ON STATE REDUCTION IN LINEAR ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Linear attention offers a computationally efficient yet expressive alternative to softmax attention. However, recent empirical results indicate that the hidden state of trained linear attention models often exhibits a low-rank structure, suggesting that these models underexploit their capacity in practice. To understand this phenomenon, we analyze how the keys and values shape the rank of the recurrent state, providing a theoretical perspective on memory utilization in linear attention. In addition to these theoretical insights, we conjecture that the low-rank states can be substantially reduced post-training with only minimal performance degradation. To this end, we propose a hardware-aware approach that structurally prunes key and query matrices, reducing the state size while retaining compatibility with existing kernels. We adapt several existing pruning strategies to fit our framework and, building on our theoretical analysis, propose a robust structured pruning method based on a rank-revealing QR decomposition. Our empirical evaluations across models of varying sizes and on various downstream tasks, reveal that the state of pretrained linear attention models can be halved with only a minor decrease of next token prediction capabilities.

1 INTRODUCTION

Linear Attention (Katharopoulos et al., 2020; Schlag et al., 2021; Sun et al., 2023; Peng et al., 2023; Gu & Dao, 2024; Yang et al., 2024; Dao & Gu, 2024; Yang et al., 2025a; Team et al., 2025) has emerged as an efficient alternative to softmax attention (Vaswani et al., 2017), enabling high-throughput chunkwise parallel training with linear time complexity and constant memory inference (Hua et al., 2022; Lingle, 2024; Yang et al., 2023). These efficiency gains have driven the development of large hybrid models (Blakeman et al., 2025; Qwen Team, 2026) which predominantly employ linear attention layers, interspersed with only a few softmax attention layers.

Despite their impressive performance, prior work indicates that linear attention models still underutilize their capacity in practice (Siems et al., 2026; Parnichkun et al., 2025). In particular, their matrix-valued hidden states exhibit a low-rank structure after training. This state acts, through the lens of linear associative memories (Ramsauer et al., 2020; Wang et al., 2025), as a storage of sequence history. Thus, low effective rank indicates that the model might be using its memory inefficiently, effectively wasting capacity. In this work, we first provide a theoretical perspective on this phenomenon. We then investigate whether the hidden state of pretrained linear attention models can be reduced to improve inference speed and memory efficiency.

Towards this end, we propose a structured pruning framework that relies on the observation that linear attention models are invariant under orthogonal transformations applied jointly to the queries and keys. Since the per-channel convolutions typically employed in efficient recurrent models are not compatible with rotations (Chollet, 2017; So et al., 2021; Fu et al., 2022), we restrict the space of applicable transformations to key-channel selection methods. Within this framework, we provide a reformulation of several established pruning strategies, including those based on parameter magnitude and gradient saliency, to reduce the dimension of the hidden state. In addition, motivated by theoretical insights into the role of rank in linear attention, we introduce a novel structured pruning approach that targets higher state rank utilization through rank-revealing channel selection.

Our results show that approximately 50% of the key and query channels can often be removed with only modest degradation on language-modeling evaluations, even before recovery fine-tuning (Hu et al., 2021; Ma et al., 2023; Ashkboos et al., 2024a). In addition, our long-context results further

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

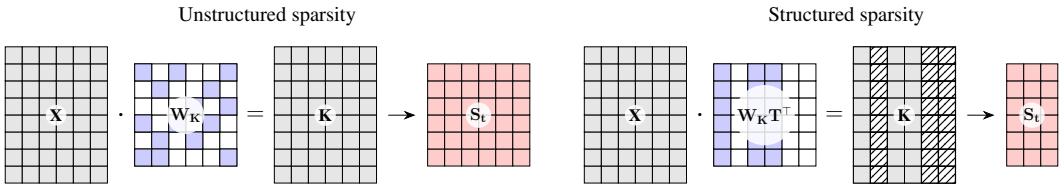


Figure 1: **Left:** Unstructured pruning yields a sparse weight matrix $\mathbf{W_K}$ yet preserves the column dimension of $\mathbf{K}$, leaving the domain of the state matrix $\mathbf{S}_t \in \mathbb{R}^{d_v, d_k}$ invariant. **Right:** Structured pruning eliminates basis vectors, mapping keys to a lower-dimensional space $\mathbb{R}^{d'_k}$ where $d'_k < d_k$. This results in a compressed state $\mathbf{S}_t \in \mathbb{R}^{d_v, d'_k}$. Figure inspired by Ashkboos et al. (2024a, Figure 1).

confirm a known weakness of linear attention models: purely recurrent models struggle with long horizon recall (Arora et al., 2024a;b), whereas the hybrid Qwen3.5 9B (Qwen Team, 2026) preserves retrieval, potentially because its attention layers retain an explicit long-context path.

In summary, our **main contributions** are:

- We provide theoretical insights into the role of rank in linear attention (Section 2.1). In particular, we show that rank utilization affects readout error and can amplify query noise.
- Motivated by the low rank utilization observed in practice, we formulate a post-training state-size reduction framework (Section 3). Specifically, we show that the state can be reduced substantially by jointly selecting subsets of channels from the keys and queries.
- We adapt several existing pruning strategies to fit our framework and, building on our theoretical analysis, propose a structured pruning method based on rank-revealing QR decompositions (Section 3.3).
- We demonstrate state reduction across recurrent models from 370M to 7B parameters and a 9B hybrid model, including end-to-end efficiency and long-context retrieval evaluations (Section 5).

2 Theoretical Insights

This section establishes the theoretical foundations for our analysis. We begin by introducing the linear attention mechanism and its interpretation as an associative memory, before developing a framework to characterize the effect of rank inefficiencies within this setting.

2.1 Background on Linear Attention

In its simplest form, linear attention (Katharopoulos et al., 2020; Schlag et al., 2021) arises by replacing the exponential kernel in self attention with a linear one and discarding the normalization (Vaswani et al., 2017). Let $\mathbf{X} \in \mathbb{R}^{T,h}$ be an input sequence that gets mapped to queries $\mathbf{Q} = \mathbf{X}\mathbf{W_Q} \in \mathbb{R}^{T,d_k}$, keys $\mathbf{K} = \mathbf{X}\mathbf{W_K} \in \mathbb{R}^{T,d_k}$, and values $\mathbf{V} = \mathbf{X}\mathbf{W}_v \in \mathbb{R}^{T,d_v}$ via linear projections. By the associativity of matrix multiplication, linear attention can then be expressed recurrently as

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{v}_t\mathbf{k}_t^\top, \tag{1}$$

where $\mathbf{S}_t \in \mathbb{R}^{d_v,d_k}$ is a matrix-valued hidden state used to compute the output $\mathbf{o}_t = \mathbf{S}_t\mathbf{q}_t$, and $\mathbf{S}_0 = \mathbf{0}$. Katharopoulos et al. (2020) provide a detailed derivation. The state $\mathbf{S}_t$ implements a linear associative memory (Ramsauer et al., 2020; Wang et al., 2025) that is addressed using the queries. Throughout this work, we will refer to the update rule of $\mathbf{S}_t$ as the sequence mixer.

The mechanism defined in Equation (1) can only write into the associative memory. DeltaNet (Schlag et al., 2021; Yang et al., 2024) addresses this limitation by explicitly erasing old information correlated with the current key before writing the new value to $\mathbf{S}_t$,

$$\mathbf{S}_t = \mathbf{S}_{t-1}(\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^\top) + \beta_t\mathbf{v}_t\mathbf{k}_t^\top. \tag{2}$$

Gated DeltaNet (Yang et al., 2025a) further refines this mechanism by introducing a data-dependent decay term.

2.2 On the Role of the Rank

Linear attention compresses the sequence history into a fixed-size hidden state, which generally entails a loss of information. This compression is commonly viewed as a limitation relative to softmax attention, whose growing KV cache allows each query to directly attend to individual past tokens. However, recent empirical results indicate that pretrained linear attention models don't manage this memory well (Siems et al., 2026; Parnichkun et al., 2025), exhibiting an effective low-rank structure in practice. We confirm this finding for DeltaNet 370M. Its hidden state has a characteristic tail (Figure 2). In the remainder of this section, we develop a framework that illuminates this phenomenon.

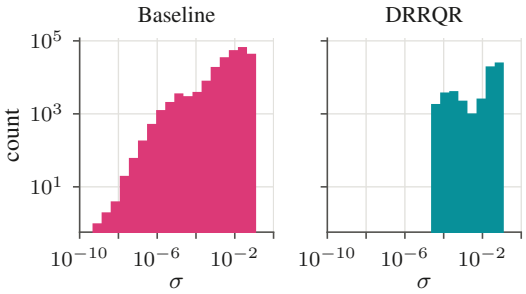


Figure 2: Singular value spectra for a DeltaNet 370M head (FineWeb, $T = 2048$, first 128 tokens skipped). **Left:** Uncompressed baseline. **Right:** DRRQR at 75% compression (pre-RFT).

By the nature of linear attention, the subspace spanned by the associative memory is governed by the keys and values. Formally, for a family of linear attention models, the row and column spaces of the associative memory satisfy

$$\operatorname{row} \mathbf{S}_t \subseteq \operatorname{span}(\mathbf{k}_1, \ldots, \mathbf{k}_t), \operatorname{col} \mathbf{S}_t \subseteq \operatorname{span}(\mathbf{v}_1, \ldots, \mathbf{v}_t).$$

It thus follows that

$$\operatorname{rank} \mathbf{S}_t \leq \min\left(\operatorname{rank} \mathbf{K}_t, \operatorname{rank} \mathbf{V}_t\right) \leq t, \tag{3}$$

where $\mathbf{K}_t = [\mathbf{k}_1, \ldots, \mathbf{k}_t]$ and $\mathbf{V}_t = [\mathbf{v}_1, \ldots, \mathbf{v}_t]$ are the matrices obtained by stacking the keys and values, respectively. The proof is presented in Appendix F.2.

Equation (3) shows that the algebraic rank of the associative memory is bounded by the algebraic ranks of the keys and values. However, algebraic rank is too rigid of a metric for noisy real-world data. For a non-zero matrix $\mathbf{S} \in \mathbb{R}^{d_v, d_k}$, we therefore use its *effective rank* (or *stable rank*) (Rudelson & Vershynin, 2007; Tropp, 2015; Vershynin, 2019; Ipsen & Saibaba, 2025), defined as

$$\operatorname{er}(\mathbf{S}) \coloneqq \frac{\|\mathbf{S}\|_F^2}{\|\mathbf{S}\|_2^2} = \frac{\sum_i \sigma_i^2}{\sigma_1^2},$$

where $\sigma_1 \geq \ldots \geq \sigma_{\min(d_v, d_k)} \geq 0$ are the singular values of $\mathbf{S}$. Intuitively, $\operatorname{er}(\mathbf{S})$ measures the skewness of the singular-value spectrum of $\mathbf{S}$.

An effective rank analogue of the bound in Equation (3) relates the effective rank of $\mathbf{S}$ to the conditioning of the keys. In particular, for the linear attention recurrence $\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top$, there exists a scalar quantity $\nu(\mathbf{V}_t)$ such that

$$\frac{\nu(\mathbf{V}_t)}{\kappa^2(\mathbf{K}_t)} \leq \operatorname{er}(\mathbf{S}_t), \tag{4}$$

where κ denotes the l^2 condition number. Appendix F.1 proves this bound. It extends to DeltaNet after a simple reparameterization. Define the pseudo-value

$$\mathbf{u}_t \coloneqq \beta_t(\mathbf{v}_t - \mathbf{S}_{t-1}\mathbf{k}_t).$$

Then Equation (2) becomes $\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{u}_t \mathbf{k}_t^\top$. Equation (4) therefore holds with $\nu(\mathbf{U}_t)$ in place of $\nu(\mathbf{V}_t)$. This bound motivates considering key conditioning when selecting channels.

2.2.1 Rank Utilization

While the effective rank measures the effective dimensionality of the state matrix, it does not capture how efficiently a recurrent model uses its available memory. For a non-zero state $\mathbf{S} \in \mathbb{R}^{d_v, d_k}$ with maximum capacity $d \coloneqq \min(d_k, d_v)$, we define its *rank utilization* as

$$\mathrm{u}(\mathbf{S}) \coloneqq \operatorname{er}(\mathbf{S})/d. \tag{5}$$

Rank utilization sets effective rank in relation to the theoretical maximum capacity and thus serves as a measure for memory utilization in linear attention models. We identify two edge cases:

- **Low Utilization** ($\mathrm{u} \ll 1$)**:** The memory suffers from rank collapse. Its energy is concentrated in a few principal components. Information is stored in an effectively low-dimensional subspace.
- **High Utilization** ($\mathrm{u} = 1$)**:** The memory is isotropic. Energy is distributed evenly across all dimensions.

Dividing Equation (4) by d yields

$$\frac{\nu(\mathbf{V}_t)}{d\kappa^2(\mathbf{K}_t)} \leq \mathrm{u}(\mathbf{S}_t),$$

which establishes a connection between the conditioning of the keys and the rank utilization of the corresponding memory.

2.2.2 Why Does Rank Utilization Matter?

The previous sections introduced effective rank and rank utilization as measures of the isotropy of an associative memory. We now connect these measures to noise sensitivity during readout. Intuitively, a memory with a skewed singular spectrum responds more strongly to some directions than to others. As a result, a small relative change in the query can lead to a larger relative change in the output.

To formalize this phenomenon, consider a perturbed query $\tilde{\mathbf{q}} = \mathbf{q}^* + \boldsymbol{\epsilon}$, $\|\mathbf{q}^*\|_2 = 1$, with outputs $\mathbf{o}^* = \mathbf{S}\mathbf{q}^*$ and $\mathbf{o} = \mathbf{S}\tilde{\mathbf{q}}$, where we drop the time index for brevity. Define the relative error amplification

$$R(\mathbf{q}^*, \boldsymbol{\epsilon}) \coloneqq \frac{\|\mathbf{o} - \mathbf{o}^*\|_2 / \|\mathbf{o}^*\|_2}{\|\boldsymbol{\epsilon}\|_2 / \|\mathbf{q}^*\|_2}. \tag{6}$$

Values above one indicate amplification, while values below one indicate attenuation of the query noise. The amplification ratio satisfies

$$R(\mathbf{q}^*, \boldsymbol{\epsilon}) = \frac{\|\mathbf{S}\boldsymbol{\epsilon}\|_2 / \|\boldsymbol{\epsilon}\|_2}{\|\mathbf{S}\mathbf{q}^*\|_2} \leq \frac{\sigma_1(\mathbf{S})}{\|\mathbf{S}\mathbf{q}^*\|_2}.$$

This bound is attained when the perturbation $\boldsymbol{\epsilon}$ follows the strongest right singular direction of the memory $\mathbf{S}$. Its denominator is smallest when the query follows the weakest right singular direction, giving $\|\mathbf{S}\mathbf{q}^*\|_2 = \sigma_{d_k}(\mathbf{S})$. The worst case combines a strong response to noise with a weak query signal:

Proposition 2.1 (Rank Utilization and Readout Sensitivity)**.** *Let $\mathbf{S} \in \mathbb{R}^{d_v, d_k}$ have full column rank. The worst-case relative amplification satisfies*

$$\max_{\|\mathbf{q}^*\|_2 = 1} \max_{\boldsymbol{\epsilon} \neq \mathbf{0}} R(\mathbf{q}^*, \boldsymbol{\epsilon}) = \kappa(\mathbf{S}) \geq \frac{1}{\sqrt{u(\mathbf{S})}}. \tag{7}$$

The proof of this statement is provided in Appendix F.3. The argument shows that amplification error is governed by the alignment between the state and the noise compared to the query. The proposition then shows that low rank utilization leads to large worst-case relative error amplification.

We distinguish two regimes:

- **Low Utilization** ($u(\mathbf{S}) \ll 1$)**:** The memory's skewed singular value spectrum permits large error amplification when noise follows a strong singular direction and the query follows a weak one.
- **High Utilization** ($u(\mathbf{S}) = 1$)**:** The memory becomes isotropic. All dimensions carry approximately the same energy and relative error is preserved.

Beyond amplifying readout error, low rank utilization also adversely affects the conditioning of the query weight gradients during the backward pass. See Appendix F.4 for details.

3 The Proposed Pruning Framework

Motivated by the above insights into the role of rank in linear attention, we propose a structured pruning framework to reduce the per-head key dimension of linear attention models. Besides revealing redundancy in pretrained linear attention models, our pruning strategy yields a strictly smaller hidden state with larger throughput.

