# UNLOCKING FEATURE LEARNING IN GATED DELTA NETWORKS AT SCALE

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Training and scaling Large Language Models demand enormous computational resources, motivating both efficient sub-quadratic architectures and principled hyperparameter tuning methods. While the Maximal Update Parametrization ($\mu$P) has enabled zero-shot hyperparameter transfer for standard Transformers, its extension to linear models, particularly those with structured state transitions and complicated architectures, remains largely unexplored. By rigorously propagating coordinate-size estimates through the forward pass, gating mechanisms, and recurrent state dynamics, we derive the scaling rules for Gated Delta Network. Experiments on language-model pre-training confirm that our configurations enable stable learning-rate transfer across model widths under both AdamW and SGD, whereas standard parametrization fails to transfer, providing empirical support and practical utility of our analysis.

## 1 INTRODUCTION

The rapid development of Large Language Models (LLMs) has demonstrated remarkable capabilities across a wide range of downstream tasks (Brown et al., 2020; Touvron et al., 2023; Radford et al., 2019; Vaswani, 2017). However, scaling these models to larger sizes introduces two challenges. First, empirical scaling laws show that optimal performance requires increased model size, while the computational budget required for training grows steeply with model scale (Kaplan et al., 2020; Hoffmann et al., 2022). Second, the efficiency of standard Transformer architecture is limited by the quadratic complexity of softmax self-attention with respect to sequence length, making it increasingly costly for long-context inference and training (Katharopoulos et al., 2020).

Linear models have been proposed to address these issues. The original linear attention (Katharopoulos et al., 2020) rewrites softmax attention as a linear kernel, enabling recurrent-form inference at constant per-step cost. Structured state space models (SSMs) such as S4 (Gu et al., 2022), Mamba (Gu & Dao, 2024) and Mamba-2 (Dao & Gu, 2024) utilize recurrent state spaces to represent long-range dependencies within linear structures. A particularly promising family of linear recurrent models is based on the delta rule (Widrow & Hoff, 1960), which updates a fast-weight matrix by subtracting the prediction error of the current key-value pair. Furthermore, the DeltaNet (Yang et al., 2024c) introduced a hardware-efficient parallel training algorithm for delta-rule Transformers, enabling scaling to large language models. Afterwards, Gated Delta Network (Yang et al., 2025) augmented DeltaNet with the data-dependent gating mechanism of Mamba-2, which achieves strong language modeling performance while maintaining linear-time inference.

Simultaneously, training deep networks requires careful selection of hyperparameters such as learning rates, which are expensive to tune through grid search (Snoek et al., 2012; 2015), and whose optimal values often change greatly with model scale. Meta-learning approaches have been explored to transfer hyperparameters across tasks and datasets (Yogatama & Mann, 2014; Perrone et al., 2018; Horváth et al., 2021; Akiba et al., 2019). A more principled solution is offered by the Maximal Update Parametrization ($\mu$P) (Yang & Hu, 2021), which identifies the valid parametrization of a neural network that supports feature learning in the infinite-width limit, as formalized through the Tensor Programs framework (Yang et al., 2022; Yang & Littwin, 2023; Yang et al., 2024a). $\mu$P theories demonstrated that hyperparameters tuned on small proxy models transfer zero-shot to large target models, with extensions to adaptive optimizers (Yang & Littwin, 2023; Ishikawa & Karakida, 2024; Everett et al., 2024) and a spectral reformulation (Yang et al., 2023). Subsequent work has successfully applied $\mu$P to other fields (Blake et al., 2025; Dey et al., 2024; Hajjar et al., 2024), and even industrial models (Meta AI, 2025; Team et al., 2025).



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.

Despite the development of efficient linear architectures, how to properly parametrize them for feature learning at scale has received very limited attention. The core challenge is that their recurrent state is updated through the sequence dimension, which does not fit the standard feedforward or attention-based derivations of $\mu$P. The only prior work on this is Vankadara et al. (2024), which shows that vanilla $\mu$P and spectral scaling conditions both fail to support feature learning in diagonal SSMs like Mamba, and proposes a corrected scaling rule for them. However, the Gated Delta Network (also named as "Gated DeltaNet") differs from diagonal SSMs fundamentally, since its recurrent state is a full matrix updated with additional data-dependent scalar gating through two separate weight matrices. These differences make the SSM-specific analysis of Vankadara et al. (2024) inapplicable, leaving the $\mu$P parametrization of Gated DeltaNet an open problem.

In this paper, we formally derive the complete $\mu$P formulation for Gated DeltaNet. Our main contributions are:

- We theoretically derive coordinate-size estimates through the full forward pass. We also derive principled initialization variances, forward multipliers, and learning-rate scalings for all weight classes. We discover that the gating weight matrices require a non-standard $\Theta(1/\sqrt{d})$ learning-rate scaling, and the scalar gating parameters require a $\Theta(\sqrt{d})$ scaling, both of which deviate from standard $\mu$P setting.
- We pretrain Gated DeltaNet language models across multiple widths and show that our $\mu$P formulation enables zero-shot learning-rate transfer under both AdamW and SGD optimizers, while standard parametrization fails to transfer, confirming both the theoretical derivation and its practical efficiency.

## 2 Related Work

**Efficient Sequence Models** The standard Transformer (Vaswani, 2017) and its variants (Radford et al., 2019; Brown et al., 2020; Touvron et al., 2023) have become the dominant architecture for large-scale language modeling, but their quadratic attention complexity limits its application in much larger language models. Linear attention (Katharopoulos et al., 2020) replaces the softmax with a kernel function that permits rewriting attention as a linear RNN, enabling $O(1)$ per-step inference. Structured state space models (SSMs) improve by integrating recurrent state spaces. S4 (Gu et al., 2022) introduces HiPPO-based (Gu et al., 2020) structured matrices for long-range sequence modeling, Mamba (Gu & Dao, 2024) adds input-selective state transitions for improved performance, and Mamba-2 (Dao & Gu, 2024) unifies SSMs with structured matrix attention via state-space duality. Other notable architectures include RetNet (Sun et al., 2023), RWKV (Peng et al., 2023), Gated Linear Attention (Yang et al., 2024b), HGRN (Qin et al., 2023) and its expansion HGRN2 (Qin et al., 2024). Recently, numerous models incorporate the delta rule (Widrow & Hoff, 1960), which subtracts the prediction error rather than accumulating outer products. This was formalized in the fast-weight programmer framework (Schlag et al., 2021), with recurrent extensions in Irie et al. (2021). DeltaNet (Yang et al., 2024c) further introduced a hardware-efficient parallel training algorithm, and Gated Delta Networks (Yang et al., 2025) combined the delta-rule state update with the data-dependent gating of Mamba-2 with improved performance. Our work focuses precisely on this architecture and derives its $\mu$P formulation.

**Hyperparameter Transfer** Prior work has explored approaches to reduce the cost of hyperparameter search of training deep learning models (Snoek et al., 2012; 2015; Jamieson & Talwalkar, 2016; Akiba et al., 2019). Some have also explored the methods to transfer learning between different tasks or datasets (Horváth et al., 2021; Perrone et al., 2018; Yogatama & Mann, 2014). Moreover, based on standard parametrization (SP, such as Xavier initialization (Glorot & Bengio, 2010) and Kaiming initialization (He et al., 2015)), Yang & Hu (2021) proposed Maximal Update Parametrization ($\mu$P) based on abc-parameterization framework, which unifies previous parametrization methods such as SP, Neural Tangent Kernel (NTK) (Jacot et al., 2018) and Mean Field parametrization (Chizat & Bach, 2018; Mei et al., 2018; Sirignano & Spiliopoulos, 2020; Rotskoff & Vanden-Eijnden, 2022). It enables feature learning in the infinite-width limit. And based on $\mu$P, Yang et al. (2022) proposed *$\mu$Transfer* that can implement zero-shot transfer of hyperparameters to large models from a much smaller proxy model, and generalized it to different architectures and optimizers (Yang & Littwin, 2023), such as SGD, Adagrad (Duchi et al., 2011) and Adam (Kingma & Ba, 2015). Following that, they also reformulated $\mu$P from the perspective of spectral norm (Yang et al., 2023). Recently, many researchers tried to further generalize $\mu$P in other fields or successfully scale LLMs with $\mu$P (Blake et al., 2025; Meta AI, 2025; Haas et al., 2024; Dey et al., 2024; Hajjar et al., 2024; Team et al., 2025).

However, the question of how to properly parametrize these models for feature learning at scale has received very little attention. The core challenge is that original $\mu$P designs cannot be directly applied to these architectures with recurrent state transitions and structured matrix operations. To our knowledge, the only work that formally addresses $\mu$P-style parametrization for this class of models is Vankadara et al. (2024), which studies the scaling behavior of structured SSMs like Mamba. Their analysis reveals that both vanilla $\mu$P and spectral scaling conditions fail to support feature learning in SSMs, and they derive the scaling rule for SSMs that recovers feature learning. The parametrization of Gated Delta Net, however, differs from the diagonal SSMs greatly, since its state is updated via outer-product delta rules rather than scalar recurrences with matrix-valued hidden states. This work is, to our knowledge, the first to derive and validate a $\mu$P-consistent parametrization for Gated Delta Network.

# 3 PRELIMINARIES

## 3.1 GATED DELTANET

Proposed by Yang et al. (2025), Gated DeltaNet is a variant of linear transformer (Katharopoulos et al., 2020), based on the Mamba 2 architecture (Dao & Gu, 2024). For the query, key and value vectors $\mathbf{q}_t, \mathbf{k}_t$ and $\boldsymbol{b}_t$ similar to the original Transformer, the update rule of the latent state is shown as:

$$\mathbf{S}_t = \mathbf{S}_{t-1}(\alpha_t(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top)) + \beta_t \boldsymbol{b}_t \mathbf{k}_t^\top, \tag{3.1}$$

where $\alpha_t \in (0, 1)$ is the data-dependent gating scale and $\beta_t \in (0, 1)$ is the "writing strength" of the current input at time $t$, as proposed in Widrow & Hoff (1960); Schlag et al. (2021). And the output is just direct readout of the latent state on the query:

$$\mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t. \tag{3.2}$$

Different from the Transformer, Gated DeltaNet added a short convolution after query, key and value projections, followed by a SiLU activation layer. There are also L2 Normalization layer for queries and keys. And there is also an RMSNorm layer before the output projection to stabilize the training. As discussed in the original paper, these norm are crucial to the performance of Gated Delta Net.

## 3.2 $\mu$P THEORY

In deep learning, models are frequently scaled by increasing their hidden dimension or width $d$. Under the Standard Parameterization (SP), including He (He et al., 2015) or Xavier (Glorot & Bengio, 2010) initialization, hidden weights are typically initialized with entries drawn from $\mathcal{N}(0, \sigma^2/d)$ and optimized using a uniform learning rate $\eta$ across all layers. However, as $d$ goes to infinity, SP encounters fundamental limitations. If the learning rate remains constant, the network's activations and gradients diverge. To prevent this instability, $\eta$ must be scaled down by $\mathcal{O}(1/d)$, which forces the network into the Neural Tangent Kernel (NTK) or "lazy training" regime Yang & Hu (2021), where the intermediate representations (features) seldom evolve from their initialized state, meaning the network fails to perform real feature learning.

To resolve the trade-off between stability and feature learning in the infinite-width limit, Yang & Hu (2021) proposed the *Maximal Update Parameterization* ($\mu$P) using the Tensor Programs framework. $\mu$P provides rigorous configurations for scaling weight initializations and learning rates as a function of the width $d$ (sometimes a width-dependent multiplier on the weight is required; refer to Tables 2 and 1 for AdamW and SGD configurations) to ensure feature learning. In this setting, feature updates at every layer remain bounded and non-vanishing (i.e., $\Delta h = \Theta(1)$) as the model expands to infinity width. To further illustrate this, the definition of coordinate size should be first introduced:

**Definition 3.1.** A vector $v \in \mathbb{R}^d$ has $\Theta(d^a)$-sized coordinates if $\|v\|^2/d = \Theta(d^{2a})$, i.e., each entry of $v$ has variance $\Theta(d^{2a})$ as $d \to \infty$. When $d$ is large, the coordinates of the vectors being studied are regarded as roughly i.i.d. Gaussian.

Based on the definition above, $\mu$P theory proposes three desiderata. Firstly, every (pre)activation vector should have $\Theta(1)$-sized coordinates; and the output of a network should be $O(1)$; moreover, all parameters should be updated as much as possible without leading to divergence. And based on these desiderata and the assumption of feature learning, there are some derivatives. For example, the gradient to a hidden state is with $\Theta(1/d)$ coordinate size when optimized with SGD optimizer.

# 4 THE $\mu$P FORWARD ANALYSIS OF GATED DELTA NET

In this section, we will derive the scaling law of this architecture.

We derive the Maximal Update Parametrization ($\mu$P) conditions for Gated DeltaNet by propagating coordinate-size estimates through the forward pass and the gating mechanisms, then conclude with the implications for the AdamW optimizer.

**Notation and standing assumptions.** Following Yang et al. (2022), we say a vector $\mathbf{z} \in \mathbb{R}^d$ has $\Theta(1)$ coordinate size if $\|\mathbf{z}\|_2 = \Theta(\sqrt{d})$, i.e. each coordinate is of order $\Theta(1)$ in magnitude. Equivalently, the per-coordinate variance of $\mathbf{z}$ is $\Theta(1)$. For a matrix $\mathbf{A} \in \mathbb{R}^{d\times d}$, we say it has $\Theta(c)$ coordinate size if each entry is of order $\Theta(c)$ in magnitude.

We assume throughout that the hidden state $\mathbf{x}_t \in \mathbb{R}^d$ satisfies the $\mu$P feature-learning condition, namely

$$\|\mathbf{x}_t\|_2 = \Theta(\sqrt{d}), \qquad \|\Delta\mathbf{x}_t\|_2 = \Theta(\sqrt{d}),$$

so that $\mathbf{x}_t$ has $\Theta(1)$ coordinate size and its update is of the same order. To isolate the effect of the parametrization we temporarily ignore the SiLU activations (see Remark 4.1 below).

