA is non-dispersive (Proof in Appendix A.2).

Theorem 4.4 (TRA is Non-Dispersive)**.** *Under Assumptions 4.1 and 4.2, the Threshold Rectified Attention (TRA) is non-dispersive.*

4.2 Threshold Differential Attention (TDA)

While TRA bounds the *number* of spurious exceedances per row, a single view thresholded map can still admit occasional high-magnitude noise. To suppress these, we incorporate the key idea of *differential attention* (Ye et al., 2025; Kong et al., 2025): we compute an *excitatory* thresholded view and subtract an *inhibitory* thresholded view. Intuitively, a large similarity measure $\boldsymbol{q}^\top \boldsymbol{k}$ can also occur spuriously due to shared, uninformative structure. In TDA, the inhibitory TRA is trained to capture such non-selective exceedances.

Definition. We employ two independent sets of projections $\{\boldsymbol{q}^{(t)}, \boldsymbol{k}^{(t)}\}_{t\in\{1,2\}}$. We normalize queries and keys in each view and compute scores $\boldsymbol{s}_{ij}^{(t)} := \langle \tilde{\boldsymbol{q}}_i^{(t)}, \tilde{\boldsymbol{k}}_j^{(t)} \rangle$. Using the same length-dependent threshold τ_i as in TRA (Equation (1)), we compute the differential weights:

$$\begin{aligned} \boldsymbol{a}_{ij}^{(t)} &:= (\boldsymbol{s}_{ij}^{(t)} - \tau_i)_+^p, \qquad t \in \{1,2\}, \\ \Delta \boldsymbol{a}_{ij} &:= \boldsymbol{a}_{ij}^{(1)} - \lambda \boldsymbol{a}_{ij}^{(2)} \qquad \lambda \in (0,1), \\ \boldsymbol{o}_i &:= \mathrm{Norm}\left(\sum_{j=1}^{i} \Delta \boldsymbol{a}_{ij} \boldsymbol{v}_j \right). \end{aligned}$$

Here λ is a learned scalar controling inhibition strength. Similar to Ye et al. (2025), $\Delta \boldsymbol{a}_{ij}$ can be negative, yielding a signed attention signal.

Properties. We define *consensus spurious survivors* C_i as the number of noise keys exceeding the threshold in *both* views simultaneously.

$$C_i := \sum_{j \in \mathcal{N}(i)} \mathbf{1}\left(\boldsymbol{s}_{ij}^{(1)} > \tau_i\right) \mathbf{1}\left(\boldsymbol{s}_{ij}^{(2)} > \tau_i\right)$$

Assumption 4.5 (Independence of noise views[2])**.** Fix a query position i. For any noise key $j \in \mathcal{N}(i)$, the similarities $\boldsymbol{s}_{ij}^{(1)}$ and $\boldsymbol{s}_{ij}^{(2)}$ are independent.

Theorem 4.6 (Consensus spurious survivors vanish)**.** *Under Assumption 4.1 and Assumption 4.5 for both views (with the same σ), for all $i \geq 1$, $\mathbb{E}[C_i] \leq \frac{\kappa^2}{i+1}$. Thus, $\lim_{i\to\infty} \mathbb{E}[C_i] = 0$.*

[2]We assume independent noise for tractability; with positive dependence, joint exceedances may increase, and TDA then relies more on differential cancellation than filtration.

This formalizes the benefit of TDA: while each view may admit $O(1)$ spurious exceedances, the probability of them overlapping on the same noise token vanishes as context length increases (Proof in Appendix A.3). Additionally, as a linear combination of two TRA views, TDA naturally inherits the non-dispersive property (Proof in Appendix A.4):

Corollary 4.7 (TDA is Non-Dispersive)**.** *Under Assumptions 4.1, 4.2, and 4.5 for both views (with same σ), then TDA is non-dispersive.*

We provide additional empirical diagnostics for all the assumptions in Appendix B.

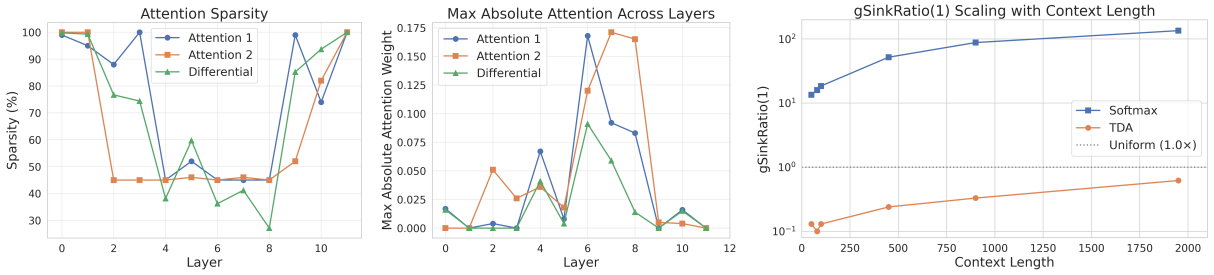
5 Mechanistic Analysis of TDA

We begin by mechanistically diagnosing the behavior of TDA to illustrate its theoretical properties. For these analyses, we use a modified GPT-2 architecture (Radford et al., 2019) where the standard absolute positional embeddings are replaced with Rotary Positional Embeddings (RoPE) (Su et al., 2021) and the Softmax attention is replaced by TDA. We use the sentence *"The quick brown fox jumps over the lazy dog"* as input to visualize internal attention dynamics and inhibition patterns.

Attention Sparsity. A defining feature of TDA is that it produces intrinsically sparse attention maps without explicit top-k truncation. To quantify this, we mark an attention entry as *inactive* if its magnitude is exact 0, and report sparsity as the fraction of inactive entries per layer (Figure 1a). We observe a depth-dependent profile: early and late layers are highly sparse, whereas middle layers are substantially more active (near-zero rate drops to $\sim$50%). This "active core" is consistent with representations in intermediate layers, producing stronger and query–key alignments, so a larger fraction of pairs exceed the row-wise threshold and contribute to value aggregation; outside this region, most interactions are gated off, and further cancel differential subtraction, yielding near-all-zero maps.

Attention Sink. Figure 1b shows the layerwise maximum absolute attention weight in each view and in the effective differential map $\Delta \boldsymbol{A} = \boldsymbol{A}^{(1)} - \lambda \boldsymbol{A}^{(2)}$. Across layers, the differential map exhibits substantially smaller peaks than either individual view, indicating that inhibition cancels large common-mode exceedances and prevents any single interaction from dominating the aggregation.

This bounded-peak behavior directly translates to robustness against attention sinks. As shown in



(a) **Attention Sparsity.** (b) **Max Absolute Attention.** (c) **Attention Sink Scaling.**

Figure 1: **Mechanistic diagnostics for TDA.** We visualize (a) the sparsity of attention weights across layers, (b) the maximum absolute attention values, and (c) the first token attention sink ratio $\mathrm{gSinkRatio}(1)$ as context length increases. Attention 1 indicates an excitatory view, while Attention 2 indicates an inhibitory view.

