When Good Enough Is Optimal: Multiplication-Only Matrix Inversion Approximation for Quantized Gated DeltaNet

Luoming Zhang [*1] **Yuwei Ren** [*1] **Kui Zhang** [1] **Tian Liu** [1] **Lingjuan Ge** [1] **Denghao Li** [1]
Matthew Harper Langston [1] **Yin Huang** [1] **Weiliang Will Zeng** [1] **Liang Zhang** [1]

Abstract

Matrix inversion in chunk-wise parallel linear attention is a major bottleneck for long-context modeling, particularly on NPUs, where forward-substitution-based methods exhibit limited parallelism and poor hardware utilization. We propose a fast, Matrix Multiplication (MatMul)-based algorithm tailored for strictly lower-triangular matrices arising in chunk-wise linear attention. Motivated by the rapid growth of Neumann-series terms and the diagonal concentration of the inverse matrix, we employ a truncated Neumann expansion with structural masking and parallel residual correction to eliminate sequential dependencies. We further extend our method to low-bits INT by mitigating the dynamic range expansion arising from repeated matrix power operations, and adapt the approximation order and residual step to the chunk size to minimize computational cost while preserving the model's accuracy. Experiments on Qwen3.5-family models demonstrate up to 5× kernel-level speedup and a 20% reduction in decode-layer overhead, while preserving accuracy under both floating-point and low-precision inference. Our method offers an efficient and hardware-friendly solution for scalable linear attention.

1. Introduction

As LLM context lengths grow (Zhang et al., 2025; Qwen Team, 2026; Google DeepMind, 2026), standard attention (Vaswani et al., 2017) suffers from quadratic memory and runtime costs. Linear attention methods (Katharopoulos et al., 2020; Hua et al., 2022; Sun et al., 2023; Gu & Dao, 2023) address this issue by maintaining a fixed-size recurrent state, avoiding the quadratic $\mathcal{O}(T^2)$ cost. GatedDeltaNet (Yang et al., 2025a), adopted by recent large-scale models such as QWen3.5 (Qwen Team, 2026) and KiMi (Zhang et al., 2025), achieves strong long-context capability, but its sequential state update remains a key bottleneck for parallel hardware execution.

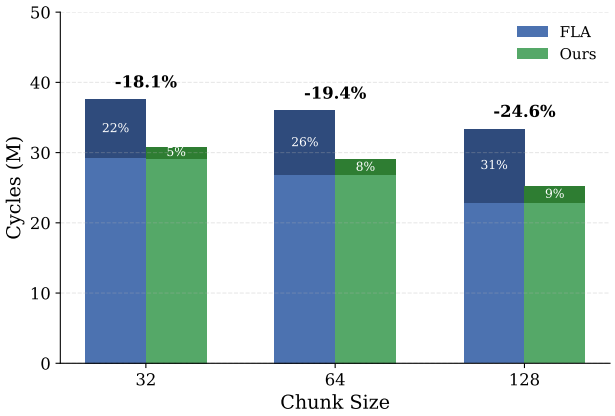

Figure 1. Cycle breakdown across chunk sizes at fixed sequence length ($L = 128$) on a GatedDeltaNet layer from Qwen3.5-4B. The lighter segment denotes base computation, while the darker segment highlights matrix-inverse overhead.

To address this bottleneck, Yang et al. (2024) reformulate token-level recurrent updates in Gated Linear Attention into chunk-level recurrences, enabling parallel computation within each chunk while preserving recurrent semantics across chunks. When applied to GatedDeltaNet, this strategy removes the outer token-by-token dependency and exposes locally parallel low-rank linear solves. However, it also introduces a new inner bottleneck: triangular matrix inversion. As shown in Figure 1, matrix inversion is not a negligible overhead. On a GatedDeltaNet layer from Qwen3.5-4B with fixed sequence length ($L = 128$), it accounts for 22.3%–31.4% of the total cycle cost across chunk sizes, and its relative contribution increases as the chunk size grows. This overhead stems from the forward-substitution-based inversion procedure, whose limited parallelism and vector-heavy operations are poorly matched to NPU execu-

[*]Equal contribution [1]Qualcomm AI Research, an initiative of Qualcomm Technologies, Inc. Correspondence to: Luoming Zhang <luomzhan@qti.qualcomm.com>, Yuwei Ren <ren@qti.qualcomm.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

tion, leaving matrix processing units underutilized.