3.1 Invariant Transformations for Linear Attention

Decomposing $\mathbf{S}_t$ row-wise reveals that the sequence mixer simulates d_v parallel linear time-varying (LTV) dynamical systems. Indeed, let $\mathbf{h}_t^{(i)} \in \mathbb{R}^{d_k}$ denote the transpose of the i-th row of $\mathbf{S}_t$. Each value channel $i \in \{1, \dots, d_v\}$ follows the vector-valued dynamics

$$\mathbf{h}_t^{(i)} = \mathbf{A}_t \mathbf{h}_{t-1}^{(i)} + \mathbf{B}_t v_{t,i}, \tag{8}$$

where the system matrices $\mathbf{A}_t = \mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top$ and $\mathbf{B}_t = \beta_t \mathbf{k}_t$ are shared across all value channels. The readout is computed via the vector $\mathbf{C}_t = \mathbf{q}_t^\top$.

The dynamical system in Equation (8) is invariant under the choice of basis in state-space (Chahine et al., 2026; Chen, 1984). That is, every transformation

$$(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t) \to (\mathbf{T}\mathbf{A}_t\mathbf{T}^{-1}, \mathbf{T}\mathbf{B}_t, \mathbf{C}_t\mathbf{T}^{-1})$$

derived from an invertible matrix $\mathbf{T} \in GL(d_k)$ leaves the input-output mapping invariant. If the transformed system should also be expressible in the same functional form as the original recurrence, requiring invertibility is not sufficient. Indeed, the transformed transition matrix

$$\tilde{\mathbf{A}}_t = \mathbf{T}\mathbf{A}_t\mathbf{T}^{-1} = \mathbf{I} - \beta_t(\mathbf{T}\mathbf{k}_t)(\mathbf{T}^{-\top}\mathbf{k}_t)^\top,$$

is generally not symmetric and can thus not be expressed as a DeltaNet. If additionally $\mathbf{T} \in O(d_k)$ is an orthogonal change of basis, this transformation can be absorbed to preserve the structure of the recurrence:

Proposition 3.1 (Orthogonal Invariance of Sequence Mixing)**.** *Let $\mathbf{T} \in O(d_k)$ be an orthogonal matrix. Then the (Gated) DeltaNet attention mechanism is invariant under the* simultaneous *transformation $(\mathbf{k}_t, \mathbf{q}_t) \mapsto (\mathbf{T}\mathbf{k}_t, \mathbf{T}\mathbf{q}_t)$.*

Our structured pruning framework, presented in the subsequent section, makes use of this result by applying a semi-orthogonal transformation jointly to queries and keys.

3.2 State Size Reduction Requires Structured Pruning

Existing pruning methods for Large Language Models generally rely on unstructured or semi-structured sparsity (Frantar & Alistarh, 2023; Sun et al., 2024). However, these methods do not reduce the state dimension of the dynamical system, as they leave the query and key vectors dense (see Figure 1). To address this issue, we focus on structured pruning.

Formally, given a semi-orthogonal matrix $\mathbf{T} \in \mathbb{R}^{d_k', d_k}$ with target dimension $d_k' < d_k$, we apply it jointly to the keys and queries before computing the dynamical system:

$$(\mathbf{k}_t, \mathbf{q}_t) \to (\mathbf{T}\mathbf{k}_t, \mathbf{T}\mathbf{q}_t) \quad \forall t = 1, \dots, T.$$

By Proposition 3.1, if $d_k' = d_k$, this transformation yields an equivalent realization of the dynamical system. Our goal is finding a $\mathbf{T}$ with $d_k' \ll d_k$ that entails a small error. A natural framework for deriving such a transformation would be Principal Component Analysis (PCA). By computing the empirical covariance $\hat{\mathbf{\Sigma}} = \frac{1}{N}\sum_t \mathbf{k}_t\mathbf{k}_t^\top$ over a calibration set, one could derive a projection matrix $\mathbf{T}$ from the top-d_k' principal components. However, while theoretically optimal for key-reconstruction (Eckart & Young, 1936; Mirsky, 1960), this approach is not compatible with the causal convolutions employed in linear attention architectures:

The Convolution Constraint. To achieve actual wall-clock speedup, we aim to absorb $\mathbf{T}$ into the weights $\mathbf{W_K}$ and $\mathbf{W_Q}$. However, linear attention models typically apply causal convolutions (Chollet, 2017; So et al., 2021; Fu et al., 2022; Yang et al., 2023; Gu & Dao, 2024; Dao & Gu, 2024; Yang et al., 2024) to queries and keys. While the sequence mixer is compatible with orthogonal transformations (Proposition 3.1), depthwise convolutions are not. A dense semi-orthogonal matrix $\mathbf{T}$ (such as one derived from PCA) would mix channels, misaligning them with their corresponding convolution filters. Preserving the dynamics under such rotations would require converting the efficient diagonal convolution filters into expensive dense matrices. Therefore, we effectively constrain our framework to axis-aligned transformations. The case of general orthogonal matrices is discussed in Appendix B.3.

3.3 AXIS-ALIGNED METHODS

To circumvent the aforementioned intricacies, we focus on transformations that preserve the channel-wise convolution filters. Formally, this restricts transformations to axis-aligned semi-orthogonal matrices: Let $\mathcal{I} = \{i_1, \ldots, i_{d'_k}\} \subseteq \{1, \ldots, d_k\}$ be a set of distinct indices with cardinality $d'_k < d_k$. We define the corresponding axis-aligned projection $\mathbf{P}_\mathcal{I} \in \mathbb{R}^{d'_k, d_k}$ by selecting the associated standard basis vectors as rows:

$$\mathbf{P}_\mathcal{I} \coloneqq \begin{bmatrix} \mathbf{e}_{i_1} & \cdots & \mathbf{e}_{i_{d'_k}} \end{bmatrix}^\top.$$

This matrix is semi-orthogonal. Applying it corresponds to a structural pruning operation that selects the subset of key-channels $\mathcal{I}$ and discards the rest. Crucially, this operation preserves the independence of the remaining channels and can be obtained by slicing the corresponding columns from the projection matrices $\mathbf{W_K}$ and $\mathbf{W_Q}$, along with the corresponding entries of the convolution filters.

In the following paragraphs, we present a plethora of different strategies to select the index set $\mathcal{I}$ for each head, ranging from simple magnitude-based heuristics to a novel rank-revealing approach. By Proposition 3.1, every transformation must be applied to the queries and keys simultaneously. Consequently, the proposed column selection mechanisms incorporate both queries and keys.

3.3.1 PROPOSED PRUNING METHODS

Weight-Magnitude-based ($\mathbf{L^1}$). As a baseline, we implement a weight-magnitude based pruner, defining the importance of the j-th key channel as the sum of the l^1 norms of the corresponding columns in the query and key projection matrices:

$$s_j = \|\mathbf{W}_{\mathbf{Q},:j}\|_1 + \|\mathbf{W}_{\mathbf{K},:j}\|_1.$$

We rank these scores locally within each head and select the top-d'_k indices to form the retained set $\mathcal{I}$.

S-Wanda. We adapt Wanda (Sun et al., 2024) to our structured setting (coining the method S-Wanda), and define the saliency of the j-th channel by aggregating the element-wise Wanda scores across the channel dimension:

$$s_j = \sum_{i=1}^{h} \left(|\mathbf{W}_{q,ij}| + |\mathbf{W}_{k,ij}|\right) \|\mathbf{X}_{:i}\|_2,$$

where $\|\mathbf{X}_{:i}\|_2$ is the l^2 norm of the i-th input feature computed over a calibration set. Unlike L^1, this metric accounts for the distribution of the input.

Sensitivity-based. We employ a gradient-based saliency criterion (LeCun et al., 1989; Wang et al., 2019; Ma et al., 2023) to identify dimensions that maximally influence the training objective. It is our only method that takes into account information beyond the current layer and is task-aware. We quantify the importance of the j-th key dimension using the first-order Taylor expansion of the loss on a calibration set:

$$s_j = \sum_i \left(\left|\mathbf{W}_{\mathbf{Q},ij} \nabla_{\mathbf{W}_{\mathbf{Q},ij}} \mathcal{L}\right| + \left|\mathbf{W}_{\mathbf{K},ij} \nabla_{\mathbf{W}_{\mathbf{K},ij}} \mathcal{L}\right|\right).$$

Rank-based (DRRQR). Motivated by the bound in Equation (4), we propose a simple algorithm that uses conditioning to guide channel selection. *Deep Rank Revealing QR (DRRQR)* applies a Strong Rank Revealing QR factorization (Gu & Eisenstat, 1996) to the activation statistics $\mathbf{M} = [\mathbf{K}; \mathbf{Q}]$ to select a subset of d'_k feature channels. Strong RRQR selects feature channels with a provable bound on their condition number. The algorithm (see Algorithm 2) starts from a QR decomposition with column pivoting and then iteratively permutes columns between a current chosen basis and a set of candidates to satisfy numerical stability bounds. We detail the specific swap-gain metric, update rules, and conditioning bound in Appendix B.2. Unlike heuristic approaches, DRRQR relies on deterministic guarantees for rank revelation of the joint key-query activation matrix, rendering the method tractable and interpretable.

4 RELATED WORK

Rank Considerations. Recent empirical results show that the memory of linear attention models often exhibits a low-rank structure (Siems et al., 2026; Parnichkun et al., 2025), indicating that they

do not manage their associative memory well. Our work builds on these observations, forcing the model to operate in a lower-dimensional space with increased rank utilization and a minimal decrease in effective rank (Ipsen & Saibaba, 2025).

Rank collapse is a well-known phenomenon in transformers. Dong et al. (2021), for instance, show that skip connections help alleviate it, while Noci et al. (2022) reveal that collapsed queries and keys hinder gradient flow at initialization. In this work, we tie the rank of the queries and keys to that of the hidden state in linear attention and show how a skewed spectrum of the associative memory can amplify query noise during readout.

Pruning Methods. Conventional pruning methods are usually either unstructured or semi-structured (LeCun et al., 1989; Frantar & Alistarh, 2023; Sun et al., 2024) and thus require specialized kernels to realize sparsity-induced speedups. SparseGPT (Frantar & Alistarh, 2023) frames pruning as a local reconstruction problem. Sun et al. (2024) propose a simpler, gradient-free metric based on the product of weight magnitudes and input activation norms. Several works have sought to enhance this metric by re-incorporating gradients. GBLM (Das et al., 2023) and Pruner-Zero (Dong et al., 2024) utilize gradients derived from full-model backpropagation to refine pruning scores. Wanda++ (Yang et al., 2025b) introduces regional gradients to decrease the computational cost.

A challenge in structured pruning is the handling of coupled structures, where removing a neuron or head in one layer breaks dimensional consistency in subsequent layers. Ma et al. (2023) address this by constructing dependency graphs to identify groups of parameters that must be excised simultaneously. Similarly, our work addresses the structural coupling of linear attention models, specifically the dependency between the projection matrices of queries and keys and the depthwise convolutions. SliceGPT (Ashkboos et al., 2024a) prunes a model's backbone. This affects the rows of $\mathbf{W_Q}$ and $\mathbf{W_K}$. In particular, the attention mechanism operates in the same space pre- and post-slicing. Our work can be considered complementary to this approach. By pruning columns of $\mathbf{W_Q}$ and $\mathbf{W_K}$, we explicitly reduce the dimension of the attention mechanism. ThinK (Xu et al., 2025a) prunes KV cache channels in softmax Transformers based on attention weight magnitudes. While conceptually related to our framework, this method was designed for the KV cache of softmax attention layers, not the state of linear attention ones.

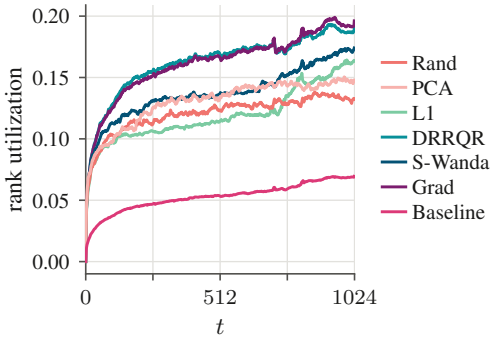


Figure 3: Rank utilization of DeltaNet 370M as a function of the token index for a random sample of Fineweb, averaged over layers and heads, at a compression ratio of 75% pre-RFT. All models except for the baseline have the same state dimensions and are therefore directly comparable.

Quantization Methods. Recent Transformer quantization approaches like QuaRot (Ashkboos et al., 2024b) and SpinQuant (Liu et al., 2025) apply orthogonal rotations to both the residual stream and the state-space to disperse outliers. While effective for softmax Transformers, these state-space rotations are incompatible with the causal convolutions typically employed in linear attention models.

Recent works on quantizing linear attention variants, including Quamba (Chiang et al., 2025b;a), MambaQuant (Xu et al., 2025b), and SSDi8 (Kim et al., 2026), further demonstrate the community's interest in compressing the sequence mixer of linear attention models. Low-rank factorization methods such as FWSVD (Hsu et al., 2022) and ASVD (Yuan et al., 2023) approximate weight matrices but do not reduce the state dimension: the recurrence still operates in the original d_k-dimensional space, yielding no speedup in the sequence mixer. Our framework instead utilizes axis-aligned transformations, which are compatible with causal convolutions and directly reduce the state size.

5 Experiments

We use flame (Zhang & Yang, 2025) for recovery fine-tuning (RFT, Ashkboos et al. (2024a)) and for pre-training the 370M parameter models. RFT uses LoRA (Hu et al., 2021). The details on our

Table 1: Post-RFT evaluation of Gated DeltaNet 1.3B. We report WikiText-2 and Lambada perplexity and average zero-shot accuracy (ZS, Gao et al. (2024)); best results are **bold** and second-best results are underlined.

	75%			50%			30%		
METHOD	WIKI ↓	LMB ↓	ZS ↑	WIKI ↓	LMB ↓	ZS ↑	WIKI ↓	LMB ↓	ZS ↑
RAND	19.3	18.1	54.6	16.9	11.5	57.8	16.1	10.2	58.7
L1	18.6	16.8	55.3	16.8	13.0	57.2	16.3	11.3	58.3
S-WANDA	18.2	15.1	55.4	16.6	11.7	57.8	16.1	11.0	58.6
GRAD	**17.8**	**12.5**	**56.8**	16.4	**10.5**	**58.3**	**15.9**	**9.9**	**59.0**
DRRQR	17.9	14.7	56.2	**16.3**	12.0	58.0	**15.9**	10.7	58.8
BASELINE	16.8	9.7	59.4	16.8	9.7	59.4	16.8	9.7	59.4

experimental setup can be found in Appendix A. Besides the channel selection methods introduced in Section 3, we also report a Rand baseline which randomly selects key channels to drop.

Accuracy We evaluate our structured pruning methods on DeltaNet and Gated DeltaNet (Yang et al., 2024; 2025a) with both 370M and 1.3B parameters, as well as Mamba-Codestral-7B[1] and Qwen3.5-9B (Qwen Team, 2026). Specifically, we compare the optimization-based approaches DRRQR (rank-based) and Grad (gradient-based) with the magnitude-based strategies.

Table 2: DeltaNet 370M WikiText-2 perplexity at 75% state reduction before RFT. Each calibration set contains 128 sequences of length 2048.

Corpus	DRRQR	Grad
FineWeb	41.6	44.9
PG19	41.5	53.8
Pile	41.7	51.0

Figure 3 shows the rank utilization during a forward pass through DeltaNet 370M. It shows that the two best performing compression methods, Grad and DRRQR, have the highest rank utilization. A fine-grained per-head analysis (Appendix C.1) reveals that rank utilization is heterogeneous across heads, and that DRRQR lifts it consistently. Figure 2 shows how DRRQR removes the tail of the hidden states' spectrum.

Surprisingly, we find that pruning half of the key dimension causes only modest increase in WikiText-2 and Lambada perplexity as well as zero-shot language-model evaluations, especially under Grad and DRRQR. Table 1 reports the post-RFT results across compression ratios. Table 20 in the appendix additionally compares pre- and post-RFT performance across model families, showing that pre-RFT degradation depends on the model and pruning method. These results indicate that the evaluated models use substantially less than their nominal recurrent capacity for next-token prediction.

Table 3: Mamba-Codestral-7B before RFT at 50% state reduction.

Method	Wiki ↓	ZS ↑
Baseline	14.5	64.4
Grad	16.8	59.5
DRRQR	17.8	56.8
L1	25.7	48.8

Notably, DRRQR and Grad often perform similarly well. DRRQR is thus competitive with the only task-aware, gradient-based method. Table 2 isolates the practical distinction between the two selectors. DRRQR transfers across calibration corpora. Grad is competitive in-domain but deteriorates under distribution shifts.