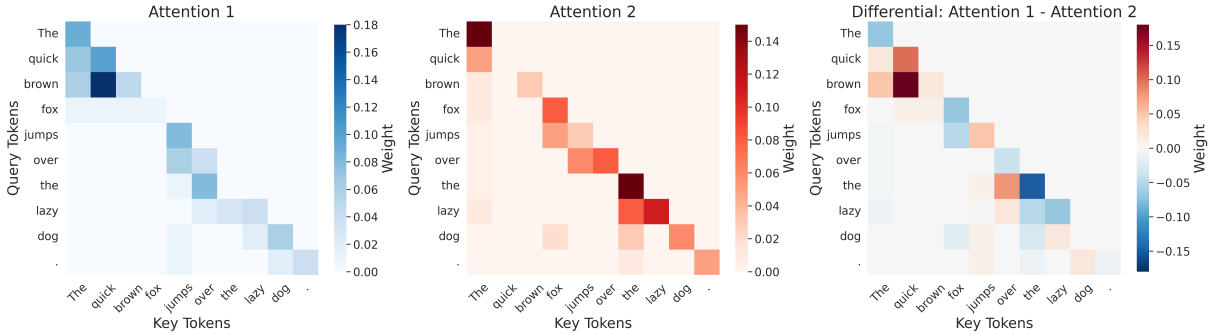


Figure 2: **Inhibition in TDA.** Per-token attention maps for a representative head (Layer 6, Head 0) on the sequence *"The quick brown fox jumps over the lazy dog"*. The differential attention maps show $\Delta \boldsymbol{A} = \boldsymbol{A}^{(1)} - \lambda \boldsymbol{A}^{(2)}$ where $\lambda = 1$; negative values indicate *inhibition*, while positive values indicate *excitation*.

Figure 1c, under Softmax, the first-token sink ratio $\mathrm{gSinkRatio}(1)$ increases sharply as the sequence grows, reflecting the well-known tendency for early positions to become globally attractive "sinks." In contrast, TDA maintains a sink ratio near (or below) the uniform baseline as context length increases, confirming that differential inhibition effectively prevents the formation of attention sinks.

Inhibition Behavior. The differential panel in Figure 2 visualizes the signed weights $\Delta \boldsymbol{a}_{ij} := \boldsymbol{a}_{ij}^{(1)} - \lambda\, \boldsymbol{a}_{ij}^{(2)}$. Negative values correspond to *inhibition*: key positions whose contribution is actively subtracted in the value aggregation. In the example head, the high-frequency preposition *"the"* is broadly inhibited across many queries, consistent with suppressing globally attractive but semantically weak keys. By contrast, content tokens such as *"quick"* and *"brown"* show *query-dependent* inhibition: they contribute only for the subset of queries where they are useful, and are inhibited elsewhere. Overall, the inhibition behavior enables fine-grained suppression of redundant keys while preserving selective interactions.

6 Experiments

We pretrain all models on the FineWebEdu-10B dataset from scratch, a high-quality educational subset of the FineWeb dataset (Penedo et al., 2024). The dataset consists of 10B tokens, and we reserve the first 100M tokens for validation. We train a variant of GPT-2-162M (Radford et al., 2019) model, replacing the learnable positional encoding with RoPE (Su et al., 2021). Throughout the experiments, we set $\kappa = 1, \beta = 1$, and $p = 2$ for TRA and TDA. When extending models to longer context lengths, we employ NTK-aware scaling for RoPE-based models (Peng et al., 2024; bloc97, 2023), and further train them on the pretraining dataset for 500 additional steps. We conduct all inference using `lm-evaluation-harness` library (Gao et al., 2023). All experiments were conducted on 8 NVIDIA A100-80GB GPUs. See further experimental details in Appendix E.

6.1 Language Modeling

Setup. For language modelling, we report the zero-shot performance (accuracy and length-normalized accuracy) on a standard suite of multiple-choice commonsense and science QA

Method	Val. Loss ↓	HellaSwag		ARC-Easy		ARC-Challenge		OpenBookQA		PIQA		Winogrande	Sparsity ↑
		Acc	Acc-Norm	Acc	Acc-Norm	Acc	Acc-Norm	Acc	Acc-Norm	Acc	Acc-Norm	Acc	
Softmax	3.1196	**0.345**	0.409	**0.526**	0.487	0.223	0.245	0.180	0.304	0.641	0.621	0.490	0%
Gated Softmax	3.1489	0.330	0.382	0.474	0.436	0.194	0.224	0.162	0.284	0.620	0.586	0.500	0%
SSMax	3.1369	0.324	0.387	0.472	0.462	0.191	0.231	0.144	0.302	0.620	0.590	0.508	0%
Entmax	3.1941	0.342	0.391	0.508	0.472	0.194	0.245	0.198	0.304	0.632	0.609	0.523	43%
LSSAR	3.1676	0.330	0.378	0.521	0.470	0.217	**0.259**	0.192	0.314	0.652	0.621	**0.532**	0%
ReLA	3.1657	0.329	0.394	0.512	0.468	**0.226**	0.250	0.194	0.306	0.634	0.621	0.509	94%
Diff Softmax	3.1941	0.336	**0.423**	0.509	**0.496**	0.225	0.252	0.178	0.316	**0.648**	0.619	0.514	0%
Dex	3.1349	0.339	0.395	0.492	0.466	0.215	0.241	0.172	0.282	0.640	0.608	0.519	0%
Diff ReLA	3.1294	0.331	0.391	0.514	0.472	0.220	0.248	0.192	0.298	0.636	0.623	0.494	96%
TRA (p=2, β=1)	3.1320	0.330	0.401	0.516	0.471	**0.226**	0.247	0.194	0.278	0.637	**0.626**	0.496	92%
TDA (p=2, β=1)	**3.1190**	0.337	0.415	0.524	0.488	0.220	0.239	**0.216**	**0.320**	0.628	**0.626**	0.489	**99%**

Table 2: **Language modeling results.** We report validation loss and both accuracy (Acc) and length-normalized accuracy (Acc-Norm). Winogrande reports Acc only. **Sparsity** (see Section E.1) is the fraction of attention weights that are exactly 0, averaged over layers, heads, and instances. We **bold** the first and underscore the second place.

benchmarks: HellaSwag (Zellers et al., 2019), ARC-Easy/Challenge (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), and Winogrande (Sakaguchi et al., 2020). We compare against (i) *Softmax* attention (Vaswani et al., 2017) and two length/activation variants: *Gated Softmax* (Qiu et al., 2025) and *Scalable Softmax (SSMax)* (Nakanishi, 2025); (ii) non-softmax baselines including *Entmax* (Martins and Fernández Astudillo, 2016; Peters et al., 2019), *LSSAR* (Gao and Spratling, 2025), and *ReLA* (Zhang et al., 2021); and (iii) differential baselines *Differential Softmax* (Ye et al., 2025) and *Dex* (Kong et al., 2025). Finally, we include *Differential ReLA* as a straightforward combination of ReLA with a differential construction, and our proposed *TRA* and *TDA*. For a fair comparison, we retrained all baselines and our proposed methods under the same experimental configurations.

Results. The results on zero-shot common sense reasoning are summarized in Table 2. TDA achieves the lowest validation loss and 99% sparsity while maintaining competitive accuracy across all tasks. Standard Softmax remains a reliable baseline. While variants like Gated and Scalable Softmax aim to improve length robustness, they do not surpass the baseline in the standard context regime.

Among non-softmax methods, unconstrained ReLA suffers from performance degradation due to noise accumulation. Our single-view TRA significantly closes this gap, validating that length-dependent thresholding effectively manages the noise floor. Entmax also shows that sparsity helps with reasoning, achieving competitive accuracy. LSSAR performs impressively on ARC-Challenge and Winogrande but remains fully dense, foregoing the efficiency benefits of exact zeros.

Finally, Differential Softmax is the strongest baseline, validating the utility of inhibitory noise cancellation, yet it remains fully dense. Dex attempts efficient correction but trails in accuracy without achieving sparsity, while Differential ReLA gains sparsity at the cost of performance. TDA bridges this gap, matching the high accuracy of Differential Softmax while achieving extreme sparsity by actively filtering noise via thresholding rather than just suppressing it.