In GatedDeltaNet, the matrix inversion in fragments takes the form $T = (I - A)^{-1}, A \in \mathbb{R}^{k\times k}$. Since A is a strict lower-triangular matrix, its inverse can be efficiently approximated by using the Neumann series expansion (Golub & Van Loan, 2013). However, directly expanding the Neumann series for larger chunks can introduce rapidly growing intermediate values, increasing the risk of numerical instability under finite-precision arithmetic. Previous work (Yang et al., 2024; Huawei CSL, 2026) adopts block-wise matrix inversion, decomposing a large matrix inversion into multiple smaller ones. While this improves numerical behavior, it also restricts the matrix size of each computation block, reducing available parallelism and limiting hardware utilization. These observations raise a central question: *how can we design a MatMul-based algorithm for fast inversion of large matrices to accelerate the overall computation?*

We answer this question by reformulating such triangular inversion as an *approximate-but-sufficient* MatMul-only computation. Our key observation is that exact inversion is unnecessary: although high-order Neumann terms may produce large intermediate values, their impact mainly lies on deeper sub-diagonals, while the inverse energy is concentrated near the main diagonal. Thus, a low-order Neumann approximation with structured masking and parallel residual correction preserves accuracy while mapping the dominant computation to matrix multiplications.

Our main contributions are summarized as follows:

- **Approximate-but-sufficient MatMul-only inversion.** We replace sequential triangular inversion in chunk-wise GatedDeltaNet with a low-order Neumann approximation, structured diagonal masking, and parallel residual correction, preserving accuracy under floating-point and integer quantization while mapping the computation to MatMul kernels.
- **Hardware-efficient NPU execution.** The proposed formulation removes triangular-solve dependencies, exposes dense matrix multiplications with controllable truncation and correction depth, and achieves up to $5\times$ kernel speedup on NPUs without accuracy degradation.

2. Background

2.1. Matrix inversion in Gated DeltaNet

Let $A \in \mathbb{R}^{k\times k}$ be a strictly lower-triangular matrix, i.e., $T_{ij} = 0$ for $j \geq i$. We consider the inversion of a matrix of the form

$$T = (I - A)^{-1}. \tag{1}$$

In Gated DeltaNet (Yang et al., 2025a), the matrix A is constructed from the key representations via a scaled inner product of the form $A = KK^\top$. Since an ℓ_2-normalization layer is applied after the key projection, which bounds the magnitude of the entries in $\rho(A) < 1$. More details about GatedDeltaNet are included in Appendix B.

2.2. Forward Substitution

In previous methods, the matrix inverse is computed via a forward-substitution procedure:

$$T_{i,0:i} \leftarrow T_{i,0:i} + T_{i,0:i}\, T_{0:i,0:i}, \qquad i = 1, \ldots, k-1, \tag{2}$$

followed by an identity augmentation

$$T \leftarrow T + I. \tag{3}$$

As shown in Eq. (2), forward substitution requires $k - 1$ sequential iterations for a $k \times k$ matrix, where each step depends on all previous results. These loop-carried dependencies hinder efficient parallelization and limit scalability on modern hardware. Flash Linear Attention (Yang & Zhang, 2024) addresses this by decomposing a large matrix inverse into multiple smaller inverses computed jointly; details are provided in Appendix F.

2.3. Neumann Series for Strictly Lower-Triangular Matrix Inversion

When the spectral radius of A satisfies $\rho(A) < 1$, the inverse can be expressed by the Neumann series as

$$(I - A)^{-1} = \sum_{n=0}^{\infty} A^n = I + A + A^2 + \cdots. \tag{4}$$

Moreover, since A is strictly lower triangular, it is nilpotent and satisfies

$$A^k = 0. \tag{5}$$

Therefore, the above series terminates after finitely many terms, yielding the exact finite expansion

$$(I - A)^{-1} = \sum_{n=0}^{k-1} A^n. \tag{6}$$

This finite-series form underpins matrix-inverse approximation in chunk-wise Gated DeltaNet and motivates Neumann-series implementations.

3. Method

3.1. Low-Order Truncation for Efficient Inversion

Although the inverse $(I - A)^{-1}$ admits the exact finite expansion with $\sum_{n=0}^{k-1} A^n$, computing all k terms is inefficient for large chunk size. A full expansion is often unnecessary in practice. Because the inverse energy is concentrated near the main diagonal.