## 4.1 COORDINATE SIZES OF THE PROJECTED FEATURES

Let $\widetilde{\mathbf{q}}_t = \text{ShortConv}(\mathbf{W}_q\,\mathbf{x}_t)$, $\widetilde{\mathbf{k}}_t = \text{ShortConv}(\mathbf{W}_k\,\mathbf{x}_t)$, and $\boldsymbol{b}_t = \text{ShortConv}(\mathbf{W}_v\,\mathbf{x}_t)$, where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{d\times d}$ are the query, key, and value projection matrices, respectively. Under the $\mu$P initialization of hidden weights (Yang et al., 2022), the products $\mathbf{W}_q\,\mathbf{x}_t$, $\mathbf{W}_k\,\mathbf{x}_t$, $\mathbf{W}_v\,\mathbf{x}_t$ each have $\Theta(1)$ coordinate size; the short convolution preserves this order, so $\widetilde{\mathbf{q}}_t, \widetilde{\mathbf{k}}_t$ and $\boldsymbol{b}_t$ each have $\Theta(1)$ coordinate size. The L2-normalized query and key are

$$\mathbf{q}_t = \frac{\widetilde{\mathbf{q}}_t}{\|\widetilde{\mathbf{q}}_t\|_2},\ \mathbf{k}_t = \frac{\widetilde{\mathbf{k}}_t}{\|\widetilde{\mathbf{k}}_t\|_2}.$$

Since $\|\widetilde{\mathbf{q}}_t\|_2 = \Theta(\sqrt{d})$, each coordinate of $\mathbf{q}_t$ and $\mathbf{k}_t$ is of order $\Theta(1)/\Theta(\sqrt{d}) = \Theta(1/\sqrt{d})$, i.e., $\mathbf{q}_t$ and $\mathbf{k}_t$ both have $\Theta(1/\sqrt{d})$ coordinate size.

## 4.2 COORDINATE SIZE OF THE LATENT STATE

The rank-one write update in equation 3.1 is $\mathbf{U}_t = \beta_t\,\boldsymbol{b}_t\mathbf{k}_t^\top$. Since $\beta_t \in (0,1)$ as a bounded scalar and combining the $\Theta(1)$ coordinate size of $\boldsymbol{b}_t$ with the $\Theta(1/\sqrt{d})$ coordinate size of $\mathbf{k}_t$, each entry of $\mathbf{U}_t$ satisfies

$$(\mathbf{U}_t)_{ij} = \beta_t\,(\boldsymbol{b}_t)_i\,(\mathbf{k}_t)_j = \Theta(1)\cdot\Theta\Big(\frac{1}{\sqrt{d}}\Big) = \Theta\Big(\frac{1}{\sqrt{d}}\Big).$$

For the cumulative latent state $\mathbf{S}_t$, we apply the argument of Vankadara et al. (2024)[1]: unless the write update $\mathbf{U}_t$ perfectly cancels the residual term in equation 3.1 at every step $t$, the steady-state variance of $\mathbf{S}_t$ matches that of $\mathbf{U}_t$. More precisely, the spectral contraction factor of the map $\mathbf{S} \mapsto \mathbf{S}\,\alpha_t(\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^\top)$ is at most $\alpha_t(1 - \beta_t\|\mathbf{k}_t\|_2^2) \le \alpha_t$, which is strictly less than 1 when $\alpha_t, \beta_t \in (0,1)$ are both bounded away from 0 and 1. We assume this condition holds throughout the analysis. Under this assumption the geometric sum of write updates converges and $\mathbf{S}_t$ has $\Theta(1/\sqrt{d})$ coordinate size.

## 4.3 COORDINATE SIZE OF THE READOUT

The output is $\mathbf{o}_t = \mathbf{S}_t\,\mathbf{q}_t$, so the $j$-th coordinate is

$$(\mathbf{o}_t)_j = \sum_{i=1}^{d} (\mathbf{S}_t)_{ji}\,(\mathbf{q}_t)_i.$$

Treating the entries of $\mathbf{S}_t$ and $\mathbf{q}_t$ as approximately independent and zero-mean, each with variance $\Theta(1/d)$, the variance of the sum is

$$\text{Var}\big[(\mathbf{o}_t)_j\big] = \sum_{i=1}^{d} \text{Var}\big[(\mathbf{S}_t)_{ji}\big]\cdot\text{Var}\big[(\mathbf{q}_t)_i\big] = d\cdot\Theta\Big(\frac{1}{d}\Big)\cdot\Theta\Big(\frac{1}{d}\Big) = \Theta\Big(\frac{1}{d}\Big),$$

[1]The detailed derivations can be found in Appendix A.2.

so $\mathbf{o}_t$ has $\Theta(1/\sqrt{d})$ coordinate size. Although the subsequent RMSNorm forces its output to $\Theta(1)$ coordinate size, this implicit rescaling would disrupt the gradient scaling required by $\mu$P. We therefore recommend inserting a $\sqrt{d}$-multiplier before RMSNorm so that the input to RMSNorm is already $\Theta(1)$:

$$\mathbf{o}_t \mapsto \sqrt{d}\mathbf{o}_t, \qquad \|\sqrt{d}\mathbf{o}_t\|_2 = \Theta(\sqrt{d}).$$

An equivalent alternative is to replace the L2-Normalization on $\mathbf{q}_t$ with an RMSNorm layer, which absorbs the same $\sqrt{d}$-factor. With either modification, the standard $\mu$P formulation applies to all projection weights other than those governing $\alpha_t$ and $\beta_t$.

### 4.4 First-order analysis of the gating scalars

The gating parameters are defined as

$$\beta_t = \sigma(\mathbf{W}_\beta \mathbf{x}_t) \in (0,1), \qquad \mathbf{W}_\beta \in \mathbb{R}^{1\times d},$$

and

$$\alpha_t = e^{g_t} \in (0,1), \qquad g_t = -e^{a_{\log}} \ln(1 + e^{\mathbf{W}_\alpha \mathbf{x}_t + b}),$$

with $\mathbf{W}_\alpha \in \mathbb{R}^{1\times d}$ a trainable weight row and $a_{\log}, b \in \mathbb{R}$ scalar parameters shared within each head. Because both $\alpha_t$ and $\beta_t$ are nonlinear transformations of a Gaussian-distributed pre-activation, they are not themselves Gaussian, and traditional $\mu$P theory does not directly apply.

For $\alpha_t$, since it is bounded by $(0,1)$, it is naturally $\Theta(1)$. Define $z_{\alpha,t} = \mathbf{W}_\alpha \mathbf{x}_t + b$, when $z_{\alpha,t} + a_{\log} \ll 0$, $|\partial\alpha_t/\partial z_{\alpha,t}| = \alpha_t e^{z_{\alpha,t}+a_{\log}}/(1+e^{z_{\alpha,t}})$ is also $\Theta(1)$. Under the original initialization of Yang et al. (2025), namely $a_{\log} \sim \mathrm{Uniform}(0,16)$ and $b = b_0 + \ln(1-e^{-b_0})$ with $b_0 = 10^{2\epsilon_b - 3}$, $\epsilon_b \sim \mathrm{Uniform}(0,1)$, the gradient $|\partial\alpha_t/\partial z_{\alpha,t}| = \alpha_t \cdot e^{z_{\alpha,t}+a_{\log}}/(1+e^{z_{\alpha,t}}) = e^{z_{\alpha,t}+a_{\log}}/(1+e^{z_{\alpha,t}})^{e^{a_{\log}}+1}$ is bounded by 1.

*Proof.* Denote $k = e^{a_{\log}} > 0, p = e^{z_{\alpha,t}} > 0$, and $f(k,p) = \log(\frac{kp}{(1+p)^{k+1}}) = \log(kp) - (k+1)\log(1+p)$. Then we have $\frac{\partial f}{\partial p} = \frac{1}{p} - \frac{1+k}{1+p} = \frac{1-pk}{p(1+p)}$. Therefore, $\forall k > 0$, $f(k,p) \le f(k, 1/k) = \log(\frac{1}{(1+1/k)^{k+1}}) < 0$, and $|\partial\alpha_t/\partial z_{\alpha,t}| = e^{f(k,p)} < 1$. □

For $\beta_t$, since $z_{\beta,t} := \mathbf{W}_\beta \mathbf{x}_t$ is $\Theta(1)$, $\beta_t \in (0,1)$ does not saturate and $\beta_t = \sigma(z_{\beta,t})$, therefore, $\frac{\partial\beta_t}{\partial z_{\beta,t}} = \beta_t(1-\beta_t) = \Theta(1)$.

Combining the two analyses, $\Theta(1)$ coordinate-size behavior of both $\alpha_t$ and $\beta_t$ is maintained under $\mu$P initialization. Consequently, $\mathbf{W}_\alpha$ and $\mathbf{W}_\beta$ may be treated as hidden weights with initial variance $1/d$, while the scalar parameters $a_{\log}$ and $b$ are assigned constant (i.e., width-independent) initial variance.

**Remark 4.1** (Effect of SiLU activations)**.** The analysis above assumes that the SiLU activations $\mathrm{SiLU}(x) = x/(1+e^{-x})$ following the short convolutions are suppressed. In practice, these activations introduce non-Gaussian statistics in $\widetilde{\mathbf{q}}_t$, $\widetilde{\mathbf{k}}_t$, and $\boldsymbol{b}_t$. Therefore, the derivations above are only an approximation to the ideal $\mu$P conditions. The approximation quality degrades if the pre-activations are far from zero, but remains adequate for the initialization in $\mu$P.

## 5 The $\mu$P Analysis of Gated DeltaNet under SGD

In this section, we will derive the backward process of the $\mu$P configuration and the learning rates for each module accordingly. Due to page limitation, we postpone the analysis for AdamW to Appendix B, where AdamW optimizer benefits from a key simplification: it normalizes the gradient by its coordinate-wise second moment, so the effective update magnitude is $\Theta(\eta)$ per coordinate regardless of the raw gradient scale. Consequently, all weight classes share the same $\Theta(1)$ update magnitude, and the learning-rate schedule needs only to account for the number of terms in the activation sum (i.e. the fan-in $n_{\ell-1}$).

Under plain SGD this normalization is absent. The update rule is

$$W^\ell \leftarrow W^\ell - \eta^\ell \nabla_{W^\ell} \mathcal{L},$$

so the magnitude of the change $\Delta z^\ell = (\Delta W^\ell) h^{\ell-1}$ is proportional to both the learning rate and the raw gradient magnitude. Different weights in Gated DeltaNet receive different gradient magnitudes, so they require different learning-rate scalings to satisfy the $\mu$P feature-learning condition $\|\Delta h^\ell\|_2 = \Theta(\sqrt{d})$.

### 5.1 NOTATION AND ASSUMPTIONS

We retain all conventions and assumptions from Section 4: the hidden state $\mathbf{x}_t$ satisfies $\|\mathbf{x}_t\|_2 = \Theta(\sqrt{d})$, all forward-pass estimates carry over unchanged, and SiLU activations are suppressed (see Remark 4.1). And we write $\eta^\ell$ for the per-layer SGD learning rate.

**Assumption 5.1** (Short effective memory and domination of direct gradient)**.** The loss gradient with respect to the latent state satisfies

$$\frac{\partial \mathcal{L}}{\partial \mathbf{S}_t} = \mathbf{g}_t \mathbf{q}_t^\top + \underbrace{\sum_{\tau=t+1}^{T} \Big( \prod_{s=t+1}^{\tau} \alpha_s \Big) \mathbf{g}_\tau \mathbf{q}_\tau^\top \Big( \prod_{s=t+1}^{\tau} (\mathbf{I} - \beta_s \mathbf{k}_s \mathbf{k}_s^\top) \Big)}_{\text{BPTT tail}},$$

where the first term is the direct contribution from the readout at step $t$ and the second term accumulates gradients from all future readouts through the recurrent state chain. Each BPTT (Backpropagation-through-time) term is individually of the same $d$-order as the direct term: its $(a, b)$-entry has magnitude $\Theta(1/d)$, identical to $(\mathbf{g}_t)_a(\mathbf{q}_t)_b$. However, the BPTT sum contains up to $T - t$ terms.

We assume throughout this section that the effective memory length $L_{\text{eff}} := \sum_{\tau=t}^{T} \prod_{s=t+1}^{\tau} \alpha_s = O(1)$, i.e. the gating values $\alpha_s$ decay the past state sufficiently fast so that $\bar{\alpha} := \mathbb{E}[\alpha_t]$ satisfies $(1 - \bar{\alpha})^{-1} = O(1)$. Under this assumption the BPTT tail is $O(1)$ times the direct term in the $\Theta(\cdot)$ sense, and all gradient estimates below use only the direct term $\partial \mathcal{L}/\partial \mathbf{S}_t = \mathbf{g}_t \mathbf{q}_t^\top$ without loss of $d$-scaling accuracy.

When the model is used in a long-context scenario with slow forgetting ($\alpha_t \to 1$), $L_{\text{eff}}$ can grow as $O(T)$. In this case, the $\mu$P learning-rate prescriptions should be scaled down by $1/L_{\text{eff}}$ accordingly. Due to page limitations, we postpone the derivations of the gradients of query, key and value projections and the detailed derivations of the BPTT term in Appendix A.

### 5.2 BACKWARD ERROR AT THE READOUT

The readout is $\mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t$, followed by a $\sqrt{d}$-multiplier and RMSNorm. The loss-gradient at $\mathbf{o}_t$ satisfies

$$\mathbf{g}_t := \frac{\partial \mathcal{L}}{\partial \mathbf{o}_t} = \sqrt{d} \cdot \frac{\partial \mathcal{L}}{\partial (\sqrt{d}\, \mathbf{o}_t)}.$$

If we assume that all the architectures outside Gated DeltaNet follow $\mu$P rules, then it is assumed that $\frac{\partial \mathcal{L}}{\partial (\sqrt{d}\, \mathbf{o}_t)}$ has $\Theta(1/d)$ coordinate size and $\mathbf{g}_t$ is with $\Theta(1/\sqrt{d})$ coordinate size.

### 5.3 GRADIENT OF THE GATING

We now show that the gating weights require a different learning rate scaling from the projection matrices, which is a distinctive feature of the SGD formulation absent in the AdamW case.

**Gating weight $\mathbf{W}_\alpha$.** Note that $\alpha_t$ affects not only $\mathbf{S}_t$ but also all future states $\mathbf{S}_{t'}$ for $t' > t$ through the recurrence chain equation 3.1. However, under Assumption 5.1 those future contributions are $O(L_{\text{eff}}) = O(1)$ times the direct contribution and do not change the $d$-scaling of the gradient.