Table 3 demonstrates the applicability of our pruning framework to a 7B Mamba-2 model before RFT. DRRQR and Grad degrade substantially less than magnitude based pruning. Appendix C.3 reports more compression ratios and the full task breakdown.

Recurrent models are known to perform poorly in long-context recall (Arora et al., 2024a;b). Our experiments confirm that observation. We furthermore find that, at 50% reduction, the purely recurrent Gated DeltaNet loses recall accuracy. At the same time, the hybrid Qwen3.5-9B matches its baseline (Table 4) under a 50% reduction of its linear layers. This contrast supports the hypothesis that exact attention layers preserve a long-range memory path. Appendix C.4 reports moderate compression ratios and more long-context results.

Speedup We benchmark end-to-end inference speedup under our framework using flash-linear-attention kernels (Yang & Zhang, 2024). Each configuration uses 10 warmup steps and 100 measured

[1] https://huggingface.co/mistralai/Mamba-Codestral-7B-v0.1

steps, and we report the median of three repetitions. We show Gated DeltaNet 1.3B in the main text. Appendix G.1 provides the complete results for both Gated DeltaNet and DeltaNet 1.3B.

Figure 4 (a) shows that decode speedup increases with state reduction and batch size, reaching up to $\sim 40\%$ at 50% compression with batch size $B = 32$. Figure 4 (b) reports prefill speedup at 2K and 8K tokens. At 50% state reduction, gains are $\sim 15\%$ for $B \geq 4$, also at 16K tokens (Appendix G.1). Gains are negligible at 2K with $B \leq 2$. Together, both panels establish end-to-end benefits for both inference regimes. The appendix additionally isolates the sequence mixer to expose kernel-level compute and memory reductions.

Table 4: Long-context retrieval after RFT at 50% state reduction. Only the recurrent components of Qwen3.5-9B are updated. The softmax attention layers remain frozen.

Model	Method	RULER ↑ 4K	NIAH ↑ 4K
GDN 1.3B	Baseline	38.8	81.3
	DRRQR	27.5	61.9
Qwen3.5-9B	Baseline	93.9	100.0
	DRRQR	93.6	100.0

6 Discussion

Pretrained linear attention models do not utilize their memory well, as indicated by the low-rank structure of their hidden state. We provide a rigorous analysis of the role of rank in linear attention, analyzing how the keys and values impact the effective rank and can thus amplify query-noise during readout from said memory. As a consequence, we show that structured pruning of the keys and queries allows discarding at least half of the state while entailing only a minor increase in perplexity. At the same time, our framework exposes the brittleness of linear attention in long-context recall. More specifically, we hypothesize that, in hybrid language models, the softmax layers carry the burden of modeling long-range interactions.

In addition to adapting several pruning strategies, we introduce a method with a provable conditioning bound for the selected joint key-query activations. We present extensive empirical evaluations demonstrating the practical efficiency and effectiveness of our structured pruning framework.

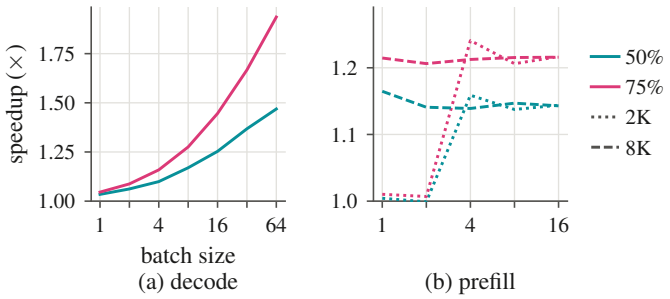


Figure 4: End-to-end speedup of Gated DeltaNet 1.3B on an H100, relative to the uncompressed baseline. **(a)** Decode, prompt length 512. **(b)** Prefill, at context length 2K and 8K. Complete results for Gated DeltaNet and DeltaNet are provided in Appendix G.1.

We note that our proposed framework has two primary limitations. First, the achievable speed-up depends on the batch size, with smaller batch sizes yielding more modest gains. Second, state reduction is generally not lossless, and competitive downstream performance typically requires RFT after pruning.

AI use statement

In this work, we have used generative AI tools to formulate mathematical claims, provide critical ingredients for proving them, to design or give feedback on our methodology and experiments, to create or edit code, implement methods, to propose or refine hypotheses, to support qualitative and thematic data analysis, interpret results, and to do a literature review. Additionally, we used generative AI tools for LaTeX formatting, to draft parts of the appendix, and for brainstorming. We have not used generative AI tools for generating synthetic datasets, to help develop theoretical models or conceptual frameworks, to assist with translation, or to clean or reformat datasets. We have reviewed all AI-assisted work, checked the code as well as the reformatted LaTeX tables. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff. *arXiv preprint arXiv:2402.18668*, 2024a.

Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: closing the recall gap for recurrent language models. *arXiv preprint arXiv:2407.05483*, 2024b.

Saleh Ashkboos, Maximilian Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations*, volume 2024, pp. 11682–11701, 2024a.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024b.

Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, et al. Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models. *arXiv preprint arXiv:2504.03624*, 2025.

Makram Chahine, Philipp Nazari, Daniela Rus, and T Konstantin Rusch. The curious case of in-training compression of state space models. In *International Conference on Learning Representations*, volume 2026, pp. 111759–111798, 2026.

Chi-Tsong Chen. *Linear system theory and design*, volume 301. Holt, Rinehart and Winston New York, 1984.

Hung-Yueh Chiang, Chi-Chih Chang, Natalia Frumkin, Kai-Chiang Wu, Mohamed S Abdelfattah, and Diana Marculescu. Quamba2: A robust and scalable post-training quantization framework for selective state space models. *arXiv preprint arXiv:2503.22879*, 2025a.

Hung-Yueh Chiang, Chi-Chih Chang, Natalia Frumkin, Kai-Chiang Wu, and Diana Marculescu. Quamba: A post-training quantization recipe for selective state space models. In *International Conference on Learning Representations*, volume 2025, pp. 101328–101354, 2025b.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 1800–1807. Ieee, 2017.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Rocktim Jyoti Das, Mingjie Sun, Liqun Ma, and Zhiqiang Shen. Beyond size: How gradients shape pruning decisions in large language models. *arXiv preprint arXiv:2311.04902*, 2023.

Peijie Dong, Lujun Li, Zhenheng Tang, Xiang Liu, Xinglin Pan, Qiang Wang, and Xiaowen Chu. Pruner-zero: Evolving symbolic pruning metric from scratch for large language models. *arXiv preprint arXiv:2406.02924*, 2024.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International conference on machine learning*, pp. 2793–2803. PMLR, 2021.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pp. 10323–10337. PMLR, 2023.

Daniel Y Fu, Tri Dao, Khaled K Saab, Armin W Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. *arXiv preprint arXiv:2212.14052*, 2022.

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, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024. URL https://arxiv.org/abs/2312.00752.

Ming Gu and Stanley C Eisenstat. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.

Roger A Horn and Charles R Johnson. *Topics in matrix analysis*. Cambridge Cambridge University Press, 1991.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language model compression with weighted low-rank factorization. *arXiv preprint arXiv:2207.00112*, 2022.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In *International conference on machine learning*, pp. 9099–9117. PMLR, 2022.

Ilse CF Ipsen and Arvind K Saibaba. Stable rank and intrinsic dimension of real and complex matrices. *SIAM Journal on Matrix Analysis and Applications*, 46(3):1988–2007, 2025.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Hyunwoo Kim, Byoungchan Ko, Minseok Kang, Minwoo Kim, Dongjin Lee, Jaehoon Lee, Sungroh Yoon, and Dahuin Jung. Ssdi8: Accurate and efficient 8-bit quantization for state space duality. In *International Conference on Learning Representations*, volume 2026, pp. 97014–97042, 2026.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Lucas D Lingle. Transformer-vq: Linear-time transformers via vector quantization. In *International Conference on Learning Representations*, volume 2024, pp. 6121–6150, 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: Llm quantization with learned rotations. In *International Conference on Learning Representations*, volume 2025, pp. 92009–92032, 2025.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.

Leon Mirsky. Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1):50–59, 1960.

Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. *Advances in Neural Information Processing Systems*, 35:27198–27211, 2022.

Rom N Parnichkun, Neehal Tumma, Armin W Thomas, Alessandro Moro, Qi An, Taiji Suzuki, Atsushi Yamashita, Michael Poli, and Stefano Massaroli. Quantifying memory utilization with effective state-size. *arXiv preprint arXiv:2504.19561*, 2025.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, et al. Rwkv: Reinventing rnns for the transformer era. In *Findings of the association for computational linguistics: EMNLP 2023*, pp. 14048–14077, 2023.

Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

Mark Rudelson and Roman Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)*, 54(4):21–es, 2007.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9355–9366. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/schlag21a.html.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. Deltaproduct: Improving state-tracking in linear rnns via householder products. *Advances in Neural Information Processing Systems*, 38:153738–153782, 2026.

David So, Wojciech Mańke, Hanxiao Liu, Zihang Dai, Noam Shazeer, and Quoc V Le. Searching for efficient transformers for language modeling. *Advances in neural information processing systems*, 34:6010–6022, 2021.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 4942–4964, 2024.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, et al. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

Joel A Tropp. An introduction to matrix concentration inequalities. *Foundations and trends® in machine learning*, 8(1-2):1–230, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Roman Vershynin. High-dimensional probability. *Cambridge Series in Statistical and Probabilistic Mathematics*, 47, 2019.

Chaoqi Wang, Roger Grosse, Sanja Fidler, and Guodong Zhang. Eigendamage: Structured pruning in the kronecker-factored eigenbasis. In *International conference on machine learning*, pp. 6566–6575. PMLR, 2019.

Ke Alexander Wang, Jiaxin Shi, and Emily B Fox. Test-time regression: a unifying framework for designing sequence models with associative memory. *arXiv preprint arXiv:2501.12352*, 2025.

Yuhui Xu, Zhanming Jie, Hanze Dong, Lei Wang, Xudong Lu, Aojun Zhou, Amrita Saha, Caiming Xiong, and Doyen Sahoo. Think: Thinner key cache by query-driven pruning. In *International Conference on Learning Representations*, volume 2025, pp. 56691–56709, 2025a.

Zukang Xu, Yuxuan Yue, Xing Hu, Zhihang Yuan, Zixu Jiang, Zhixuan Chen, Jiangyong Yu, Chen Xu, Sifan Zhou, and Dawei Yang. Mambaquant: Quantizing the mamba family with variance aligned rotation methods, 2025b. URL https://arxiv.org/abs/2501.13484.

Songlin Yang and Yu Zhang. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, January 2024. URL https://github.com/fla-org/flash-linear-attention.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. *Advances in neural information processing systems*, 37: 115491–115522, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *International Conference on Learning Representations*, volume 2025, pp. 29687–29707, 2025a.

Yifan Yang, Kai Zhen, Bhavana Ganesh, Aram Galstyan, Goeric Huybrechts, Markus Müller, Jonas M. Kübler, Rupak Vignesh Swaminathan, Athanasios Mouchtaris, Sravan Babu Bodapati, Nathan Susanj, Zheng Zhang, Jack FitzGerald, and Abhishek Kumar. Wanda++: Pruning large language models via regional gradients, 2025b. URL https://arxiv.org/abs/2503.04992.

Zhihang Yuan, Yuzhang Shang, Yue Song, Dawei Yang, Qiang Wu, Yan Yan, and Guangyu Sun. Asvd: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*, 2023.

Yu Zhang and Songlin Yang. Flame: Flash language modeling made easy, January 2025. URL https://github.com/fla-org/flame.

A Training Details

We use the flame (Zhang & Yang, 2025) library for recovery fine-tuning (RFT, (Ashkboos et al., 2024a)) and pre-training the 370M parameter models. The latter use 10 billion tokens of FineWeb. The larger DeltaNet and Gated DeltaNet models are taken from fla-hub [2] and m-a-p [3], respectively. The scale experiments use Mamba-Codestral-7B-v0.1 [4] and Qwen3.5-9B-Base. Mamba-Codestral results are pre-RFT. For Qwen3.5, we prune only the Gated DeltaNet layers; during RFT, the exact-attention layers remain frozen.

We perform RFT using LoRA (Hu et al., 2021) with rank $r = 32$ and $\alpha = 64$, using $32k$ samples of Fineweb. During training, we use a batch size of 16 and sequences of length 2048. After warming up for 5% of the total steps, we decay the learning rate from 10^{-4} down to 10^{-5}. We furthermore unfreeze the causal convolutions and all normalization layers, which make up just a fraction of the total parameters. We furthermore apply knowledge-distillation during RFT using the original model as a teacher.

For the compression methods requiring a calibration set, we use 128 samples of FineWeb. DRRQR uses a random subsample of 5K keys and queries each. The calibration ablation varies this set from 8 to 512 sequences of length 2048, changes the random seed, and replaces FineWeb with PG19 or the Pile while holding the evaluation fixed.

B Additional Material

Algorithm 1 describes a typical DeltaNet (Yang et al., 2024) layer.

Algorithm 1 A Typical DeltaNet Layer

Input: Hidden states $\mathbf{X} \in \mathbb{R}^{T,h}$, H heads, Previous state $\mathbf{S}_0$ per head
Per-head parameters: $\mathbf{W}_q, \mathbf{W}_k \in \mathbb{R}^{h,d_k}$, $\mathbf{W}_v \in \mathbb{R}^{h,d_v}$, $\mathbf{W}_\beta \in \mathbb{R}^{h,1}$.
Output projection: $\mathbf{W}_o \in \mathbb{R}^{Hd_v,h}$.

// 1. Projections and Local Mixing (per head)
$\mathbf{q} \leftarrow \mathrm{SiLU}(\mathrm{Conv1D}(\mathbf{X}\mathbf{W}_q))$
$\mathbf{k} \leftarrow \mathrm{SiLU}(\mathrm{Conv1D}(\mathbf{X}\mathbf{W}_k))$
$\mathbf{v} \leftarrow \mathrm{SiLU}(\mathrm{Conv1D}(\mathbf{X}\mathbf{W}_v))$
$\beta \leftarrow \sigma(\mathbf{X}\mathbf{W}_\beta)$

// 2. Normalization
$\mathbf{q} \leftarrow \mathbf{q}/\|\mathbf{q}\|_2$
$\mathbf{k} \leftarrow \mathbf{k}/\|\mathbf{k}\|_2$

// 3. Delta Rule
for $t = 1$ **to** T **do**
  $\mathbf{S}_t \leftarrow \mathbf{S}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) + \beta_t \mathbf{v}_t \mathbf{k}_t^\top$
  $\mathbf{o}_t \leftarrow \mathbf{S}_t \mathbf{q}_t$
end for
Concatenate heads to form $\mathbf{O} \in \mathbb{R}^{T,Hd_v}$

// 4. Output Projection
$\mathbf{O} \leftarrow \mathrm{RMSNorm}(\mathbf{O})$
$\mathbf{Y} \leftarrow \mathbf{O}\mathbf{W}_o$

Return: $\mathbf{Y}$

[2] https://huggingface.co/collections/fla-hub/deltanet
[3] https://huggingface.co/m-a-p/1.3B-100B-GatedDeltaNet-pure
[4] https://huggingface.co/mistralai/Mamba-Codestral-7B-v0.1

B.1 Structured Pruning from the Test-Time Regression Perspective

Linear attention, and specifically DeltaNet, is usually interpreted as performing gradient descent on a linear regression objective (Schlag et al., 2021; Yang et al., 2024), where the hidden state $\mathbf{S}_t$ acts as the fast weight matrix trained to map keys $\mathbf{k}_t$ (inputs) to values $\mathbf{v}_t$ (targets). In this framework, our proposed structured pruning strategy admits another interpretation, functioning as a feature selection step applied to the input of the online learner. By restricting the input features to the subspace spanned by the transformation matrix $\mathbf{T}$, we effectively constrain the hypothesis class of the regression. This forces the fast weights to ignore the null space of $\mathbf{T}$, acting as a form of regularization.

Algorithm 2 Deep Rank Revealing QR (DRRQR). Adapted from Algorithm 4 of Gu & Eisenstat (1996).