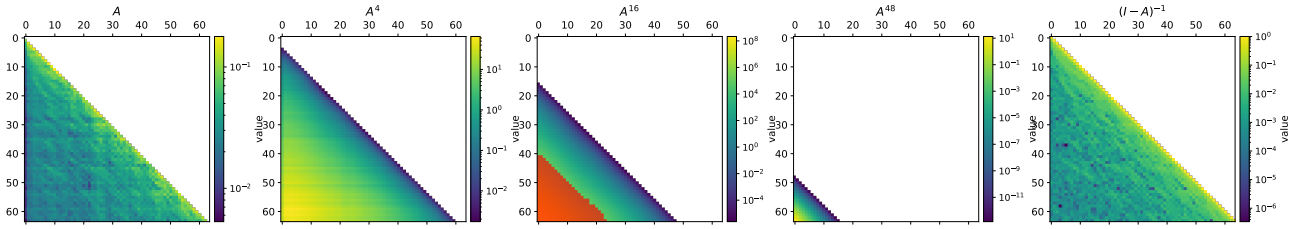

Figure 2. Distribution of A^n over 100 samples. Values exceeding the FP16 limit (65,504) are highlighted in red; 2 samples exhibit overflow, indicating heavy-tailed growth in higher-order terms.

Lemma 3.1 (Diagonal Localization of Neumann Series for strictly lower-triangular matrix)**.** *Let $A \in \mathbb{R}^{k\times k}$ be strictly lower triangular, i.e., $A_{ij} = 0$ for $i \le j$. Then for any $n \ge 0$,*

$$(A^n)_{ij} = 0 \;\Rightarrow\; i - j < n.$$

Based on Lemma 3.1, each power A^n only contributes to the n-th sub-diagonal and below, a low-order truncation already captures most of the useful structure of the inverse. Therefore, the Neumann series can be approximated by its first N terms:

$$T^{(0)} = \sum_{n=0}^{N} A^n, \qquad N \ll k, \tag{7}$$

as the initial approximation. The truncation error is

$$E_N = \sum_{n=N+1}^{k-1} A^n, \tag{8}$$

which decays rapidly as N increases when $\|A\| < 1$.

Lemma 3.2 (Truncation Error of Finite Neumann Expansion)**.** *Let $A \in \mathbb{R}^{k\times k}$ be a strictly lower-triangular matrix, for any truncation order $N < k-1$ and for any sub-multiplicative matrix norm $\|\cdot\|$ and $\|A\| < 1$, then*

$$\|E_N\| \le \frac{\|A\|^{N+1}}{1-\|A\|}. \tag{9}$$

This lemma and empirical study, in Appendix C, shows that a much lower truncation order($N \approx 30$) already captures most of the inverse structure, making full expansion unnecessary in practice, although $(I-A)^{-1}$ admits an exact finite expansion for strictly lower-triangular A.

3.2. Numerical Overflow in Truncated Neumann Series

We analyze the distribution of individual entries in the Neumann series by examining average across 100 samples. Fig. 2 illustrates the matrices at different truncation orders. Based on these empirical observations, we derive the following two findings.

Although the entries of A are bounded in $[0,1]$, the magnitudes of higher-order terms in the Neumann series grow rapidly with increasing powers. In particular, as n increases, the distribution of A^n exhibits heavier-tailed distributions, with many entries exceeding the FP16 range. This effect, particularly visible in higher-order terms, shows that overflow can occur even for well-conditioned matrices. Thus, forward-substitution FP16 Neumann implementations are limited by element-wise growth rather than spectral properties alone.

Lemma 3.3 (Exact Entrywise Growth of Powers of Strictly Lower-Triangular Matrices)**.** *Let $A \in \mathbb{R}^{k\times k}$ be a strictly lower-triangular matrix satisfying $\rho(A) < 1$*

For indices $i > j$, define the diagonal distance $d \triangleq i - j$. Then, for any integer $n \ge 1$,

$$(A^n)_{ij} \;\le\; \binom{d-1}{n-1}.$$

Based on Lemma 3.3, we can directly derive the numerical stability bounds for the Neumann-series-based inversion. For a 64×64 matrix, the worst-case upper bound is 6 under FP16 precision and 32 under FP32 precision.

