THE KEY TO STATE REDUCTION IN LINEAR ATTENTION: A RANK-BASED PERSPECTIVE

Philipp Nazari
Max Planck Institute for Intelligent Systems &
ETH Zürich &
ELLIS Institute Tübingen
philipp.nazari@tuebingen.mpg.de

T. Konstantin Rusch
ELLIS Institute Tübingen &
Max Planck Institute for Intelligent Systems &
Tübingen AI Center &
Liquid AI
tkrusch@tue.ellis.eu

ABSTRACT

Linear attention offers a computationally efficient yet expressive alternative to softmax attention, maintaining a recurrent state that functions as a linear associative memory. However, recent empirical results indicate that the associative memory of trained linear attention models often exhibits a low-rank structure, suggesting that these models underexploit their capacity in practice. To illuminate this phenomenon, we provide a theoretical analysis of the role of rank in linear attention, revealing that low effective rank can affect retrieval error by amplifying query noise, as well as poorly condition query gradients. In addition to these theoretical insights, we conjecture that the low-rank states can be substantially reduced post-training with only minimal performance degradation, yielding faster and more efficient models. To this end, we propose a hardware-aware approach that structurally prunes key and query matrices, effectively reducing the state size. We adapt existing pruning strategies to fit our framework and, building on our theoretical analysis, propose a structured pruning method based on a rank-revealing QR decomposition. Our empirical results demonstrate the effectiveness of this associative memory reduction framework. We highlight that it enables the removal of 50% of the query and key channels at only a minor increase in perplexity. The code for this project can be found at https://github.com/camail-official/LinearAttentionPruning.

1 INTRODUCTION

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

Despite their impressive performance, prior work indicates that linear attention models underutilize their capacity in practice (Siems et al., 2025; Parnichkun et al., 2025). Specifically, their matrix-valued hidden state often exhibits a low-rank structure. While rank collapse is a known phenomenon in standard Transformers (Dong et al., 2021; Noci et al., 2022), its implications for the state of linear attention remain underexplored. Viewed through the lens of *associative memory* (Ramsauer et al., 2020; Wang et al., 2025), our observation of low effective rank suggests that the model inefficiently stores redundant memory patterns, not utilizing its capacity properly.

Towards more effective associative memories, we propose a structured pruning framework to reduce the per-head key dimension of linear attention models. We find that we can consistently remove approximately 50%–75% of the key/query channels at only a minor increase in perplexity, even before recovery fine-tuning. This observation points to a fundamental inefficiency of the associative memory implemented by linear attention models.

2 THEORETICAL INSIGHTS

Background on Linear Attention. Given an input sequence $\mathbf{X} \in \mathbb{R}^{T,h}$, linear attention forms queries $\mathbf{Q} = \mathbf{X}\mathbf{W}_\mathbf{Q} \in \mathbb{R}^{T,d_k}$, keys $\mathbf{K} = \mathbf{X}\mathbf{W}_\mathbf{K} \in \mathbb{R}^{T,d_k}$, and values $\mathbf{V} = \mathbf{X}\mathbf{W}_v \in \mathbb{R}^{T,d_v}$. Exploiting the associativity of matrix multiplication, (Schlag et al., 2021; Yang et al., 2024b; Dao & Gu, 2024; Yang et al., 2024a; Siems et al., 2025) the linear attention mechanism can be written as a recurrence

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

where $\mathbf{S}_t \in \mathbb{R}^{d_v,d_k}$ is a matrix-valued hidden state used to compute the output $\mathbf{o}_t = \mathbf{S}_t\mathbf{q}_t$. The state $\mathbf{S}_t$ acts as a *linear associative memory* (Wang et al., 2025). We will refer to its update rule as the *sequence mixer*. The one expressed in Equation (1) can only write *into* the memory. DeltaNet (Schlag et al., 2021; Yang et al., 2024b) addresses this limitation by explicitly erasing information associated with the current key before writing the new value into $\mathbf{S}_t$,

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

where β_t is a scalar writing-strength. Gated DeltaNet (Yang et al., 2024a) further refines this mechanism by introducing a data-dependent decay term.

On the Role of the Rank. A commonly identified weakness of linear attention models compared to their softmax counterparts is their fixed-sized associative memory. However, recent empirical results indicate that these models do not manage this memory well (Siems et al., 2025; Parnichkun et al., 2025) and effectively exhibit a low-rank structure in practice. In the following, we build a framework that illuminates this phenomenon.

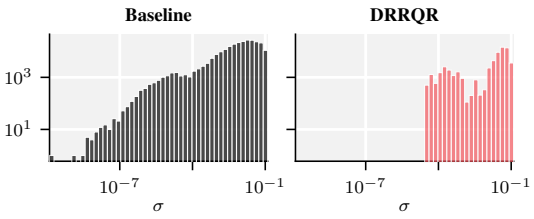

Figure 1: Singular value spectra of DeltaNet 370M hidden states at a random head (Fineweb-Edu, $T = 2048$, skipping the first 128 tokens), aggregated over tokens. **Left:** Uncompressed baseline. **Right:** DRRQR at 75% compression (pre-RFT).

Notably, the algebraic rank of the associative memory of a family of linear attention models is bounded by the rank of the keys and values (see Appendix E.2),

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

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. However, the algebraic rank is too rigid to serve as a meaningful measure for noisy real-world data. To address this, we consider the *effective rank* (or *stable rank*) (Rudelson & Vershynin, 2007; Tropp et al., 2015; Vershynin, 2018; Ipsen & Saibaba, 2025), a surrogate for the numerical rank; given a matrix $\mathbf{A} \in \mathbb{R}^{m,n}$, its *effective rank* is defined as

$$\operatorname{er}(\mathbf{A}) \coloneqq \|\mathbf{A}\|_F^2 / \|\mathbf{A}\|_2^2.$$

It measures the skewness of the singular value spectrum and can also be computed as $\operatorname{er}(\mathbf{A}) = \sum_i \sigma_i^2/\sigma_1^2$, where $\sigma_1 \geq \ldots \geq \sigma_{\min(m,n)} \geq 0$ are the singular values of $\mathbf{A}$. For us, it will serve as a measure for how a model manages its hidden state.

The following proposition generalizes Equation (3), relating the effective rank of the associative memory to the l^2 condition number κ of the keys:

Proposition 2.1. *Consider 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 the effective rank of the memory is lower bounded:*

$$\nu(\mathbf{V}_t)/\kappa^2(\mathbf{K}_t) \leq \operatorname{er}(\mathbf{S}_t).$$

The proof of this proposition may be found in Appendix E.1. Although this statement holds for the specific case of plain linear attention, we use it as an approximation for (Gated) DeltaNet. Towards our goal of improving the memory utilization of linear attention models, this proposition shows that improving the conditioning of the keys improves the effective rank of the memory.