Input: Activation Matrix $\mathbf{A} \in \mathbb{R}^{2N,d_k}$, Target Rank d'_k, Tolerance $f \geq 1$
Output: Selected Indices $\mathcal{I}$

// 1. Initialization: QR with Column Pivoting
$[\mathbf{Q}, \mathbf{R}, \boldsymbol{\pi}] \leftarrow \mathrm{QRCP}(\mathbf{A})$, with column index vector $\boldsymbol{\pi}$
Partition $\mathbf{R} = \begin{pmatrix} \mathbf{A}_{d'_k} & \mathbf{B}_{d'_k} \\ \mathbf{0} & \mathbf{C}_{d'_k} \end{pmatrix}$, where $\mathbf{A}_{d'_k} \in \mathbb{R}^{d'_k,d'_k}$
Initialize $\omega_i(\mathbf{A}_{d'_k}) = 1/\|(\mathbf{A}^{-1}_{d'_k})_{i,:}\|_2$ and $\gamma_j(\mathbf{C}_{d'_k}) = \|(\mathbf{C}_{d'_k})_{:,j}\|_2$

// 2. Iterative Swapping for Stability
while True **do**
  Compute $\mathbf{U} = \mathbf{A}^{-1}_{d'_k}\mathbf{B}_{d'_k}$

  // Calculate swap gain metric ρ_{ij} for all pairs
  // Checks if U_{ij} is large or if residual γ_j is large relative to basis ω_i
  $\rho_{ij} \leftarrow \sqrt{|U_{ij}|^2 + (\gamma_j(\mathbf{C}_{d'_k})/\omega_i(\mathbf{A}_{d'_k}))^2}$

  Let $(i^*, j^*) = \mathrm{argmax}_{i,j}\, \rho_{ij}$
  if $\rho_{i^*j^*} \leq f$ **then**
    break // Strong RRQR condition met
  end if

  Swap columns i^* and $d'_k + j^*$ of $\mathbf{R}$
  Swap entries π_{i^*} and $\pi_{d'_k+j^*}$
  Retriangularize $\mathbf{R}$ and update $\omega(\mathbf{A}_{d'_k})$ and $\gamma(\mathbf{C}_{d'_k})$
end while

$\mathcal{I} \leftarrow \{\pi_1, \ldots, \pi_{d'_k}\}$
Return: $\mathcal{I}$

B.2 Details on Rank Revealing QR (RRQR)

In this section, we detail the *Strong Rank-Revealing QR* (RRQR) algorithm (see Algorithm 2) proposed by Gu & Eisenstat (1996), which forms the basis of our DRRQR pruning method.

Mathematical Formulation. Let $\mathbf{M} \in \mathbb{R}^{m,n}$ be the input matrix of concatenated keys and queries (in the main text, $m = 2N$ and $n = d_k$). We seek a permutation $\boldsymbol{\Pi}$ and a target rank k (in our main text denoted as d'_k) such that the QR factorization

$$\mathbf{M}\boldsymbol{\Pi} = \mathbf{Q}\begin{pmatrix} \mathbf{A}_k & \mathbf{B}_k \\ \mathbf{0} & \mathbf{C}_k \end{pmatrix} \tag{9}$$

satisfies specific bounds on the singular values of the leading principal submatrix $\mathbf{A}_k \in \mathbb{R}^{k,k}$ and the trailing submatrix $\mathbf{C}_k \in \mathbb{R}^{(m-k),(n-k)}$. Assume $\mathrm{rank}(\mathbf{M}) \geq k$. For tolerance $f \geq 1$, the stopping condition below guarantees (Gu & Eisenstat, 1996, Theorem 3.2)

$$\sigma_i(\mathbf{A}_k) \geq \frac{\sigma_i(\mathbf{M})}{c_f} \quad (1 \leq i \leq k), \qquad \|\mathbf{C}_k\|_2 \leq c_f \sigma_{k+1}(\mathbf{M}),$$

where $c_f = \sqrt{1 + f^2 k(n-k)}$. The selected columns share the singular values of $\mathbf{A}_k$. Since $\sigma_1(\mathbf{A}_k) \leq \sigma_1(\mathbf{M})$, their condition number satisfies

$$\kappa(\mathbf{A}_k) \leq c_f \frac{\sigma_1(\mathbf{M})}{\sigma_k(\mathbf{M})}.$$

Geometric Intuition. The volume of the parallelotope formed by the selected columns is $|\det(\mathbf{A}_k)| = \prod_{i=1}^{k} \sigma_i(\mathbf{A}_k)$. Each accepted column swap increases this volume. The procedure stops when no single-column swap increases it by a factor greater than f, ensuring the bounds above.

The Swap Criterion. Let $\mathbf{\Pi}$ be the current column permutation matrix. To determine if swapping the i-th column of the basis (where $1 \leq i \leq k$) with the j-th column of the residual (where $1 \leq j \leq n-k$) improves the factorization, we analyze the ratio of the new determinant to the current determinant.

Gu & Eisenstat (1996) derive an efficiently computable metric for this ratio. Let $\mathbf{U} = \mathbf{A}_k^{-1}\mathbf{B}_k$. We define:

- $\gamma_j(\mathbf{C}_k) = \|(\mathbf{C}_k)_{:,j}\|_2$: The ℓ_2-norm of the j-th column of the residual block.
- $\omega_i(\mathbf{A}_k) = 1/\|(\mathbf{A}_k^{-1})_{i,:}\|_2$: The reciprocal of the ℓ_2-norm of the i-th row of the inverse basis.

By Lemma 3.1 of Gu & Eisenstat (1996), the potential gain ρ_{ij} from swapping basis column i with candidate column j is given by:

$$\rho_{ij} = \sqrt{|U_{ij}|^2 + \left(\frac{\gamma_j(\mathbf{C}_k)}{\omega_i(\mathbf{A}_k)}\right)^2}. \tag{10}$$

If $\rho_{ij} > f$ for a chosen tolerance factor $f \geq 1$, swapping these columns guarantees an increase in $|\det(\mathbf{A}_k)|$ by a factor of at least ρ_{ij}. The first term, $|U_{ij}|^2$, captures the linear dependence of the candidate vector on the current basis vector, while the second term captures the magnitude of the candidate relative to the stability of the basis vector.

Algorithm and Update Rules. The DRRQR procedure (Algorithm 2) proceeds as follows:

1. **Initialization:** Compute an initial factorization using standard QRCP. This provides a baseline $\mathbf{\Pi}$, $\mathbf{A}_k$, $\mathbf{B}_k$, and $\mathbf{C}_k$.
2. **Identification:** Search for a pair of indices (i, j) such that $\rho_{ij} > f$. Efficient search strategies maximize over j for fixed i, or simply identify the first valid pair.
3. **Update:** If a valid pair is found:
   (a) Swap columns i and $j + k$ of $\mathbf{R}$ and update the permutation $\mathbf{\Pi}$ accordingly.
   (b) Retriangularize the matrix $\mathbf{R}$ using Givens rotations to restore the upper-triangular structure of $\mathbf{A}_k$.
   (c) Update the auxiliary vectors $\omega(\mathbf{A}_k)$ and $\gamma(\mathbf{C}_k)$ using the formulas provided in Section 4 of Gu & Eisenstat (1996).
4. **Termination:** The process repeats until no pair (i, j) satisfies $\rho_{ij} > f$, ensuring the matrix satisfies the strong rank-revealing condition.

B.3 Adapting Convolutions to General Rotations

Handling the more general case of (semi-) orthogonal, non-axis-aligned transformations is more intricate than the axis-aligned one (see Section 3.3). If one wishes to employ general orthogonal transformations $\mathbf{T} \in O(d_k)$ (such as those derived from PCA) to prune the sequence mixer, the convolution layers must be adapted.

Depthwise convolutions operate independently on each channel. Rotating the input space via $\mathbf{T}$ generally turns the exact filter matrices into dense matrices. To retain depthwise convolutions, we approximate these rotated filters with diagonal matrices.

B.4 Optimal Diagonal Filter Approximation

We choose the per-channel filters that minimize the sum of squared Frobenius distances to the exact rotated filter matrices.

Proposition B.1 (Optimal Diagonal Filter Approximation)**.** *Let $\mathbf{T} \in O(d)$ be an orthogonal transformation and let $\mathbf{W} = [\mathbf{w}^{(0)}, \dots, \mathbf{w}^{(l-1)}] \in \mathbb{R}^{d,l}$ contain the original convolution filters for d channels and kernel size l. Writing $\mathbf{W}' = [\mathbf{w}'^{(0)}, \dots, \mathbf{w}'^{(l-1)}]$, the optimization problem*

$$\min_{\mathbf{W}' \in \mathbb{R}^{d,l}} \sum_{j=0}^{l-1} \left\| \operatorname{diag}(\mathbf{w}'^{(j)}) - \mathbf{T} \operatorname{diag}(\mathbf{w}^{(j)}) \mathbf{T}^\top \right\|_F^2$$

has the unique solution

$$\mathbf{W}' = (\mathbf{T} \odot \mathbf{T})\mathbf{W},$$

where $\odot$ denotes the Hadamard (element-wise) product.

Proof. For rotated inputs $\tilde{\mathbf{x}}_t = \mathbf{T}\mathbf{x}_t$, the exact rotated convolution is

$$\mathbf{T}(\mathbf{W} * \mathbf{x})_t = \sum_{j=0}^{l-1} \mathbf{M}^{(j)} \tilde{\mathbf{x}}_{t-j}, \qquad \mathbf{M}^{(j)} \coloneqq \mathbf{T} \operatorname{diag}(\mathbf{w}^{(j)}) \mathbf{T}^\top.$$

For each lag j, the squared Frobenius error decomposes as

$$\left\| \operatorname{diag}(\mathbf{w}'^{(j)}) - \mathbf{M}^{(j)} \right\|_F^2 = \sum_{a=1}^{d} \left(w_a'^{(j)} - M_{aa}^{(j)} \right)^2 + \sum_{a \neq b} \left(M_{ab}^{(j)} \right)^2.$$

The second sum is independent of $\mathbf{W}'$. Hence the unique minimizer satisfies

$$w_a'^{(j)} = M_{aa}^{(j)} = \sum_{b=1}^{d} T_{ab}^2 w_b^{(j)} = \left((\mathbf{T} \odot \mathbf{T}) \mathbf{w}^{(j)} \right)_a.$$

Stacking the coefficients across all lags gives $\mathbf{W}' = (\mathbf{T} \odot \mathbf{T})\mathbf{W}$. □

Each adapted kernel is a weighted average of the original kernels, with nonnegative weights T_{ab}^2 that sum to one. This optimality concerns filter approximation in Frobenius norm and requires no assumptions on the activation distribution. We evaluate its effect on model outputs empirically in Appendix C.6.

B.5 Shared Convolutions

An alternative approach to facilitate general rotations is to constrain the model architecture itself. If we enforce that the convolution filters are shared across all channels within a head, the convolution operation becomes a scalar multiplication at each lag, which commutes with any linear transformation.

Lemma B.2 (Commutativity of Shared Convolutions)**.** *Let $\mathbf{w} \in \mathbb{R}^l$ be a filter shared across all d channels, such that the convolution kernel matrix $\mathbf{W} \in \mathbb{R}^{d,l}$ has identical rows $\mathbf{W}_{k,:} = \mathbf{w}$ for all k. For any linear transformation matrix $\mathbf{T} \in \mathbb{R}^{d,d}$ (including orthogonal rotations), the convolution commutes with the transformation:*

$$\mathbf{W} * (\mathbf{T}\mathbf{x}) = \mathbf{T}(\mathbf{W} * \mathbf{x}).$$

Proof. For a shared filter, the convolution operation on the vector $\mathbf{x}_t$ can be written as a scalar convolution applied element-wise: $(\mathbf{W} * \mathbf{x})_t = \sum_{j=0}^{l-1} w_j \mathbf{x}_{t-j}$. Applying the transformation $\mathbf{T}$ first:

$$\mathbf{W} * (\mathbf{T}\mathbf{x})_t = \sum_{j=0}^{l-1} w_j (\mathbf{T}\mathbf{x}_{t-j}) = \mathbf{T}\left(\sum_{j=0}^{l-1} w_j \mathbf{x}_{t-j}\right) = \mathbf{T}(\mathbf{W} * \mathbf{x})_t.$$

□

Shared convolutions commute with rotations without filter adaptation. This equivariance does not extend to the full layer when SiLU is present, since SiLU generally does not commute with dense rotations.

C Additional Experimental Results

C.1 Per-Head Rank Utilization

We compute the rank utilization of each (layer, head) pair for DeltaNet 370M at 75% compression (pre-RFT), averaged over the sequence and batch axes for a sample from FineWeb. Table 5 summarizes the results across all (layer, head) pairs. Three findings emerge. First, heads are heterogeneous: rank utilization varies by an order of magnitude across heads, suggesting that the model allocates memory capacity unevenly. Second, DRRQR lifts rank utilization consistently across heads, while magnitude-based methods such as L1 exhibit weaker correlation with the baseline per-head structure ($\rho = 0.68$). Third, every method increases rank utilization at the reduced capacity, consistent with the effective-rank bound in Equation (4).

Table 5: Per-head rank utilization statistics across all (layer, head) pairs for DeltaNet 370M at 75% compression (pre-RFT). ρ denotes the Pearson correlation between baseline and post-pruning per-head rank utilization.

Method	**Mean**	**Std**	**Min**	**Max**	$\boldsymbol{\rho}$
Baseline ($d_k = 128$)	0.053	0.028	0.01	0.16	1.00
DRRQR ($d_k = 32$)	0.170	0.060	0.03	0.40	0.96
L1 ($d_k = 32$)	0.124	0.036	0.03	0.30	0.68

C.2 DRRQR, Grad, and Calibration Robustness

DRRQR and Grad share a median of 26 of the 32 retained channels per head on DeltaNet 370M at 75% compression, whereas random selection shares 8. Thus, the loss gradient implicitly identifies much of the same rank-preserving subspace as DRRQR.

Table 6 reports per-head medians for the retained columns of the joint $[\mathbf{Q}; \mathbf{K}]$ activation matrix. DRRQR raises $\sigma_{\min}$ while preserving $\sigma_{\max}$. L1 and S-Wanda obtain smaller condition numbers only by removing high-energy directions; the condition-number column is the median of per-head ratios and therefore need not equal the ratio of the displayed medians.

Table 6: Selected key/query subspace on DeltaNet 370M at 75% compression.

Selection	$\sigma_{\min}$	$\sigma_{\max}$	κ
Baseline ($d_k = 128$)	0.10	49.4	529
DRRQR	0.96	49.4	65
Grad	0.62	48.5	89
S-Wanda	0.65	22.5	42
L1	0.64	15.3	32
Rand	0.16	13.3	116

We next vary calibration sample count, seed, and corpus, always using sequences of 2048 tokens and evaluating before RFT. Table 7 shows that DRRQR is effectively invariant across all three axes. Grad

is stable in-domain but substantially less robust to distribution shift because its score depends on the calibration loss.

Table 7: Calibration sensitivity on DeltaNet 370M at 75% compression (WikiText-2 perplexity, pre-RFT).

Calibration	DRRQR	Grad
FineWeb, $n = 8$	41.7	44.9
FineWeb, $n = 32$	41.6	45.0
FineWeb, $n = 128$	41.6	44.9
FineWeb, $n = 512$	41.6	44.8
FineWeb, $n = 128$, seed 111	41.6	44.8
FineWeb, $n = 128$, seed 1234	41.5	44.6
PG19, $n = 128$	41.5	53.8
Pile, $n = 128$	41.7	51.0

C.3 Generalization to Mamba-2

To assess generality beyond DeltaNet, we apply our framework to Mamba-2 (Dao & Gu, 2024), where $\mathbf{B}$ and $\mathbf{C}$ play the roles of keys and queries. Table 8 gives the full zero-shot task breakdown for a 370M model at 75% reduction. Table 9 adds a 50% setting and Mistral's Mamba-2 checkpoint Mamba-Codestral-7B-v0.1. All results are pre-RFT and therefore isolate channel-selection quality. Grad and DRRQR degrade substantially more gracefully than magnitude pruning at 7B scale, although Grad is strongest in this setting.

Table 8: Mamba-2 370M at 75% state reduction ($128 \rightarrow 32$). Zero-shot accuracy and WikiText-2 perplexity (pre-RFT).

Method	ARC-E	ARC-C	HellaS	WinoG	PIQA	Wiki ↓
Baseline	49.8	27.1	40.2	52.4	65.7	29.2
Grad	47.0	26.7	36.8	50.6	64.1	44.6
DRRQR	44.4	26.4	36.1	51.0	63.5	47.3
S-Wanda	41.7	25.3	35.4	53.0	62.2	52.8
L1	40.3	24.9	34.6	51.7	62.0	54.3
Rand	41.0	25.0	34.7	50.6	61.5	52.8

Table 9: Mamba-2 results across scale before RFT. Zero-shot is the mean accuracy over ARC-Easy, ARC-Challenge, HellaSwag, WinoGrande, PIQA, and LAMBADA.