Method	QMSum	SummScreenFD	GovReport	Qasper
Softmax	10.29	7.25	3.78	8.82
SSMax	11.22	8.47	2.68	9.70
Diff Softmax	10.57	8.08	3.08	11.23
Entmax	**11.52**	**10.16**	4.24	**11.54**
ReLA	11.20	9.14	4.42	10.77
TRA (p=2, β=1)	11.18	9.47	**5.61**	11.09
TDA (p=2, β=1)	11.46	9.13	5.24	11.41

Table 3: **Long-context evaluation on SCROLLS.** We report ROUGE-1 for QMSum, SummScreen, and GovReport, and F1 for Qasper. We **bold** the first and underscore the second place.

6.2 Long-Context Language Modeling

Setup. We evaluate long-context generalization on four tasks from the SCROLLS benchmark (Shaham et al., 2022): QMSum, SummScreenFD, GovReport, and Qasper. We report ROUGE-1 for QMSum, SummScreenFD, and GovReport, and F1 for Qasper, following SCROLLS. We compare Softmax and representative alternatives: Scalable Softmax, Differential Softmax, Entmax, and ReLA, against our proposed TRA and TDA.

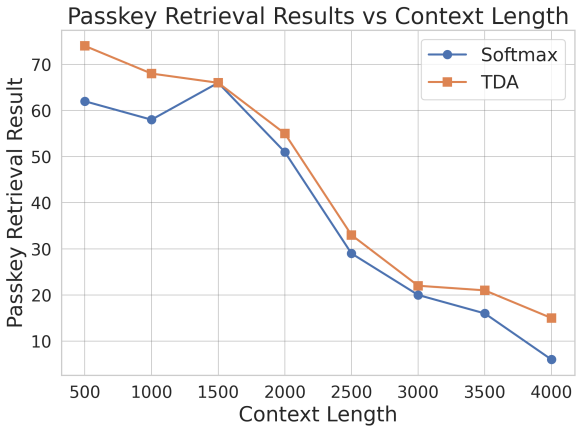


Figure 3: **Passkey retrieval results.** We report correct runs over 100 trials for each context length, with randomly positioned passkey per trial.

Results. Table 3 shows that our methods are consistently competitive: TDA is the second-best method on QMSum and Qasper, while TRA achieves the best GovReport score and ranks second on SummScreenFD. While Entmax achieves the best overall long-context performance across the four SCROLLS tasks, it is known to be substantially more expensive at long context. Overall, both TRA and TDA provide strong long-context performance while avoiding projection-based sparse attention overheads.

6.3 Passkey Retrieval Test

Setup. We evaluate long-context retrieval using the *passkey retrieval* stress test (Mohtashami and Jaggi, 2023; Kamradt, 2023), in which a short numeric key is inserted at a random position inside a long span of irrelevant text, and the model is later queried to retrieve the key. We test target context lengths from 500 to 4000 tokens in increments of 500, and run 100 trials with independently sampled passkeys and random insertion locations for each length. See Appendix D for details.

Results. Figure 3 reports the number of successful trials (out of 100) as a function of context length. Both Softmax and TDA degrade as the amount of irrelevant text grows, but TDA is consistently more robust across nearly all lengths. Specifically, TDA consistently yields higher retrieval accuracy than Softmax in shorter contexts, and this advantage persists at long contexts: at 4000 tokens, TDA achieves 15% correct vs. 6% for Softmax. Overall, these results suggest that TDA's length-aware thresholding and inhibitory view better suppress chance matches from irrelevant tokens, showing improved retrieval under heavy long-context noise.

Context	2 Needles		4 Needles	
	Softmax	TDA	Softmax	TDA
500	48.0	**72.0**	18.0	**62.0**
1000	40.0	**64.0**	8.0	**28.0**
2000	4.0	**70.0**	0.0	**2.0**
4000	0.0	**82.0**	0.0	**20.0**

Table 4: **Multi-needle passkey retrieval.** Avg accuracy of multiple key-value pairs which are inserted at widely separated positions in the context.

6.4 Multi-needle Retrieval Test

Setup. To further test whether extreme sparsity harms retrieval from multiple distant locations, we also evaluate a *multi-needle* variant in which the model must retrieve several key-value pairs placed at widely separated positions in the context.

Results. Table 4 shows that TDA consistently outperforms Softmax in both the 2-needle and 4-needle settings, with the advantage becoming especially pronounced at longer contexts. These results suggest that TDA's sparsity does not eliminate useful global interactions; instead, by filtering irrelevant matches, it preserves the ability to retrieve multiple distant pieces of information under heavy long-context noise.

6.5 Practical Efficiency

Setup. A practical advantage of the ultra-sparsity induced by TDA and TRA is the potential reduction in value-aggregation cost. We therefore evaluate the runtime of the proposed TRA kernel under BF16, which is the standard precision for modern LLM training and inference. Table 5 compares TRA against FlashAttention-2 (Dao, 2024) using identical tensor layouts, hardware, and CUDA-event median timing across sequence lengths up to 65k tokens. Additional implementation details and FP32 benchmarks are provided in Appendix C.

Results. TRA is slightly slower at very short sequence lengths ($\leq$2k) due to fixed kernel overhead, becomes competitive around 4k, and achieves consistent speedups at longer contexts ($\geq$8k), reaching up to 1.29$\times$ at 32k tokens. These gains arise from the sparsity induced by thresholding, which reduces value aggregation cost and becomes more pronounced as sequence length increases.

Context Length	512	1024	2048	4096	8192	16384	32768	65536
FlashAttn-2	0.108	**0.124**	**0.213**	**0.568**	2.009	7.781	31.713	135.616
TRA	**0.099**	0.140	0.224	0.577	**1.773**	**6.240**	**24.641**	**109.803**
Speedup	1.09×	0.89×	0.95×	0.98×	1.13×	1.25×	1.29×	1.24×

Table 5: **BF16 latency (ms) and speedup vs. FlashAttention-2.**

p	Val. Loss ↓	Avg Accuracy ↑
1	3.2068	0.3945
2	**3.1190**	**0.4023**
3	3.1408	0.4020
5	3.1412	0.3922

(a) **Different power** p (fix $\beta = 1$).

β	Val. Loss ↓	Avg Accuracy ↑
1.0	3.1190	**0.4023**
0.8	**3.1140**	0.4015
0.5	3.1288	0.4018

(b) **Different threshold scaling** β (fix $p = 2$).

Table 6: **TDA hyperparameter study.** Avg Accuracy is averaged over HellaSwag, ARC-Easy, ARC-Challenge, OpenBookQA, PIQA, and Winogrande.

For completeness, we also report additional FP32 benchmarks comparing our fused Triton implementation against naive PyTorch baselines and fused SDPA operator in Appendix C.

6.6 Hyperparameter Study

Effect on Power p. Table 6a studies the power p applied in TRA $(\boldsymbol{s}_{ij} - \tau_i)_+^p$. We find that a mild nonlinearity is important: p=2 yields the best validation loss and average accuracy, while p=1 is noticeably worse, likely because it removes the nonlinearity and thus reduces expressive power, which is consistent with prior observations (Gao and Spratling, 2025). Larger powers ($p \geq 3$) also slightly degrade performance, likely because aggressively amplifying increases gradient variance

Effect on Threshold Scaling Parameter β. Table 6b varies the threshold scale β, which controls the selectivity of the length-aware gate τ_i: smaller β admits more noise, whereas larger β can overprune and increase the risk of near-empty rows before the model adapts. Empirically, performance is fairly robust across the tested range, with β=1.0 achieving the best validation loss and the highest average accuracy, while β=0.5 remains close in accuracy but is slightly worse in loss.