While effective rank measures the raw dimensionality of the stored information, it does not capture how *efficiency* a model manages its memory. Given a non-zero matrix $\mathbf{S} \in \mathbb{R}^{d_v,d_k}$, we thus define its *rank utilization* as the ratio of its effective rank and its theoretically maximal rank:

$$\mathrm{u}(\mathbf{S}) \coloneqq \operatorname{er}(\mathbf{S}) / \min(d_k, d_v).$$

Rank utilization combines effective rank and theoretically maximum capacity $d \coloneqq \min(d_k, d_v)$ and thus serves as a measure for memory occupancy. We identify the following two edge cases: **Low Utilization** ($\mathrm{u} \ll 1$)**:** The memory suffers from *rank collapse*. Its energy is concentrated in a few principal components. The vast majority of information stored in the state is redundant. **High Utilization** ($\mathrm{u} \approx 1$)**:** The memory is *isotropic*. Energy is distributed evenly across all dimensions.

Why does Rank Utilization Matter? Let $\mathbf{S} = \sum_{i=1}^{d} \sigma_i \mathbf{u}_i \mathbf{w}_i^\top$ be the SVD of the associative memory at a given time (we drop the subscript t here for brevity), with singular values $\sigma_1 \geq \cdots \geq \sigma_d > 0$. Moreover, consider a noisy query $\tilde{\mathbf{q}} = \mathbf{q}^* + \mathbf{n}$, where $\mathbf{q}^*$ is the pure query and $\mathbf{n}$ is noise. To analyze how the memory structure affects the output error, define two coefficients that capture the alignment δ of the noise and of the signal (denoted by γ) with the principal axis:

$$\delta \coloneqq |\mathbf{n}^\top \mathbf{w}_1| / \|\mathbf{n}\|_2 \quad \text{and} \quad \gamma \coloneqq |\mathbf{q}^{*\top} \mathbf{w}_1| / \|\mathbf{q}^*\|_2.$$

The following theorem establishes that the relative retrieval error is governed by the effective rank:

Theorem 2.2 (Effective Rank Governs Retrieval Error)**.** *The ratio of the relative output error to the input noise-to-signal ratio is governed by the effective rank of the memory:*

$$\frac{\delta}{\sqrt{\mathrm{er}(\mathbf{S})}} \leq \frac{\|\mathbf{o} - \mathbf{o}^*\|_2 / \|\mathbf{o}^*\|_2}{\|\mathbf{n}\|_2 / \|\mathbf{q}^*\|_2} \leq \frac{\sqrt{\mathrm{er}}(\mathbf{S})}{\gamma}, \tag{4}$$

The proof may be found in Appendix E.4. Equation (4) can also be expressed in terms of rank utilization, since $\mathrm{u}(\mathbf{S}) = \mathrm{er}(\mathbf{S})/d$ for a given maximum capacity $d = \min(d_k, d_v)$. In this formulation, Theorem 2.2 reveals the influence of rank utilization on the retrieval error. We highlight two regimes: **Low Utilization** ($u(\mathbf{S}) \ll 1$)**:** The system is highly sensitive to noise, unless the noise is orthogonal to the principal component of the memory ($\delta \approx 0$). Simultaneously, the upper bound is small only if the signal is aligned with the principal component ($\gamma \approx 1$). The rank utilization acts as a strong multiplier to the alignment of the noise with the principal component. **High Utilization** ($u(\mathbf{S}) \approx 1$)**:** The memory becomes isotropic, all dimensions carry the same energy. Rank utilization acts only as a weak multiplier on the alignment of the noise and signal with the principal component.

Besides amplifying retrieval error, low rank utilization also leads to poorly conditioned gradients for $\mathbf{W_Q}$ during the backwards-pass, since

$$\kappa\left(\frac{\partial \mathbf{o}_t}{\partial \mathrm{vec}(\mathbf{W}_\mathbf{Q}^\top)}\right) = \kappa(\mathbf{x}_t^\top \otimes \mathbf{S}_t) = \kappa(\mathbf{S}_t).$$

The proof may be found in Appendix E.5.

The results presented in this section shed light on the role of the associative memory's effective rank and provide a theoretical justification for our structured pruning framework. To this end, the following section develops methods for reducing the dimension of the queries and keys.

3 The Proposed Pruning Approach

Existing pruning methods for LLMs generally rely on unstructured or semi-structured sparsity (Frantar & Alistarh, 2023; Sun et al., 2023a). However, these methods do not reduce the state dimension of the dynamical system underlying linear attention models. Instead, we focus on *structured* pruning.

Towards this end, note that the sequence mixer of (Gated) DeltaNet is *invariant under simultaneous orthogonal transformations* of the queries and keys, $(\mathbf{k}_t, \mathbf{q}_t) \mapsto (\mathbf{T}\mathbf{k}_t, \mathbf{T}\mathbf{q}_t)$ for some $\mathbf{T} \in O(d_k)$ (see Appendix E.3).

Dimension reduction is achieved by applying a *semi-orthogonal* matrix $\mathbf{T} \in \mathbb{R}^{d_k', d_k}$ with target dimension $d_k' < d_k$ *jointly* to the keys and queries before computing the sequence mixer. The goal is to find a $\mathbf{T}$ with $d_k' \ll d_k$ that entails a small approximation error. To achieve actual wall-clock speedup, $\mathbf{T}$ must be absorbed into the weight matrices $\mathbf{W_K}$ and $\mathbf{W_Q}$. We thus restrict our focus to transformations that preserve the channel-wise convolutions (Chollet, 2017; So et al., 2021; Fu et al., 2022; Yang et al., 2023; Gu & Dao, 2024; Dao & Gu, 2024; Yang et al., 2024b). Formally, this restricts transformations to *axis-aligned* semi-orthogonal matrices:

Definition 3.1 (Axis-Aligned Transformations)**.** 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 axis-aligned projection matrix $\mathbf{P}_\mathcal{I} \in \mathbb{R}^{d'_k, d_k}$ as the matrix whose rows are the standard basis vectors corresponding to $\mathcal{I}$:

$$\mathbf{P}_\mathcal{I} \coloneqq [\mathbf{e}_{i_1}; \ldots; \mathbf{e}_{i_{d'_k}}]^\top.$$

This matrix is semi-orthogonal. Applying it corresponds to a structural pruning operation that selects the subset of channels $\mathcal{I}$ and discards the rest. Crucially, this operation preserves the independence of the remaining channels:

Proposition 3.2 (Compatibility with Depthwise Convolutions)**.** *Let* $\mathrm{Conv1D}(\mathbf{X}, \mathbf{W})$ *denote a depthwise convolution on input* $\mathbf{X} \in \mathbb{R}^{T, d_k}$ *with per-channel filters* $\mathbf{W} \in \mathbb{R}^{d_k, l}$ *of size* l*. Let* $\mathbf{P}_\mathcal{I}$ *be an axis-aligned semi-orthogonal transformation. Then:*

$$\mathrm{Conv1D}(\mathbf{X}, \mathbf{W})\mathbf{P}_\mathcal{I}^\top = \mathrm{Conv1D}(\mathbf{X}\mathbf{P}_\mathcal{I}^\top, \mathbf{P}_\mathcal{I}\mathbf{W}).$$

This construction allows pruning query/key dimensions by simply removing the corresponding columns from the projection matrices $\mathbf{W_K}$ and $\mathbf{W_Q}$, along with the corresponding entries of the convolution weights. In the following paragraphs, we present strategies for selecting the index set $\mathcal{I}$.