Model	Method	Compression	Wiki ↓	Zero-shot ↑
Mamba-2 370M	Baseline	0%	29.2	44.5
	Grad	50%	34.4	42.7
	DRRQR	50%	35.9	41.4
	L1	50%	35.4	40.8
Mamba-Codestral 7B	Baseline	0%	14.5	64.4
	Grad	50%	16.8	59.5
	DRRQR	50%	17.8	56.8
	L1	50%	25.7	48.8
	Grad	75%	22.7	52.5
	DRRQR	75%	27.0	48.2
	L1	75%	58.1	40.8

C.4 Long-Context Retrieval and Hybrid Models

We evaluate post-RFT Gated DeltaNet 1.3B checkpoints on RULER (Hsieh et al., 2024), including single-needle retrieval, and on SQuAD- and SWDE-completion. The models are trained at sequence

Table 10: Zero-shot performance of Mamba-Codestral-7B before RFT, evaluated using the lm-eval-harness (Gao et al., 2024), given different compression ratios.

Method	**Compr.**	Wiki. ↓ ppl	LMB. ↓ ppl	ARC-e ↑ acc_n	ARC-c ↑ acc_n	Hella. ↑ acc_n	Wino. ↑ acc	PIQA ↑ acc_n	LMB. ↑ acc	Avg ↑
Grad	50%	**16.8**	**14.0**	67.4	43.5	68.0	57.9	74.6	45.8	**59.5**
DRRQR	50%	17.8	29.9	66.4	41.6	66.4	58.3	74.5	33.8	56.8
L1	50%	25.7	73.7	55.5	30.9	56.7	53.9	71.5	23.9	48.8
Grad	75%	**22.7**	**67.8**	63.4	38.1	62.5	53.0	73.6	24.3	**52.5**
DRRQR	75%	27.0	290.3	60.2	36.5	59.1	51.2	71.9	10.4	48.2
L1	75%	58.1	1157.9	46.1	27.5	46.1	52.2	67.3	5.5	40.8
Baseline	0%	14.5	6.7	71.5	44.8	70.5	64.2	77.0	58.2	64.4

length 2K and evaluated at both 2K and 4K. Table 11 shows that 20–30% reduction incurs a moderate cost, whereas at 50% the finite recurrent capacity begins to bind.

Table 11: Long-context and recall results for Gated DeltaNet 1.3B after RFT.

Model	RULER ↑ 2K	RULER ↑ 4K	NIAH ↑ 2K	NIAH ↑ 4K	SQuAD ↑	SWDE ↑
Baseline	47.7	38.8	93.1	81.3	40.5	51.2
DRRQR 20%	44.6	35.4	91.1	75.9	39.9	45.5
Grad 20%	46.1	34.0	92.3	77.1	39.8	47.1
DRRQR 30%	43.1	32.6	89.2	72.5	39.3	43.8
Grad 30%	45.0	33.9	90.7	75.6	39.3	45.8
DRRQR 50%	37.9	27.5	81.9	61.9	36.4	37.0
Grad 50%	38.4	28.1	83.7	65.7	37.2	37.7

We repeat the 50% experiment on Qwen3.5-9B-Base, a hybrid with softmax attention and Gated DeltaNet layers. RFT updates only the recurrent components; softmax attention layers remain unchanged. Table 12 shows that the hybrid preserves long-context retrieval at the same compression ratio that degrades the purely recurrent model. This supports the hypothesis that exact-attention layers carry recall.

Table 12: Qwen3.5-9B-Base long-context results after RFT of only the Gated DeltaNet components.

Configuration	RULER ↑ 2K	RULER ↑ 4K	NIAH ↑ 2K	NIAH ↑ 4K	SQuAD ↑	SWDE ↑
Baseline	94.7	93.9	100.0	100.0	65.3	92.7
DRRQR 50%	94.3	93.6	100.0	100.0	66.0	92.3

The pre-RFT results in Table 13 show that the rank-aware selector is important: DRRQR preserves most zero-shot accuracy at 50%, whereas L1 collapses. We have not combined state reduction directly with intra-layer token routing or sparse caches such as STILL and LoLA; those methods act on the token axis and remain complementary future work.

C.5 Singular Value Spectrum

Figure 5 illustrates the singular value spectrum of a randomly selected head's hidden state, aggregated across all tokens. We compare the uncompressed baseline against DRRQR, Grad and L1 methods at a 75% compression ratio (prior to recovery fine-tuning). Notably, while the uncompressed model exhibits a heavy tail of singular values, the compressed models display a much sharper spectral truncation. For some heads, we observe the emergence of a spectral gap. This gap can also develop during recovery fine-tuning (see Figure 6). We consider studying this phenomenon an interesting branch for future research.

Table 13: Qwen3.5-9B-Base before RFT.

Configuration	Wiki ↓	LAMBADA ↓	Zero-shot ↑
Baseline	9.0	3.8	73.3
DRRQR 25%	9.7	4.3	73.0
DRRQR 50%	11.6	5.7	71.4
L1 25%	18.8	7.1	61.3
L1 50%	38.0	33.4	50.9

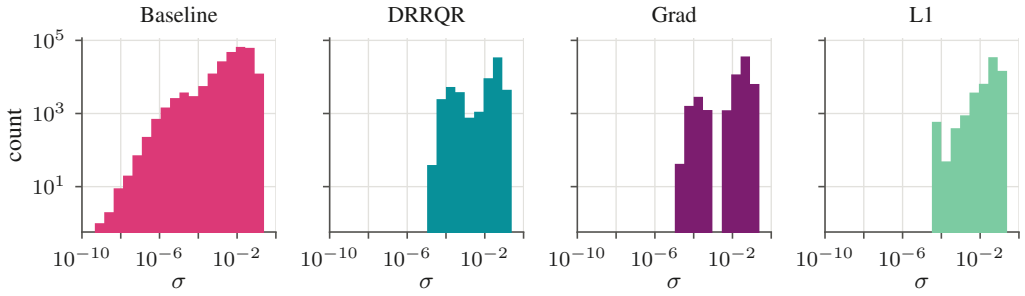


Figure 5: Singular value spectrum of a DeltaNet 370M head's hidden state (Fineweb, $T = 2048$, first 128 tokens skipped). We compare the uncompressed *Baseline* against compressed models at a 75% compression ratio (pre-RFT).

C.6 On PCA and Convolutions

In this section, we provide extended experimental results on the PCA-based pruning strategy. We furthermore analyze the impact of depthwise convolutions on the applicability and performance of semi-orthogonal structured pruning.

As mentioned in Section 3, depthwise convolutions are generally not invariant under orthogonal transformations. When pruning via PCA, the features are rotated, causing a misalignment with the per-channel convolution filters. Appendix B.3 considers two approaches: shared convolutions, which commute with rotations, and filter mixing, which gives the closest diagonal approximation to the rotated filter matrices in Frobenius norm (Proposition B.1).

C.6.1 Shared Convolutions

To understand the impact of those two approaches, we train DeltaNet 370M variants using shared convolutions, where the convolution filter is tied across all channels within a head. As shown in Lemma B.2, the shared convolution operator is equivariant under rotations.

Table 14: DeltaNet 370M models trained on $10B$ tokens evaluated on commonsense zero-shot reasoning tasks. "Shared Conv" indicates whether convolution filters are shared across heads and/or channels (for example, $\times\checkmark$ means that filters are shared across channels inside of a head but not across heads).

Shared Conv	Wiki. ↓ ppl	LMB. ↓ ppl	ARC-e ↑ acc_n	ARC-c ↑ acc_n	Hella. ↑ acc_n	Wino. ↑ acc	PIQA ↑ acc_n	LMB. ↑ acc	Avg ↑
$\times\times$	29.8	37.0	51.1	27.6	38.1	52.2	65.0	31.3	44.2
$\times\checkmark$	29.5	40.4	49.7	27.1	38.3	52.4	64.7	30.4	43.8
$\checkmark\checkmark$	29.3	40.4	49.9	26.4	37.8	49.9	65.2	29.9	43.2

Filter Similarity. Figure 7 visualizes the learned filters of a standard DeltaNet (non-shared). We observe high similarity between filters within specific heads, suggesting that the model naturally learns to share dynamics across channels. This could serve as a justification for sharing filters among channels inside each head.

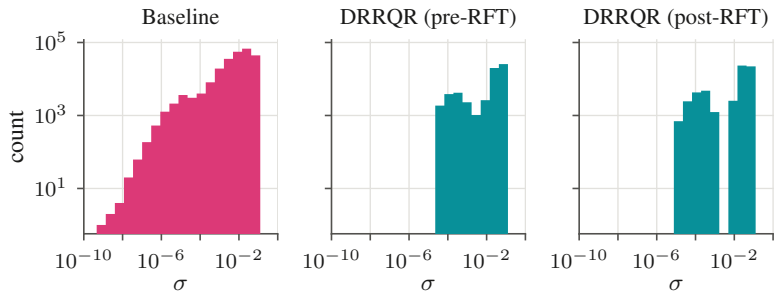


Figure 6: Impact of recovery fine-tuning (RFT) on the singular value spectrum ($T = 2048$, first 128 tokens skipped). We compare the uncompressed *Baseline* with *DRRQR* (75% compression) both before (*pre*) and after (*post*) RFT.

Performance of shared convolutions. We include results on pre-trained DeltaNet models with and without shared convolutions in Table 14. It shows that, while models with shared convolutions are competitive (especially when just shared inside of a head and not across heads), there is a slight drop-off.

Table 15: Comparison of post-compression (pre-RFT) perplexity on WikiText-2 for DeltaNet 370M. We compare standard (non-shared) vs. shared convolutions under different pruning strategies. "PCA (adv.)" removes the highest variance components (keeping noise), while "PCA (prop.)" removes the lowest variance components (keeping signal).

Comp.	Method	Non Shared	Shared
75%	PCA (adv.)	$3.1 \cdot 10^5$	$2.1 \cdot 10^5$
	PCA (prop.)	$3.3 \cdot 10^5$	184.48
	Grad	46.7	69.78
50%	PCA (adv.)	$2.4 \cdot 10^5$	$6.1 \cdot 10^4$
	PCA (prop.)	$3.2 \cdot 10^5$	155.76
	Grad	31.75	33.73
0%	–	29.83	29.55

C.6.2 Impact of Convolutions on PCA-based Pruning

Towards quantifying the two approaches of handling the per-channel convolutions, we compare a "Proper PCA" method, where we retain the principal components with the highest variance, against an "Adversarial PCA" baseline, where we deliberately retain the dimensions with the lowest variance. In a system robust to rotation, proper PCA should outperform the adversarial baseline.

Table 15 presents the perplexity on WikiText-2 for DeltaNet 370M (pre-RFT). For the shared convolutions, proper PCA generally yields way better perplexity than the adversarial baseline. However, in the standard, non-shared case, the filter averaging does not suffice to make up for the misalignment post-transformation. Furthermore, the Grad pruning method still outperforms PCA even when using shared convolutions.

C.7 On Coupled Selection

We compare selecting indices based on (i) the sum of scores derived from both projections ($\mathbf{K}$, $\mathbf{Q}$), (ii) keys only ($\mathbf{K}$), and (iii) queries only ($\mathbf{Q}$). Results are reported in Table 16.

We observe that selecting columns based on query projections ($\mathbf{Q}$) consistently outperforms selection based solely on keys ($\mathbf{K}$) for magnitude-based methods. Since queries govern readout, pruning based on $\mathbf{Q}$ ensures we discard dimensions with minimal contribution to the output. In contrast, pruning based on $\mathbf{K}$ risks removing information that the model attempts to access with a strong query, leading to significant readout errors.

Table 16: Consolidated ablation study on DeltaNet and Gated DeltaNet models (370M and 1.3B) measuring WikiText-2 perplexity. We compare the impact of picking just keys ($\mathbf{K}$), just queries ($\mathbf{Q}$), or both ($\mathbf{K}, \mathbf{Q}$) for feature selection at a 50% compression ratio.

	DeltaNet						Gated DeltaNet					
	370M			1.3B			370M			1.3B		
Method	$\mathbf{K}, \mathbf{Q}$	**K**	**Q**	$\mathbf{K}, \mathbf{Q}$	**K**	**Q**	$\mathbf{K}, \mathbf{Q}$	**K**	**Q**	$\mathbf{K}, \mathbf{Q}$	**K**	**Q**
L1	3843.5	425218.3	33.0	66.1	1596.3	34.9	41.8	61.0	49.9	26.5	29.75	24.5
DRRQR	31.4	286035.7	32.1	20.5	566.9	22.9	31.6	56.9	43.5	17.3	21.9	22.0
Grad	31.7	17709.6	31.9	18.3	20.3	19.5	33.3	36.2	34.6	17.4	18.3	17.7
S-Wanda	915.9	376074.0	33.0	60.0	1604.4	29.1	40.0	61.3	43.5	21.9	22.5	21.5
Baseline	*29.8*			*16.7*			*28.8*			*16.8*		

Curiously, magnitude-based methods often perform better when selecting just based on queries ($\mathbf{Q}$) than when selecting based on both keys and queries ($\mathbf{K}, \mathbf{Q}$). Since they add the scores of keys and queries ($s_j = \|\mathbf{W}_{k,:j}\| + \|\mathbf{W}_{q,:j}\|$), this suggests the model contains large key weights whose corresponding query weights have lower magnitude. DRRQR avoids this by targeting the effective rank of the joint subspace.

The results in Table 16 reveal a clear difference between the studied pruning methods. Magnitude-based methods (L1, Wanda) are unstable when targeting keys, often performing best when restricted to queries. In contrast, optimization-based methods (Grad, DRRQR) consistently achieve the lowest perplexity using joint selection ($\mathbf{K}, \mathbf{Q}$). This indicates that the associative memory's effective rank requires accounting for the coupled interaction between keys and queries, rather than treating them in isolation.

D The Effective Rank

In this section we present some properties of the effective rank. Ipsen & Saibaba (2025) provide more details.

Definition D.1 (effective rank)**.** For a non-zero matrix $\mathbf{A} \in \mathbb{R}^{m,n}$, the effective rank is defined as:

$$\mathrm{er}(\mathbf{A}) := \frac{\|\mathbf{A}\|_F^2}{\|\mathbf{A}\|_2^2} = \frac{\sum_i \sigma_i^2(\mathbf{A})}{\sigma_{\max}^2(\mathbf{A})}.$$

Unlike the algebraic rank, which is discontinuous, the effective rank is a continuous function of the matrix entries. This implies that small perturbations to the memory state $\mathbf{S}_t$ (e.g., from gradient noise or quantization) result in bounded changes to $\mathrm{er}(\mathbf{S}_t)$.

Proposition D.2 (Invariance under Transposition)**.** *The effective rank is invariant under transposition. For any matrix* $\mathbf{A}$*:*

$$\mathrm{er}(\mathbf{A}) = \mathrm{er}(\mathbf{A}^\top).$$

Proposition D.3 (Invariance under Unitary Transformations and Scaling)**.** *The effective rank is invariant under unitary transformations and scalar multiplication. For any unitary matrices* $\mathbf{U}, \mathbf{V}$ *and non-zero scalar* $c \in \mathbb{R}$*:*

$$\mathrm{er}(c\mathbf{U}\mathbf{A}\mathbf{V}^\top) = \mathrm{er}(\mathbf{A}).$$

Proposition D.4 (Bounds and Relation to Algebraic Rank)**.** *The effective rank is bounded by the algebraic rank:*

$$1 \le \mathrm{er}(\mathbf{A}) \le \mathrm{rank}(\mathbf{A}) \le \min(m, n).$$

The lower bound is achieved if and only if $\mathbf{A}$ *has rank 1. The upper bound is achieved if and only if all non-zero singular values are equal.*

Proposition D.5 (Relation to Condition Number)**.** *Let* $\mathbf{A}$ *be a (non-zero) matrix. Then*

$$\kappa^2(\mathbf{A}) \ge \frac{\mathrm{rank}(\mathbf{A})}{\mathrm{er}(\mathbf{A})}.$$

In the specific case where $\mathbf{A}$ *has full rank, this implies*

$$\kappa^2(\mathbf{A}) \geq \frac{1}{\mathrm{u}(\mathbf{A})},$$

where u *is the rank utilization defined in Equation* (5)*.*

Edge Case (Isotropy): For a full-rank matrix $\mathbf{A}$ with $\kappa(\mathbf{A}) = 1$, let $r = \mathrm{rank}(\mathbf{A}) = \min(m, n)$. The inequality becomes an equality:

$$1 \geq \frac{r}{\mathrm{er}(\mathbf{A})} \implies \mathrm{er}(\mathbf{A}) \geq r.$$

Since we know $\mathrm{er}(\mathbf{A}) \leq r$, this forces $\mathrm{er}(\mathbf{A}) = \mathrm{rank}(\mathbf{A})$. This confirms that for isotropic matrices (where all non-zero singular values are equal), the effective rank and algebraic rank coincide. Conversely, a large gap between r and $\mathrm{er}(\mathbf{A})$ is a sufficient condition for ill-conditioning.