7 Conclusion

We introduced *Threshold Differential Attention* (TDA), a drop-in non-softmax attention mechanism for language modeling that addresses the structural pathologies of attention sinks and dispersion. TDA combines a length-aware extreme-value threshold with a differential view, yielding signed, ultra-sparse, and sink-free attention. Theoretically, we prove that spurious survivors per row remains bounded by $O(1)$ and that consensus spurious exceedances across views vanish as context grows. Empirically, TDA achieves $> 99\%$ exact-zero sparsity while maintaining competitive performance. Furthermore, we provide a fused Triton kernel that translates this into significant runtime and memory gains. In summary, TDA provides a practical path toward long-context Transformers by improving robustness against dispersion and sinks.

Limitations

We note that due to hardware constraints, the evaluation in this work is primarily at a small scale - scaling TDA to larger models remains an important future work. While we observe consistent behavior across these settings, it remains to be validated whether the same sparsity patterns, training stability, and efficiency gains hold at larger scales (e.g., multi-billion parameter models).

Additionally, overly aggressive thresholding can also cause *dead heads* - heads with no survivors. While head inactivity is not unique to TDA and may in some cases function as an explicit idle state, in the extreme, widespread dead heads can effectively disable multi-head attention in certain layers, reducing the model's expressive capacity for long-range information routing and making performance more sensitive to threshold hyperparameters. Future work should investigate layer-/head-wise adaptive threshold schedules that preserve ultra-sparsity while preventing head collapse, especially in initial and late layers.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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

Federico Barbero, Álvaro Arroyo, Xiangming Gu, Christos Perivolaropoulos, Michael Bronstein, Petar Veličković, and Razvan Pascanu. 2025. Why do llms attend to the first token? In *COLM*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: Reasoning about physical commonsense in natural language. In *AAAI*.

bloc97. 2023. Ntk-aware scaled rope allows llama models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation. Reddit. Accessed: 2026-01-04.

Siwan Boufadène and François-Xavier Vialard. 2025. Sliced ReLU attention: Quasi-linear contextual expressivity via sorting. *arXiv preprint arXiv:2512.11411*.

Shi Chen, Zhengjiang Lin, Yury Polyanskiy, and Philippe Rigollet. 2025. Critical attention scaling in long-context transformers. *arXiv preprint arXiv:2510.05554*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Bo Gao and Michael W. Spratling. 2025. Softplus attention with re-weighting boosts length extrapolation in large language models. In *AAAI*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2023. A framework for few-shot language model evaluation.

Nuno Gonçalves, Marcos V. Treviso, and Andre Martins. 2025. AdaSplash: Adaptive sparse flash attention. In *ICML*.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. 2025. When attention sink emerges in language models: An empirical view. In *ICLR*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Armand Joulin, Moustapha Cissé, David Grangier, Hervé Jégou, and 1 others. 2017. Efficient softmax approximation for gpus. In *International conference on machine learning*, pages 1302–1310. PMLR.

Greg Kamradt. 2023. Llmtest_needleinahaystack. https://github.com/gkamradt/LLMTest_NeedleInAHaystack. GitHub repository, accessed 2026-01-02.

Chaerin Kong, Jiho Jang, and Nojun Kwak. 2025. Understanding differential transformer unchains pretrained self-attentions. In *NeurIPS*.

André F. T. Martins and Ramón Fernández Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Random-access infinite context length for transformers. In *NeurIPS*, volume 36.

Ken M. Nakanishi. 2025. Scalable-softmax is superior for attention. *arXiv preprint arXiv:2501.19399*.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A. Raffel, Leandro Von Werra, Thomas Wolf, and 1 others. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *NeurIPS*.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. Yarn: Efficient context window extension of large language models. In *ICLR*.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2019. Sparse sequence-to-sequence models. In *ACL*.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *NeurIPS*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *AAAI*.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, and 1 others. 2022. Scrolls: Standardized comparison over long language sequences. *arXiv preprint arXiv:2201.03533*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Kai Shen, Junliang Guo, Xu Tan, Siliang Tang, Rui Wang, and Jiang Bian. 2023. A study on ReLU and softmax in transformer. *arXiv preprint arXiv:2302.06461*.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. In *Proceedings of the 2021 Conference of the Association for Computational Linguistics (ACL)*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Pavlo Vasylenko, Hugo Pitorro, André F. T. Martins, and Marcos Treviso. 2025. Long-context generalization with sparse attention. *arXiv preprint arXiv:2506.16640*.

Petar Veličković, Christos Perivolaropoulos, Federico Barbero, and Razvan Pascanu. 2025. softmax is not enough (for sharp out-of-distribution). In *ICML*.

Roman Vershynin. 2018. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. 2025. Differential transformer. In *ICLR*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and 1 others. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *ACL*.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2021. Sparse attention with linear units. In *EMNLP*.

Zayd M. K. Zuhri, Erland Hilman Fuadi, and Alham Fikri Aji. 2025. Softpick: No attention sink, no massive activations with rectified softmax. *arXiv preprint arXiv:2504.20966*.

A Proofs

Assumption A.1 (Sub-Gaussian noise per row (restated Assumption 4.1))**.** Fix a query position i and consider the visible set $\mathcal{S}(i) = \{1, \dots, i\}$. Let the noise key lies in a subset $\mathcal{N} \subseteq \mathcal{S}$. For any *noise* key $j \in \mathcal{N}(i)$, the similarity $\boldsymbol{s}_{ij} = \langle \tilde{\boldsymbol{q}}_i, \tilde{\boldsymbol{k}}_j \rangle$ is mean-zero and σ^2/d-sub-Gaussian:

$$\mathbb{E}[\exp(t\boldsymbol{s}_{ij})] \le \exp\Big(\frac{\sigma^2}{2d}t^2\Big), \qquad \forall t \in \mathbb{R}.$$

Assumption A.2 (Bounded relevant survivors (restated Assumption 4.2))**.** Fix a query position i with visible set $\mathcal{S}(i) = \{1, \dots, i\}$. Let $\mathcal{R}(i) := \mathcal{S}(i) \setminus \mathcal{N}(i)$ denote the relevant (non-noise) keys. Then the number of relevant keys that exceed the threshold is uniformly bounded:

$$R_i := \sum_{j \in \mathcal{R}(i)} \mathbf{1}(\boldsymbol{s}_{ij} > \tau_i) \le r \quad \text{for all } i.$$

Assumption A.3 (Two-view independence for noise (restated Assumption 4.5))**.** For a fixed query position i and any noise key $j \in \mathcal{N}(i)$, the similarities $\boldsymbol{s}_{ij}^{(1)}$ and $\boldsymbol{s}_{ij}^{(2)}$ are independent.