Weight-Magnitude-based ($\mathbf{L^1}$). As a baseline, we implement a weight-magnitude based pruner, defining the importance of the j-th channel as the sum of the l^1 norms of the corresponding columns in the query and key projection matrices: $s_j = \|\mathbf{W_{Q},:j}\|_1 + \|\mathbf{W_{K},:j}\|_1$. We rank these scores locally within each head and select the top-d'_k indices to form the retained set $\mathcal{I}$.

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. We quantify the importance of the j-th key dimension using the first-order Taylor expansion of the loss on a calibration set:

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

Rank-Based (DRRQR). In light of our theoretical insights into the role of rank in linear attention models (Section 2), we propose an algorithm that explicitly improves the conditioning of the queries and keys. By Proposition 2.1, this increases the effective rank of the associative memory. *Deep Rank Revealing QR* (*DRRQR*) applies a *Strong Rank Revealing QR* factorization (Gu & Eisenstat, 1996) to the activation statistics $\mathbf{M} = [\mathbf{K}, \mathbf{Q}] \in \mathbb{R}^{2N, d_k}$ to select a subset of d'_k columns that form a well-conditioned set of channels (details in Appendix B.2).

4 Experiments

The details on our experimental setup may be found in Appendix A. We evaluate the introduced structured pruning methods on Gated DeltaNet (Yang et al., 2024a) at the 370M and 1.3B parameter scales, before and after recovery fine-tuning (RFT) (see Appendix C for more results, also on DeltaNet).

Figure 2 shows the typical rank utilization during a forward pass through DeltaNet 370M at a compression ratio of 75%. We can see that the two most powerful compression methods, *Grad* and *DRRQR*, have the largest rank utilization. Figure 1 furthermore shows how DRRQR removes the tail of the hidden states' spectrum.

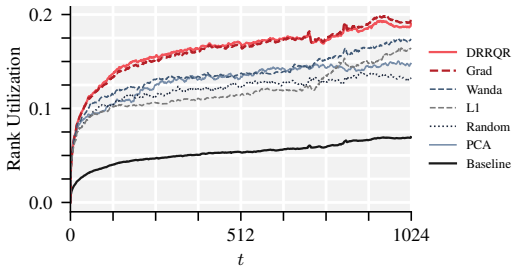

Figure 2: Rank utilization of DeltaNet 370M as a function of the token index for a random sample of Fineweb-Edu of length 1024, averaged over layers and heads, at a compression ratio of 75% pre-RFT.

Table 1: Comprehensive post-RFT evaluation of Gated DeltaNet 1.3B. We report Wikitext and Lambada perplexities, the average zero-shot generation accuracy (**ZS**, Gao et al. (2024)) as well as average retrieval accuracy (**Ret**, Arora et al. (2024)) across varying compression ratios (best results in **bold**, second best underlined).

Method	75% Compression				50% Compression				30% Compression			
	Wiki ↓	LMB ↓	ZS ↑	Ret ↑	Wiki ↓	LMB ↓	ZS ↑	Ret ↑	Wiki ↓	LMB ↓	ZS ↑	Ret ↑
L1	18.6	16.8	55.3	25.7	16.8	13.0	57.2	32.9	16.3	11.3	58.3	35.4
Grad	**17.8**	**12.5**	**56.8**	**26.9**	16.4	**10.5**	**58.3**	**34.3**	**15.9**	**9.9**	**59.0**	**38.0**
DRRQR	17.9	14.7	56.2	26.1	**16.3**	12.0	58.0	33.0	**15.9**	10.7	58.8	36.5
Baseline	16.8	9.7	59.4	40.3	———				———			

Language Modeling. We observe that pruning 50% of the key dimension entails minimal degradation in Wikitext2 and Lambada perplexity, especially when using the *Grad* and *DRRQR* pruners, even before RFT (see Table 2). For example, Gated DeltaNet 1.3B sees Wikitext perplexity go from 16.8 to 17.4 at 50% compression ratio. Even at 75%, the perplexity only goes up by about two points. This finding suggests that pre-trained linear attention models seem to effectively use at most half of their available capacity for next token prediction.

We generally find the rank-based method *DRRQR* to perform competitively with the gradient-saliency based *Grad*, even though it is local and task-agnostic. *Grad*, on the other hand, is non-local and removes weight in a way that entails a minimal increase in perplexity.

The zero-shot common sense reasoning scores remain robust under compression. For instance, Gated DeltaNet 1.3B maintains an average of 58.3 at 50% compression when compressed via *Grad*, around a one-point drop from the 59.4 baseline. The retrieval tasks show higher sensitivity to state reduction, especially *FDA* and *SWDE* (see for example Table 16). However, for more conservative pruning ratios around 30%, the retrieval capabilities stay competitive.

Method	370M		1.3B	
	Wiki ↓	LMB ↓	Wiki ↓	LMB ↓
L1	41.8	156.6	26.5	34.0
Grad	33.3	**39.2**	17.4	**10.1**
DRRQR	**31.6**	39.4	**17.3**	14.2
Base	*28.8*	*35.9*	*16.8*	*9.7*

Table 2: Pre-RFT perplexity of DeltaNet models at 50% compression on Wikitext and Lambada.

Speedup. We benchmark the sequence mixer throughput and peak VRAM usage on an NVIDIA H100 GPU (see Table 7). Comparing the baseline ($d_k = 128$) against compressed variants ($d_k \in \{64, 32\}$): For DeltaNet, a 50% reduction yields a $\sim 1.33\times$ speedup in terms of throughput. A 75% reduction yields a speedup of $\sim 1.58\times$. Peak VRAM usage decreases by 28% and 42%, respectively.

5 Discussion

Motivated by the low-rank structure of the associative memory in trained linear attention models, we propose a state-size reduction framework that selects a subset of informative key and query columns while discarding the remainder. In addition to adapting several heuristic pruning strategies to this framework, we introduce a principled structured pruning method explicitly designed to improve effective rank and rank utilization. We further provide an analysis of the role of rank in linear attention, showing that low-rank structure can amplify query noise and govern retrieval error, as well as poorly condition the query gradients. Finally, we present extensive empirical evaluations demonstrating the effect of rank collapse of the associative memory, as well as the practical effectiveness of our structured pruning framework.

Our empirical results show that the sequence mixer can be compressed by up to 50% while incurring only a minor increase in perplexity. However, our findings also highlight a limitation of the proposed framework: reducing the state size can lead to performance drops on some recall-intensive tasks. This observation is well known for linear attention models (Arora et al., 2024) and is a primary motivation for hybrid architectures that combine linear and softmax attention. Accordingly, a promising direction for future work would be to apply our structured pruning approach to hybrid models, where the additional softmax attention layers may help mitigate performance losses on recall-intensive tasks.

ACKNOWLEDGEMENTS

Philipp Nazari was supported by the Max Planck ETH Center for Learning Systems. This work was supported by the Hector Foundation.