E Dynamical Systems Perspective

In this section, we derive the DeltaNet recurrence rule from a continuous-time dynamical systems perspective. We show that the sequence mixer can be interpreted as a discretization of a continuous gradient flow minimizing a linear regression objective.

Continuous-Time Dynamics. Consider a time-continuous associative memory $\mathbf{S}(t) \in \mathbb{R}^{d_v, d_k}$ receiving a stream of keys $\mathbf{k}(t) \in \mathbb{R}^{d_k}$ and values $\mathbf{v}(t) \in \mathbb{R}^{d_v}$. We define the instantaneous regression loss at time t as:

$$\mathcal{L}(\mathbf{S}(t)) = \frac{1}{2}\|\mathbf{v}(t) - \mathbf{S}(t)\mathbf{k}(t)\|_2^2.$$

The dynamics of the state $\mathbf{S}(t)$ are governed by the gradient flow minimizing this objective:

$$\dot{\mathbf{S}}(t) = -\nabla_{\mathbf{S}}\mathcal{L}(\mathbf{S}(t)) \tag{11}$$

$$= -(\mathbf{S}(t)\mathbf{k}(t) - \mathbf{v}(t))\mathbf{k}(t)^\top \tag{12}$$

$$= \mathbf{v}(t)\mathbf{k}(t)^\top - \mathbf{S}(t)\mathbf{k}(t)\mathbf{k}(t)^\top. \tag{13}$$

Equation (13) represents a linear time-varying (LTV) ordinary differential equation (ODE) of the form $\dot{\mathbf{S}}(t) = \mathbf{S}(t)\mathbf{A}(t) + \mathbf{B}(t)$, where $\mathbf{A}(t) = -\mathbf{k}(t)\mathbf{k}(t)^\top$ is the state-transition matrix acting on the right, and $\mathbf{B}(t) = \mathbf{v}(t)\mathbf{k}(t)^\top$ is the input forcing term.

Euler Discretization. To obtain the discrete-time update rule employed by DeltaNet, we apply the forward Euler method to Equation (13). Let Δt be the step size, which corresponds to the gating factor β_t in the DeltaNet formulation. The discretization yields:

$$\frac{\mathbf{S}_t - \mathbf{S}_{t-1}}{\beta_t} \approx \mathbf{v}_t\mathbf{k}_t^\top - \mathbf{S}_{t-1}\mathbf{k}_t\mathbf{k}_t^\top.$$

Rearranging the terms to solve for the next state $\mathbf{S}_t$:

$$\begin{aligned}\mathbf{S}_t &= \mathbf{S}_{t-1} + \beta_t(\mathbf{v}_t\mathbf{k}_t^\top - \mathbf{S}_{t-1}\mathbf{k}_t\mathbf{k}_t^\top) \\ &= \mathbf{S}_{t-1}(\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^\top) + \beta_t\mathbf{v}_t\mathbf{k}_t^\top.\end{aligned}$$

This exactly recovers the DeltaNet update rule (Equation (2)).

System Stability. The stability of this dynamical system is determined by the spectral properties of the transition matrix $(\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^\top)$. For the system to be stable (non-divergent), the singular values of this operator must lie within the unit circle. This implies the condition $|1 - \beta_t\|\mathbf{k}_t\|_2^2| \leq 1$. In standard DeltaNet implementations, keys are normalized ($\|\mathbf{k}_t\|_2 = 1$), and β_t is the output of a sigmoid function ($\beta_t \in (0, 1)$), strictly satisfying the stability condition and ensuring the memory decays appropriately over time.

F PROOFS

In this section, we provide proofs of statements presented in the main paper.

F.1 PROOF OF THE EFFECTIVE-RANK BOUND

We prove Equation (4). Using the column-stacking convention of the main text, write the associative memory as $\mathbf{S} = \mathbf{V}\mathbf{K}^\top$, where $\mathbf{V} \in \mathbb{R}^{d_v,T}$ and $\mathbf{K} \in \mathbb{R}^{d_k,T}$. Assume $\mathbf{K}$ has full row rank.

To handle potential misalignment between the subspaces of values and keys, we decompose the values $\mathbf{V}$ into two orthogonal components relative to the row space of the keys $\mathbf{K}$:

$$\mathbf{V} = \mathbf{V}_\parallel + \mathbf{V}_\perp,$$

where the rows of $\mathbf{V}_\parallel$ lie in $\mathrm{row}(\mathbf{K})$, and the rows of $\mathbf{V}_\perp$ are orthogonal to it. Consequently, $\mathbf{V}_\perp\mathbf{K}^\top = \mathbf{0}$, and the memory state simplifies to:

$$\mathbf{S} = (\mathbf{V}_\parallel + \mathbf{V}_\perp)\mathbf{K}^\top = \mathbf{V}_\parallel\mathbf{K}^\top.$$

Assume $\mathbf{V}_\parallel \neq \mathbf{0}$. We define the scalar quantity $\nu(\mathbf{V})$ appearing in the main text as the effective rank of the projected values:

$$\nu(\mathbf{V}) \coloneqq \mathrm{er}(\mathbf{V}_\parallel) = \frac{\|\mathbf{V}_\parallel\|_F^2}{\|\mathbf{V}_\parallel\|_2^2}.$$

The quantity $\nu(\mathbf{V})$ also depends on $\mathbf{K}$ through this projection. Since multiplication by $\mathbf{K}^\top$ is injective on $\mathrm{row}(\mathbf{K})$, these assumptions ensure $\mathbf{S} \neq \mathbf{0}$. We now derive the lower bound for the effective rank $\mathrm{er}(\mathbf{S}) = \|\mathbf{S}\|_F^2/\|\mathbf{S}\|_2^2$.

First, we bound the numerator (Frobenius norm) from below. We use the property $\|\mathbf{AB}\|_F \geq \|\mathbf{A}\|_F\sigma_{\min}(\mathbf{B})$, which holds here because the rows of $\mathbf{V}_\parallel$ lie in $\mathrm{col}(\mathbf{K}^\top)$:

$$\|\mathbf{S}\|_F^2 = \|\mathbf{V}_\parallel\mathbf{K}^\top\|_F^2 \geq \|\mathbf{V}_\parallel\|_F^2\sigma_{\min}^2(\mathbf{K}).$$

Next, we bound the denominator (Spectral norm) from above using the standard sub-multiplicative property $\|\mathbf{AB}\|_2 \leq \|\mathbf{A}\|_2\|\mathbf{B}\|_2$:

$$\|\mathbf{S}\|_2^2 = \|\mathbf{V}_\parallel\mathbf{K}^\top\|_2^2 \leq \|\mathbf{V}_\parallel\|_2^2\|\mathbf{K}\|_2^2.$$

Combining these inequalities yields the lower bound:

$$\mathrm{er}(\mathbf{S}) = \frac{\|\mathbf{S}\|_F^2}{\|\mathbf{S}\|_2^2} \geq \frac{\|\mathbf{V}_\parallel\|_F^2\sigma_{\min}^2(\mathbf{K})}{\|\mathbf{V}_\parallel\|_2^2\|\mathbf{K}\|_2^2} = \mathrm{er}(\mathbf{V}_\parallel)\frac{1}{\kappa^2(\mathbf{K})} = \frac{\nu(\mathbf{V})}{\kappa^2(\mathbf{K})}.$$

□

F.2 PROOF OF ALGEBRAIC RANK

We first prove the following proposition:

Proposition F.1. *Let $\mathbf{S}_0 = 0$ and $\mathbf{S}_t$ be a matrix satisfying the recursion*

$$\mathbf{S}_t = \mathbf{S}_{t-1}(\alpha_t\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^\top) + \gamma_t\mathbf{v}_t\mathbf{k}_t^\top$$

for some non-zero scalars $\alpha_t, \beta_t, \gamma_t$ and some vectors $\mathbf{v}_t$, $\mathbf{k}_t$. Then

$$\mathrm{row}\,\mathbf{S}_t \subseteq \mathrm{span}\langle\mathbf{k}_1, \ldots, \mathbf{k}_t\rangle \quad \textit{and} \quad \mathrm{col}\,\mathbf{S}_t \subseteq \mathrm{span}\langle\mathbf{v}_1, \ldots, \mathbf{v}_t\rangle.$$

Consequently,

$$\mathrm{rank}\,\mathbf{S}_t \leq \min\left(\mathrm{rank}\,\mathbf{K}_t, \mathrm{rank}\,\mathbf{V}_t\right)$$

where $\mathbf{K}_t = (\mathbf{k}_1, \ldots, \mathbf{k}_t)$ and $\mathbf{V}_t = (\mathbf{v}_1, \ldots, \mathbf{v}_t)$ are the matrices obtained by stacking the $\mathbf{k}$- and $\mathbf{v}$-vectors, respectively.

Proof. We start by showing the first claim by induction. For $t = 1$, we have $\mathbf{S}_1 = \gamma_1 \mathbf{v}_1 \mathbf{k}_1^\top$. Consider $t > 1$ and assume the claim is true for every $s < t$. Then

$$\begin{aligned}\mathbf{S}_t &= \mathbf{S}_{t-1}(\alpha_t \mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) + \gamma_t \mathbf{v}_t \mathbf{k}_t^\top \\ &= \alpha_t \mathbf{S}_{t-1} + (\gamma_t \mathbf{v}_t - \beta_t \mathbf{S}_{t-1}\mathbf{k}_t)\mathbf{k}_t^\top \\ &= \alpha_t \mathbf{S}_{t-1} + \mathbf{u}_t \mathbf{k}_t^\top,\end{aligned}$$

where we defined $\mathbf{u}_t \coloneqq \gamma_t \mathbf{v}_t - \beta_t \mathbf{S}_{t-1}\mathbf{k}_t$. Next, we use that for two matrices $\mathbf{A}$ and $\mathbf{B}$,

$$\text{row}(\mathbf{A} + \mathbf{B}) \subseteq \text{span}(\text{row}(\mathbf{A}), \text{row}(\mathbf{B}))$$

and thus

$$\begin{aligned}\text{row}(\alpha_t \mathbf{S}_{t-1} + \mathbf{u}_t \mathbf{k}_t^\top) &\subseteq \text{span}(\text{row}(\alpha_t \mathbf{S}_{t-1}), \text{row}(\mathbf{u}_t \mathbf{k}_t^\top)) \\ &\subseteq \text{span}(\text{row}(\mathbf{S}_{t-1}), \mathbf{k}_t) \\ &\subseteq \text{span}(\mathbf{k}_1, ..., \mathbf{k}_t).\end{aligned}$$

This concludes the first claim.

For the second claim, we proceed analogously.

Indeed, we again show this claim by induction. The case $t = 1$ is clear. For any $t > 1$, we compute, using $\text{col}(\mathbf{AB}) \subseteq \text{col}(\mathbf{A})$,

$$\begin{aligned}\text{col}(\mathbf{S}_t) &\subseteq \text{col}(\mathbf{S}_{t-1}(\alpha_t \mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top), \gamma_t \mathbf{v}_t \mathbf{k}_t^\top) \\ &\subseteq \text{col}(\mathbf{S}_{t-1}, \gamma_t \mathbf{v}_t \mathbf{k}_t^\top) \\ &\subseteq \text{col}(\mathbf{v}_1, \ldots, \mathbf{v}_t).\end{aligned}$$

This concludes the proof. □

Interestingly, PCA-based transformations are guaranteed not to decrease the rank utilization of the keys:

Lemma F.2 (Monotonicity of Rank Utilization)**.** *For a non-zero key matrix with at least d_k samples, pruning the low-variance directions via PCA increases or maintains rank utilization. That is, for any $1 \leq d_k' < d_k$:*

$$u(\mathbf{K}') \geq u(\mathbf{K}).$$

Proof. Assume towards a contradiction that the utilization decreases, i.e., $u(\mathbf{K}') < u(\mathbf{K})$. This means that

$$\frac{1}{d_k'} \sum_{i=1}^{d_k'} \sigma_i^2 < \frac{1}{d_k} \sum_{i=1}^{d_k} \sigma_i^2.$$

In words, the average energy of the top d_k' principal components is strictly less than the average energy of the full spectrum. This is a contradiction, as the singular values are non-increasing ($\sigma_1 \geq \cdots \geq \sigma_{d_k}$). Thus, our assumption must have been wrong. □

F.3 Proof of Proposition 2.1

Assume $\mathbf{S} \in \mathbb{R}^{d_v, d_k}$ has full column rank, so $d = d_k$ and $\sigma_{d_k} > 0$. Since R is invariant to rescaling the query, take $\|\mathbf{q}^*\|_2 = 1$. For nonzero $\boldsymbol{\epsilon}$, write

$$R(\mathbf{q}^*, \boldsymbol{\epsilon}) = \frac{\|\mathbf{S}\boldsymbol{\epsilon}\|_2 / \|\boldsymbol{\epsilon}\|_2}{\|\mathbf{S}\mathbf{q}^*\|_2} \leq \frac{\sigma_1}{\|\mathbf{S}\mathbf{q}^*\|_2}.$$

Worst-Case Amplification. Let $\mathbf{w}_i$ be right singular vectors of $\mathbf{S}$, ordered by decreasing singular value. For each fixed query, the upper bound is attained by taking $\boldsymbol{\epsilon} = \mathbf{w}_1$. Hence

$$\max_{\boldsymbol{\epsilon} \neq \mathbf{0}} R(\mathbf{q}^*, \boldsymbol{\epsilon}) = \frac{\sigma_1}{\|\mathbf{S}\mathbf{q}^*\|_2}.$$

For unit-norm queries, $\|\mathbf{S}\mathbf{q}^*\|_2 \geq \sigma_{d_k}$, with equality at $\mathbf{q}^* = \mathbf{w}_{d_k}$. Maximizing over the query therefore gives

$$\max_{\|\mathbf{q}^*\|_2=1} \max_{\boldsymbol{\epsilon}\neq\mathbf{0}} R(\mathbf{q}^*, \boldsymbol{\epsilon}) = \frac{\sigma_1}{\sigma_{d_k}} = \kappa(\mathbf{S}).$$

Furthermore,

$$u(\mathbf{S}) = \frac{1}{d_k}\sum_{i=1}^{d_k}\frac{\sigma_i^2}{\sigma_1^2} \geq \frac{\sigma_{d_k}^2}{\sigma_1^2} = \frac{1}{\kappa(\mathbf{S})^2}.$$

Rearranging proves Equation (7).

Isotropic Endpoint. Under the full-column-rank assumption, $u(\mathbf{S}) = 1$ holds if and only if all d_k singular values equal σ_1. Then $\mathbf{S}^\top\mathbf{S} = \sigma_1^2\mathbf{I}_{d_k}$, so $\|\mathbf{S}\mathbf{x}\|_2 = \sigma_1\|\mathbf{x}\|_2$ for every $\mathbf{x}$, and $R = 1$ for all nonzero queries and perturbations. □

F.4 Conditioning of Query Weight Gradients

For the linear query map $\mathbf{q} = \mathbf{W}_\mathbf{Q}^\top\mathbf{x}$, hold the state $\mathbf{S}$ and nonzero input $\mathbf{x}$ fixed. The readout $\mathbf{o} = \mathbf{S}\mathbf{W}_\mathbf{Q}^\top\mathbf{x}$ has Jacobian

$$\mathbf{J} := \frac{\partial \mathbf{o}}{\partial \operatorname{vec}(\mathbf{W}_\mathbf{Q})} = \mathbf{S} \otimes \mathbf{x}^\top.$$

For a scalar loss $\mathcal{L}$ with output gradient $\mathbf{g} = \nabla_\mathbf{o}\mathcal{L}$, the backward map is

$$\nabla_{\operatorname{vec}(\mathbf{W}_\mathbf{Q})}\mathcal{L} = \mathbf{J}^\top\mathbf{g} = (\mathbf{S}^\top \otimes \mathbf{x})\mathbf{g}.$$

If $\mathbf{S}$ has full column rank, the nonzero singular values of this map are $\|\mathbf{x}\|_2\sigma_i(\mathbf{S})$, for $i = 1, \ldots, d_k$. Their ratio is therefore

$$\frac{\sigma_{\max}(\mathbf{J}^\top)}{\sigma_{\min}^+(\mathbf{J}^\top)} = \kappa(\mathbf{S}) \geq \frac{1}{\sqrt{u(\mathbf{S})}},$$

where $\sigma_{\min}^+$ denotes the smallest nonzero singular value. Thus low rank utilization makes the backward map ill-conditioned on its nonzero singular subspaces. This statement concerns the local linear readout; its effect on a particular gradient depends on the direction of $\mathbf{g}$.