The data distribution of A^n and $(I-A)^{-1}$ exhibits a strong block-wise structure. The dominant contributions of the inverse matrix $(I-A)^{-1}$ are primarily concentrated along the diagonal and near-diagonal entries. While values that exceed the representable precision range are primarily concentrated in the lower-left (strictly lower-triangular) region of the matrix.

By Lemma 3.1, the n-th Neumann term contributes only to the n-th sub-diagonal and below, leading to strong diagonal concentration in $(I-A)^{-1}$. Consequently, truncated Neumann series initialization can efficiently obtain an accurate approximation. As analyzed in Appendix C, the Neumann series in practice typically requires more than 20 terms to achieve over 0.99 power ratio. Therefore, maintaining a small truncation order for numerical stability leads to noticeable accuracy degradation. To address this trade-off, we introduce a residual correction stage following the truncated Neumann approximation, as described in Sec. 3.4.

3.3. Effective Diagonal Masking

Low-order truncation preserves most of the meaningful values near the diagonal, which exactly match the corresponding entries of $(I - A)^{-1}$. However, the truncation error increases with the magnitude of the Neumann-series terms, by Lemma 3.2 . Such artifacts can adversely affect subsequent computations based on the approximate inverse and are particularly problematic under INT quantization, where min–max calibration is sensitive to outliers.

Based on the observed block-wise structure of A^n and $(I - A)^{-1}$, we apply a diagonal mask to separate the already well-resolved components from those with large truncation errors. The diagonal mask $M_{ij}^{(K)}$ is defined as

$$M_{ij}^{(K)} = \begin{cases} 1, & i - j \leq K, \\ 0, & otherwise. \end{cases} \tag{10}$$

Where the K is set equal to the Neumann series truncation order N. This masking strategy effectively removes extraneous large-magnitude values, improving numerical stability and making the approximation quantization-friendly.

3.4. Residual Correction

After applying a low–order Neumann series expansion together with the diagonal mask, we obtain an initial approximation of $(I - A)^{-1}$, denoted as $T^{(0)}$:

$$T^{(0)} = M^{(N)} \sum_{n=0}^{N} A^n. \tag{11}$$

Although diagonal masking suppresses large outliers, non-negligible approximation errors remain in the lower-triangular region, necessitating further refinement. Following a matrix-calculation–based design rule, we adopt a residual correction scheme that iteratively improves the approximation without increasing the Neumann order.

Given the current estimate $T^{(m)}$, the residual and update rule are defined as

$$\begin{aligned} R^{(m)} &= I - (I - A)T^{(m)}, \\ T^{(m+1)} &= T^{(m)} + T^{(m)}R^{(m)}. \end{aligned} \tag{12}$$

This formulation applies a first-order multiplicative correction to compensate for the remaining inversion error. Empirically, our ablation study shows that only 2–3 iterations are sufficient for convergence.

However, the iterative process is inherently sequential: $R^{(m)}$ depends on $T^{(m)}$, and $T^{(m+1)}$ depends on its predecessor. This step-wise dependency limits hardware efficiency, more details can be found in Appendix G. To address this issue, we reformulate residual correction as an accumulation of matrix powers, eliminating explicit iteration. Specifically, we define

$$\begin{aligned} E &= I - (I - A)T^{(0)}, \\ T &\approx T^{(0)} \sum_{s=0}^{S} E^s. \end{aligned} \tag{13}$$

With $S = 2^M$ (typically $S = 4$–8), this approach enables MatMul-only execution, improving hardware efficiency while maintaining correction accuracy.

Algorithm 1 Masked Neumann Initialization with Iterative Residual Correction

Require: Strictly lower-triangular $A \in \mathbb{R}^{k \times k}$; Neumann order N; residual iterations $m_{\max}$
Ensure: Approximate inverse $T \approx (I - A)^{-1}$

```
Construct mask M^(N) where M_ij^(N) = 1 if i − j ≤ N, else 0
T^(0) ← I,   P ← I
for n = 1 to N do
    P ← PA
    T^(0) ← T^(0) + P
end for
T^(0) ← M^(N) ⊙ T^(0)
E ← I − (I − A)T^(0)
T ← I, P ← I
for s = 0 to s_max − 1 do
    P ← PE
    T ← T + P
end for
return T ← TT^(0)
```

3.5. Good-Enough under Low-Bit Quantization

The *good-enough* principle seeks the minimal Neumann truncation order that preserves numerical stability while introducing Non (or negligible) accuracy loss. This rationale is particularly strong under low-bit quantization (e.g., INT8), where the iterative Neumann structure repeatedly applies quantization to the same values, causing quantization noise to dominate the overall error. In this regime, the additional approximation error from low-order truncation becomes insignificant, making reduced computation a preferable trade-off for improved hardware efficiency and lower inference latency.