REFERENCES

Simran Arora, Aman Timalsina, Aaryan Singhal, 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*, 2024.

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress Large Language Models by Deleting Rows and Columns. *arXiv preprint arXiv:2401.15024*, 2024.

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*, 2026.

Chi-Tsong Chen. *Linear System Theory and Design*. Saunders college publishing, 1984.

François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 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.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not All Aou Need: Pure Attention Loses Rank Doubly Exponentially with Depth. In *International conference on machine learning*, pp. 2793–2803. PMLR, 2021.

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. In *First conference on language modeling*, 2024.

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 university press, 1994.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022.

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 *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

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

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling Foundation Models with Lightning Attention. *arXiv preprint arXiv:2501.08313*, 2025.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A Hybrid Transformer-Mamba Language Model. *arXiv preprint arXiv:2403.19887*, 2024.

Lucas D Lingle. Transformer-VQ: Linear-Time Transformers via Vector Quantization. *arXiv preprint arXiv:2309.16354*, 2023.

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.

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, Matteo Grella, et al. RWKV: Reinventing RNNs for the Transformer Era. *arXiv preprint arXiv:2305.13048*, 2023.

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

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 *International conference on machine learning*, pp. 9355–9366. PMLR, 2021.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. DeltaProduct: Improving State-Tracking in Linear RNNs via Householder Products. *arXiv preprint arXiv:2502.10297*, 2025.

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 J Zico Kolter. A Simple and Effective Pruning Approach for Large Language Models. *arXiv preprint arXiv:2306.11695*, 2023a.

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*, 2023b.

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.

Qwen Team. Qwen3-Next: Towards Ultimate Training & Inference Efficiency, 2025.

Lloyd N Trefethen and David Bau. *Numerical Linear Algebra*. SIAM, 2022.

Joel A Tropp et al. 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: An Introduction with Applications in Data Science*, volume 47 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, 2018.

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.

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, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with Delta Rule. *arXiv preprint arXiv:2412.06464*, 2024a.

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, 2024b.

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. (2024)) and pre-training the 370M parameter models. The latter uses 10 billion tokens of Fineweb-Edu. The larger DeltaNet and Gated DeltaNet models are taken from fla-hub [1] and m-a-p [2], respectively.

We perform RFT on a single H100 GPU using LoRA (Hu et al., 2022) with rank $r = 16$ $\alpha = 32$, using $32k$ samples of Fineweb-Edu. During training, we use: a batch size of 16, training on 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, which make up just a fraction of the total parameters. As suggested by Ma et al. (2023), we furthermore apply knowledge-distillation during RFT using the original model.

For the compression methods requiring a calibration set, we use 128 samples of Fineweb-Edu. *DRRQR* uses a random subsample of $5k$ keys and queries each.

B Additional Material

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

Algorithm 1 A Typical DeltaNet Layer

Input: Hidden states $\mathbf{X} \in \mathbb{R}^{T,d}$, Previous state $\mathbf{S}_0$
Parameters: $\mathbf{W}_q, \mathbf{W}_k \in \mathbb{R}^{d_k,h}$, $\mathbf{W}_v \in \mathbb{R}^{d_v,h}$, $\mathbf{W}_o \in \mathbb{R}^{h,d_v}$, $\mathbf{W}_\beta \in \mathbb{R}^{1,h}$.

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

// 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 $\mathbf{O} \in \mathbb{R}^{T,d}$

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

Return: $\mathbf{Y}$

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

Linear attention, and specifically DeltaNet, are usually interpreted as performing gradient descent on a linear regression objective (Schlag et al., 2021; Yang et al., 2024b), 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.

[1] https://huggingface.co/collections/fla-hub/deltanet
[2] https://huggingface.co/m-a-p/1.3B-100B-GatedDeltaNet-pure

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
$[\mathbf{Q}, \mathbf{R}, \mathbf{\Pi}] \leftarrow \mathrm{QRCP}(\mathbf{A})$
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 column i^* of $\mathbf{A}_{d'_k}$ with column j^* of $\mathbf{C}_{d'_k}$
  Update $\mathbf{R}$, $\omega(\mathbf{A}_{d'_k})$, and $\gamma(\mathbf{C}_{d'_k})$
end while

$\mathcal{I} \leftarrow \mathbf{\Pi}[1 : d'_k]$
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 $\mathbf{\Pi}$ and a target rank k (in our main text denoted as d'_k) such that the QR factorization

$$\mathbf{M\Pi} = \mathbf{Q} \begin{pmatrix} \mathbf{A}_k & \mathbf{B}_k \\ \mathbf{0} & \mathbf{C}_k \end{pmatrix} \tag{5}$$

satisfies specific bounds on the singular values of the leading principal submatrix $\mathbf{A}_k \in \mathbb{R}^{k,k}$ and the trailing submatrix $\mathbf{C}_k \in \mathbb{R}^{(m-k),(n-k)}$. Specifically, a Strong RRQR factorization guarantees that $\sigma_{\min}(\mathbf{A}_k)$ is bounded away from zero and $\|\mathbf{C}_k\|_2 = \sigma_{\max}(\mathbf{C}_k)$ is small. This implies that $\mathbf{A}_k$ is well-conditioned.

Geometric Intuition. The algorithm aims to select k columns that maximize the volume of the parallelotope formed by the selected column vectors. Since $|\det(\mathbf{A}_k)| = \prod_{i=1}^{k} \sigma_i(\mathbf{A}_k)$, maximizing the determinant pushes the smallest singular values upward, thereby minimizing the condition number $\kappa(\mathbf{A}_k)$.

The Swap Criterion. Let $\mathbf{\Pi}$ be the current permutation 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:

$$\rho_{ij} = \sqrt{|U_{ij}|^2 + \left(\frac{\gamma_j(\mathbf{C}_k)}{\omega_i(\mathbf{A}_k)}\right)^2}. \tag{6}$$

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) Permute columns to swap indices i and $j + k$.
   (b) Retriangularize the matrix $\mathbf{R}$ using Givens rotations to restore the upper-triangular structure of $\mathbf{A}_k$. This costs $O(k(n-k))$ operations rather than the $O(n^3)$ of a full factorization.
   (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 Derivation of Tighter Error Bounds

In this section, we derive tighter bounds for the retrieval error ratio. The proof of Theorem 2.2 relies on the loose upper bound $\|\mathbf{Sn}\|_2 \leq \|\mathbf{S}\|_F \|\mathbf{n}\|_2$.