By the chain rule via the latent state $\mathbf{S}_t$:

$$\frac{\partial \mathcal{L}}{\partial \alpha_t} = \text{Tr}\left( \left( \frac{\partial \mathcal{L}}{\partial \mathbf{S}_t} \right)^\top \frac{\partial \mathbf{S}_t}{\partial \alpha_t} \right)$$

Since $\frac{\partial \mathcal{L}}{\partial \mathbf{S}_t} = \mathbf{g}_t \mathbf{q}_t^\top$ and $\frac{\partial \mathbf{S}_t}{\partial \alpha_t} = \mathbf{S}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top)$, we have

$$\frac{\partial \mathcal{L}}{\partial \alpha_t} = \text{Tr}\left( \mathbf{q}_t \mathbf{g}_t^\top \mathbf{S}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) \right) = \underbrace{\mathbf{g}_t^\top \mathbf{S}_{t-1} \mathbf{q}_t}_{\text{Term 1}} - \underbrace{\beta_t (\mathbf{g}_t^\top \mathbf{S}_{t-1} \mathbf{k}_t) \langle \mathbf{k}_t, \mathbf{q}_t \rangle}_{\text{Term 2}}.$$

For Term 1, $(\mathbf{S}_{t-1}\mathbf{q}_t)_i = \sum_j (\mathbf{S}_{t-1})_{ij} (\mathbf{q}_t)_j$. Since both are zero-mean independent variables of order $\Theta(1/\sqrt{d})$, we have $\mathbf{S}_{t-1}\mathbf{q}_t = \Theta(1/\sqrt{d})$ per coordinate. Moreover, since $\mathbf{g}_t$ is $\Theta(1/\sqrt{d})$ per

coordinate, finally we arrive that Term 1 is $\Theta(1/\sqrt{d})$. For Term 2, similarly we have $\mathbf{g}_t^\top \mathbf{S}_{t-1}\mathbf{k}_t = \Theta(1/\sqrt{d})$. Multiplying by $\langle \mathbf{k}_t, \mathbf{q}_t\rangle = \Theta(1/\sqrt{d})$ yields $\Theta(1/d)$. Therefore,

$$\frac{\partial \mathcal{L}}{\partial \alpha_t} = \Theta\left(\frac{1}{\sqrt{d}}\right).$$

Since $\partial \alpha_t / \partial z_{\alpha,t} = \Theta(1)$, finally we have

$$\frac{\partial \mathcal{L}}{\partial z_{\alpha,t}} = \frac{\partial \mathcal{L}}{\partial \alpha_t} \cdot \frac{\partial \alpha_t}{\partial z_{\alpha,t}} = \Theta\left(\frac{1}{\sqrt{d}}\right) \cdot \Theta(1) = \Theta\left(\frac{1}{\sqrt{d}}\right) \tag{5.1}$$

Then the gradient per entry of $\mathbf{W}_\alpha$ is

$$\frac{\partial \mathcal{L}}{\partial (\mathbf{W}_\alpha)_j} = \frac{\partial \mathcal{L}}{\partial z_{\alpha,t}} \cdot (\mathbf{x}_t)_j = \Theta\Big(\frac{1}{\sqrt{d}}\Big) \cdot \Theta(1) = \Theta\Big(\frac{1}{\sqrt{d}}\Big).$$

To enable feature learning, the scalar pre-activation $z_\alpha$ must move by $\Theta(1)$ after one gradient step. The change in $z_\alpha$ is

$$\Delta z_\alpha = \Delta \mathbf{W}_\alpha \mathbf{x}_t = -\eta_\alpha \sum_{j=1}^{d} \frac{\partial \mathcal{L}}{\partial (\mathbf{W}_\alpha)_j} (\mathbf{x}_t)_j = -\eta_\alpha \frac{\partial \mathcal{L}}{\partial z_{\alpha,t}} \sum_{j=1}^{d} (\mathbf{x}_t)_j^2.$$

Because $\sum_j (\mathbf{x}_t)_j^2 = \|\mathbf{x}_t\|_2^2 = \Theta(d)$, we have

$$\Delta z_\alpha = \eta_\alpha \cdot \Theta\left(\frac{1}{\sqrt{d}}\right) \cdot \Theta(d) = \eta_\alpha \Theta(\sqrt{d}).$$

For $\Delta z_\alpha = \Theta(1)$, the learning rate must be set to

$$\boxed{\eta_\alpha = \Theta\Big(\frac{1}{\sqrt{d}}\Big) = \Theta\Big(\frac{1}{\sqrt{n_{\ell-1}}}\Big).}$$

**Gating weight $\mathbf{W}_\beta$.** Similarly, $\beta_t$ affects all future states through the recurrence, but under Assumption 5.1 we need only the direct contribution to $\mathbf{S}_t$. The gradient of $\beta_t$ combines contributions from both the write term and the erase term of equation 3.1:

$$\frac{\partial \mathcal{L}}{\partial \beta_t} = \sum_{a,b} \frac{\partial \mathcal{L}}{\partial (\mathbf{S}_t)_{ab}} \cdot \frac{\partial (\mathbf{S}_t)_{ab}}{\partial \beta_t} = \underbrace{\langle \mathbf{g}_t, \mathbf{v}_t\rangle \langle \mathbf{q}_t, \mathbf{k}_t\rangle}_{\text{Write Term}} - \underbrace{\alpha_t \langle \mathbf{g}_t, \mathbf{S}_{t-1}\mathbf{k}_t\rangle \langle \mathbf{q}_t, \mathbf{k}_t\rangle}_{\text{Erase Term}}.$$

Since $\langle \mathbf{g}_t, \boldsymbol{b}_t\rangle \langle \mathbf{q}_t, \mathbf{k}_t\rangle = \Theta(1) \cdot \Theta(1/\sqrt{d}) = \Theta(1/\sqrt{d})$, the write contribution is $\partial (\mathbf{S}_t)_{ab} / \partial \beta_t \big|_{\text{write}} = (\boldsymbol{b}_t)_a (\mathbf{k}_t)_b$. And since $\langle \mathbf{g}_t, \mathbf{S}_{t-1}\mathbf{k}_t\rangle \langle \mathbf{q}_t, \mathbf{k}_t\rangle = \Theta(1/\sqrt{d}) \cdot \Theta(1/\sqrt{d}) = \Theta(1/d)$, the erase term is $\partial (\mathbf{S}_t)_{ab} / \partial \beta_t \big|_{\text{erase}} = -\alpha_t \sum_c (\mathbf{S}_{t-1})_{ac} (\mathbf{k}_t)_c (\mathbf{k}_t)_b = -\alpha_t (\mathbf{S}_{t-1}\mathbf{k}_t)_a (\mathbf{k}_t)_b$. The write term therefore dominates and

$$\frac{\partial \mathcal{L}}{\partial \beta_t} = \Theta\Big(\frac{1}{\sqrt{d}}\Big).$$

For $z_{\beta,t} := \mathbf{W}_\beta \mathbf{x}_t$, since $\frac{\partial \beta_t}{\partial z_{\beta,t}} = \Theta(1)$, then the gradient of the pre-activation is

$$\frac{\partial \mathcal{L}}{\partial z_{\beta,t}} = \frac{\partial \beta_t}{\partial z_{\beta,t}} \cdot \frac{\partial \mathcal{L}}{\partial \beta_t} = \Theta(1) \cdot \Theta\Big(\frac{1}{\sqrt{d}}\Big) = \Theta\Big(\frac{1}{\sqrt{d}}\Big).$$

Finally, the gradient per entry of $\mathbf{W}_\beta$ is

$$\frac{\partial \mathcal{L}}{\partial (\mathbf{W}_\beta)_j} = \frac{\partial \mathcal{L}}{\partial z_{\beta,t}} \cdot (\mathbf{x}_t)_j = \Theta\Big(\frac{1}{\sqrt{d}}\Big) \cdot \Theta(1) = \Theta\Big(\frac{1}{\sqrt{d}}\Big),$$

and the same feature-learning analysis as $\mathbf{W}_\alpha$ gives

$$\boxed{\eta_\beta = \Theta\Big(\frac{1}{\sqrt{d}}\Big) = \Theta\Big(\frac{1}{\sqrt{n_{\ell-1}}}\Big).}$$

**Scalar parameters.** Both $a_{\log}$ and $b$ enter through the gating pre-activations as additive scalars. Using the exact chain rule on $\alpha_t = e^{g_t}$,

$$\frac{\partial \alpha_t}{\partial a_{\log}} = \frac{\partial e^{g_t}}{\partial a_{\log}} = e^{g_t} \frac{\partial g_t}{\partial a_{\log}} = \alpha_t g_t = \Theta(1),$$

so $\partial \mathcal{L}/\partial a_{\log} = (\partial \mathcal{L}/\partial \alpha_t) \cdot \Theta(1)$. From the argument in equation 5.1, $\partial \mathcal{L}/\partial \alpha_t = \Theta(1/\sqrt{d})$. Hence $\partial \mathcal{L}/\partial a_{\log} = \Theta(1/\sqrt{d})$, and an SGD step changes $a_{\log}$ by $\eta_{\text{scal}} \cdot \Theta(1/\sqrt{d})$. Since $a_{\log}$ enters $\alpha_t$ multiplicatively, a $\Theta(1)$ change in $\alpha_t$ via $a_{\log}$ requires $\eta_{\text{scal}} = \Theta(\sqrt{d})$.

### 5.4 Summary

According to the $\mu$P assumptions, the learning rate for other parameters can be set in the same way as Table 8 in Yang et al. (2022). In summary, the complete $\mu$P formulation for Gated Delta Networks with SGD is summarized in Table 1.

Table 1: $\mu$P formulation of Gated DeltaNet under SGD. Weights have shape $\mathbb{R}^{n_\ell \times n_{\ell-1}}$; for input weights and biases, $n_{\ell-1} = \Theta(1)$. All initialization variances and forward multipliers are identical to the AdamW table (Table 2), and only the learning-rate row differs. The gray factors are the differences between original scaling law in Yang et al. (2022) and our proposed formulation.

| | Input weights & all other biases | Output weights | Hidden weights (except $\mathbf{W}_\alpha, \mathbf{W}_\beta$) | $\mathbf{W}_\alpha, \mathbf{W}_\beta$ | $a_{\log}$ and $b$ |
|---|---|---|---|---|---|
| Initial variance | $\frac{1}{n_{\ell-1}}$ | 1 | $\frac{1}{n_{\ell-1}}$ | $\frac{1}{n_{\ell-1}}$ | 1 |
| Multiplier | 1 | $\frac{1}{n_{\ell-1}}$ | 1 | 1 | 1 |
| SGD LR | $n_\ell$ | $n_{\ell-1}$ | 1 | $\frac{1}{\sqrt{n_{\ell-1}}}$ (1) | $\sqrt{n_{\ell-1}}$ (1) |

## 6 Experiments

### 6.1 Experiment details

We implement LLM pre-training experiments to validate our $\mu$P derivation. All models use 8 layers and 6 attention heads. We test four model widths $d \in \{256, 512, 1024, 1536\}$ for AdamW (Loshchilov, 2017) and $d \in \{256, 512, 768, 1024\}$ for SGD optimizer, which correspond to parameter counts ranging from approximately 21M to 342M (non-embedding).

**Architectural parameters.** We refer to (Yang et al., 2025) and its official repository for the implementation of GDN and re-implement it on `nanoGPT` training framework (Karpathy, 2022). In detail, we set the head dimension of queries and keys to $d/8$ and that of values to $d/4$. And we set the kernel size of short convolutions in queries and keys to 4. Additionally, the intermediate size of MLP is set to $4d$, and we tied the input and output embeddings.

**Initialization and optimizer.** For the base model with $d_0 = 256$, the embedding layer and all input projections are initialized with standard deviation 0.02, which also applies to larger models for SP. In contrast, we initialize large models under original $\mu$P and our proposed $\mu$P according to Tables 2 and 1. The scalar gating parameters $a_{\log}$ and $b$ follow the scheme of Yang et al. (2025): $a_{\log} \sim \text{Uniform}(0, 16)$ and $b$ is set as described in Section 4. We apply a gradient clipping to 1.0 and set Dropout ratio (Srivastava et al., 2014) to 0.0. The minimum learning rate is fixed to 5e-5 throughout. All runs use a cosine learning-rate schedule with 2,000 warmup steps.

For AdamW experiments, we use a weight decay of 0.1 and set $(\beta_1, \beta_2) = (0.9, 0.95)$. And for SGD experiments, we use SGD with Nesterov momentum (Nesterov, 1983) with a momentum of 0.98, since we notice there is great instability when using original SGD optimizer. For both optimizers, we use the same learning rates for all the modules in models with $d_0 = 256$ and all models under SP, and applies different learning rates according to the scaling laws in Tables 2 and 1 in $\mu$P experiments.

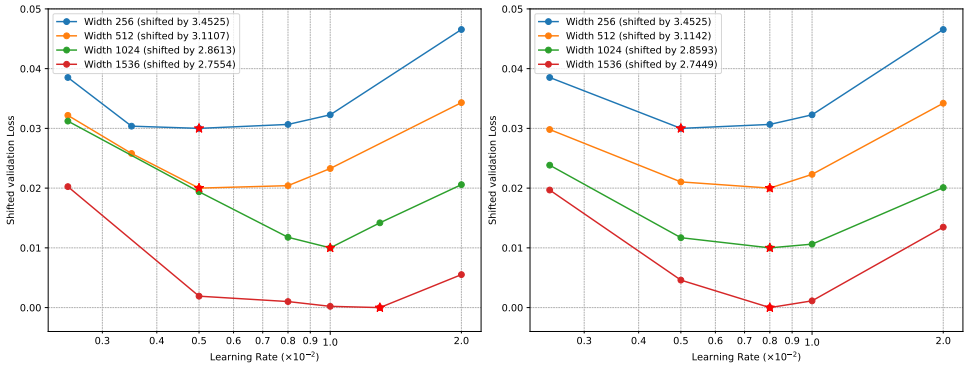
We train models with each width at 5-7 different learning rates log-spaced with increased density near optimal learning rates. The learning rate search grid ranges between 1e-3 and 2e-2 for AdamW and between 1e-1 and 1 for SGD experiments. And we fix the training seed to 42.

**Data and compute.** We train on the FineWeb-Edu 100B dataset (Lozhkov et al., 2024) for 20k steps with a global batch size of 480 sequences and a sequence length of 1024 (approximately 9.83B tokens in total). Moreover, we use 1 NVIDIA H100 80GB HBM3 GPU for all the experiments.