Derivation. The vectorization identity (Petersen et al., 2008, Equation (520)) gives

$$\mathbf{o} = \operatorname{vec}(\mathbf{o}^\top) = \operatorname{vec}(\mathbf{x}^\top\mathbf{W}_\mathbf{Q}\mathbf{S}^\top) = (\mathbf{S} \otimes \mathbf{x}^\top)\operatorname{vec}(\mathbf{W}_\mathbf{Q}).$$

Differentiating yields $\mathbf{J}$, and the chain rule gives the backward map. The singular values of a Kronecker product are the pairwise products of those of its factors (Horn & Johnson, 1991, Theorem 4.2.15). Since $\mathbf{x}^\top$ has the single nonzero singular value $\|\mathbf{x}\|_2$, the stated singular values follow. Finally, the proof of Proposition 2.1 gives $\kappa(\mathbf{S}) \geq 1/\sqrt{u(\mathbf{S})}$. □

G Extended Results

G.1 End-to-End Efficiency

We benchmark DeltaNet and Gated DeltaNet 1.3B end to end on an H100-SXM with CUDA-graph capture. All configurations use the same unmodified flash-linear-attention kernels, including the standard chunked kernels for prefill. State reduction changes only d_k'. We use 10 warmup steps and 100 measured steps and report the median across three independent trials. Table 17 reports the complete decode results for prompt length 512, and Table 18 the complete prefill grid.

For comparison, Table 19 isolates the sequence mixer. These kernel-level measurements expose the computational benefit of reducing d_k, but exclude projections, normalization, feed-forward layers, and other full-model costs captured by Tables 17 and 18.

Table 17: Complete full-model decode results on an H100-SXM for DeltaNet and Gated DeltaNet 1.3B (prompt length 512, batch size B). Base is ms/token; relative entries are speedup.

	GDN			DN		
B	BASE	50%	75%	BASE	50%	75%
1	2.72	1.03×	1.05×	2.22	1.03×	1.04×
2	2.85	1.06×	1.09×	2.31	1.06×	1.07×
4	3.18	1.10×	1.16×	2.56	1.09×	1.14×
8	3.79	1.17×	1.28×	2.99	1.14×	1.23×
16	4.96	1.25×	1.45×	3.88	1.23×	1.38×
32	7.20	1.37×	1.67×	5.48	1.31×	1.57×
64	11.61	1.47×	1.94×	8.68	1.41×	1.78×

Table 18: Complete full-model prefill speedup grid on an H100-SXM for DeltaNet and Gated DeltaNet 1.3B, over all measured batch sizes B and context lengths T. Cells absent from the grid are configurations where the chunked kernel exceeds device memory.

$B \times T$	GDN 50%	GDN 75%	DN 50%	DN 75%
1 × 2K	1.00×	1.01×	1.04×	1.01×
2 × 2K	1.00×	1.01×	1.03×	1.00×
4 × 2K	1.16×	1.24×	1.12×	1.19×
8 × 2K	1.14×	1.21×	1.12×	1.18×
16 × 2K	1.14×	1.22×	1.12×	1.18×
32 × 2K	1.14×	1.21×	1.12×	1.19×
64 × 2K	1.14×	1.22×	1.12×	1.19×
128 × 2K	1.14×	1.22×	1.12×	1.19×
1 × 8K	1.16×	1.21×	1.14×	1.19×
2 × 8K	1.14×	1.21×	1.12×	1.18×
4 × 8K	1.14×	1.21×	1.11×	1.18×
8 × 8K	1.15×	1.22×	1.12×	1.18×
16 × 8K	1.14×	1.22×	1.12×	1.18×
32 × 8K	1.14×	1.22×	1.12×	1.18×
1 × 16K	1.15×	1.22×	1.12×	1.18×
2 × 16K	1.14×	1.20×	1.12×	1.18×
4 × 16K	1.15×	1.22×	1.12×	1.18×
8 × 16K	1.14×	1.21×	1.12×	1.18×
16 × 16K	1.14×	1.22×	1.12×	1.18×

Table 19: Gated DeltaNet 1.3B sequence-mixer-only prefill speedup and peak-VRAM ratio relative to the uncompressed baseline at the same batch size. Measurements use an NVIDIA H100 ($T = 2048$, 16 heads, $h = 2048$, $d_v = 128$).

Batch size	State reduction	Speedup	VRAM ratio
8	50%	1.26×	0.73×
	75%	1.32×	0.60×
16	50%	1.28×	0.73×
	75%	1.50×	0.59×
32	50%	1.28×	0.72×
	75%	1.51×	0.59×

G.2 Language Modeling

This subsection contains the extensive zero-shot (Gao et al., 2024) task evaluations. Table 20 contains averaged results for all models at a fixed compression ratio of 50%.

Table 20: Comparison of compression methods at a 50% compression ratio across all model sizes, both pre- and post-RFT. We report perplexity for WikiText (Wiki) and Lambada (LMB), as well as average accuracy for LM Evals (best results in **bold**, second best underlined).

		Pre-RFT			Post-RFT		
Model	Method	Wiki ↓	LMB ↓	LM Evals Avg ↑	Wiki ↓	LMB ↓	LM Evals Avg ↑
DeltaNet 370M	L1	3843.5	57304.8	33.0	32.5	56.8	43.0
	Rand	1032.6	2882.8	34.3	32.6	49.9	42.6
	DRRQR	**31.4**	36.0	44.7	**29.4**	36.6	**44.5**
	Grad	31.7	**33.3**	**44.9**	**29.4**	**36.3**	44.4
	S-Wanda	915.9	6118.0	35.1	32.1	53.2	43.4
	Baseline	29.8	37.0	44.2	29.8	37.0	44.2
DeltaNet 1.3B	L1	66.1	298.7	41.6	19.0	16.1	48.2
	Rand	41.2	68.0	44.4	19.4	14.4	48.2
	DRRQR	20.5	47.2	45.7	17.5	11.8	49.7
	Grad	**18.3**	**15.4**	**48.8**	**17.2**	**11.3**	**50.3**
	S-Wanda	60.0	361.3	40.8	18.5	15.9	48.3
	Baseline	16.7	11.9	50.0	16.7	11.9	50.0
Gated DeltaNet 370M	L1	41.8	156.6	40.4	29.2	44.6	43.5
	Rand	35.5	50.7	43.4	29.3	46.5	43.6
	DRRQR	**31.6**	39.4	**44.0**	**28.7**	40.6	**43.8**
	Grad	33.3	**39.2**	43.8	**28.7**	**39.4**	43.7
	S-Wanda	40.0	141.5	40.9	29.2	44.6	43.5
	Baseline	28.8	35.9	44.4	28.8	35.9	44.4
Gated DeltaNet 1.3B	L1	26.5	34.0	53.3	16.8	13.0	57.2
	Rand	18.8	14.4	56.5	16.9	11.5	57.8
	DRRQR	**17.3**	14.2	57.5	**16.3**	12.0	58.0
	Grad	17.4	**10.1**	**58.6**	16.4	**10.5**	**58.3**
	S-Wanda	21.9	24.1	55.1	16.6	11.7	57.8
	Baseline	16.8	9.7	59.4	16.8	9.7	59.4

Table 21: Zero-shot performance of DeltaNet 370M models, evaluated using the lm-eval-harness (Gao et al., 2024), given different compression ratios. Pre- and post RFT.

RFT	Method	Compr.	Wiki. ↓ ppl	LMB. ↓ ppl	ARC-e ↑ acc_n	ARC-c ↑ acc_n	Hella. ↑ acc_n	Wino. ↑ acc	PIQA ↑ acc_n	LMB. ↑ acc	Avg ↑
×	L1	75%	100083.0	9661610.6	27.2	28.0	27.7	52.2	50.9	0.0	31.0
×	Rand	75%	136955.3	5879873.3	27.0	27.4	26.6	48.9	51.4	0.0	30.2
×	DRRQR	75%	**42.0**	74.1	51.9	27.5	38.7	50.9	64.6	25.3	43.1
×	Grad	75%	46.7	**58.0**	52.0	27.0	38.5	49.8	64.7	29.1	**43.5**
×	S-Wanda	75%	33512.5	4955134.6	28.5	26.8	27.7	51.1	48.9	0.0	30.5
✓	L1	75%	36.3	146.7	46.0	26.2	36.1	51.0	63.0	17.1	39.9
✓	Rand	75%	40.3	129.9	46.8	26.1	35.8	51.9	62.8	17.7	40.2
✓	DRRQR	75%	**31.4**	43.7	52.0	27.5	38.1	52.4	64.6	29.7	44.0
✓	Grad	75%	31.5	**42.5**	52.1	28.0	38.0	51.6	64.8	30.2	**44.1**
✓	S-Wanda	75%	35.4	101.7	46.5	26.7	37.1	50.9	63.3	20.3	40.8
×	L1	50%	3843.5	57304.8	35.8	23.6	31.2	50.8	55.0	1.6	33.0
×	Rand	50%	1032.6	2882.8	37.8	23.7	30.3	50.5	56.6	6.9	34.3
×	DRRQR	50%	**31.4**	36.0	52.1	27.6	38.7	52.0	65.1	32.7	44.7
×	Grad	50%	31.7	**33.3**	51.8	27.9	38.7	52.2	65.5	33.4	**44.9**
×	S-Wanda	50%	915.9	6118.0	39.9	24.3	33.5	50.1	57.4	5.4	35.1
✓	L1	50%	32.5	56.8	49.2	27.0	37.7	53.0	64.6	26.4	43.0
✓	Rand	50%	32.6	49.9	49.2	27.9	37.6	49.0	64.9	27.3	42.6
✓	DRRQR	50%	**29.4**	36.6	51.8	27.7	38.1	52.1	65.3	31.7	**44.5**
✓	Grad	50%	**29.4**	**36.3**	51.0	27.7	38.1	52.1	65.5	32.0	44.4
✓	S-Wanda	50%	32.1	53.2	51.1	26.1	37.9	52.6	65.0	27.7	43.4
×	L1	40%	1232.0	5431.6	39.6	23.9	34.0	50.2	56.4	5.7	34.9
×	Rand	40%	121.3	329.1	42.8	25.4	34.2	49.8	61.5	15.0	38.1
×	DRRQR	40%	**30.2**	33.4	52.6	28.0	38.7	51.9	65.2	33.1	**44.9**
×	Grad	40%	30.4	**32.7**	52.2	27.7	38.7	51.8	65.3	33.4	**44.9**
×	S-Wanda	40%	276.8	603.4	45.3	25.4	35.8	50.3	60.3	13.4	38.4
✓	L1	40%	31.7	53.9	50.8	26.7	37.7	53.0	65.1	26.5	43.3
✓	Rand	40%	31.0	43.6	49.1	27.6	38.0	52.5	65.2	29.2	43.6
✓	DRRQR	40%	**29.0**	36.3	51.4	27.8	38.2	52.0	65.4	32.0	**44.5**
✓	Grad	40%	29.1	**35.7**	51.5	27.4	38.2	51.5	65.5	32.1	44.4
✓	S-Wanda	40%	31.2	48.3	50.7	26.9	38.0	52.4	64.8	28.8	43.6
×	L1	30%	270.1	307.0	45.6	24.7	36.2	50.2	60.1	18.5	39.2
×	Rand	30%	52.7	103.2	48.4	26.8	36.1	50.4	62.9	22.7	41.2
×	DRRQR	30%	**29.3**	**32.7**	52.4	27.7	38.6	52.7	65.0	33.5	**45.0**
×	Grad	30%	29.4	33.0	51.7	27.9	38.6	52.5	64.7	33.4	44.8
×	S-Wanda	30%	138.9	133.9	46.3	26.0	37.1	50.0	62.0	22.7	40.7
✓	L1	30%	30.8	50.9	51.1	27.0	37.9	53.6	64.7	27.4	43.6
✓	Rand	30%	30.0	40.7	50.1	27.6	38.1	52.2	65.5	30.3	44.0
✓	DRRQR	30%	**28.8**	**36.0**	51.3	27.6	38.2	52.1	65.1	32.3	44.4
✓	Grad	30%	**28.8**	**36.0**	51.3	27.7	38.1	51.9	65.5	32.3	**44.5**
✓	S-Wanda	30%	30.7	45.4	51.2	27.2	38.0	52.6	64.9	29.6	43.9
Baseline	–	0%	29.8	37.0	51.1	27.6	38.1	52.2	65.0	31.3	44.2

Table 22: Zero-shot performance of DeltaNet 1.3B models, evaluated using the lm-eval-harness (Gao et al., 2024), given different compression ratios. Pre- and post RFT.

RFT	Method	Compr.	Wiki. ↓ ppl	LMB. ↓ ppl	ARC-e ↑ acc_n	ARC-c ↑ acc_n	Hella. ↑ acc_n	Wino. ↑ acc	PIQA ↑ acc_n	LMB. ↑ acc	Avg ↑
×	L1	75%	2860.8	6728.9	36.0	26.4	41.5	51.9	62.4	2.6	36.8
×	Rand	75%	53664.2	3585.3	37.4	26.9	36.5	50.2	60.3	3.6	35.8
×	DRRQR	75%	43.9	375.5	43.6	27.7	49.1	51.3	69.5	9.9	41.8
×	Grad	75%	**27.0**	**61.5**	49.7	27.0	49.3	52.3	69.8	22.4	**45.1**
×	S-Wanda	75%	3160.0	6856.0	37.0	27.4	40.5	51.8	64.3	2.0	37.2
✓	L1	75%	20.2	26.6	48.1	25.8	47.3	51.1	68.7	33.1	45.7
✓	Rand	75%	22.9	22.9	47.8	27.1	46.5	51.5	69.7	35.3	46.3
✓	DRRQR	75%	19.2	17.5	50.5	27.3	48.7	53.5	69.6	40.5	48.4
✓	Grad	75%	**19.0**	**14.7**	50.7	26.8	49.1	52.4	69.9	43.6	**48.8**
✓	S-Wanda	75%	20.0	23.1	48.6	26.4	47.2	51.9	69.5	35.6	46.5
×	L1	50%	66.1	298.7	42.4	27.2	47.9	54.4	67.1	10.6	41.6
×	Rand	50%	41.2	68.0	46.9	27.8	47.9	53.7	69.5	20.4	44.4
×	DRRQR	50%	20.5	47.2	46.5	27.8	51.3	53.0	69.9	25.5	45.7
×	Grad	50%	**18.3**	**15.4**	48.8	27.5	50.9	53.4	70.2	42.2	**48.8**
×	S-Wanda	50%	60.0	361.3	40.8	27.0	47.1	53.7	66.4	9.9	40.8
✓	L1	50%	19.0	16.1	48.9	27.1	48.6	53.7	69.7	41.1	48.2
✓	Rand	50%	19.4	14.4	48.5	26.2	48.6	52.5	69.9	43.4	48.2
✓	DRRQR	50%	17.5	11.8	51.0	26.6	49.7	53.3	69.7	48.2	49.7
✓	Grad	50%	**17.2**	**11.3**	51.2	26.5	49.9	54.5	70.7	48.7	**50.3**
✓	S-Wanda	50%	18.5	15.9	49.7	27.5	48.5	52.0	70.2	41.5	48.3
×	L1	40%	45.2	194.4	44.1	27.3	48.2	53.8	68.5	13.6	42.6
×	Rand	40%	23.9	30.0	49.3	27.6	49.1	52.3	69.3	30.6	46.4
×	DRRQR	40%	19.3	37.8	47.8	28.3	52.0	54.4	69.5	28.4	46.7
×	Grad	40%	**17.9**	**13.0**	48.8	26.9	51.0	53.5	70.3	45.6	**49.4**
×	S-Wanda	40%	44.7	189.0	42.6	27.6	47.8	54.1	68.3	13.7	42.4
✓	L1	40%	18.5	13.7	49.8	27.1	48.8	52.2	69.8	44.6	48.7
✓	Rand	40%	18.6	13.0	49.6	26.4	48.9	52.8	70.2	44.8	48.8
✓	DRRQR	40%	**17.1**	11.1	51.5	27.0	49.9	53.9	69.7	49.2	50.2
✓	Grad	40%	**17.1**	**10.4**	50.8	26.8	50.1	54.5	70.7	50.8	**50.6**
✓	S-Wanda	40%	18.2	13.7	50.7	27.2	48.9	51.8	70.3	45.0	49.0
×	L1	30%	37.6	128.6	44.5	27.1	48.7	53.4	68.6	16.6	43.2
×	Rand	30%	19.8	22.9	48.9	27.3	49.3	53.7	69.6	36.7	47.6
×	DRRQR	30%	18.3	27.4	48.5	29.0	52.3	55.2	70.4	33.2	48.1
×	Grad	30%	**17.2**	**11.9**	50.3	26.8	50.8	54.6	70.8	47.5	**50.1**
×	S-Wanda	30%	36.9	119.0	44.4	26.1	48.6	52.6	68.7	17.4	43.0
✓	L1	30%	18.1	12.8	50.0	27.0	49.3	52.9	70.0	45.9	49.2
✓	Rand	30%	17.4	12.0	50.4	26.9	49.6	53.9	70.3	46.6	49.6
✓	DRRQR	30%	**16.8**	10.5	51.1	27.4	50.0	53.9	70.0	50.2	50.4
✓	Grad	30%	16.9	**10.1**	51.3	26.5	50.4	54.8	70.6	51.1	**50.8**
✓	S-Wanda	30%	18.0	13.0	50.5	27.0	49.2	54.0	70.3	45.4	49.4
Baseline	–	0%	16.7	11.9	51.3	26.1	50.6	53.3	70.5	48.4	50.0

Table 23: Zero-shot performance of Gated DeltaNet 370M models, evaluated using the lm-eval-harness (Gao et al., 2024), given different compression ratios. Pre- and post RFT.