However, if we allow using the condition number $\kappa(\mathbf{S}) = \sigma_1/\sigma_d$ as a measure for the anisotropy of the associative memory, it follows readily from standard perturbation theory that

$$\frac{1}{\kappa(\mathbf{S})} \leq \frac{\|\mathbf{o} - \mathbf{o}^*\|_2/\|\mathbf{o}^*\|_2}{\|\mathbf{n}\|_2/\|\mathbf{q}^*\|_2} \leq \kappa(\mathbf{S}). \tag{7}$$

Indeed, it holds that (Trefethen & Bau, 2022, Lecture 12) $\sigma_d\|\mathbf{x}\|_2 \leq \|\mathbf{Sx}\|_2 \leq \sigma_1\|\mathbf{x}\|_2$ for any vector $\mathbf{x}$. Applying these inequalities to $\mathbf{Sn}$ and $\mathbf{Sq}^*$ yields Equation (7). This allows deriving bounds on the expected error under isotropic Gaussian noise (assuming again $\|q^*\|_2 = 1$ for simplicity):

$$\frac{1}{\kappa(\mathbf{S})}\mu \leq \|\mathbf{o} - \mathbf{o}^*\|_2/\|\mathbf{o}^*\|_2 \leq \kappa(\mathbf{S})\mu.$$

B.4 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. In particular, Proposition 3.2 does not hold anymore. 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. When the input space is rotated via $\mathbf{T}$, the original basis-aligned filters become misaligned with the new principal components. Towards maintaining learned structures after pruning, one must find new convolution kernels that best approximate the original dynamics.

B.5 Optimal Diagonal Adaptation

We formalize this adaptation as an optimization problem: finding the optimal diagonal (channel-wise) filters in the new basis that minimize the reconstruction error of the original convolution output.

Proposition B.1 (Optimal Diagonal Adaptation)**.** *Let $\mathbf{x}_t$ be the input signal and let $\mathbf{T} \in O(d)$ be an orthogonal matrix, yielding features $\tilde{\mathbf{x}}_t = \mathbf{T}\mathbf{x}_t$. Let $\mathbf{W} \in \mathbb{R}^{d,l}$ be the original learnable filters for d channels and kernel size l.*

The optimal diagonal per-channel weights $\mathbf{W}' \in \mathbb{R}^{d,l}$ that minimize the expected squared reconstruction error:

$$\min_{\mathbf{W}'} \mathbb{E}_{\mathbf{x}} \left[\|\mathbf{W}' * \tilde{\mathbf{x}} - \mathbf{T}(\mathbf{W} * \mathbf{x})\|_F^2 \right]$$

are given by the energy-weighted projection:

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

where $*$ denotes the depthwise convolution and $\odot$ is the Hadamard (element-wise) product.

Proof. We seek to find new depthwise separable convolutions with filter weights $\mathbf{W}'$ that minimize the error between the rotated input convolved with the new weights and the rotated original output.

Recall that a depthwise convolution with filter matrix $\mathbf{W}'$ acting on input $\tilde{\mathbf{x}}$ can be written as:

$$\mathbf{W}' * \tilde{\mathbf{x}} = \sum_{j=0}^{l-1} \mathbf{W}'^{(j)} \tilde{\mathbf{x}}_{t-j},$$

where $\mathbf{W}'^{(j)} = \mathrm{diag}(\mathbf{w}'^{(j)})$ is the diagonal filter matrix at time lag j. Similarly, the rotated original output is:

$$\mathbf{T}(\mathbf{W} * \mathbf{x}) = \mathbf{T} \sum_{j=0}^{l-1} \mathbf{W}^{(j)} \mathbf{x}_{t-j} = \sum_{j=0}^{l-1} \mathbf{T}\mathbf{W}^{(j)}\mathbf{T}^\top \tilde{\mathbf{x}}_{t-j},$$

where we used $\mathbf{x} = \mathbf{T}^\top \tilde{\mathbf{x}}$.

The error term is then $\sum_{j=0}^{l-1} (\mathbf{W}'^{(j)} - \mathbf{T}\mathbf{W}^{(j)}\mathbf{T}^\top)\tilde{\mathbf{x}}_{t-j}$. Thus, the ideal filter in the new basis is the dense matrix $\mathbf{M}^{(j)} \coloneqq \mathbf{T}\mathbf{W}^{(j)}\mathbf{T}^\top$. However, to maintain the efficiency of depthwise convolutions, we are constrained to approximate this dense matrix with a diagonal matrix $\mathbf{W}'^{(j)}$.

Let us focus on the error for a specific lag j (omitting j for brevity) and channel k. The error vector is $\mathbf{e} = (\mathbf{W}' - \mathbf{M})\tilde{\mathbf{x}}$. The k-th component is:

$$e_k = (W'_{kk} - M_{kk})\tilde{x}_k - \sum_{n \neq k} M_{kn}\tilde{x}_n.$$

Squaring and taking the expectation, assuming the features in the rotated basis $\tilde{\mathbf{x}}$ are decorrelated (which is true if $\mathbf{T}$ is the PCA transformation matrix) such that $\mathbb{E}[\tilde{x}_k \tilde{x}_n] = 0$ for $n \neq k$:

$$\mathbb{E}[e_k^2] = (W'_{kk} - M_{kk})^2 \mathbb{E}[\tilde{x}_k^2] + \sum_{n \neq k} M_{kn}^2 \mathbb{E}[\tilde{x}_n^2].$$

To minimize this error with respect to the diagonal weight W'_{kk}, we must set the first term to zero:

$$W'_{kk} = M_{kk} = (\mathbf{T}\,\mathrm{diag}(\mathbf{w})\mathbf{T}^\top)_{kk}.$$

Expanding this matrix multiplication:

$$W'_{kk} = \sum_{m=1}^{d} T_{km} w_m T_{km} = \sum_{m=1}^{d} (T_{km})^2 w_m = ((\mathbf{T} \odot \mathbf{T})\mathbf{w})_k.$$

Extending this to all channels and lags, we obtain the matrix form $\mathbf{W}' = (\mathbf{T} \odot \mathbf{T})\mathbf{W}$. □

Intuitively, the new kernel for a principal component is a weighted average of the original kernels, weighted by the energy (squared contribution) each original dimension contributes to that component.

B.6 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:

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

□

This lemma implies that for models trained with shared convolutions, the optimal filter in the rotated basis $\mathbf{W}'$ is identical to the original filter $\mathbf{W}$. This architectural choice would render the model naturally robust to basis changes, enabling rotation-based pruning methods like PCA-based truncation without the need for filter adaptation or approximation errors.

C Additional Experimental Results

C.1 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. In Appendix B.4, we lay out a framework to fix this misalignment, either by introducing *shared convolutions* or by *mixing filters* (see Proposition B.1).

C.1.1 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, this architecture is equivariant under rotations.

Table 3: DeltaNet 370M models trained on $10B$ tokens evaluated on common sense 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

Performance of Shared Convolutions. We include results on pre-trained DeltaNet models with and without shared convolutions in Table 3. 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 4: 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.1.2 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 4 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 methods still outperforms PCA even when using shared convolutions.

C.2 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 5.

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 retrieval, 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.

Curiously, magnitude-based methods seem to 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

Table 5: 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	$\mathbf{K}, \mathbf{Q}$	$\mathbf{K}$	$\mathbf{Q}$	$\mathbf{K}, \mathbf{Q}$	$\mathbf{K}$	$\mathbf{Q}$	$\mathbf{K}, \mathbf{Q}$	$\mathbf{K}$	$\mathbf{Q}$	$\mathbf{K}, \mathbf{Q}$	$\mathbf{K}$	$\mathbf{Q}$
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
Baseline	*29.8*			*16.7*			*28.8*			*16.8*		

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 5 reveal a clear difference between the studied pruning methods. Magnitude-based methods (*L1*, *Wanda*) are unstable when targeting keys, 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. For more details, please refer to (Ipsen & Saibaba, 2025).