### 6.2 Experiment results

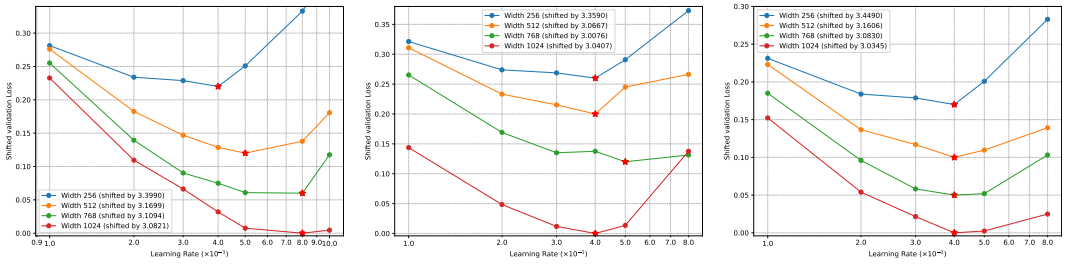
The final validation losses for models with different widths and peak learning rates under the $\mu$P and SP configurations are shown in Figures 1 and 2 for AdamW and SGD, respectively. To remove the trivial width-dependence of the absolute loss, we report *shifted* validation loss, defined as the difference from the optimal loss value among all the experiments on the models with the same width but different learning rates.

For AdamW, the optimal learning rate is consistently the same across all 4 model widths under $\mu$P, demonstrating zero-shot learning-rate transfer. While under SP, the optimal learning rate shifts substantially with width, confirming that SP fails to preserve LR transfer at scale. SGD experiments show the same qualitative pattern. Under SP, the optimal learning rate does not transfer across widths and under original $\mu$P configuration, it varies a lot. And in our $\mu$P configuration, the optimal learning rate transfers across model widths. These results validate that our theory works well in practice.




(a) Standard Parametrization (SP) (b) $\mu$P configuration

Figure 1: Shifted validation loss for Gated DeltaNet trained with AdamW under varying peak learning rates and model widths.




(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 2: Shifted validation loss for Gated DeltaNet trained with SGD under varying peak learning rates and model widths.

## 7 Conclusion

We have derived the $\mu$P-style parametrization for Gated Delta Networks. Our analysis reveals that under SGD, scalings of the gating weight matrices and the scalar gating parameters are different from the standard $\mu$P law. LLM pre-training experiments confirm that our $\mu$P formulation achieves zero-shot learning-rate transfer under both AdamW and SGD, while standard parametrization fails to transfer, empirically providing empirical support of our theoretical derivation. More broadly, our analysis provides a template for deriving width-dependent parameterizations for other linear recurrent and hybrid architectures with structured state transitions.

### AI USE STATEMENT

In this work, we have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Charlie Blake, Constantin Eichenberg, Josef Dean, Lukas Balles, Luke Yuri Prince, Björn Deiseroth, Andres Felipe Cruz-Salinas, Carlo Luschi, Samuel Weinbach, and Douglas Orr. u-$\mu$P: The Unit-Scaled Maximal Update Parametrization. In *The Thirteenth International Conference on Learning Representations*, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning*, pp. 10041–10071. PMLR, 2024.

Nolan Dey, Shane Bergsma, and Joel Hestness. Sparse maximal update parameterization: A holistic approach to sparse training dynamics. *Advances in Neural Information Processing Systems*, 37: 33836–33862, 2024.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Katie E Everett, Lechao Xiao, Mitchell Wortsman, Alexander A Alemi, Roman Novak, Peter J Liu, Izzeddin Gur, Jascha Sohl-Dickstein, Leslie Pack Kaelbling, Jaehoon Lee, et al. Scaling exponents across parameterizations and optimizers. In *International Conference on Machine Learning*, pp. 12666–12700. PMLR, 2024.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*, 2024.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33: 1474–1487, 2020.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

Moritz Haas, Jin Xu, Volkan Cevher, and Leena Chennuru Vankadara. Effective sharpness aware minimization requires layerwise perturbation scaling. In *High-dimensional Learning Dynamics 2024: The Emergence of Structure and Reasoning*, 2024.

Karl Hajjar, Lénaïc Chizat, and Christophe Giraud. Training integrable parameterizations of deep neural networks in the infinite-width limit. *Journal of Machine Learning Research*, 25(196):1–130, 2024.

Ali Hatamizadeh, Yejin Choi, and Jan Kautz. Gated deltanet-2: Decoupling erase and write in linear attention. *arXiv preprint arXiv:2605.22791*, 2026.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Samuel Horváth, Aaron Klein, Peter Richtárik, and Cédric Archambeau. Hyperparameter transfer learning with adaptive complexity. In *International conference on artificial intelligence and statistics*, pp. 1378–1386. PMLR, 2021.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. Going beyond linear transformers with recurrent fast weight programmers. In *Advances in Neural Information Processing Systems*, volume 34, pp. 7703–7717, 2021.

Satoki Ishikawa and Ryo Karakida. On the parameterization of second-order optimization effective towards the infinite width. In *The Twelfth International Conference on Learning Representations*, 2024.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*, pp. 240–248. PMLR, 2016.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Andrej Karpathy. NanoGPT. https://github.com/karpathy/nanoGPT, 2022.

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.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, et al. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. Fineweb-edu: the finest collection of educational content, 2024. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu.

Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

Meta AI. Llama 4: Advancing multimodal intelligence. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, apr 2025. URL https://ai.meta.com/blog/llama-4-multimodal-intelligence/. Accessed: 2025-04-10.

Yurii Nesterov. A method for solving the convex programming problem with convergence rate o (1/k2). In *Dokl akad nauk Sssr*, volume 269, pp. 543, 1983.

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.

Valerio Perrone, Rodolphe Jenatton, Matthias W Seeger, and Cédric Archambeau. Scalable hyperparameter transfer learning. *Advances in neural information processing systems*, 31, 2018.

Zhen Qin, Songlin Yang, and Yiran Zhong. Hierarchically gated recurrent neural network for sequence modeling. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. Hgrn2: Gated linear rnns with state expansion. In *First Conference on Language Modeling*, 2024.

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

Grant Rotskoff and Eric Vanden-Eijnden. Trainability and accuracy of artificial neural networks: An interacting particle system approach. *Communications on Pure and Applied Mathematics*, 75(9): 1889–1935, 2022.

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, Riccardo Grazzi, Korbinian Pöppel, Jaisidh Singh, Arber Zela, Timur Carstensen, Jenia Jitsev, Frank Hutter, Volkan Cevher, Antonio Orvieto, and Aaron Klein. Complex kda: Understanding and enhancing the expressivity of kimi delta attention. *arXiv preprint arXiv:2609.24797*, 2026.

Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752, 2020.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pp. 2171–2180. PMLR, 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

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

Meituan LongCat Team, Bei Li, Bingye Lei, Bo Wang, Bolin Rong, Chao Wang, Chao Zhang, Chen Gao, Chen Zhang, Cheng Sun, et al. Longcat-flash technical report. *arXiv preprint arXiv:2509.01322*, 2025.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Leena Chennuru Vankadara, Jin Xu, Moritz Haas, and Volkan Cevher. On feature learning in structured state space models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

B Widrow and ME Hoff. Adaptive switching circuits, wescon conv. *Rec., pt*, 4, 1960.

Greg Yang and Edward J. Hu. Tensor programs IV: Feature learning in infinite-width neural networks. 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. 11727–11737. PMLR, 18–24 Jul 2021.

Greg Yang and Etai Littwin. Tensor programs ivb: Adaptive optimization in the infinite-width limit. *arXiv preprint arXiv:2308.01814*, 2023.

Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.

Greg Yang, James B Simon, and Jeremy Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2023.

Greg Yang, Dingli Yu, Chen Zhu, and Soufiane Hayou. Tensor programs VI: feature learning in infinite depth neural networks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024a.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *International Conference on Machine Learning*. PMLR, 2024b.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems*, volume 37, 2024c.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*, 2025.

Dani Yogatama and Gideon Mann. Efficient transfer learning method for automatic hyperparameter tuning. In *Artificial intelligence and statistics*, pp. 1077–1085. PMLR, 2014.

# A Additional derivations in the backward process for SGD

Due to page limitations, we postpone the derivations of the gradients for query, key and value projections, as well as the detailed analysis of BPTT term in this section.

## A.1 Gradient of query, key and value projections

**Gradient of the value projection $\mathbf{W}_v$.** Recall from equation 3.1 and equation 3.2 that $\mathbf{S}_t$ depends on $\boldsymbol{b}_t$ through the rank-one write update $\beta_t \boldsymbol{b}_t \mathbf{k}_t^\top$. Under Assumption 5.1, the gradient of $\mathcal{L}$ with respect to $(\boldsymbol{b}_t)_i$ receives a contribution from the direct readout at time $t$:

$$\frac{\partial \mathcal{L}}{\partial (\boldsymbol{b}_t)_i} = \beta_t \sum_j \frac{\partial \mathcal{L}}{\partial (\mathbf{S}_t)_{ij}} \cdot (\mathbf{k}_t)_j = \beta_t \sum_j (\mathbf{g}_t)_i (\mathbf{q}_t)_j \cdot (\mathbf{k}_t)_j = \beta_t \, (\mathbf{g}_t)_i \, \langle \mathbf{q}_t, \mathbf{k}_t \rangle, \tag{A.1}$$

where we used $\partial \mathcal{L} / \partial (\mathbf{S}_t)_{ij} = (\mathbf{g}_t)_i (\mathbf{q}_t)_j$ from the readout equation 3.2 under the short-memory assumption (Assumption 5.1). Note also that $\boldsymbol{b}_t$ does not propagate gradient back to earlier states $\mathbf{S}_{t'}$

with $t' < t$; those states depend on $\boldsymbol{b}_{t'}$, not $\boldsymbol{b}_t$. However, $\mathbf{S}_t$ itself contributes to future states $\mathbf{S}_{t''}$ for $t'' > t$, which is precisely the BPTT tail handled by Assumption 5.1.

Since $\mathbf{q}_t$ and $\mathbf{k}_t$ are L2-normalized to unit vectors in $\mathbb{R}^d$, with independent entries each of order $\Theta(1/\sqrt{d})$ at initialization, the inner product satisfies $\big|\langle \mathbf{q}_t, \mathbf{k}_t \rangle\big| = \Theta\big(\frac{1}{\sqrt{d}}\big)$. In equation A.1 the three factors are: $\beta_t = \Theta(1)$, $(\mathbf{g}_t)_i = \Theta(1/\sqrt{d})$ per coordinate, and $\langle \mathbf{q}_t, \mathbf{k}_t \rangle = \Theta(1/\sqrt{d})$. Hence:

$$\frac{\partial \mathcal{L}}{\partial (\boldsymbol{b}_t)_i} = \Theta(1) \cdot \Theta\Big(\frac{1}{\sqrt{d}}\Big) \cdot \Theta\Big(\frac{1}{\sqrt{d}}\Big) = \Theta\Big(\frac{1}{d}\Big).$$

By treating ShortConv as the identity for scale estimates, the gradient of the value projection $\mathbf{W}_v \in \mathbb{R}^{d \times d}$ is

$$\frac{\partial \mathcal{L}}{\partial (\mathbf{W}_v)_{ij}} = \frac{\partial \mathcal{L}}{\partial (\boldsymbol{b}_t)_i} \cdot (\mathbf{x}_t)_j = \Theta\Big(\frac{1}{d}\Big) \cdot \Theta(1) = \Theta\Big(\frac{1}{d}\Big).$$

The SGD update is $\Delta(\mathbf{W}_v)_{ij} = -\eta_v \Theta(1/d)$. The resulting change in $\boldsymbol{b}_t$ is

$$(\Delta \boldsymbol{b}_t)_i = \sum_{j=1}^{d} \Delta(\mathbf{W}_v)_{ij} (\mathbf{x}_t)_j = d \cdot \Theta\Big(\frac{\eta_v}{d}\Big) = \Theta(\eta_v).$$

The feature-learning condition requires $\Delta \boldsymbol{b}_t$ with $\Theta(1)$ coordinate size, therefore,

$$\boxed{\eta_v = \Theta(1).}$$

**Gradient of the key projection $\mathbf{W}_k$.** The key $\mathbf{k}_t = \widetilde{\mathbf{k}}_t / \|\widetilde{\mathbf{k}}_t\|_2$ enters the latent-state update both through the write term ($\beta_t \boldsymbol{b}_t \mathbf{k}_t^\top$) and the erase term ($\alpha_t \mathbf{S}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top)$). The gradient of $\mathcal{L}$ with respect to the normalized $\mathbf{k}_t$ from the write term alone is

$$\Big(\frac{\partial \mathcal{L}}{\partial \mathbf{k}_t}\Big)_l \Big|_{\text{write}} = \beta_t \sum_i \frac{\partial \mathcal{L}}{\partial (\mathbf{S}_t)_{il}} \cdot (\boldsymbol{b}_t)_i = \beta_t \, (\mathbf{q}_t)_l \sum_i (\mathbf{g}_t)_i (\boldsymbol{b}_t)_i = \beta_t \, (\mathbf{q}_t)_l \, \langle \mathbf{g}_t, \boldsymbol{b}_t \rangle. \tag{A.2}$$

Here $\boldsymbol{b}_t$ has $\Theta(1)$ coordinate size while $\mathbf{g}_t$ has $\Theta(1/\sqrt{d})$ coordinate size, so $|\langle \mathbf{g}_t, \boldsymbol{b}_t \rangle| = \Theta(1)$. Since $\mathbf{q}_t$ and $\mathbf{k}_t$ have $\Theta(1/\sqrt{d})$ coordinate size, and $\mathbf{g}_t$ has $\Theta(1/\sqrt{d})$ coordinate size, according to equation A.2, we have

$$\Big(\frac{\partial \mathcal{L}}{\partial \mathbf{k}_t}\Big)_l \Big|_{\text{write}} = \Theta(1) \cdot \Theta\Big(\frac{1}{\sqrt{d}}\Big) \cdot \Theta(1) = \Theta\Big(\frac{1}{\sqrt{d}}\Big).$$

We now compute the erase-term contribution. The erase term is $\mathbf{E}_t = \alpha_t \mathbf{S}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top)$. Differentiating $(\mathbf{S}_t)_{al} = (\mathbf{E}_t)_{al} + \beta_t (\boldsymbol{b}_t)_a (\mathbf{k}_t)_l$ with respect to $(\mathbf{k}_t)_l$ gives two sub-terms:

(1) column of $\mathbf{S}_{t-1}$ contracted with gradient:

$$\Big(\frac{\partial \mathcal{L}}{\partial \mathbf{k}_t}\Big)_l \Big|_{\text{erase,1}} = -\alpha_t \beta_t \sum_a \frac{\partial \mathcal{L}}{\partial (\mathbf{S}_t)_{al}} \sum_j (\mathbf{S}_{t-1})_{aj} (\mathbf{k}_t)_j = -\alpha_t \beta_t (\mathbf{q}_t)_l \, \langle \mathbf{g}_t, \mathbf{S}_{t-1} \mathbf{k}_t \rangle.$$

Since $(\mathbf{S}_{t-1}\mathbf{k}_t)_i$ has variance $d \cdot \Theta(1/d) \cdot \Theta(1/d) = \Theta(1/d)$, i.e. $\Theta(1/\sqrt{d})$ per coordinate, the inner product $\langle \mathbf{g}_t, \mathbf{S}_{t-1}\mathbf{k}_t \rangle$ has variance $d \cdot \Theta(1/d) \cdot \Theta(1/d) = \Theta(1/d)$, so it is $\Theta(1/\sqrt{d})$. Combined with $(\mathbf{q}_t)_l = \Theta(1/\sqrt{d})$, this sub-term is $\Theta(1/d)$.