A.1 Proof of Theorem 4.3

Theorem A.4 (TRA keeps $O(1)$ spurious survivors per row (restated Theorem 4.3))**.** *Assume Assumption 4.1. Define the number of* spurious survivors *in row i as*

$$S_i \;:=\; \sum_{j \in \mathcal{S}(i)} \mathbf{1}(\boldsymbol{s}_{ij} > \tau_i)\,,$$

(where the sum ranges over noise keys; for a worst-case statement, you may interpret all keys in $\mathcal{S}(i)$ as noise). If the row-wise threshold is

$$\tau_i \;:=\; \sigma\sqrt{\frac{2\log\big(\frac{i+1}{\kappa}\big)}{d}}, \qquad \kappa > 0,$$

then for all $i \ge 1$, $\mathbb{E}[S_i] \le \kappa$. More specifically, for $\tau_i = \beta\sqrt{\frac{2\log(i+1)}{d}}$ with $\beta > 0$,

$$\mathbb{E}[S_i] \;\le\; (i+1)^{\,1-\beta^2/\sigma^2}.$$

Proof. Fix a query position i and a noise key $j \in \mathcal{N}(i)$. By the sub-Gaussian tail implied by Assumption 4.1, for any $x \in \mathbb{R}$,

$$\mathbb{P}(\boldsymbol{s}_{ij} > x) \;\le\; \exp\Big(-\frac{dx^2}{2\sigma^2}\Big).$$

By linearity of expectation (no independence across keys is needed),

$$\mathbb{E}[S_i] = \sum_{j \in \mathcal{S}(i)} \mathbb{P}(\boldsymbol{s}_{ij} > \tau_i) \;\le\; i \cdot \exp\Big(-\frac{d\tau_i^2}{2\sigma^2}\Big).$$

For $\tau_i = \sigma\sqrt{\frac{2\log((i+1)/\kappa)}{d}}$, we have $\frac{d\tau_i^2}{2\sigma^2} = \log\big(\frac{i+1}{\kappa}\big)$, hence

$$\mathbb{E}[S_i] \le i\exp\Big(-\log\Big(\frac{i+1}{\kappa}\Big)\Big) = \kappa \cdot \frac{i}{i+1} \le \kappa.$$

For $\tau_i = \beta\sqrt{2\log(i+1)/d}$,

$$\exp\Big(-\frac{d\tau_i^2}{2\sigma^2}\Big) = (i+1)^{-\beta^2/\sigma^2},$$

so $\mathbb{E}[S_i] \le i(i+1)^{-\beta^2/\sigma^2} \le (i+1)^{1-\beta^2/\sigma^2}$. □

A.2 Proof of Corollary 4.4

Theorem A.5 (TRA is Non-Dispersive (restated Theorem 4.4))**.** *Under Assumption 4.1 and Assumption 4.2, the Threshold Rectified Attention (TRA) mechanism is non-dispersive.*

Proof. Let S_i be the number of non-zero entries (survivors) in the TRA weight vector $\boldsymbol{a}_i$. If $\boldsymbol{a}_i$ has support size S_i, then the effective entropy (defined on the ℓ_1-normalized absolute weights) is maximized by the uniform distribution on that support, hence

$$H(\boldsymbol{a}_i) \le \log(1 + S_i),$$

where the $+1$ handles the case $S_i = 0$.

Write $S_i = R_i + N_i$, where R_i is the number of non-noise (relevant) survivors and N_i is the number of noise survivors. By Assumption 4.2, $R_i \le r$ for all i, and by Theorem 4.3, if $\beta \ge \sigma$ then $\mathbb{E}[N_i] \le \kappa$. Applying Jensen's inequality (since $x \mapsto \log x$ is concave),

$$\mathbb{E}[H(\boldsymbol{a}_i)] \le \mathbb{E}[\log(1 + S_i)] \le \log\big(1 + \mathbb{E}[S_i]\big) \le \log\big(1 + r + \kappa\big).$$

We now take the limit of the dispersion ratio as context length $i \to \infty$:

$$\lim_{i\to\infty} \frac{\mathbb{E}[H(\boldsymbol{a}_i)]}{\log i} \le \lim_{i\to\infty} \frac{\log(1 + r + \kappa)}{\log i} = 0.$$

Thus, TRA is non-dispersive. □

A.3 Proof of Theorem 4.6

Theorem A.6 (Consensus spurious survivors vanish in TDA (restated Theorem 4.6))**.** *Under Assumptions 4.1 and 4.5 for both views (with the same σ). Set $\beta = \sigma$, and*

$$\tau_i := \beta\sqrt{\frac{2\log\big(\frac{i+1}{\kappa}\big)}{d}}, \qquad \kappa > 0,$$

and define the number of consensus *spurious survivors*

$$C_i := \sum_{j\in\mathcal{N}(i)} \mathbf{1}\Big(\boldsymbol{s}_{ij}^{(1)} > \tau_i\Big)\, \mathbf{1}\Big(\boldsymbol{s}_{ij}^{(2)} > \tau_i\Big).$$

Then for all $i \ge 1$, $\mathbb{E}[C_i] \le \kappa^2/(i+1)$, hence $\mathbb{E}[C_i] \to 0$ as $i \to \infty$.

Proof. Fix a row i and a noise key $j \in \mathcal{N}(i)$. By Assumption 4.1, each view is mean-zero and σ^2/d-sub-Gaussian, hence

$$\mathbb{P}\Big(\boldsymbol{s}_{ij}^{(t)} > \tau_i\Big) \le \exp\Big(-\frac{d\tau_i^2}{2\sigma^2}\Big), \qquad t \in \{1, 2\}.$$

By Assumption 4.5, for a noise key j we have independence across views, so

$$\mathbb{P}\Big(\boldsymbol{s}_{ij}^{(1)} > \tau_i,\ \boldsymbol{s}_{ij}^{(2)} > \tau_i\Big) = \mathbb{P}\Big(\boldsymbol{s}_{ij}^{(1)} > \tau_i\Big)\,\mathbb{P}\Big(\boldsymbol{s}_{ij}^{(2)} > \tau_i\Big) \le \exp\Big(-\frac{d\tau_i^2}{\sigma^2}\Big).$$

Taking expectation and using linearity,

$$\mathbb{E}[C_i] = \sum_{j\in\mathcal{N}(i)} \mathbb{P}\Big(\boldsymbol{s}_{ij}^{(1)} > \tau_i,\ \boldsymbol{s}_{ij}^{(2)} > \tau_i\Big) \le |\mathcal{N}(i)| \exp\Big(-\frac{d\tau_i^2}{\sigma^2}\Big) \le (i+1)\exp\Big(-\frac{d\tau_i^2}{\sigma^2}\Big).$$

With $\tau_i = \sigma\sqrt{2\log((i+1)/\kappa)/d}$, we have $\frac{d\tau_i^2}{\sigma^2} = 2\log((i+1)/\kappa)$, hence

$$\mathbb{E}[C_i] \le (i+1)\exp\Big(-2\log\Big(\frac{i+1}{\kappa}\Big)\Big) = (i+1)\Big(\frac{\kappa}{i+1}\Big)^2 = \frac{\kappa^2}{i+1}.$$

□

A.4 Proof of Theorem 4.7

Corollary A.7 (TDA is Non-Dispersive (restated Corollary 4.7))**.** *Under Assumption 4.1, Assumption 4.2, and Assumption 4.5 for both views (with the same σ), Threshold Differential Attention (TDA) is non-dispersive.*

Proof. TDA computes $\Delta\boldsymbol{a} = \boldsymbol{a}^{(1)} - \lambda\boldsymbol{a}^{(2)}$. If $\Delta\boldsymbol{a}_{ij} \neq 0$, then at least one of $\boldsymbol{a}_{ij}^{(1)}, \boldsymbol{a}_{ij}^{(2)}$ is non-zero, hence the support size satisfies

$$S_i^{\Delta} \leq S_i^{(1)} + S_i^{(2)}.$$

Write $S_i^{(t)} = R_i^{(t)} + N_i^{(t)}$, where $R_i^{(t)}$ is the number of non-noise (relevant) survivors and $N_i^{(t)}$ is the number of noise survivors in view t. By Assumption 4.2, $R_i^{(t)} \leq r$ for all i and $t \in \{1, 2\}$. By Theorem 4.3 (applied to each view) with $\beta \geq \sigma$, we have $\mathbb{E}[N_i^{(t)}] \leq \kappa$. Therefore,

$$\mathbb{E}[S_i^{\Delta}] \leq \mathbb{E}[S_i^{(1)}] + \mathbb{E}[S_i^{(2)}] \leq 2(r + \kappa).$$