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*

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

where u *is the rank utilization.*

Edge Case (Isotropy): In the specific case where the matrix is perfectly conditioned on its support (i.e., $\kappa(\mathbf{A}) = 1$), the inequality becomes an equality:

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

Since we known $\mathrm{er}(\mathbf{A}) \leq r$, 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 PROOFS

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

E.1 PROOF OF PROPOSITION 2.1

We consider the matrix form of the associative memory $\mathbf{S} = \mathbf{V}^\top \mathbf{K}$, where $\mathbf{V} \in \mathbb{R}^{T,d_v}$ and $\mathbf{K} \in \mathbb{R}^{T,d_k}$.

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

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

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

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

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

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

Since the columns of $\mathbf{V}_\parallel$ lie entirely within the column space of $\mathbf{K}$, and assuming $\mathbf{K}$ has full column rank, the matrix multiplication acts as a bijection on the row space of $\mathbf{V}_\parallel^\top$. 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 strictly here because the rows of $\mathbf{V}_\parallel^\top$ align with the range of $\mathbf{K}$:

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

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

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

Combining these inequalities yields the lower bound:

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

□

E.2 PROOF OF ALGEBRAIC RANK

We first prove the following proposition:

Proposition E.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*

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

Consequently,

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

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 $\mathbf{B}$,

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

and thus

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

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 $\operatorname{col}(\mathbf{AB}) \subseteq \operatorname{col}(\mathbf{A})$,

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

This concludes the proof. □

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

Lemma E.2 (Monotonicity of Rank Utilization)**.** *Pruning the low-variance directions via PCA strictly increases (or maintains) rank utilization. That is, for any* $d_k' < d_k$*:*

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

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. □

E.3 Proof of Invariance Property

Decomposing $\mathbf{S}_t$ row-wise reveals that the sequence mixer simulates d_v parallel, independent linear time-varying (LTV) 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

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

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 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, Chapter 4.4). That is, every transformation

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

derived from an invertible matrix $\mathbf{T} \in GL(d_k)$ leaves the input-output mapping invariant. However, only requiring invertibility of $\mathbf{T}$ is not enough, as the transformed state transition matrix

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

is 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 directly into the keys and queries as $\tilde{\mathbf{k}}_t = \mathbf{T}\mathbf{k}_t$ and $\tilde{\mathbf{q}}_t = \mathbf{T}\mathbf{q}_t$, preserving the structure of the DeltaNet recurrence:

Proposition E.3 (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.

E.4 Proof of Theorem 2.2

We analyze the error amplification ratio R, defined as the relative output error divided by the input noise-to-signal ratio:

$$R = \frac{\|\mathbf{o} - \mathbf{o}^*\|_2 / \|\mathbf{o}^*\|_2}{\|\mathbf{n}\|_2 / \|\mathbf{q}^*\|_2} = \frac{\|\mathbf{S}\mathbf{n}\|_2}{\|\mathbf{n}\|_2} \cdot \frac{\|\mathbf{q}^*\|_2}{\|\mathbf{S}\mathbf{q}^*\|_2}.$$

This expression represents the Rayleigh quotient of the noise divided by the Rayleigh quotient of the signal.

We start by showing the lower bound. To find the minimum error, we start by projecting the response to the noise onto $\mathbf{u}_1$:

$$\|\mathbf{S}\mathbf{n}\|_2 \geq |\mathbf{u}_1^\top \mathbf{S}\mathbf{n}| = \sigma_1 |\mathbf{w}_1^\top \mathbf{n}| = \sigma_1 \delta \|\mathbf{n}\|_2.$$

Next, we upper bound the response to the true signal, using Cauchy-Schwarz:

$$\|\mathbf{S}\mathbf{q}^*\|_2 \leq \|\mathbf{S}\|_F \|\mathbf{q}^*\|_2.$$

Substituting these into R:

$$R \geq \frac{\sigma_1 \delta \|\mathbf{n}\|_2}{\|\mathbf{n}\|_2} \cdot \frac{\|\mathbf{q}^*\|_2}{\|\mathbf{S}\|_F \|\mathbf{q}^*\|_2} = \delta \frac{\sigma_1}{\|\mathbf{S}\|_F} = \delta \frac{1}{\sqrt{\mathrm{er}(\mathbf{S})}}.$$

Using the definition $\mathrm{er}(\mathbf{S}) = d \cdot u(\mathbf{S})$, we obtain the lower bound:

$$R \geq \frac{\delta}{\sqrt{d \cdot u(\mathbf{S})}}.$$

Next, we show the upper bound. To find the maximum error, we first upper bound the response of the system to the noise:

$$\|\mathbf{S}\mathbf{n}\|_2 \leq \|\mathbf{S}\|_F \|\mathbf{n}\|_2.$$

Next, we lower bound the response to the true query by projecting onto $\mathbf{u}_1$:

$$\|\mathbf{S}\mathbf{q}^*\|_2 \geq |\mathbf{u}_1^\top \mathbf{S}\mathbf{q}^*| = \sigma_1 |\mathbf{w}_1^\top \mathbf{q}^*| = \sigma_1 \gamma \|\mathbf{q}^*\|_2.$$

Table 6: Throughput measurements (thousands of tokens/second) and relative speedup factors for DeltaNet and Gated DeltaNet models on a single H100. Baselines (0%) are uncompressed models with head key dimension $d_k = 128$ for DeltaNet and $d_k = 256$ for Gated DeltaNet.

	DeltaNet 370M		**DeltaNet 1.3B**		**DeltaNet 2.7B**		**Gated DeltaNet 370M**		**Gated DeltaNet 1.3B**	
d_k	Throughput	Speedup	Throughput	Speedup	Throughput	Speedup	Throughput	Speedup	Throughput	Speedup
0%	416.2	1.00×	150.7	1.00×	81.5	1.00×	396.5	1.00×	138.4	1.00×
50%	474.8	1.14×	169.4	1.12×	91.7	1.12×	469.0	1.18×	157.8	1.14×
75%	505.9	1.22×	180.4	1.20×	98.6	1.21×	504.5	1.27×	170.1	1.22×
87.5%	521.6	1.25×	185.9	1.23×	99.3	1.22×	521.9	1.32×	176.3	1.26×

Again, substituting these into R:

$$R \leq \frac{\|\mathbf{S}\|_F \|\mathbf{n}\|_2}{\|\mathbf{n}\|_2} \cdot \frac{\|\mathbf{q}^*\|_2}{\sigma_1 \gamma \|\mathbf{q}^*\|_2} = \frac{1}{\gamma} \frac{\|\mathbf{S}\|_F}{\sigma_1} = \frac{\sqrt{\mathrm{er}(\mathbf{S})}}{\gamma}.$$

Using $\mathrm{er}(\mathbf{S}) = d \cdot u(\mathbf{S})$, we obtain the upper bound:

$$R \leq \frac{\sqrt{d \cdot u(\mathbf{S})}}{\gamma}.$$

□

E.5 Calculus

We first need to show that

$$\partial_{\mathrm{vec}(\mathbf{B})} \mathbf{ABx} = \mathbf{x}^\top \otimes \mathbf{A}$$

for matrices $\mathbf{A}, \mathbf{B}$ and a vector $\mathbf{x}$. But this follows immediately from the matrix identity (Petersen et al., 2008, Equation (520))

$$\mathbf{ABx} = (\mathbf{x}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{B})$$

and then taking the derivative.