RFT	Method	Compr.	Wiki. ↓ ppl	LMB. ↓ ppl	ARC-e ↑ acc_n	ARC-c ↑ acc_n	Hella. ↑ acc_n	Wino. ↑ acc	PIQA ↑ acc_n	LMB. ↑ acc	Avg ↑
×	L1	75%	116.4	1399.7	36.8	27.6	34.7	50.3	59.7	8.5	36.3
×	Rand	75%	135.3	1078.1	41.5	26.5	34.4	51.2	60.4	11.2	37.5
×	DRRQR	75%	**45.9**	**97.8**	49.7	27.8	38.4	50.1	63.7	23.8	**42.3**
×	Grad	75%	79.6	215.6	41.5	27.9	37.4	50.0	62.9	20.1	40.0
×	S-Wanda	75%	96.6	1205.5	37.2	26.5	35.5	50.4	60.6	8.8	36.5
✓	L1	75%	32.2	60.0	49.6	27.4	38.2	49.3	64.6	25.2	42.4
✓	Rand	75%	33.5	70.9	49.5	26.4	37.9	51.9	65.0	22.5	42.2
✓	DRRQR	75%	**31.4**	53.5	50.8	27.1	38.5	49.9	65.3	26.3	43.0
✓	Grad	75%	**31.4**	**49.1**	50.3	27.4	38.5	50.6	65.8	27.1	**43.3**
✓	S-Wanda	75%	32.1	58.2	50.0	27.0	38.0	49.3	64.5	25.6	42.4
×	L1	50%	41.8	156.6	45.6	28.3	38.8	47.8	64.0	17.9	40.4
×	Rand	50%	35.5	50.7	47.1	27.0	39.1	53.1	64.9	29.4	43.4
×	DRRQR	50%	**31.6**	39.4	49.8	27.9	39.9	49.5	65.2	31.5	**44.0**
×	Grad	50%	33.3	**39.2**	48.2	27.6	40.0	49.2	65.8	31.8	43.8
×	S-Wanda	50%	40.0	141.5	45.5	28.7	38.9	49.6	64.0	18.9	40.9
✓	L1	50%	29.2	44.6	50.2	27.8	39.0	49.6	65.3	29.0	43.5
✓	Rand	50%	29.3	46.5	50.3	26.6	38.9	51.9	66.1	28.2	43.6
✓	DRRQR	50%	**28.7**	40.6	50.9	27.5	39.2	50.4	65.3	29.8	**43.8**
✓	Grad	50%	**28.7**	**39.4**	50.3	27.4	39.0	50.6	65.7	29.6	43.7
✓	S-Wanda	50%	29.2	44.6	50.3	27.3	39.1	50.2	65.0	29.0	43.5
×	L1	40%	36.9	96.4	46.1	28.3	38.7	49.0	65.0	21.5	41.4
×	Rand	40%	32.2	42.2	48.8	27.3	39.2	53.0	65.3	30.5	44.0
×	DRRQR	40%	**30.1**	38.3	50.5	28.0	40.0	49.6	65.4	31.7	44.2
×	Grad	40%	30.7	**34.6**	49.5	27.5	40.2	50.4	65.8	33.0	**44.4**
×	S-Wanda	40%	35.9	95.8	46.7	28.0	39.2	50.2	64.1	22.1	41.7
✓	L1	40%	28.6	40.6	50.8	28.2	39.1	50.1	65.4	30.8	44.0
✓	Rand	40%	28.6	43.1	50.7	26.5	38.9	50.9	65.9	29.5	43.7
✓	DRRQR	40%	**28.2**	39.5	51.1	27.8	39.4	50.0	65.4	30.5	44.0
✓	Grad	40%	**28.2**	**38.0**	50.6	27.7	39.2	50.7	65.6	30.5	**44.1**
✓	S-Wanda	40%	28.5	40.6	50.9	27.8	38.8	49.6	65.2	30.8	43.9
×	L1	30%	33.9	65.1	47.7	28.2	39.4	49.3	64.9	25.1	42.4
×	Rand	30%	29.9	38.4	48.9	26.5	39.6	51.3	65.2	31.3	43.8
×	DRRQR	30%	**29.0**	36.6	51.1	28.2	39.9	50.4	65.3	32.4	**44.5**
×	Grad	30%	**29.0**	**32.4**	50.6	27.0	40.2	50.4	64.7	33.7	44.4
×	S-Wanda	30%	33.2	66.5	47.4	27.7	39.5	50.0	65.2	25.3	42.5
✓	L1	30%	28.1	38.5	50.6	27.9	39.1	49.9	65.7	31.1	44.1
✓	Rand	30%	28.2	40.9	50.8	27.0	39.3	50.5	65.4	29.7	43.8
✓	DRRQR	30%	**27.8**	38.0	51.2	27.9	39.4	50.0	65.5	31.4	44.2
✓	Grad	30%	**27.8**	**35.8**	51.3	27.7	39.4	50.6	65.5	31.7	**44.4**
✓	S-Wanda	30%	28.0	38.8	50.5	28.0	39.1	49.0	65.3	30.7	43.8
Baseline	–	0%	28.8	35.9	51.5	28.2	39.6	50.4	65.5	31.5	44.4

Table 24: Zero-shot performance of Gated DeltaNet 1.3B models, evaluated using the lm-eval-harness (Gao et al., 2024), given different compression ratios. Pre- and post RFT.

RFT	Method	Compr.	Wiki. ↓ ppl	LMB. ↓ ppl	ARC-e ↑ acc_n	ARC-c ↑ acc_n	Hella. ↑ acc_n	Wino. ↑ acc	PIQA ↑ acc_n	LMB. ↑ acc	Avg ↑
×	L1	75%	68.8	279.5	54.4	32.8	47.2	55.4	66.2	15.9	45.3
×	Rand	75%	32.8	37.8	56.0	34.3	51.8	54.5	70.2	30.8	49.6
×	DRRQR	75%	22.5	23.6	62.6	36.5	55.7	59.8	72.1	37.0	54.0
×	Grad	75%	**22.3**	**17.7**	64.0	38.3	57.1	60.2	71.4	42.5	**55.6**
×	S-Wanda	75%	30.6	77.0	59.7	36.3	54.3	58.9	69.9	24.6	50.6
✓	L1	75%	18.6	16.8	66.1	37.8	57.0	58.0	72.4	40.7	55.3
✓	Rand	75%	19.3	18.1	64.7	36.5	56.6	57.8	72.5	39.5	54.6
✓	DRRQR	75%	17.9	14.7	64.0	37.8	58.4	61.1	73.1	43.1	56.2
✓	Grad	75%	**17.8**	**12.5**	65.1	37.5	58.4	60.9	72.9	45.8	**56.8**
✓	S-Wanda	75%	18.2	15.1	63.9	37.7	57.5	59.0	72.6	41.7	55.4
×	L1	50%	26.5	34.0	64.8	36.5	56.3	58.2	71.6	32.3	53.3
×	Rand	50%	18.8	14.4	65.5	38.3	58.8	59.7	73.2	43.7	56.5
×	DRRQR	50%	**17.3**	14.2	66.2	39.2	59.9	62.5	73.1	44.0	57.5
×	Grad	50%	17.4	**10.1**	65.9	39.8	60.5	61.2	72.7	51.3	**58.6**
×	S-Wanda	50%	21.9	24.1	65.1	38.5	58.7	60.0	72.1	36.1	55.1
✓	L1	50%	16.8	13.0	66.0	39.2	59.2	59.8	73.6	45.3	57.2
✓	Rand	50%	16.9	11.5	66.6	38.9	58.8	61.2	73.6	47.8	57.8
✓	DRRQR	50%	**16.3**	12.0	66.8	39.4	59.4	61.1	73.7	47.4	58.0
✓	Grad	50%	16.4	**10.5**	65.7	39.3	59.6	61.1	73.9	49.9	**58.3**
✓	S-Wanda	50%	16.6	11.7	66.6	39.2	59.4	60.5	73.8	47.3	57.8
×	L1	40%	23.7	25.3	64.3	38.5	57.8	58.6	71.7	36.1	54.5
×	Rand	40%	18.3	11.0	65.7	39.3	59.7	60.5	73.9	49.7	58.1
×	DRRQR	40%	**16.7**	11.9	67.2	39.4	60.5	60.9	73.5	47.8	58.2
×	Grad	40%	**16.7**	**9.4**	67.8	40.5	60.6	62.3	73.5	52.6	**59.6**
×	S-Wanda	40%	20.7	21.6	65.4	39.6	59.5	60.3	72.4	37.4	55.8
✓	L1	40%	16.5	11.9	67.2	38.8	59.6	59.8	74.0	47.5	57.8
✓	Rand	40%	16.5	10.7	66.4	39.2	59.6	61.3	73.8	49.5	58.3
✓	DRRQR	40%	**16.1**	11.0	67.2	40.0	59.8	61.3	73.9	49.8	**58.7**
✓	Grad	40%	**16.1**	**10.1**	66.7	39.4	59.9	61.2	74.2	50.5	58.6
✓	S-Wanda	40%	16.4	11.5	66.8	39.5	59.9	60.8	74.2	47.7	58.1
×	L1	30%	22.2	24.3	64.5	38.1	58.4	61.0	71.9	35.4	54.9
×	Rand	30%	16.7	11.4	65.2	39.2	60.2	60.2	73.9	48.7	57.9
×	DRRQR	30%	**16.3**	11.6	67.0	39.7	60.5	61.0	73.7	47.5	58.2
×	Grad	30%	**16.3**	**9.2**	68.5	40.4	60.7	62.4	73.7	52.8	**59.7**
×	S-Wanda	30%	19.8	18.4	65.5	39.8	60.1	61.5	73.1	39.7	56.6
✓	L1	30%	16.3	11.3	67.5	39.2	59.7	61.2	74.1	48.3	58.3
✓	Rand	30%	16.1	10.2	66.5	39.6	59.8	61.6	74.4	50.4	58.7
✓	DRRQR	30%	**15.9**	10.7	67.3	39.9	59.9	61.8	74.3	49.7	58.8
✓	Grad	30%	**15.9**	**9.9**	66.9	40.4	60.0	61.6	74.4	50.9	**59.0**
✓	S-Wanda	30%	16.1	11.0	67.6	40.3	60.0	60.5	74.0	49.0	58.6
Baseline	–	0%	16.8	9.7	67.7	41.0	60.1	62.2	74.0	51.5	59.4

H MORE PLOTS

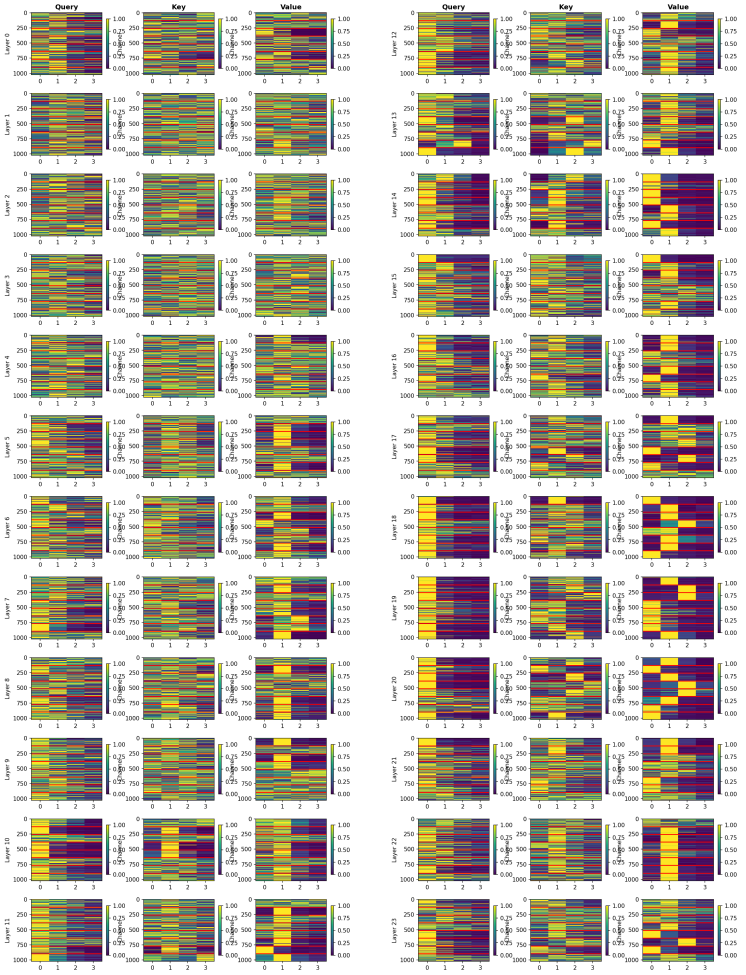


Figure 7: Convolution filters for queries, keys, and values of a standard DeltaNet 370M (non-shared). High similarity within heads (separated by red lines) suggests implicit sharing.

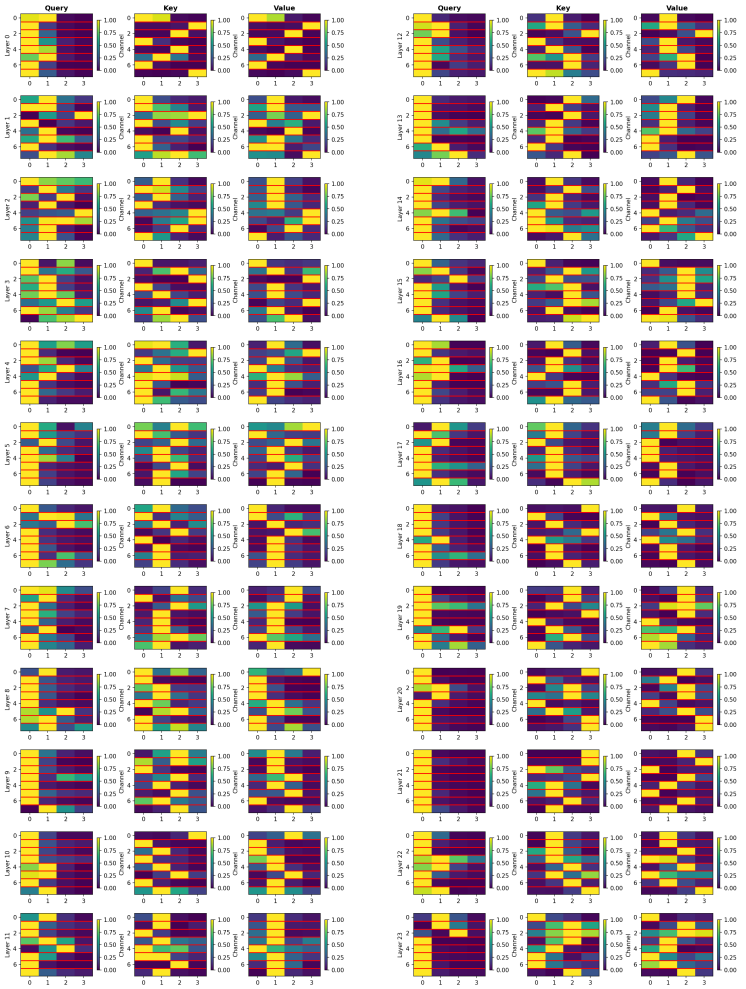


Figure 8: Convolution filters for DeltaNet 370M with explicitly shared convolutions. The model learns distinct patterns for Q, K, and V.