As shown in Fig. 3, increasing the truncation order from $n = 3$ to $n = 4$ dramatically enlarges the dynamic range: the maximum absolute value expands from 978 to 16216. This expansion is highly non-uniform: a few entries grow explosively while most remain tightly concentrated, yielding a pronounced heavy tail. Under uniform quantization, the scaling factors are therefore dictated by these rare extreme values rather than the representative bulk. Conse-

quently, most entries are compressed into a severely limited effective resolution, which exacerbates rounding errors and leads to a significant degradation in numerical fidelity under INT8/INT16 precision constraints.

As a result, higher-order truncation introduces substantial quantization instability while providing only marginal approximation gains. We therefore limit the truncation order ($N \leq 3$) and compensate with residual correction, achieving an efficient matrix inverse that balances numerical stability, accuracy, and efficiency under low-precision settings.

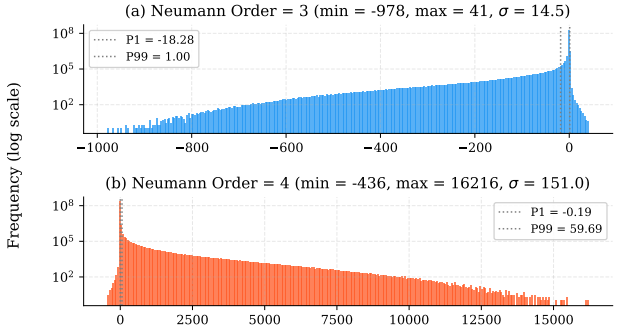

Figure 3. Activation distribution under Neumann truncation for order=3 and 4 for a 64×64 matrix Neumann Series.

4. Experiments

4.1. Experiments Setting

We select the Qwen3Next (Team, 2025) and Qwen3.5 (Qwen Team, 2026) model families, to complete full accuracy and on-target latency study. Unless otherwise stated, all experiments are conducted with a chunk size $k = 64$, a Neumann series order $N = 3$, and $S = 8$ residual correction steps.

Single-kernel accuracy analysis. We conduct experiments using 100 samples from the WikiText-v2 (Merity et al., 2017) training set with a sequence length of 4K, based on Qwen3-Next-80B-A3B-Instruct model [1]. The experiments in Sec. 3.2 are also conducted using the same data.

End-to-end accuracy analysis. Based on multiple model scales of the Qwen3.5 family, the evaluation includes perplexity (PPL) measured on the WikiText-v2 validation, and also assess downstream task accuracy on benchmarks such as MMLU (Hendrycks et al., 2021), CSR (Clark et al., 2019; 2018; Bisk et al., 2020; Zellers et al., 2019) and multi-modality RealWorldQA (xAI, 2024)[2]. In the quantization, we apply W4A16 to the LLM decoder and W8A16 to the visual decoder. Detailed quantization configurations are provided in the Appendix H.

[1]https://huggingface.co/Qwen/Qwen3-Next-80B-A3B-Instruct

[2]Thinking mode is disabled for efficient evaluation, resulting in a slight drop in RealWorldQA accuracy.

On-target performance analysis. We profile the runtime cost of a single matrix inversion operation and a single GatedDeltaNet layer on the Snapdragon 8 Elite Gen 5 platform. We compare our approach against chunk-wise parallel implementation from Flash Linear Attention.

4.2. Accuracy Experiments

Single kernel experiment. During inference, we collect the inputs and outputs of each matrix inversion operation from all layers and quantify the approximation error of the proposed matrix inversion method using the signal-to-noise ratio (SNR). As shown in Table 1, FP16 and INT16 introduce only marginal MSE increases compared to FP32 while maintaining high SNR. INT16 remains comparable to FP16, demonstrating stable single-kernel behavior under low-precision execution.