Next, we need to show that

$$\kappa(\mathbf{x} \otimes \mathbf{A}) = \kappa(\mathbf{A}).$$

But this is follows from (Horn & Johnson, 1994, Theorem 4.2.15) and writing κ as a fraction of singular values, so that

$$\kappa(\mathbf{x} \otimes \mathbf{A}) = \kappa(\mathbf{x})\kappa(\mathbf{A}) = \kappa(\mathbf{A}).$$

□

F Extended Results

F.1 Throughput Measurements

Table 6 shows the speedup achieved by compressing the state space. Compared to Table 7, it measures the throughput of the whole model during training, not just the sequence mixer layer.

F.2 Language Modeling

This subsection contains the extensive zero-shot (Gao et al., 2024) and real-world retrieval (Arora et al., 2024) task evaluations. Table 8 contains averaged results for all models at a fixed compression ratio of 50%. Tables 9-16.

Table 7: Training throughput (**TPS**, tokens per second) of sequence mixers on an NVIDIA H100 (batch size 32, sequence length 2048, number of heads 16, hidden dimension 2048). We fix the per-head value dimension to $d_v = 128$.

Model	d_k	**TPS**	**Speedup**	**Memory**	**Ratio**
DeltaNet	128	8.1M	1.00×	6.33GiB	1.00
	64	10.8M	1.34×	4.57GiB	0.72
	32	12.9M	1.60×	3.70GiB	0.58
Gated DeltaNet	128	7.9M	1.00×	6.35GiB	1.00
	64	10.4M	1.32×	4.59GiB	0.72
	32	12.2M	1.55×	3.71GiB	0.58

Table 8: Comparison of compression methods at a 50% compression ratio pre-RFT across all model sizes, both pre- and post-RFT. We report Perplexity for WikiText2 (**Wiki**) and Lambada (**LMB**), as well as the average accuracy for Zero-Shot (**ZS**) and Retrieval (**Ret**) tasks (best results in **bold**, second best in underlined).

		PRE-RFT			POST-RFT			
MODEL	METHOD	WIKI ↓	LMB ↓	ZS AVG ↑	WIKI ↓	LMB ↓	ZS AVG ↑	RET AVG ↑
DELTANET 370M	L1	3843.5	57304.8	33.0	32.5	56.8	43.0	15.8
	RAND	1032.6	2882.8	34.3	32.6	49.9	42.6	16.1
	DRRQR	**31.4**	36.0	44.7	**29.4**	36.6	**44.5**	**17.8**
	GRAD	31.7	**33.3**	**44.9**	**29.4**	**36.3**	44.4	17.8
	BASELINE	29.8	37.0	44.2	29.8	37.0	44.2	20.8
DELTANET 1.3B	L1	66.1	298.7	41.6	19.0	16.1	48.2	29.1
	RAND	41.2	68.0	44.4	19.4	14.4	48.2	30.3
	DRRQR	20.5	47.2	45.7	17.5	11.8	49.7	31.1
	GRAD	**18.3**	**15.4**	**48.8**	**17.2**	**11.3**	**50.3**	**33.3**
	BASELINE	16.7	11.9	50.0	16.7	11.9	50.0	40.1
GATED DELTANET 370M	L1	41.8	156.6	40.4	29.2	44.6	43.5	18.1
	RAND	35.5	50.7	43.4	29.3	46.5	43.6	18.2
	DRRQR	**31.6**	39.4	**44.0**	**28.7**	40.6	**43.8**	18.3
	GRAD	33.3	**39.2**	43.8	**28.7**	**39.4**	43.7	**19.0**
	BASELINE	28.8	35.9	44.4	28.8	35.9	44.4	23.3
GATED DELTANET 1.3B	L1	26.5	34.0	53.3	16.8	13.0	57.2	32.9
	RAND	18.8	14.4	56.5	16.9	11.5	57.8	32.1
	DRRQR	**17.3**	14.2	57.5	**16.3**	12.0	58.0	33.0
	GRAD	17.4	**10.1**	**58.6**	16.4	**10.5**	**58.3**	**34.3**
	BASELINE	16.8	9.7	59.4	16.8	9.7	59.4	40.3

Table 9: Zero-shot performance of DeltaNet 370M models, evaluated using the *lm-eval-harness* (Gao et al., 2024), given different compression Rates. 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**
✓	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**
×	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**
✓	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
×	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**
✓	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
×	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
✓	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**
Baseline	–	0%	29.8	37.0	51.1	27.6	38.1	52.2	65.0	31.3	44.2

Table 10: Zero-shot performance of DeltaNet 1.3B models, evaluated using the *lm-eval-harness* (Gao et al., 2024), given different compression Rates. 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**
✓	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**
×	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**
✓	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**
×	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**
✓	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**
×	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**
✓	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**
Baseline	–	0%	16.7	11.9	51.3	26.1	50.6	53.3	70.5	48.4	50.0

Table 11: Zero-shot performance of Gated DeltaNet 370M models, evaluated using the *lm-eval-harness* (Gao et al., 2024), given different compression Rates. 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
✓	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**
×	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
✓	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
×	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**
✓	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**
×	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
✓	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**
Baseline	–	0%	28.8	35.9	51.5	28.2	39.6	50.4	65.5	31.5	44.4

Table 12: Zero-shot performance of Gated DeltaNet 1.3B models, evaluated using the *lm-eval-harness* (Gao et al., 2024), given different compression Rates. 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**
✓	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**
×	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**
✓	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**
×	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**
✓	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
×	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**
✓	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**
Baseline	–	0%	16.8	9.7	67.7	41.0	60.1	62.2	74.0	51.5	59.4

Table 13: Accuracy of RFT'ed DeltaNet 370M on recall-intensive retrieval tasks with input truncated to $2K$ tokens, given different compression rates. Computed using *prefix-linear-attention* (Arora et al., 2024).