Using the same entropy definition (on the ℓ_1-normalized absolute weights), we have

$$H(\Delta\boldsymbol{a}_i) \leq \log(1 + S_i^{\Delta}),$$

so by Jensen's inequality,

$$\mathbb{E}[H(\Delta\boldsymbol{a}_i)] \leq \mathbb{E}[\log(1 + S_i^{\Delta})] \leq \log\big(1 + \mathbb{E}[S_i^{\Delta}]\big) \leq \log\big(1 + 2(r + \kappa)\big).$$

Finally,

$$\lim_{i\to\infty} \frac{\mathbb{E}[H(\Delta\boldsymbol{a}_i)]}{\log i} \leq \lim_{i\to\infty} \frac{\log(1 + 2(r + \kappa))}{\log i} = 0.$$

□

Threshold Rectified Attention (Forward)

Require: $Q, K, V \in \mathbb{R}^{B\times H\times T\times D}$; threshold scale β; κ; power p; block sizes B_M, B_N
Ensure: $O \in \mathbb{R}^{B\times H\times T\times D}$

function ThresholdReLAForward($Q, K, V, \beta, p, B_M, B_N$)
  for $b \leftarrow 1$ to B **do**
    for $h \leftarrow 1$ to H **do**
      for $m \leftarrow 0$ to $T - 1$ step B_M **do** ▷ query block start
        $Q_m \leftarrow Q[b, h, m : m{+}B_M, :]$ ▷ $B_M \times D$
        Initialize $A \leftarrow \mathbf{0} \in \mathbb{R}^{B_M\times D}$ ▷ FP32 accumulator
        Compute per-row thresholds:
        for $i \leftarrow 0$ to $B_M - 1$ **do**
          $\tau_m[i] \leftarrow \beta\sqrt{\frac{2\log((m+i+1)/\kappa)}{D}}$
        end for
        for $n \leftarrow 0$ to $T - 1$ step B_N **do** ▷ key/value block start
          $K_n \leftarrow K[b, h, n : n{+}B_N, :]$ ▷ $B_N \times D$
          $V_n \leftarrow V[b, h, n : n{+}B_N, :]$ ▷ $B_N \times D$
          $S \leftarrow Q_m K_n^{\top}$ ▷ $B_M \times B_N$
          Apply causal mask (equivalent to the Triton mask):
            $S_{ij} \leftarrow 0$ **if** $(n + j) > (m + i)$
          $X \leftarrow S - \tau_m[:, \text{None}]$ ▷ broadcast τ over columns
          $R \leftarrow \max(X, 0)$ ▷ ReLU
          $W \leftarrow R^p$ ▷ elementwise power
          $A \leftarrow A + WV_n$ ▷ $(B_M \times B_N)(B_N \times D) \to (B_M \times D)$
        end for

$O[b, h, m : m{+}B_M, :] \leftarrow A$
end for
end for
end for
return O
end function

Threshold Rectified Attention (Backward)

Require: Saved $Q, K, V \in \mathbb{R}^{B\times H\times T\times D}$; upstream gradient dO; threshold scale β; κ; power p; block sizes B_M, B_N
Ensure: Gradients dQ, dK, dV with same shapes as Q, K, V
function THRESHOLDRELABACKWARD($Q, K, V, dO, \beta, p, B_M, B_N$)
Initialize $dQ \leftarrow \mathbf{0}$, $dK \leftarrow \mathbf{0}$, $dV \leftarrow \mathbf{0}$ ▷ accumulate in FP32
for $b \leftarrow 1$ to B **do**
for $h \leftarrow 1$ to H **do**
for $m \leftarrow 0$ to $T-1$ step B_M **do** ▷ query block
$Q_m \leftarrow Q[b, h, m : m{+}B_M, :]$ ▷ $B_M \times D$
$dO_m \leftarrow dO[b, h, m : m{+}B_M, :]$ ▷ $B_M \times D$
Compute thresholds for this query block:
for $i \leftarrow 0$ to $B_M - 1$ **do**
$\tau_m[i] \leftarrow \beta\sqrt{\frac{2\log((m+i+1)/\kappa)}{D}}$
end for
for $n \leftarrow 0$ to $T-1$ step B_N **do** ▷ key/value block
$K_n \leftarrow K[b, h, n : n{+}B_N, :]$ ▷ $B_N \times D$
$V_n \leftarrow V[b, h, n : n{+}B_N, :]$ ▷ $B_N \times D$
Recompute scores: $S \leftarrow Q_m K_n^\top$ ▷ $B_M \times B_N$
Apply causal mask: $S_{ij} \leftarrow 0$ if $(n + j) > (m + i)$
$X \leftarrow S - \tau_m[:, \text{None}]$
$R \leftarrow \max(X, 0)$
$W \leftarrow R^p$
Compute elementwise derivative:
$G \leftarrow p \cdot R^{p-1} \cdot \mathbb{I}[X > 0]$ ▷ $B_M \times B_N$
Gradient wrt weights (streaming):
$dW \leftarrow dO_m V_n^\top$ ▷ $(B_M \times D)(D \times B_N)$
$dS \leftarrow dW \odot G$
Accumulate dQ for this block:
$dQ[b, h, m : m{+}B_M, :] \mathrel{+}= dS K_n$ ▷ $(B_M \times B_N)(B_N \times D)$
Accumulate dV for this block:
$dV[b, h, n : n{+}B_N, :] \mathrel{+}= W^\top dO_m$ ▷ $(B_N \times B_M)(B_M \times D)$
Accumulate dK for this block:
$dK[b, h, n : n{+}B_N, :] \mathrel{+}= dS^\top Q_m$ ▷ $(B_N \times B_M)(B_M \times D)$
end for
end for
end for
end for
return dQ, dK, dV
end function

Threshold Differential Attention (TDA)

Require: $q_1, q_2, k_1, k_2, v \in \mathbb{R}^{B\times H\times T\times D}$; β; κ; λ; power p; `normalize` flag
Ensure: $O \in \mathbb{R}^{B\times H\times T\times D}$

function DIFFERENTIALTHRESHOLDRELA($q_1, q_2, k_1, k_2, v, \beta, \lambda, p,$ `normalize`)
  if `normalize` **then**
    $q_1 \leftarrow \mathrm{L2Normalize}(q_1)$; $k_1 \leftarrow \mathrm{L2Normalize}(k_1)$
    $q_2 \leftarrow \mathrm{L2Normalize}(q_2)$; $k_2 \leftarrow \mathrm{L2Normalize}(k_2)$
  end if
  $O_1 \leftarrow$ THRESHOLDRELAFORWARD($q_1, k_1, v, \beta, p, B_M, B_N$)
  $O_2 \leftarrow$ THRESHOLDRELAFORWARD($q_2, k_2, v, \beta, p, B_M, B_N$)
  $\lambda_c \leftarrow \min(1, \max(0, \lambda))$ ▷ clamp to $[0, 1]$
  $O \leftarrow O_1 - \lambda_c \cdot O_2$
  return O
end function

Layer	$\mathbb{P}(\lvert s\rvert > 2\sigma)$	$\mathbb{P}(\lvert s\rvert > 3\sigma)$	$\mathbb{P}(\lvert s\rvert > 4\sigma)$
0	0.0903	0.0423	0.0143
2	0.1145	0.0243	0.0062
4	0.0459	0.0193	0.0153
6	0.0606	0.0444	0.0181
8	0.0752	0.0142	0.0017
11	0.1574	0.0115	0.0003
$\mathcal{N}(0,1)$	0.0455	0.0027	6.33e-5

Table 7: Empirical tail probabilities $P(|s| > k\sigma)$ for raw logits $s_{ij} = q_i^\top k_j/\sqrt{d}$, where σ is the per-layer standard deviation.