Table 1. Single Kernel accuracy

METRIC	FP32	FP16	INT16
SNR	70.02	66.78	67.16

End-to-End experiments. Table 2 reports end-to-end accuracy across Qwen3.5 models from 0.8B to 9B parameters. Across all scales, our method matches the FLA baseline in perplexity, indicating no degradation in language modeling quality. On downstream tasks, including MMLU, CSR, and RealWorldQA, our approach achieves comparable performance with only minor fluctuations (within ±0.3), demonstrating that the proposed modification preserves task accuracy across model sizes. These results confirm that our method introduces no observable accuracy regression while enabling the targeted efficiency optimizations.

Table 2. End-to-end accuracy across Qwen3.5 model sizes.

MODEL	METHOD	PPL	MMLU	CSR	RWQA
0.8B	FLA	15.74	50.57	51.11	62.35
	OURS	15.74	50.61	51.17	61.70
2B	FLA	11.07	57.66	56.99	65.23
	OURS	11.07	57.72	56.93	65.62
4B	FLA	8.89	70.10	65.48	74.77
	OURS	8.89	70.11	65.45	74.64
9B	FLA	8.21	70.26	67.28	74.38
	OURS	8.21	70.23	67.30	74.12

Quantization experiments. Table 3 compares end-to-end accuracy under W4A16 quantization across Qwen3.5 model sizes. While FLA quantization leads to noticeable degradation compared with full-precision baselines, our method preserves performance more effectively, particularly on RealWorldQA, with only marginal differences in PPL, MMLU, and CSR. Across both 0.8B and 4B models, these results demonstrate that the proposed approach better maintains

task accuracy under aggressive low-precision quantization. Besides, we also explore the low bits setting (INT8) in matrix inverse, and its accuracy is almost same to INT16, shown in Appendix H.2

Table 3. quantization accuracy comparison across Qwen3.5 model

MODEL	METHOD	PPL	MMLU	CSR	RWQA
0.8B	FLA-FP	15.74	50.57	51.11	62.35
	FLA-W4A16	17.54	48.26	49.03	56.08
	OURS-W4A16	17.55	48.27	49.21	60.39
4B	FLA-FP	8.89	70.10	65.48	74.77
	FLA-W4A16	9.66	73.34	65.05	73.33
	OURS-W4A16	9.67	73.29	65.03	72.81

4.3. On-target performance experiments

For a chunk-size of $k = 32$, we found that $N = 3$ and $S = 4$ are sufficient to achieve accurate results. To test the end-to-end performance on different CL and chunk size, we included an ablation study in Appendix J.

Single kernel performance analysis. As shown in Figure 4, our method reduces single matrix-inverse cycles across chunk sizes, achieving 5.2×, 4.2×, and 4.6× improvements at 32, 64, and 128, respectively. Chunk size 32 uses 7 matmuls, while 64 requires 11. Despite $4\times$ smaller matmuls at size 32, latency does not scale proportionally, indicating dominance of non-matmul overheads. At chunk size 128, splitting and non-contiguous write-back of 64×64 submatrices introduces memory overhead, which is observed in both our method and FLA and dampens scaling efficiency.

GatedDeltaNet performance analysis. Figure 1 compares the cycle breakdown between FLA and our method across different chunk sizes. While the computation excluding matrix inversion remains comparable, FLA incurs substantial overhead from matrix inversion, accounting for 22.3%, 25.6%, and ∼31% of total cycles at chunk sizes 32, 64, and 128, respectively. In contrast, our approach reduces the matrix-inverse cost to 5.2%, 7.6%, and ∼9%, yielding overall cycle reductions of 18.1%, 19.4%, and 24.6%.

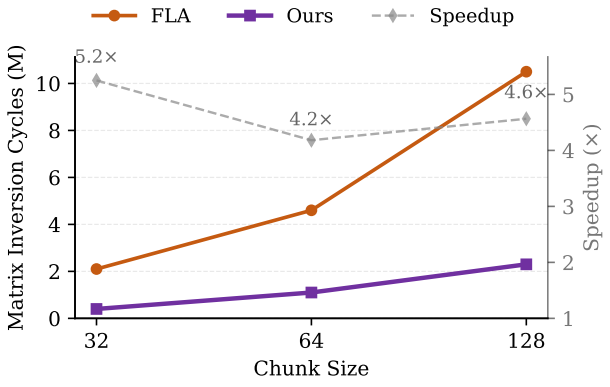

Figure 4. Plot of single kernel performance across different chunk wise. Here, H=32, D_k=128.

Combining the single-kernel results, we observe that as non-matmul operations become faster, the relative benefit of our matrix inversion method becomes more pronounced.