Method	**Compr.**	Drop cont. ↑	FDA cont. ↑	NQ cont. ↑	SQuAD cont. ↑	SWDE cont. ↑	Triv. cont. ↑	Avg
L1	75%	11.6	1.3	9.1	15.9	6.4	37.7	13.7
Rand	75%	9.6	1.3	9.2	13.8	6.5	31.9	12.1
DRRQR	75%	14.4	2.3	10.7	18.5	7.1	39.6	<u>15.4</u>
Grad	75%	14.0	2.3	10.5	18.5	7.5	40.5	**15.5**
L1	50%	13.2	2.2	12.4	18.3	8.0	40.7	15.8
Rand	50%	14.5	3.2	11.8	18.6	7.9	40.8	16.1
DRRQR	50%	15.1	4.5	13.1	21.7	9.7	43.0	**17.8**
Grad	50%	15.1	5.1	13.1	21.5	9.2	42.9	<u>17.8</u>
L1	40%	13.3	2.8	12.3	18.6	8.5	40.6	16.0
Rand	40%	13.6	4.1	12.9	20.3	9.0	42.5	17.1
DRRQR	40%	15.8	6.0	14.1	22.3	10.2	44.0	**18.7**
Grad	40%	15.4	6.9	14.2	22.7	9.7	43.2	<u>18.7</u>
L1	30%	14.1	3.3	12.8	19.6	8.7	40.7	16.5
Rand	30%	14.8	4.5	12.5	21.7	9.2	42.0	17.4
DRRQR	30%	15.6	8.3	14.1	23.2	10.4	43.7	<u>19.2</u>
Grad	30%	15.1	8.6	14.1	23.8	10.5	43.4	**19.3**
–	0%	15.2	13.1	15.0	24.9	13.0	43.5	20.8

Table 14: Accuracy of RFT'ed DeltaNet 1.3B on recall-intensive retrieval tasks with input truncated to $2K$ tokens, given different compression rates. Computed using *prefix-linear-attention* (Arora et al., 2024).

Method	**Compr.**	Drop cont. ↑	FDA cont. ↑	NQ cont. ↑	SQuAD cont. ↑	SWDE cont. ↑	Triv. cont. ↑	Avg
L1	75%	18.5	7.0	19.1	21.2	17.3	48.2	21.9
Rand	75%	17.3	7.1	17.3	22.1	15.2	46.6	20.9
DRRQR	75%	19.1	9.2	17.8	22.3	17.7	48.2	22.4
Grad	75%	18.9	15.6	20.3	26.0	20.9	51.4	**25.5**
L1	50%	20.7	23.1	21.6	27.3	29.0	53.0	29.1
Rand	50%	19.6	30.1	24.1	29.4	26.6	52.0	30.3
DRRQR	50%	20.0	31.5	23.3	28.7	29.2	53.6	<u>31.1</u>
Grad	50%	19.6	38.6	25.2	30.4	32.2	53.6	**33.3**
L1	40%	19.1	29.9	22.9	27.4	32.2	53.3	30.8
Rand	40%	21.1	32.9	24.8	30.0	31.7	52.8	32.2
DRRQR	40%	19.8	38.0	23.4	29.8	33.2	53.6	<u>33.0</u>
Grad	40%	20.3	44.6	25.7	31.1	35.6	54.4	**35.3**
L1	30%	21.5	34.2	24.9	28.4	31.6	53.9	32.4
Rand	30%	21.5	38.2	25.5	30.9	35.2	54.0	34.2
DRRQR	30%	20.6	41.7	24.9	30.5	34.9	53.8	<u>34.4</u>
Grad	30%	20.6	44.4	26.4	31.7	37.9	55.3	**36.1**
–	0%	21.0	57.8	27.2	34.6	42.8	57.5	40.1

Table 15: Accuracy of RFT'ed Gated DeltaNet 370M on recall-intensive retrieval tasks with input truncated to $2K$ tokens, given different compression rates. Computed using *prefix-linear-attention* (Arora et al., 2024).

Method	**Compr.**	Drop cont. ↑	FDA cont. ↑	NQ cont. ↑	SQuAD cont. ↑	SWDE cont. ↑	Triv. cont. ↑	Avg
L1	75%	13.9	2.2	10.0	18.2	5.3	36.9	14.4
Rand	75%	13.8	2.2	10.2	17.6	4.8	37.1	14.3
DRRQR	75%	13.7	1.9	10.3	18.7	5.5	40.6	<u>15.1</u>
Grad	75%	14.1	2.3	10.3	19.4	5.3	40.0	**15.2**
L1	50%	16.1	5.8	12.5	22.8	9.5	41.7	18.1
Rand	50%	14.8	6.8	13.3	22.6	8.9	42.7	18.2
DRRQR	50%	16.1	4.6	13.0	23.4	10.4	42.6	<u>18.3</u>
Grad	50%	16.6	6.3	13.8	23.1	9.4	45.1	**19.0**
L1	40%	16.9	8.9	13.6	23.8	11.5	42.9	19.6
Rand	40%	16.4	10.7	13.9	24.2	10.7	42.8	19.8
DRRQR	40%	16.5	6.0	13.6	23.8	11.6	43.7	19.2
Grad	40%	18.2	8.1	14.1	24.1	11.9	45.2	**20.3**
L1	30%	18.5	11.1	14.0	24.6	13.1	43.7	20.8
Rand	30%	16.4	13.0	14.7	25.0	12.6	43.4	20.8
DRRQR	30%	17.8	8.3	14.7	24.7	12.8	44.6	20.5
Grad	30%	18.4	9.6	15.1	25.2	12.8	45.6	**21.1**
–	0%	18.1	16.3	16.0	26.8	16.8	46.1	23.3

Table 16: Accuracy of RFT'ed Gated DeltaNet 1.3B on recall-intensive retrieval tasks with input truncated to $2K$ tokens, given different compression rates. Computed using *prefix-linear-attention* (Arora et al., 2024).

Method	**Compr.**	Drop cont. ↑	FDA cont. ↑	NQ cont. ↑	SQuAD cont. ↑	SWDE cont. ↑	Triv. cont. ↑	Avg
L1	75%	18.4	13.4	20.5	29.1	15.8	56.8	25.7
Rand	75%	19.2	11.2	18.0	28.0	14.1	52.8	23.9
DRRQR	75%	17.9	18.7	19.9	28.7	15.0	56.1	26.1
Grad	75%	17.8	17.3	21.1	29.1	17.2	58.7	**26.9**
L1	50%	22.8	28.9	24.3	34.9	25.4	60.9	32.9
Rand	50%	20.3	31.1	23.0	33.3	25.2	59.8	32.1
DRRQR	50%	20.5	32.1	23.5	33.6	26.5	61.6	33.0
Grad	50%	20.8	36.6	24.5	33.8	28.4	61.7	**34.3**
L1	40%	22.4	33.1	25.9	34.7	29.6	62.1	34.6
Rand	40%	22.2	39.5	24.0	34.7	26.0	61.2	34.6
DRRQR	40%	21.5	39.1	24.6	34.8	29.2	61.1	<u>35.0</u>
Grad	40%	21.7	39.8	25.1	34.7	31.1	62.7	**35.8**
L1	30%	20.9	37.2	26.0	35.4	30.5	62.4	35.4
Rand	30%	22.8	44.4	25.1	35.6	33.4	62.6	<u>37.3</u>
DRRQR	30%	21.6	42.4	25.2	35.4	32.2	61.9	36.5
Grad	30%	22.6	45.7	26.4	36.2	33.7	63.5	**38.0**
–	0%	22.1	53.7	27.0	37.0	37.5	64.5	40.3