(2) diagonal of $\mathbf{S}_{t-1}$ contracted with gradient:

The second sub-term arises from the product rule applied to $(\mathbf{S}_{t-1}\mathbf{k}_t)_a = \sum_m (\mathbf{S}_{t-1})_{am} (\mathbf{k}_t)_m$; differentiating the factor $(\mathbf{k}_t)_l$ inside the sum gives

$$\Big(\frac{\partial \mathcal{L}}{\partial \mathbf{k}_t}\Big)_l \Big|_{\text{erase,2}} = -\alpha_t \beta_t \sum_{a,c} \frac{\partial \mathcal{L}}{\partial (\mathbf{S}_t)_{ac}} \cdot (\mathbf{S}_{t-1})_{al} \cdot (\mathbf{k}_t)_c = -\alpha_t \beta_t \langle \mathbf{q}_t, \mathbf{k}_t \rangle \, (\mathbf{S}_{t-1}^\top \mathbf{g}_t)_l.$$

Since $(\mathbf{S}_{t-1}^\top \mathbf{g}_t)_l = \sum_a (\mathbf{S}_{t-1})_{al} (\mathbf{g}_t)_a$ has $\Theta(1/\sqrt{d})$ coordinate size, and $\langle \mathbf{q}_t, \mathbf{k}_t \rangle = \Theta(1/\sqrt{d})$, this sub-term is $\Theta(1/d)$ per coordinate.

Both erase sub-terms are at most $\Theta(1/d)$, smaller than the write term $\Theta(1/\sqrt{d})$. The dominant contribution therefore comes from the write term, therefore, $\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{k}_t}\right\|_2 = \Theta(1)$, i.e., with $\Theta\left(\frac{1}{\sqrt{d}}\right)$ coordinate size. Moreover, $\mathbf{k}_t$ affects $\mathbf{S}_t$ and, through the recurrent BPTT chain, all future states $\mathbf{S}_{t''}$ for $t'' > t$. Each such future contribution has the same $d$-order as the direct term above, and Assumption 5.1 ensures their sum is $O(L_{\text{eff}})$ times the direct term, which is an $O(1)$ factor under the short-memory assumption.

We now focus on the L2-normalization $\mathbf{k}_t = \widetilde{\mathbf{k}}_t/\|\widetilde{\mathbf{k}}_t\|_2$. The Jacobian of L2-normalization is $\mathbf{J}_k = (\mathbf{I} - \mathbf{k}_t\mathbf{k}_t^\top)/\|\widetilde{\mathbf{k}}_t\|_2$, which projects onto the hyperplane orthogonal to $\mathbf{k}_t$ and scales by $1/\|\widetilde{\mathbf{k}}_t\|_2 = \Theta(1/\sqrt{d})$. Since $\partial\mathcal{L}/\partial\mathbf{k}_t$ and $\mathbf{k}_t$ are approximately independent at initialization, the projection loses negligible magnitude:

$$\left\|\frac{\partial \mathcal{L}}{\partial \widetilde{\mathbf{k}}_t}\right\|_2 = \Theta\Big(\frac{1}{\sqrt{d}}\Big) \cdot \Theta(1) = \Theta\Big(\frac{1}{\sqrt{d}}\Big),$$

so each coordinate of $\partial\mathcal{L}/\partial\widetilde{\mathbf{k}}_t$ is $\Theta(1/d)$. The gradient of $\mathbf{W}_k$ is therefore

$$\frac{\partial \mathcal{L}}{\partial (\mathbf{W}_k)_{ij}} = \frac{\partial \mathcal{L}}{\partial (\widetilde{\mathbf{k}}_t)_i} \cdot (\mathbf{x}_t)_j = \Theta\Big(\frac{1}{d}\Big) \cdot \Theta(1) = \Theta\Big(\frac{1}{d}\Big),$$

and an identical feature-learning analysis as for $\mathbf{W}_v$ gives

$$\boxed{\eta_k = \Theta(1).}$$

**Gradient of the query projection $\mathbf{W}_q$.** The gradient of $\mathcal{L}$ with respect to the normalized $\mathbf{q}_t$ is

$$\frac{\partial \mathcal{L}}{\partial (\mathbf{q}_t)_j} = (\mathbf{S}_t^\top \mathbf{g}_t)_j = \sum_i (\mathbf{S}_t)_{ij} (\mathbf{g}_t)_i.$$

With $(\mathbf{S}_t)_{ij} = \Theta(1/\sqrt{d})$ and $(\mathbf{g}_t)_i = \Theta(1/\sqrt{d})$, each entry of $\mathbf{S}_t$ is approximately independent and zero-mean with variance $\Theta(1/d)$. Thus, the product $(\mathbf{S}_t)_{ij}(g_t)_i$ has variance $\Theta(1/d^2)$. Summing over $d$ terms gives $\mathrm{Var}[(\mathbf{S}_t^\top \mathbf{g}_t)_j] = d \cdot \Theta(1/d) \cdot \Theta(1/d) = \Theta(1/d)$, i.e. $\partial\mathcal{L}/\partial\mathbf{q}_t = \Theta(1/\sqrt{d})$ per coordinate. The L2-normalization Jacobian for $\mathbf{q}_t$ has the same spectral norm $\Theta(1/\sqrt{d})$ as for $\mathbf{k}_t$, therefore:

$$\frac{\partial \mathcal{L}}{\partial (\widetilde{\mathbf{q}}_t)_i} = \Theta\Big(\frac{1}{d}\Big), \qquad \frac{\partial \mathcal{L}}{\partial (\mathbf{W}_q)_{ij}} = \Theta\Big(\frac{1}{d}\Big).$$

The feature-learning condition yields

$$\boxed{\eta_q = \Theta(1),}$$

confirming that all three $d \times d$ projection matrices share the same SGD learning rate scaling $\Theta(1)$.

### A.2 Derivation of the cumulative latent space

In Section 4.2, we uses the argument of Vankadara et al. (2024) to direct achieve that $\mathbf{S}_t$ has $\Theta(1/\sqrt{d})$ coordinate size. Here we show the detailed derivations of this conclusion. The state update rule is:

$$\mathbf{S}_t = \alpha_t \mathbf{S}_{t-1} - \alpha_t \beta_t \mathbf{S}_{t-1} \mathbf{k}_t \mathbf{k}_t^\top + \mathbf{U}_t,$$

where the write matrix is $\mathbf{U}_t = \beta_t \mathbf{v}_t \mathbf{k}_t^\top$. From the $\mu$P initialization, we know that $(\mathbf{v}_t)_i = \Theta(1)$, $(\mathbf{k}_t)_j = \Theta(1/\sqrt{d})$ and $\alpha_t, \beta_t \in (0, 1)$ are bounded scalars, which can be treated as $\Theta(1)$ independent variables. And both $(\mathbf{v}_t)_i$ and $(\mathbf{k}_t)_j$ are with zero mean values. Therefore,

$$\mathbb{E}[(\mathbf{U}_t)_{ij}^2] = \mathbb{E}[\beta_t^2 (\mathbf{v}_t)_i^2 (\mathbf{k}_t)_j^2] = \Theta(1) \cdot \Theta(1) \cdot \Theta\left(\frac{1}{d}\right) = \Theta\left(\frac{1}{d}\right).$$

Therefore, $\mathbf{U}_t$ has a coordinate size of $\Theta(1/\sqrt{d})$.

Let $V_t = \mathbb{E}[(\mathbf{S}_t)_{ij}^2]$ be the element-wise variance of the state matrix. As a standard mean-field assumption at initialization, we assume $\mathbf{S}_{t-1}$ is statistically independent of the current inputs $\mathbf{v}_t$ and $\mathbf{k}_t$. Then,

$$(\mathbf{S}_t)_{ij} = \alpha_t (\mathbf{S}_{t-1})_{ij} - \alpha_t \beta_t \sum_l (\mathbf{S}_{t-1})_{il} (\mathbf{k}_t)_l (\mathbf{k}_t)_j + (\mathbf{U}_t)_{ij}.$$

Since $\mathbf{U}_t$ depends on $\mathbf{v}_t$, which has zero mean and is independent of $\mathbf{S}_{t-1}$ and $\mathbf{k}_t$, the cross-terms between $\mathbf{U}_t$ and the other terms vanish when we take the expected square. Therefore,

$$V_t = \mathbb{E}\left[\left(\alpha_t(\mathbf{S}_{t-1})_{ij} - \alpha_t\beta_t\sum_l(\mathbf{S}_{t-1})_{il}(\mathbf{k}_t)_l(\mathbf{k}_t)_j\right)^2\right] + \mathbb{E}[(\mathbf{U}_t)_{ij}^2].$$

For the squared expectation term, we have

$$\mathbb{E}[\alpha_t^2(\mathbf{S}_{t-1})_{ij}^2] = \mathbb{E}[\alpha_t^2]V_{t-1},$$

and

$$-2\mathbb{E}\left[\alpha_t^2\beta_t(\mathbf{S}_{t-1})_{ij}\sum_l(\mathbf{S}_{t-1})_{il}(\mathbf{k}_t)_l(\mathbf{k}_t)_j\right] = -2\mathbb{E}[\alpha_t^2\beta_t]\mathbb{E}[(\mathbf{S}_{t-1})_{ij}^2]\frac{1}{d} = -2\mathbb{E}[\alpha_t^2\beta_t]V_{t-1}\frac{1}{d},$$

where the second equality holds since $\mathbf{k}_t$ has independent zero-mean entries, and then the expectation over $\mathbf{k}_t$ is zero unless $l = j$. And for the remaining term, we have

$$\begin{aligned}
\mathbb{E}\left[\alpha_t^2\beta_t^2\left(\sum_l(\mathbf{S}_{t-1})_{il}(\mathbf{k}_t)_l\right)^2(\mathbf{k}_t)_j^2\right] &= \mathbb{E}\left[\alpha_t^2\beta_t^2\big(\sum_{l,m}(\mathbf{S}_{t-1})_{il}(\mathbf{S}_{t-1})_{im}(\mathbf{k}_t)_l(\mathbf{k}_t)_m\big)(\mathbf{k}_t)_j^2\right] \\
&= \mathbb{E}[\alpha_t^2\beta_t^2(\sum_l(\mathbf{S}_{t-1})_{il}^2(\mathbf{k}_t)_l^2)(\mathbf{k}_t)_j^2] \\
&= \mathbb{E}[\alpha_t^2\beta_t^2]\left(d\cdot V_{t-1}\cdot\frac{1}{d^2}\right) \\
&= \mathbb{E}[\alpha_t^2\beta_t^2]V_{t-1}\frac{1}{d},
\end{aligned}$$

where the second equality holds by the independence within the entries in $\mathbf{k}_t$. Finally we have

$$\begin{aligned}
V_t &= \mathbb{E}[\alpha_t^2]V_{t-1} - \frac{2\mathbb{E}[\alpha_t^2\beta_t] - \mathbb{E}[\alpha_t^2\beta_t^2]}{d}V_{t-1} + \Theta\left(\frac{1}{d}\right) \\
&= V_{t-1}\cdot\underbrace{\mathbb{E}\left[\alpha_t^2\left(1 - \frac{2\beta_t - \beta_t^2}{d}\right)\right]}_{\gamma} + \Theta\left(\frac{1}{d}\right).
\end{aligned}$$

Assume $\mathbb{E}[\alpha_t^2] = c^2 < 1$. Because $d$ is relatively large, the $(2\beta_t - \beta_t^2)/d$ term is small. Therefore, the contraction factor $\gamma$ is strictly dictated by $\alpha_t$, and we have $\gamma \approx \mathbb{E}[\alpha_t^2] < 1$. Then the variance $V_t$ is a simple converging geometric progression:

$$V_t = \gamma V_{t-1} + \Theta\left(\frac{1}{d}\right).$$

As $t \to \infty$, it converges to the infinite sum:

$$V_\infty = \frac{\Theta(1/d)}{1-\gamma},$$

which is $\Theta(1/d)$ since $\gamma$ is $\Theta(1)$ and strictly less than 1. Therefore, the element-wise variance of the latent state $\mathbf{S}_t$ is $\Theta(1/d)$, and the coordinate size is $\Theta(1/\sqrt{d})$.

## B Compatibility with AdamW

Under the AdamW (Loshchilov, 2017) optimizer, the parameter update at step $t$ is proportional to the coordinate-wise exponential moving average of the gradient, so the effective coordinate-wise update magnitude is $\Theta(1)$ regardless of the gradient scale. Hence the $\mu$P learning-rate scaling is similar to that for standard architectures without modification. The complete $\mu$P formulation for Gated DeltaNet trained with AdamW is summarized in Table 2. To rigorously validate the intuition, we will derive the scaling law of GDN when optimized by Adam(W) in the following.

Table 2: $\mu$P formulation of Gated DeltaNet under AdamW optimization. Weights have shape $\mathbb{R}^{n_\ell \times n_{\ell-1}}$; for input weights and biases, $n_{\ell-1} = \Theta(1)$. Under AdamW, Adam's coordinate-wise gradient normalization equalizes the effective update magnitude across all weight classes, so the gating weight matrices $\mathbf{W}_\alpha$ and $\mathbf{W}_\beta$ are subsumed into the "Hidden weights" column and require no special treatment.

| | Input weights & all biases | Output weights | Hidden weights | $a_{\log}$ and $b$ |
|---|---|---|---|---|
| Initial variance | $\frac{1}{n_{\ell-1}}$ | 1 | $\frac{1}{n_{\ell-1}}$ | 1 |
| Multiplier | 1 | $\frac{1}{n_{\ell-1}}$ | 1 | 1 |
| Learning rate | 1 | 1 | $\frac{1}{n_{\ell-1}}$ | 1 |

### B.1 Adam(W) in the Scale-Invariant Regime

Consider the pre-activation of the gating mechanism, $z_{\alpha,t} = \mathbf{W}_\alpha \mathbf{x}_t + b$. From our first-order analysis, the gradient of the loss with respect to this pre-activation is $\delta_{\alpha,t} := \frac{\partial \mathcal{L}}{\partial z_{\alpha,t}} = \Theta(1/\sqrt{d})$.