4.4. Ablation Study

Effect of each module. Table 4 ablates the key components of the Neumann-series approximation. In high precision, FP64 with $N = 64$ achieves near-perfect accuracy (178.42/96.46 dB mean/worst SNR). However, reducing precision to FP16 introduces severe instability, with the worst-case SNR dropping to -17.94 dB. Further truncation ($N = 3$) leads to substantial errors and divergence (-51.11 dB worst-case). Residual correction ($S = 8$) partially recovers mean accuracy (80.35 dB) but fails to stabilize worst-case behavior. In contrast, applying the diagonal mask markedly improves robustness, increasing mean/worst SNR to 86.91/47.98 dB. These results indicate that low-order approximation alone is insufficient, and that diagonal masking is essential for controlling worst-case errors (overflowing cases) under low-precision settings.

Table 4. Ablation study on Neumann-series approximation components.

METHOD	SNR_{mean}	SNR_{worst}
FP64 $N = 64$	178.42	96.46
⇒FP16	79.53	-17.94
⇒ $N = 3$	42.13	-51.11
+$S = 8$	80.35	-4.24
+DIAGONAL MASK	86.91	47.98

Order of Neumann series. Table 5 studies the interaction between Neumann truncation order N and residual correction steps S. Small N with insufficient correction leads to severe numerical instability, resulting in divergence (NaN) or extremely large error. Increasing S consistently stabilizes the approximation, with performance saturating around $S = 6-8$ for $N = 3$ and $N = 4$. Higher N requires fewer correction steps but worse quantization perforamnce once stability is achieved. These results suggest that INT16 quantization can't cover the dynamic range of $N = 4$. Based on these observation, we finally choose $N = 3, S = 8$ for 64×64 matrix inversion.

Table 5. Effect of Neumann series order n and residual steps s on numerical stability. Baseline W8A16 is 8.98.

	$N = 3$	$N = 4$	$N = 5$	$N = 6$
$S = 1$	NAN	NAN	NAN	469727.0
$S = 2$	NAN	NAN	62789.47	511.25
$S = 3$	NAN	NAN	1390.22	92.47
$S = 4$	NAN	169.30	61.31	144.95
$S = 5$	NAN	9.75	81.16	102.24
$S = 6$	23.56	9.54	63.52	135.83
$S = 7$	8.99	9.54	81.64	119.13
$S = 8$	**8.98**	9.54	66.03	117.79
$S = 9$	**8.98**	9.54	67.32	128.32

5. Conclusion

We identify matrix inversion in chunk-wise GatedDeltaNet as a critical bottleneck for long-context linear attention, particularly under low-precision inference. To address this, we propose a structure-aware, MatMul-based inversion algorithm that combines truncated Neumann expansion, diagonal masking, and parallel residual correction to eliminate sequential dependencies and improve numerical stability. Our method achieves up to 5× kernel speedup and 20% lower decode overhead, remaining robust under INT16 quantization and enabling efficient NPU deployment.

6. Impact Statement

This work improves the efficiency of large language models by enabling hardware-friendly, low-cost matrix inversion, facilitating deployment on resource-constrained devices such as mobile and edge platforms. By reducing compute and energy requirements, it can broaden access to long-context models and support more scalable, sustainable AI systems. However, increased accessibility of efficient LLMs may also lower the barrier for misuse, including large-scale automated content generation or deployment in sensitive domains without sufficient safeguards. Additionally, low-precision approximation may affect reliability under certain conditions. We emphasize the importance of careful evaluation and responsible deployment when applying such efficiency-oriented techniques.

References

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 7432–7439. AAAI Press, 2020. doi: 10.1609/AAAI.V34I05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

Clark, C., Lee, K., Chang, M., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 2924–2936. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1300. URL https://doi.org/10.18653/v1/n19-1300.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457, 2018. URL http://arxiv.org/abs/1803.05457.

Golub, G. H. and Van Loan, C. F. *Matrix Computations*. Johns Hopkins University Press, 2013.

Google DeepMind. Gemma 4: Frontier-level open models. https://deepmind.google/models/gemma/gemma-4/, 2026. Model card and technical documentation.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023. doi: 10.48550/ARXIV.2312.00752. URL https://doi.org/10.48550/arXiv.2312.00752.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Hua, W., Dai, Z., Liu, H., and Le, Q. V. Transformer quality in linear time. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S. (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, Proceedings of Machine Learning Research, pp. 9099–9117. PMLR, 2022. URL https://proceedings.mlr.press/v162/hua22a.html.