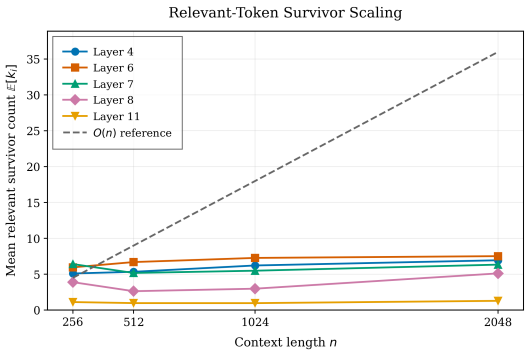


Figure 4: Relevant-token survivor scaling across context length for representative active layers.

B Empirical Validation of Assumptions

To assess whether the assumptions underlying our analysis are informative in trained models, we report three diagnostics on the trained TDA checkpoint: raw-logit tail behavior, relevant-token survivor scaling, and cross-view correlation. These diagnostics are not intended to verify the assumptions exactly; rather, they test whether the empirical behavior is broadly consistent with the theoretical picture used to motivate the length-dependent threshold and the differential construction.

B.1 Tail Behavior of Raw Logits

Our thresholding mechanism is motivated by an extreme-value theory under approximately sub-Gaussian noise (Assumption 4.1). To examine this, we analyze the raw logits before thresholding:

$$s_{ij} = q_i^\top k_j/\sqrt{d}.$$

Table 7 reports empirical tail probabilities $P(|s| > k\sigma)$ for representative layers, where σ denotes the per-layer standard deviation. While the trained logits are not exactly Gaussian, the empirical tail probabilities remain small and broadly consistent with quadratic-type decay rather than polynomial heavy-tailed behavior. We therefore interpret the sub-Gaussian assumption as an analytically useful approximation rather than an exact empirical model.

B.2 Relevant-Token Survivor Scaling

Our non-dispersion analysis assumes that the number of relevant tokens surviving the adaptive threshold remains bounded (Assumption 4.2). To examine this empirically, we compute the number of surviving keys per query while restricting attention to non-padding relevant tokens only, and evaluate this statistic for context lengths $n \in \{256, 512, 1024, 2048\}$.

Figure 4 shows the mean relevant-token survivor count as context length increases for representative active layers. Across these layers, survivor counts remain stable or grow only mildly relative to a linear $O(n)$ reference. For example, in Layer 4, the mean relevant-token survivor count increases from 5.105 at $n = 256$ to 6.949 at $n = 2048$, corresponding to only a $1.36\times$ increase as the context length grows by $8\times$. Similar behavior is observed in several other active layers. This supports the bounded-relevant-survivor assumption used in the non-dispersion analysis.

B.3 Cross-View Correlation

A key concern for differential constructions is whether the two views collapse into a redundant branch after training, which violates the independence of noise views assumption (Assumption 4.5). To assess this, we measure the Pearson correlation between the two thresholded score views and compare trained and untrained checkpoints.

Figure 5 compares layer-wise mean cross-view correlation for trained and untrained models. The trained model exhibits a modest increase in correlation relative to the untrained model, but the correlations remain low overall: the mean[3] over layer-wise values is 0.1231 for the trained checkpoint, compared with 0.0752 for the untrained checkpoint. This indicates that the two views do not collapse into a redundant copy after training.

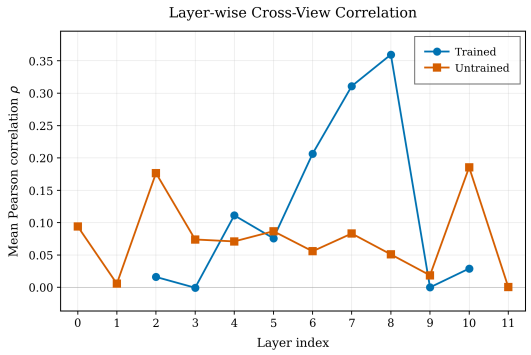


Figure 5: Layer-wise mean cross-view Pearson correlation for trained and untrained models.

C Triton Kernel for TRA and TDA

Problem setting. Given queries Q, keys K, values $V \in \mathbb{R}^{B\times H\times T\times D}$ (batch B, heads H, sequence length T, head dimension D), Threshold Rectified Attention (TRA) computes:

$$\boldsymbol{O} = \Big((\boldsymbol{Q}\boldsymbol{K}^\top - \tau)_+\Big)^p \boldsymbol{V},$$
$$\tau(i) = \beta\sqrt{\frac{2\log((i+1)/\kappa)}{D}},$$

where i is the (0-indexed) query position, β is a scalar threshold scale, κ controls the expected number of spurious survivor, and p is the power. We use causal masking, i.e., query position i can attend only to key positions $j \le i$.

We write $x_+ = \max(x, 0)$ elementwise. Tiling parameters are B_M (query block size) and B_N (key/value block size). The implementation is streaming, similar to FlashAttention (Dao et al., 2022; Dao, 2024): it never materializes the full $T \times T$ attention matrix.

Setup. We additionally benchmark end-to-end *forward+backward* time and *peak* GPU memory for a single causal attention module at sequence lengths $T \in \{512, 1024, 2048, 4096\}$ under **FP32** settings, comparing:

1. dense Softmax attention implemented with standard PyTorch operation;

2. a naive PyTorch implementation of TRA that also materializes scores;

3. PyTorch fused SDPA (`torch.nn.functional.scaled_dot_product_attention`) with Softmax;

4. our fused Triton implementation of TRA.

Our Triton kernel is streaming in the style of FlashAttention (Dao et al., 2022; Dao, 2024): it computes $\boldsymbol{Q}\boldsymbol{K}^\top$ in tiles, applies the causal mask and the row-dependent threshold τ_i, and directly accumulates $\sum_j \boldsymbol{w}_{ij}\boldsymbol{v}_j$ without storing the dense attention matrix, avoiding both the softmax reduction and quadratic activation storage.

Results. As shown in Figure 6, the naive PyTorch TRA is not competitive: despite using cheaper nonlinearities than softmax, it still pays the full cost of dense score materialization and unfused elementwise ops. In contrast, the fused Triton kernel is consistently the fastest option across all tested lengths and improves with sequence length: at T=4096 it reduces end-to-end time by several $\times$ compared to dense PyTorch attention and also outperforms fused SDPA.

Memory gains are even more pronounced: while dense PyTorch implementations scale quadratically and reach multi-GB peak usage at T=4096, our Triton kernel matches the FlashAttention-style $O(Td)$ activation footprint, staying in the tens-of-MB regime. This efficiency is crucial for training TDA at long context, where sparsity is only useful if the implementation avoids allocating dense attention tensors.