For any parameter $\theta$ with gradient $g = \partial \mathcal{L}/\partial \theta$, the Adam(W) update rule normalizes the gradient by its root-mean-square $v$. Since the gradients in our model are order $\Theta(1/\sqrt{d})$, which strictly dominates the standard Adam(W) $\epsilon$ parameter (e.g., $10^{-8}$) for practically sized $d$, Adam(W) operates in a scale-invariant regime. The update effectively reduces to a coordinate-wise sign descent:

$$\Delta\theta_j = -\eta_\theta \frac{g_j}{\sqrt{v_j} + \epsilon} \approx -\eta_\theta \text{sign}(g_j), \tag{B.1}$$

where $\eta_\theta$ is the learning rate assigned to parameter $\theta$.

### B.2 Derivation for Main Projection Weights ($\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v, \mathbf{W}_o$)

Let $\mathbf{W} \in \mathbb{R}^{d \times d}$ denote any of the main linear projection matrices, yielding a pre-activation $\mathbf{z} = \mathbf{W}\mathbf{x}_t \in \mathbb{R}^d$. Let $\delta_i = \frac{\partial \mathcal{L}}{\partial z_i}$ be the gradient arriving at the $i$-th coordinate of the output. The gradient with respect to the weight coordinate $W_{ij}$ is:

$$g_{ij} = \frac{\partial \mathcal{L}}{\partial W_{ij}} = \delta_i (\mathbf{x}_t)_j.$$

Applying the Adam update rule equation B.1, the parameter change is:

$$\Delta W_{ij} \approx -\eta_W \text{sign}\big(\delta_i (\mathbf{x}_t)_j\big) = -\eta_W \text{sign}(\delta_i) \text{sign}\big((\mathbf{x}_t)_j\big).$$

We now compute the exact shift $\Delta z_i$ in the $i$-th coordinate of the feature caused by this update:

$$\begin{aligned} \Delta z_i &= \sum_{j=1}^{d} \Delta W_{ij} (\mathbf{x}_t)_j \\ &= \sum_{j=1}^{d} \big(-\eta_W \text{sign}(\delta_i) \text{sign}\big((\mathbf{x}_t)_j\big)\big) (\mathbf{x}_t)_j \\ &= -\eta_W \text{sign}(\delta_i) \sum_{j=1}^{d} \big|(\mathbf{x}_t)_j\big|. \end{aligned} \tag{B.2}$$

Under the $\mu$P initialization, each coordinate of the hidden state $\mathbf{x}_t$ possesses a magnitude of $\Theta(1)$. Consequently, the sum of absolute values in equation B.2 (i.e., the $\ell_1$ norm of $\mathbf{x}_t$) strictly scales as $\Theta(d)$. Thus, the magnitude of the feature shift is:

$$|\Delta z_i| = \eta_W \cdot \Theta(d).$$

To enforce the feature-learning requirement $|\Delta z_i| = \Theta(1)$, the learning rate for the main projection matrices must be scaled as:

$$\eta_W = \Theta\left(\frac{1}{d}\right).$$

### B.3 Derivation for Gating Weights ($\mathbf{W}_\alpha, \mathbf{W}_\beta$)

For the gating weight matrix $\mathbf{W}_\alpha \in \mathbb{R}^{1\times d}$ (the derivation for $\mathbf{W}_\beta$ is identical), the gradient with respect to its $j$-th coordinate is:

$$g_{\alpha,j} = \frac{\partial \mathcal{L}}{\partial (\mathbf{W}_\alpha)_j} = \delta_{\alpha,t}(\mathbf{x}_t)_j.$$

Substituting this into the Adam update rule equation B.1, we obtain the exact parameter update:

$$\Delta(\mathbf{W}_\alpha)_j = -\eta_\alpha \text{sign}\big(\delta_{\alpha,t}(\mathbf{x}_t)_j\big) = -\eta_\alpha \text{sign}(\delta_{\alpha,t})\text{sign}\big((\mathbf{x}_t)_j\big).$$

To satisfy the fundamental $\mu$P feature-learning condition, a single optimization step must induce a $\Theta(1)$ shift in the pre-activation $z_{\alpha,t}$. We compute this exact shift $\Delta z_{\alpha,t}$ caused by $\Delta\mathbf{W}_\alpha$:

$$\begin{aligned}
\Delta z_{\alpha,t} &= \sum_{j=1}^{d} \Delta(\mathbf{W}_\alpha)_j (\mathbf{x}_t)_j \\
&= \sum_{j=1}^{d} \left(-\eta_\alpha \text{sign}(\delta_{\alpha,t})\text{sign}\big((\mathbf{x}_t)_j\big)\right)(\mathbf{x}_t)_j \\
&= -\eta_\alpha \text{sign}(\delta_{\alpha,t}) \sum_{j=1}^{d} \big|(\mathbf{x}_t)_j\big|. \qquad \text{(B.3)}
\end{aligned}$$

Under the $\mu$P assumptions, each coordinate of the hidden state $\mathbf{x}_t$ possesses a magnitude of $\Theta(1)$. Consequently, the sum of absolute values in equation B.3 (i.e., the $\ell_1$ norm of $\mathbf{x}_t$) strictly scales as $\Theta(d)$. Thus, the magnitude of the feature shift is:

$$|\Delta z_{\alpha,t}| = \eta_\alpha \cdot \Theta(d).$$

To enforce the feature-learning requirement $|\Delta z_{\alpha,t}| = \Theta(1)$, the learning rate must be precisely scaled as:

$$\eta_\alpha = \Theta\left(\frac{1}{d}\right).$$

This demonstrates a core property of AdamW under $\mu$P: regardless of the output dimension of the projection, the learning rate scaling is strictly dictated by the fan-in dimension ($d$). Therefore, both matrix types can be summarized as one category with a $1/n_{\ell-1}$ learning rate multiplier, aligning with Table 2.

### B.4 Derivation for Scalar Parameters ($a_{\log}$ and $b$)

We now analyze the scalar bias $b$ (and equivalently, the additive scalar $a_{\log}$). The gradient with respect to $b$ is trivially the pre-activation gradient:

$$g_b = \frac{\partial \mathcal{L}}{\partial b} = \delta_{\alpha,t} = \Theta\left(\frac{1}{\sqrt{d}}\right).$$

Applying the scale-invariant Adam update, the parameter change is:

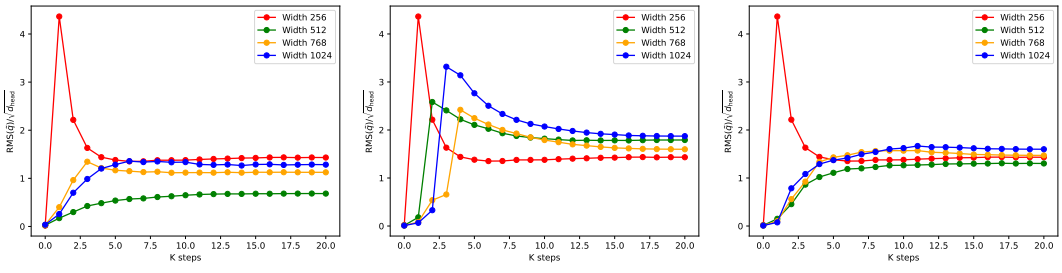
$$\Delta b \approx -\eta_b \text{sign}(\delta_{\alpha,t}).$$

Because $b$ acts as a direct additive scalar to the pre-activation ($z_{\text{new}} = \mathbf{W}_\alpha \mathbf{x}_t + b_{\text{new}}$), the resulting shift in the feature is exactly the parameter update itself:

$$|\Delta z_{\alpha,t,(\text{from bias})}| = |\Delta b| = \eta_b \cdot \Theta(1).$$

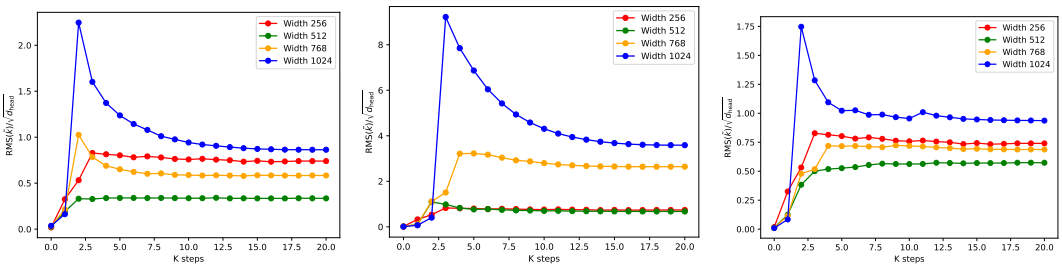
To achieve the requisite $\Theta(1)$ feature drift, the learning rate for the scalar must be independent of the width:

$$\eta_b = \Theta(1).$$




(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 3: The curves for $\mathrm{RMS}(\widetilde{\mathbf{q}})/\sqrt{d_{\text{head}}}$ for SP, original $\mu$P configuration and our $\mu$P configuration.




(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 4: The curves for $\mathrm{RMS}(\widetilde{\mathbf{k}})/\sqrt{d_{\text{head}}}$ for SP, original $\mu$P configuration and our $\mu$P configuration.

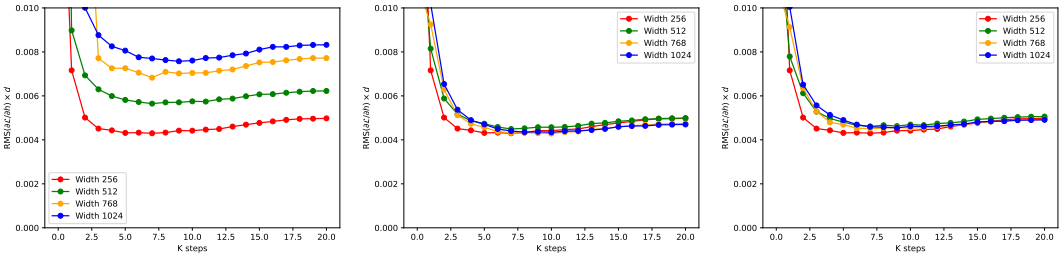
## C Dynamics analysis of $\mu$P for SGD

In Sections 4 and 5, we derived the $\mu$P formulation for the Gated DeltaNet optimized by SGD by propagating coordinate-size estimates through the forward and backward passes. To corroborate these theoretical derivations, we embed diagnostic probes into our training framework. We track the internal activations, gradients, and parameter updates of models across different widths ($d \in \{256, 512, 768, 1024\}$) during the pre-training phase with the same optimal learning rate of 0.4.

### C.1 Verification of Forward Pass Coordinate Sizes

In Section 4, we established that the query and key vectors before L2-normalization layer, $\widetilde{\mathbf{q}}_t$ and $\widetilde{\mathbf{k}}_t$, possess $\Theta(1/\sqrt{d})$ coordinate sizes. Figures 3 and 4 plot the empirical quantities $\widetilde{\mathbf{q}}/\sqrt{d_{\text{head}}} \times \sqrt{d}$ and $\widetilde{\mathbf{k}}/\sqrt{d_{\text{head}}} \times \sqrt{d}$ measured across the sampled layers, where we omitted $d = 768$ due to instability across all the configurations.

As demonstrated, the scaled quantities is approximately constant across varying widths under our $\mu$P configuration, which does not hold for SP and original $\mu$P. This directly substantiates our derivations and validates that the recurrent state $\mathbf{S}_t$ operates under stable variance conditions.



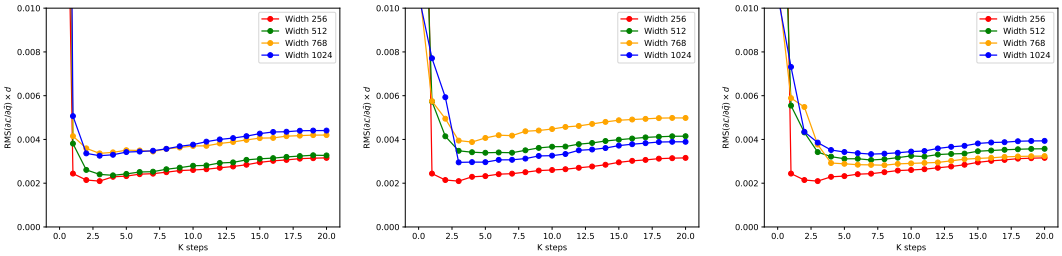

(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 5: The curves for $\mathrm{RMS}(\partial\mathcal{L}/\partial\mathbf{h}) \times d$ for SP, original $\mu$P configuration and our $\mu$P configuration, where $h$ is the hidden state before each layer.

### C.2 Verification of Backward Pass Gradient Scaling

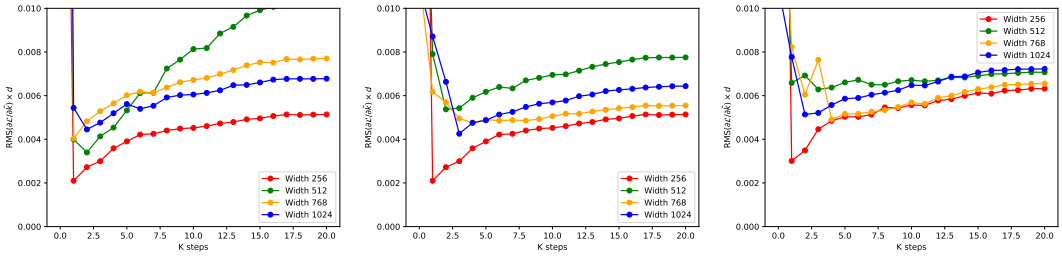
We also inspect the gradient scaling in the dynamics in backward pass. In Figure 5, we deduced that under our formulation and original $\mu$P configuration, the gradient for the hidden states $\partial\mathcal{L}/\partial\mathbf{h}$ strictly follows a $\Theta(1/d)$ coordinate size, whereas standard parametrization forces it to $\Theta(1/\sqrt{d})$, leading to scaling instability.

Moreover, in Figures 6 and 7, we observed that both $\partial\mathcal{L}/\partial\widetilde{\mathbf{q}}$ and $\partial\mathcal{L}/\partial\widetilde{\mathbf{k}}$ also follows a $\Theta(1/d)$ coordinate size in our configuration, while original $\mu$P deviates from it, making the training less stable and transfer not perfect.




(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 6: The curves for $\mathrm{RMS}(\partial\mathcal{L}/\partial\widetilde{\mathbf{q}})\times d$ for SP, original $\mu$P configuration and our $\mu$P configuration.




(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 7: The curves for $\mathrm{RMS}(\partial\mathcal{L}/\partial\widetilde{\mathbf{k}})\times d$ for SP, original $\mu$P configuration and our $\mu$P configuration.

### C.3 Stability of the Gating Dynamics

Our theoretical approximations in Section 4 assume that the data-dependent gating scalars ($\alpha_t$ and $\beta_t$) do not saturate into trivial states (e.g., vanishing completely or remaining strictly 1.0).

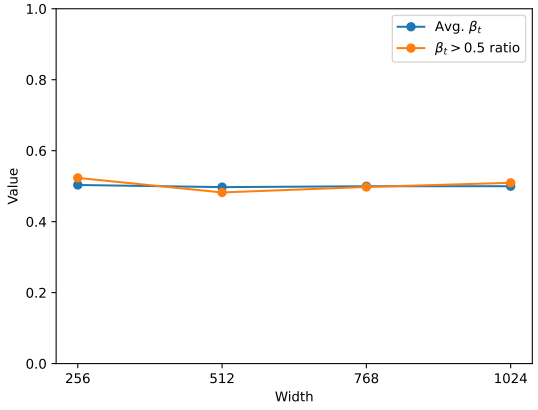
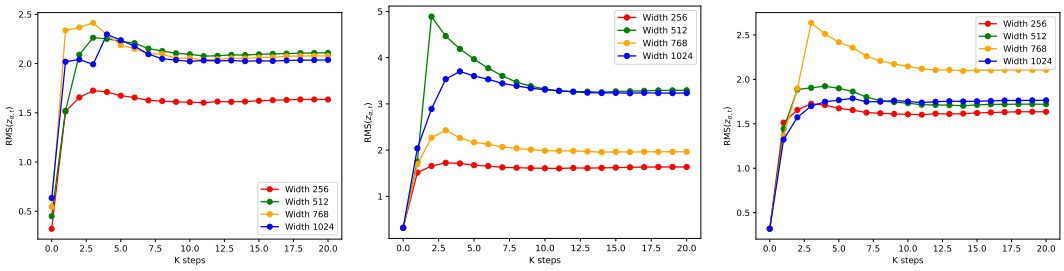
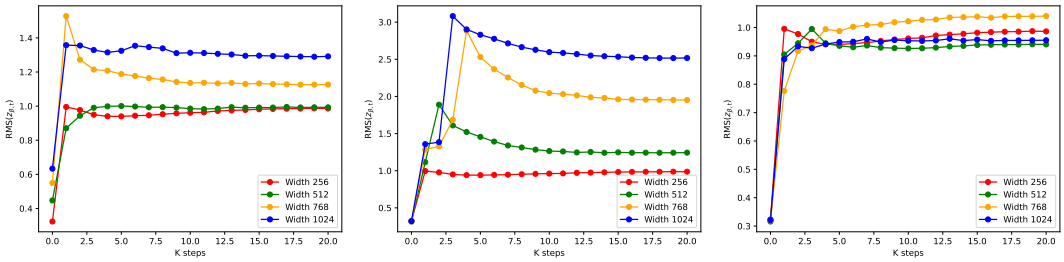



Figure 8: The average of $\beta_t$ and the ratio for strong-writing $\beta_t$ ($\beta_t > 0.5$) for GDN models trained with SGD in our $\mu$P configuration with different widths.




(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

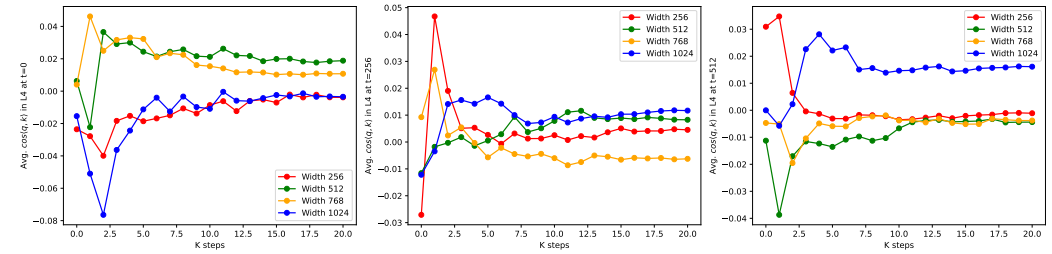
Figure 9: The curves for RMS$(z_{\alpha,t})$ for SP, original $\mu$P configuration and our $\mu$P configuration.



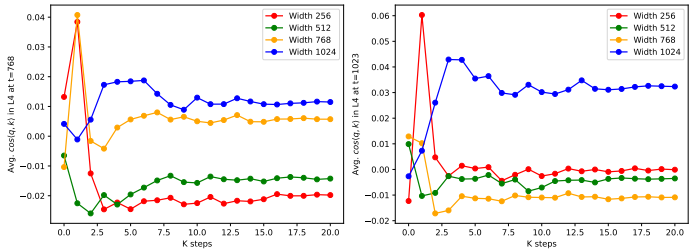

(a) Standard Parametrization (SP) (b) Original $\mu$P configuration (c) Our $\mu$P configuration

Figure 10: The curves for average standard deviation of RMS$(z_{\beta,t})$ for SP, original $\mu$P configuration and our $\mu$P configuration.

By logging the runtime statistics of the write strength $\beta = \sigma(W_\beta x_t)$ in Figure 8, we observe that its expected value remains centered around 0.5 across all model widths, with a significant proportion of tokens actively triggering strong writes. Additionally, in Figures 9 and 10 ($d = 768$ is omitted for $z_{\beta,t}$ for instability across all the configurations), the standard deviation of the pre-activations $z_{\alpha,t} = \mathbf{W}_\alpha \mathbf{x}_t + b$ and $z_{\beta,t} = \mathbf{W}_\beta \mathbf{x}_t$ remains $\Theta(1)$. These observations validate our first-order approximations and ensure that the recurrent memory updates effectively capture long-range dependencies without collapsing as the model scales.




(a) $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ at $t = 0$ (b) $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ at $t = 256$ (c) $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ at $t = 512$




(d) $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ at $t = 768$ (e) $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ at $t = 1023$

Figure 11: The curves for $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ in our $\mu$P configuration for SGD at different $t$ values at 4-th layer.

### C.4 Detecting the dynamics of state spaces

Finally, we depict the dynamics of state spaces of GDN when $t$ increases. We plotted the $\cos\langle \mathbf{q}_t, \mathbf{k}_t \rangle$ given a certain input in our $\mu$P configuration at different $t$ values at 4-th layer in Figure 11. It can be seen that all the values are around 0, showing that independent assumption of $\mathbf{q}_t$ and $\mathbf{k}_t$ still holds for GDN. And during the training process, the value also converges, showing that in our configuration, the model can indeed converge to a stable state.

"'latex

## D Extensions to Related Linear-Attention Architectures

Although the main analysis focuses on Gated DeltaNets (GDNs), the coordinate-size argument is applicable more broadly to recurrent linear-attention architectures with matrix-valued states. The resulting scaling laws are not universal: seemingly minor architectural changes, such as replacing a scalar gate by a channel-wise gate or decoupling erase and write operations, can leave the forward scaling unchanged while changing the backward scaling.

In this section, we illustrate this point on several representative architectures. Our goal is not to claim a complete $\mu$P theory for each model, but to show how the analysis developed in the main paper can be reused to obtain architecture-specific scaling predictions.

**Standing assumptions.** We use the same state orientation as in the main text:

$$\mathbf{S}_t \in \mathbb{R}^{d_v \times d_k}, \qquad \mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t.$$

We consider the same width limit as in the main analysis, with a fixed number of heads and

$$d_k = \Theta(d), \qquad d_v = \Theta(d).$$

For architectures using L2-normalized queries and keys, we assume

$$(q_t)_i, (k_t)_i = \Theta(d^{-1/2}), \qquad (v_t)_j = \Theta(1). \tag{D.1}$$

As in Section 5, after accounting for the output normalization we assume

$$(g_t)_j := \left(\frac{\partial \mathcal{L}}{\partial o_t}\right)_j = \Theta(d^{-1/2}).$$

The calculations below describe width scaling at fixed sequence length. More generally, they remain valid whenever the accumulated recurrent gain is bounded independently of width. As in Assumption 5.1, we assume that the BPTT tail changes the direct-gradient estimates only by width-independent factors. We also use the same mean-field assumption as in the main text, in particular

$$\langle \mathbf{q}_t, \mathbf{k}_t \rangle = \Theta(d^{-1/2}). \tag{D.2}$$

### D.1 Kimi Delta Attention

Kimi Team et al. (2025) introduce Kimi Delta Attention (KDA), which extends GDN by replacing the scalar retention gate with a channel-wise decay. KDA is particularly relevant to our framework because its recurrent state and delta-rule update are closely related to GDN, while the dimensionality of the forgetting gate is different.

The KDA recurrence is commonly written with a $d_k \times d_v$ state. Transposing it to match the convention of the main paper gives

$$\mathbf{S}_t = \mathbf{S}_{t-1} \mathbf{D}_t \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) + \beta_t \mathbf{v}_t \mathbf{k}_t^\top, \tag{D.3}$$

where

$$\mathbf{D}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t), \qquad \boldsymbol{\alpha}_t \in (0,1]^{d_k}.$$

This is transpose-equivalent to the formulation of Kimi Team et al. (2025).

**Forward scaling.** The rank-one write satisfies

$$(\mathbf{v}_t\mathbf{k}_t^\top)_{ji} = \Theta(1)\Theta(d^{-1/2}) = \Theta(d^{-1/2}).$$

For fixed sequence length, or more generally under a width-uniform bound on the accumulated recurrent gain, the state therefore has

$$(\mathbf{S}_t)_{ji} = \Theta(d^{-1/2}).$$

The readout then satisfies

$$(o_t)_j = \sum_{i=1}^{d_k} (\mathbf{S}_t)_{ji}(q_t)_i = \Theta(d^{-1/2}),$$

where the last equality uses the same approximate independence and mean-field cancellation as in the GDN analysis. Thus, replacing GDN's scalar retention gate by KDA's channel-wise decay does not change the leading forward width exponents.

**Gradient of the channel-wise decay.** The backward scaling is different. Define

$$\mathbf{A}_t := \mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^\top.$$

Under the direct-gradient approximation used in Section 5,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{S}_t} = \mathbf{g}_t\mathbf{q}_t^\top.$$

For the $i$-th decay coordinate,

$$\frac{\partial \mathbf{S}_t}{\partial \alpha_{t,i}} = \mathbf{S}_{t-1}\mathbf{e}_i\mathbf{e}_i^\top\mathbf{A}_t,$$

where $\mathbf{e}_i$ is the $i$-th standard basis vector. Hence

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \alpha_{t,i}} &= \left\langle \mathbf{g}_t\mathbf{q}_t^\top, \mathbf{S}_{t-1}\mathbf{e}_i\mathbf{e}_i^\top\mathbf{A}_t \right\rangle_F \\ &= \left(\mathbf{g}_t^\top\mathbf{S}_{t-1}\mathbf{e}_i\right)\left(\mathbf{e}_i^\top\mathbf{A}_t\mathbf{q}_t\right).\end{aligned}$$

For the second factor,

$$\begin{aligned}\mathbf{e}_i^\top\mathbf{A}_t\mathbf{q}_t &= (q_t)_i - \beta_t(k_t)_i\langle\mathbf{k}_t, \mathbf{q}_t\rangle \\ &= \Theta(d^{-1/2}),\end{aligned}$$

because the first term is $\Theta(d^{-1/2})$, whereas the second is $\Theta(d^{-1})$ under Eq. D.2.

For the first factor,

$$\mathbf{g}_t^\top\mathbf{S}_{t-1}\mathbf{e}_i = \sum_{j=1}^{d_v}(g_t)_j(\mathbf{S}_{t-1})_{ji} = \Theta(d^{-1/2}),$$

again using mean-field cancellation over $d_v = \Theta(d)$ terms. Therefore,

$$\boxed{\frac{\partial \mathcal{L}}{\partial \alpha_{t,i}} = \Theta(d^{-1})}. \tag{D.4}$$

This differs from the scalar GDN retention gate, for which the corresponding gradient is $\Theta(d^{-1/2})$. The reason is structural: a scalar gate changes all key directions simultaneously, whereas each coordinate of $\boldsymbol{\alpha}_t$ controls only one row/column direction of the recurrent transition.

Suppose, only for illustration, that the decay logits were generated by a single full-width projection

$$\mathbf{z}_{\alpha,t} = W_\alpha\mathbf{x}_t, \qquad \boldsymbol{\alpha}_t = f(\mathbf{z}_{\alpha,t}),$$

with $f'((z_{\alpha,t})_i) = \Theta(1)$. Then Eq. D.4 implies

$$\frac{\partial \mathcal{L}}{\partial (z_{\alpha,t})_i} = \Theta(d^{-1}).$$

Since $\|\mathbf{x}_t\|_2^2 = \Theta(d)$, one SGD step gives

$$\Delta(z_{\alpha,t})_i = -\eta_\alpha \frac{\partial \mathcal{L}}{\partial (z_{\alpha,t})_i} \|\mathbf{x}_t\|_2^2 = \Theta(\eta_\alpha).$$

A direct full-width decay projection would therefore have the maximal-update prediction

$$\boxed{\eta_\alpha = \Theta(1)}.$$

KDA itself uses a factorized/low-rank parameterization for its channel-wise decay (Kimi Team et al., 2025). For such a factorized map, the learning-rate scaling of each factor depends additionally on how the bottleneck rank scales with $d$. We therefore do not assign a parameter-level learning-rate exponent to the actual KDA decay projection without specifying this additional scaling limit. Eq. D.4, however, is independent of that choice and characterizes the gradient entering the decay parameterization.