D Details in Passkey Retrieval Tests

Each trial constructs a single prompt consisting of five blocks separated by newline characters:

1. **Task description (instruction).** A fixed instruction string:

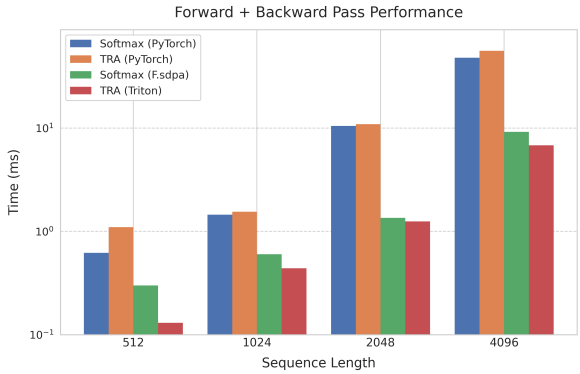
   > There is an important info hidden inside a lot of irrelevant text. Find it and memorize them. I will quiz you about the important information there.

2. **Garbage prefix.** A prefix of length $n_{\text{garbage\_prefix}}$ characters cut from a long "garbage" string:

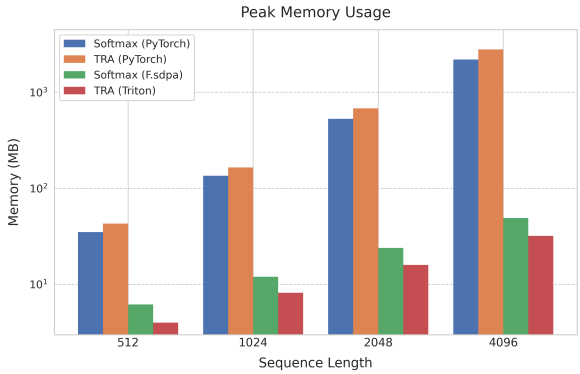
   > The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again.

   This base sentence is repeated many times to form a long buffer, and we take a random-length prefix substring.

[3] Some trained layers yield undefined correlation due to one view being nearly constant (almost all zeros). We omit those layers from the average.



(a) End-to-end forward+backward latency vs. sequence length.



(b) Peak GPU memory during forward+backward.

Figure 6: **Runtime and memory of fused Triton TRA.** We compare dense Softmax (PyTorch), naive TRA (PyTorch), fused SDPA, and our fused Triton TRA kernel under FP32.

3. **Passkey line (needle).** A single line containing the passkey, where the key is repeated to reduce ambiguity:

   > The pass key is $\{x\}$. Remember it. $\{x\}$ is the pass key.

   The passkey x is sampled uniformly as an integer $x \sim \mathrm{Unif}\{1, \ldots, 50000\}$.

4. **Garbage suffix.** A suffix of length $n_{\text{garbage\_suffix}}$ characters cut from the same repeated garbage buffer, where $n_{\text{garbage\_prefix}} + n_{\text{garbage\_suffix}} = n_{\text{garbage}}$.

5. **Final query.** A fixed question that prompts the model to output the passkey:

   > What is the pass key? The pass key is

The final prompt is the newline-joined concatenation of these blocks:

$$\texttt{prompt} = \texttt{instr} \,\|\, \texttt{garbage\_prefix} \,\|\, \texttt{needle} \,\|\, \texttt{garbage\_suffix} \,\|\, \texttt{query},$$

where $\|$ denotes concatenation with newline separators.

For each target length $T_{\text{target}} \in \{500, 1000, \ldots, 4000\}$, we construct a prompt by inserting a numeric passkey statement at a random location within a long span of irrelevant "garbage" text. We run 100 trials per length with independently sampled passkeys and insertion positions, use greedy decoding, and count a trial as correct if the generated answer matches the ground-truth passkey (exact string match after stripping whitespace, or integer match). This follows the standard needle-in-a-haystack evaluation protocol (Mohtashami and Jaggi, 2023; Kamradt, 2023).

E Further Experimental Details

E.1 Sparsity

We quantify sparsity by the fraction of attention entries that are *exactly zero*. For a given layer ℓ and head h, let $\mathbf{A}^{\ell,h} \in \mathbb{R}^{T\times T}$ denote the (masked) attention weight matrix produced by the mechanism, where causal masking enforces $\mathbf{A}^{\ell,h}_{ij} = 0$ for $j > i$. We call an entry *inactive* if $\mathbf{A}^{\ell,h}_{ij} = 0$ exactly, and define the per-head sparsity for the causal-masked transformer as

$$\mathrm{Sparsity}^{\ell,h} := \frac{\sum_{i=1}^{T}\sum_{j=1}^{T} \mathbf{1}\left[\mathbf{A}^{\ell,h}_{ij} = 0\right]}{\sum_{i=1}^{T}\sum_{j=1}^{T} \mathbf{1}[j \le i]}.$$

We then aggregate across heads and layers by averaging:

$$\mathrm{Sparsity} := \frac{1}{LH}\sum_{\ell=1}^{L}\sum_{h=1}^{H} \mathrm{Sparsity}^{\ell,h}.$$

E.2 Dataset Statistics

We report the statistics for the datasets used in our standard and long-context evaluations in Table 8 and Table 3.

For the standard language modeling evaluation, we use the following zero-shot benchmarks. These datasets typically consist of short contexts (questions or partial sentences) suitable for testing core reasoning capabilities.

For long-context evaluation, we select four diverse tasks from the SCROLLS benchmark (Shaham et al., 2022). These datasets feature input lengths significantly exceeding the training context of standard models, requiring the model to

Dataset	Train	Validation	Test	Task Type
HellaSwag	39,905	10,042	10,003	Commonsense completion
ARC-Easy	2,251	570	2,376	Multiple-choice science QA
ARC-Challenge	1,119	299	1,172	Multiple-choice science QA
OpenBookQA	4,957	500	500	Multiple-choice QA
PIQA	16,113	1,838	3,084	Physical commonsense QA
WinoGrande	40,398	1,267	1,767	Coreference resolution

Table 8: Statistics for language modelling benchmarks.

Dataset	Domain	Train	Validation	Test	Avg. Length (Words)
QMSum	Meetings	1,257	272	281	9,497
SummScreenFD	Screenplays	3,673	338	337	5,598
GovReport	Government	17,457	972	973	7,886
Qasper	Scientific papers	2,567	1,726	1,399	3,629

Table 9: Statistics for SCROLLS benchmarks used in long-context evaluation. Avg. length is reported in words.

effectively extrapolate or manage long-range dependencies.

E.3 Training Configuration

Training For standard configurations (GPT-2-162M), we use a total batch size of 524,288 tokens per gradient update step, with a mini-batch size of 16 sequences per GPU and a context length of 1024 tokens. Gradient accumulation steps are automatically calculated to achieve the desired total batch size across all GPUs. The dataset is tokenized using the GPT-2 tokenizer (tiktoken encoding) with a vocabulary size of 50,304 tokens.

All models are trained for 5 epochs with 38,146 steps per epoch, resulting in approximately 190,730 total training steps. We evaluate on the validation set every 250 steps, using 20 validation batches per evaluation.

Hyperparameters. We employ a learning rate schedule with linear warmup followed by cosine decay. The maximum learning rate is set to 1×10^{-3}, with a minimum learning rate of 1×10^{-4}. Warmup is performed over 715 steps. We use a weight decay of 0.1 and set the random seed to 1337 for reproducibility. For models with RoPE positional encoding, we use a base frequency $\theta = 10,000$. When extending models to longer context lengths, we employ NTK-aware scaling for RoPE-based models Peng et al. (2024); bloc97 (2023), and further finetune them for 500 additional steps. We report single-run results for all evaluations, as the variance is small.

E.4 Hardware and Infrastructure

All experiments were conducted on NVIDIA A100-80GB GPUs. With Triton kernel optimizations enabled for threshold-based attention mechanisms, memory usage per GPU was approximately 35–45GB.

F Use of Large Language Model

We used large language models (LLMs) to assist with writing by refining human-written text and to support code development.