**Scalar delta gate.** The scalar $\beta_t$ retains the same leading gradient order as in GDN. Differentiating Eq. D.3,

$$\frac{\partial \mathbf{S}_t}{\partial \beta_t} = (\mathbf{v}_t - \mathbf{S}_{t-1}\mathbf{D}_t\mathbf{k}_t)\,\mathbf{k}_t^\top.$$

Therefore,

$$\frac{\partial \mathcal{L}}{\partial \beta_t} = \langle \mathbf{g}_t, \mathbf{v}_t - \mathbf{S}_{t-1}\mathbf{D}_t\mathbf{k}_t \rangle \, \langle \mathbf{q}_t, \mathbf{k}_t \rangle. \tag{D.5}$$

The first factor is $\Theta(1)$: its write contribution $\langle \mathbf{g}_t, \mathbf{v}_t \rangle$ is $\Theta(1)$, while the state-read correction is of lower order under the same mean-field assumptions. Together with Eq. D.2,

$$\boxed{\frac{\partial \mathcal{L}}{\partial \beta_t} = \Theta(d^{-1/2})}.$$

Thus, if $\beta_t = \sigma(W_\beta \mathbf{x}_t)$ with a nonsaturated sigmoid, the scalar projection retains the GDN prediction

$$\boxed{\eta_\beta = \Theta(d^{-1/2})}.$$

### D.2 Complex KDA

Siems et al. (2026) extend KDA by allowing signed channel-wise gates, $\alpha_{t,i} \in [-1, 1]$, together with $\beta_t \in [0, 2]$. They show that, with unit-normalized keys, the resulting diagonal-plus-rank-one transition remains non-expansive.

For width scaling at fixed sequence length, bounded coefficients of either sign do not change the coordinate exponents in the preceding KDA calculation. More generally, if the accumulated recurrent gain remains bounded independently of width, we again obtain

$$(\mathbf{S}_t)_{ji} = \Theta(d^{-1/2}), \qquad (o_t)_j = \Theta(d^{-1/2}).$$

At the recurrence-variable level,

$$\frac{\partial \mathcal{L}}{\partial \alpha_{t,i}} = \Theta(d^{-1}), \qquad \frac{\partial \mathcal{L}}{\partial \beta_t} = \Theta(d^{-1/2}),$$

provided the corresponding gate derivatives are width-independent and non-vanishing in the operating regime.

Non-expansiveness alone should not be interpreted as a uniform-in-context stationary-state bound: an approximately norm-preserving transition may retain information for arbitrarily many steps. It is sufficient here because the present analysis concerns width scaling at fixed $T$, or, equivalently, because we separately assume a width-uniform bound on recurrent accumulation.

### D.3 DeltaNet

DeltaNet (Yang et al., 2024c) removes the explicit retention gate and uses the delta-rule recurrence

$$\mathbf{S}_t = \mathbf{S}_{t-1}\left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) + \beta_t \mathbf{v}_t \mathbf{k}_t^\top, \qquad \mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t.$$

The modern DeltaNet parameterization uses $\beta_t = \sigma(W_\beta \mathbf{x}_t)$ as a data-dependent writing strength (Yang et al., 2024c).

Under Eq. D.1, the write term again has $\Theta(d^{-1/2})$ entries. Thus, for fixed $T$,

$$(\mathbf{S}_t)_{ji} = \Theta(d^{-1/2}), \qquad (o_t)_j = \Theta(d^{-1/2}).$$

Moreover,

$$\frac{\partial \mathbf{S}_t}{\partial \beta_t} = (\mathbf{v}_t - \mathbf{S}_{t-1}\mathbf{k}_t)\,\mathbf{k}_t^\top,$$

so the same calculation as Eq. D.5 gives

$$\boxed{\frac{\partial \mathcal{L}}{\partial \beta_t} = \Theta(d^{-1/2}), \qquad \eta_\beta = \Theta(d^{-1/2})}.$$

The main difference from GDN concerns accumulation over sequence length rather than the fixed-$T$ width exponent. Since DeltaNet has no explicit multiplicative retention factor, its recurrent transition need not be strictly contractive in all directions. Joint width-context scaling therefore requires an additional analysis of the accumulated recurrent gain.

### D.4 Gated Linear Attention

Gated Linear Attention (GLA) uses a channel-wise data-dependent decay,

$$\mathbf{S}_t = \mathbf{S}_{t-1}\mathbf{D}_t + \mathbf{v}_t\mathbf{k}_t^\top, \qquad \mathbf{D}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t),$$

which is the transpose-equivalent form of the recurrence in Yang et al. (2024b). In the actual GLA architecture, $\boldsymbol{\alpha}_t$ is produced through a low-rank gate parameterization (Yang et al., 2024b).

GLA is useful for illustrating why the forward analysis should precede any parameter-level $\mu$P prescription. Unlike the GDN or KDA derivation above, one should not assume a particular normalization of its read/write features. Suppose more generally that

$$(q_t)_i = \Theta(d^{-a_q}), \qquad (k_t)_i = \Theta(d^{-a_k}), \qquad (v_t)_j = \Theta(d^{a_v}).$$

Under a width-uniform bound on recurrent accumulation,

$$(\mathbf{S}_t)_{ji} = \Theta(d^{a_v - a_k}).$$

Using mean-field cancellation in the readout,

$$\begin{aligned}(o_t)_j &= \sum_{i=1}^{d_k} (\mathbf{S}_t)_{ji}(q_t)_i \\ &= \Theta\left(d^{1/2 - a_q - a_k + a_v}\right).\end{aligned}$$

Thus, even among matrix-state linear-attention models, the state and readout exponents depend on the feature normalization convention. The gate parameterization must then be analyzed on top of these forward exponents; it cannot be inferred from the recurrence alone.

### D.5 Gated DeltaNet-2

Gated DeltaNet-2 (Hatamizadeh et al., 2026) further generalizes KDA by decoupling the erase and write operations. It introduces a channel-wise erase gate $\mathbf{b}_t \in [0,1]^{d_k}$ and a channel-wise write gate $\mathbf{w}_t \in [0,1]^{d_v}$. In the state orientation of the main paper, its recurrence is

$$\mathbf{S}_t = \mathbf{S}_{t-1}\mathbf{D}_t\left(\mathbf{I} - (\mathbf{b}_t \odot \mathbf{k}_t)\mathbf{k}_t^\top\right) + (\mathbf{w}_t \odot \mathbf{v}_t)\mathbf{k}_t^\top, \tag{D.6}$$

which is transpose-equivalent to the formulation of Hatamizadeh et al. (2026). The gates are produced by independent projections,

$$\mathbf{b}_t = \sigma(W_b \mathbf{x}_t), \qquad \mathbf{w}_t = \sigma(W_w \mathbf{x}_t).$$

The write term still has $\Theta(d^{-1/2})$ entries, so under the same fixed-$T$ or bounded-gain condition,

$$(\mathbf{S}_t)_{ji} = \Theta(d^{-1/2}), \qquad (o_t)_j = \Theta(d^{-1/2}).$$

**Write gate.** For the $j$-th value coordinate,

$$\frac{\partial \mathbf{S}_t}{\partial w_{t,j}} = (v_t)_j \mathbf{e}_j \mathbf{k}_t^\top.$$

Therefore,

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial w_{t,j}} &= (g_t)_j (v_t)_j \langle \mathbf{q}_t, \mathbf{k}_t \rangle \\ &= \boxed{\Theta(d^{-1})}.\end{aligned}$$

If the sigmoid is nonsaturated, the same scaling holds for the corresponding write-gate logit. Since $\|\mathbf{x}_t\|_2^2 = \Theta(d)$,

$$\boxed{\eta_w = \Theta(1)}.$$

**Erase gate.** Let

$$\mathbf{e}_t := \mathbf{b}_t \odot \mathbf{k}_t.$$

Differentiating Eq. D.6 with respect to $b_{t,i}$ gives

$$\frac{\partial \mathbf{S}_t}{\partial b_{t,i}} = -(k_t)_i \mathbf{S}_{t-1} \mathbf{D}_t \mathbf{e}_i \mathbf{k}_t^\top.$$

Contracting with the direct state gradient yields

$$\frac{\partial \mathcal{L}}{\partial b_{t,i}} = -(k_t)_i \left( \mathbf{g}_t^\top \mathbf{S}_{t-1} \mathbf{D}_t \mathbf{e}_i \right) \langle \mathbf{q}_t, \mathbf{k}_t \rangle.$$

The three factors have orders

$$(k_t)_i = \Theta(d^{-1/2}),$$
$$\mathbf{g}_t^\top \mathbf{S}_{t-1} \mathbf{D}_t \mathbf{e}_i = \Theta(d^{-1/2}),$$

and

$$\langle \mathbf{q}_t, \mathbf{k}_t \rangle = \Theta(d^{-1/2}),$$

respectively. Consequently,

$$\boxed{\frac{\partial \mathcal{L}}{\partial b_{t,i}} = \Theta(d^{-3/2})}.$$

For $\mathbf{b}_t = \sigma(W_b \mathbf{x}_t)$ with nonsaturated sigmoid,

$$\Delta(z_{b,t})_i = \eta_b \Theta(d^{-3/2}) \Theta(d) = \eta_b \Theta(d^{-1/2}),$$

so maximal update predicts

$$\boxed{\eta_b = \Theta(d^{1/2})}.$$

This provides a particularly useful example of why architectural role matters. The erase and write gates are both channel-wise vectors generated by full projections, yet their gradients have different width exponents:

$$\frac{\partial \mathcal{L}}{\partial w_{t,j}} = \Theta(d^{-1}), \qquad \frac{\partial \mathcal{L}}{\partial b_{t,i}} = \Theta(d^{-3/2}).$$

Consequently, their predicted SGD learning rates are different despite having the same type of parameterization.

The channel-wise decay $\boldsymbol{\alpha}_t$ in Gated DeltaNet-2 enters the recurrence in the same position as the KDA decay. Its recurrence-level gradient therefore satisfies

$$\frac{\partial \mathcal{L}}{\partial \alpha_{t,i}} = \Theta(d^{-1})$$

under the same assumptions. Since Hatamizadeh et al. (2026) generate its log-decay using a full projection, this predicts an ordinary $\Theta(1)$ SGD learning-rate scaling for that projection, provided the decay activation has a width-independent, non-vanishing derivative.

Table 3: Illustrative SGD scaling predictions obtained from the analysis framework of the main paper. The results assume $d_k, d_v = \Theta(d)$, L2-normalized query/key coordinates, the mean-field relation $\langle \mathbf{q}_t, \mathbf{k}_t \rangle = \Theta(d^{-1/2})$, and a width-uniform recurrent-gradient correction.

| Quantity | Gradient scale | SGD LR scale |
|---|---|---|
| GDN scalar decay projection | $\Theta(d^{-1/2})$ | $\Theta(d^{-1/2})$ |
| DeltaNet/GDN/KDA scalar $\beta$ projection | $\Theta(d^{-1/2})$ | $\Theta(d^{-1/2})$ |
| KDA channel decay coordinate $\alpha_i$ | $\Theta(d^{-1})$ | – |
| KDA direct full-width decay projection (hypothetical) | $\Theta(d^{-1})$ | $\Theta(1)$ |
| Gated DeltaNet-2 decay projection | $\Theta(d^{-1})$ | $\Theta(1)$ |
| Gated DeltaNet-2 write projection | $\Theta(d^{-1})$ | $\Theta(1)$ |
| Gated DeltaNet-2 erase projection | $\Theta(d^{-3/2})$ | $\Theta(d^{1/2})$ |

### D.6 Summary and General Recipe

The preceding examples suggest a general procedure for matrix-state linear-attention architectures of the form

$$\mathbf{S}_t = \mathcal{T}_t(\mathbf{S}_{t-1}) + \mathbf{U}_t, \qquad \mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t.$$

One can first determine the coordinate sizes of the read and write features, then derive the state and readout scales, control the width dependence of the recurrent BPTT contribution, and finally analyze each gate according to how it enters the recurrent transition.

Table 3 summarizes the recurrence-level results for the delta-rule architectures considered above. Parameter-level learning-rate entries are included only when the gate is generated by a single full-width projection; factorized projections require an additional specification of how the factorization rank scales with width.

These examples demonstrate two complementary points. First, the forward state scaling can be robust to substantial changes in the recurrent transition: GDN, KDA, DeltaNet, and Gated DeltaNet-2 all retain $\Theta(d^{-1/2})$ state entries under the assumptions above. Second, identical forward scaling does not imply identical backward or optimizer scaling. Changing a scalar decay into a channel-wise decay changes the gate-gradient exponent, while Gated DeltaNet-2 shows that even two channel-wise gates of identical dimensionality can require different SGD learning-rate exponents because they enter the recurrence through different algebraic paths.

Therefore, the main value of the analysis is not a single universal parameterization for all linear-attention models. Rather, it provides a systematic procedure for deriving architecture-specific maximal-update rules for recurrent matrix-state models.

## E Further experiments for validation

To validate our theories and predictions, we implement more experiments for validation. In Section 6, the experiments for SGD predicts the optimal learning rate is $0.4$ across different widths. We validates with $d = 1,536$ for SGD in our $\mu$P configuration for learning rate in $[0.3, 0.4, 0.5]$, the resulting validation loss is 2.9737, 2.9511 and 2.9540, respectively, showing that the optimal learning rate does transfer within a wider range of widths. Moreover, we implement experiments with 3 more seeds for $d = 256$ and $d = 512$ with SGD and AdamW in our $\mu$P configuration. The optimal learning rates are all 0.4 for SGD. While for AdamW, the optimal learning rate is $8 \times 10^{-3}$ for $d = 512$ for all seeds and 2 seeds of $d = 256$, but in another seed of $d = 256$, the optimal learning rate is $5 \times 10^{-3}$. Therefore, we believe that the optimal learning rate is near 0.4 for SGD and between $5 \times 10^{-3}$ and $8 \times 10^{-3}$ for AdamW, which is nearly transferrable within our $\mu$P configuration.

## F Empirical analysis of short-memory assumption

In Section 5, we proposed the assumption of short effective memory and domination of direct gradient in Assumption 5.1. To make the theories more solid, we first measured the gate-based effective memory $L_{\text{eff},t} = \sum_{\ell \geq 0} \prod_{j=1}^{\ell} \alpha_{t+j}$. Across sampled layers, heads, and token positions, its median is 62.29 and 79.65 tokens for models with hidden dimensions of 256 and 1,024, respectively, suggesting that the memory contribution saturates well below the context length of 1,024 in the regime considered in the paper.

Second, for the domination of direct gradient, we compare the full-BPTT parameter gradients with gradients obtained by stopping the recurrent-state gradient between adjacent positions. We measure the gradients for k and v when the gradient flow of BPTT tail is removed ($g_s$ using stop-gradient) and not ($g_f$), only in the 4-th layer. The relative gradient difference, $\|g_f - g_s\|_2 / \|g_f\|_2$, is 0.272 and 0.231 for $W_k$ and $W_v$ on average. Therefore, we acknowledge that the BPTT can bring about non-negligible but still relatively small affects on the gradient.