Huawei CSL. gdn-tri-inverse: Evaluation of gated delta networks with triangular matrix inversion. https://github.com/huawei-csl/gdn-tri-inverse, 2026.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research, pp. 5156–5165. PMLR, 2020. URL http://proceedings.mlr.press/v119/katharopoulos20a.html.

Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 26286–26296. IEEE, 2024. doi: 10.1109/CVPR52733.2024.02484. URL https://doi.org/10.1109/CVPR52733.2024.02484.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Peng, B., Zhang, R., Goldstein, D., Alcaide, E., Du, X., Hou, H., Lin, J., Liu, J., Lu, J., Merrill, W., Song, G., Tan, K., Utpala, S., Wilce, N., Wind, J. S., Wu, T., Wuttke, D., and Zhou-Zheng, C. RWKV-7 "goose" with expressive dynamic state evolution. *CoRR*, abs/2503.14456, 2025. doi: 10.48550/ARXIV.2503.14456. URL https://doi.org/10.48550/arXiv.2503.14456.

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

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL https://jmlr.org/papers/v21/20-074.html.

Siddegowda, S., Fournarakis, M., Nagel, M., Blankevoort, T., Patel, C., and Khobare, A. Neural network quantization with AI model efficiency toolkit (AIMET). *CoRR*, abs/2201.08442, 2022. URL https://arxiv.org/abs/2201.08442.

Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., and Wei, F. Retentive network: A successor to transformer for large language models. *CoRR*, abs/2307.08621, 2023. doi: 10.48550/ARXIV.2307.08621. URL https://doi.org/10.48550/arXiv.2307.08621.

Team, Q. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

xAI. Realworldqa: A real-world visual question answering benchmark. https://huggingface.co/datasets/xai-org/RealworldQA, 2024. Released with Grok-1.5 Vision.

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

Yang, S., Wang, B., Shen, Y., Panda, R., and Kim, Y. Gated linear attention transformers with hardware-efficient training. In Salakhutdinov, R., Kolter, Z., Heller, K. A., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pp. 56501–56523. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/yang24ab.html.

Yang, S., Kautz, J., and Hatamizadeh, A. Gated delta networks: Improving mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025a. URL https://openreview.net/forum?id=r8H7xhYPwz.

Yang, S., Shen, Y., Wen, K., Tan, S., Mishra, M., Ren, L., Panda, R., and Kim, Y. Path attention: Position encoding via accumulating householder transformations. *CoRR*, abs/2505.16381, 2025b. doi: 10.48550/ARXIV.2505.16381. URL https://doi.org/10.48550/arXiv.2505.16381.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In Korhonen, A., Traum, D. R., and Màrquez, L. (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/V1/P19-1472. URL https://doi.org/10.18653/v1/p19-1472.

Zhang, C., Lin, X., Jiang, H., Wang, Z., Li, X., Cao, Y., Zhuang, B., Men, R., Zhang, J., Zheng, B., Lin, J., Liu, D., and Zhou, J. Flashqla: Flash qwen linear attention. https://github.com/QwenLM/FlashQLA, 2026.

Zhang, Y., Lin, Z., Yao, X., Hu, J., Meng, F., Liu, C., Men, X., Yang, S., Li, Z., Li, W., Lu, E., Liu, W., Chen, Y., Xu, W., Yu, L., Wang, Y., Fan, Y., Zhong, L., Yuan, E., Zhang, D., Zhang, Y., Liu, T. Y., Wang, H., Fang, S., He, W., Liu, S., Li, Y., Su, J., Qiu, J., Pang, B., Yan, J., Jiang, Z., Huang, W., Yin, B., You, J., Wei, C., Wang, Z., Hong, C., Chen, Y., Chen, G., Wang, Y., Zheng, H., Wang, F., Liu, Y., Dong, M., Zhang, Z., Pan, S., Wu, W., Wu, Y., Guan, L., Tao, J., Fu, G., Xu, X., Wang, Y., Lai, G., Wu, Y., Zhou, X., Yang, Z., and Du, Y. Kimi linear: An expressive, efficient attention architecture. *CoRR*, abs/2510.26692, 2025. doi: 10.48550/ARXIV.2510.26692. URL https://doi.org/10.48550/arXiv.2510.26692.

Zhong, S., Xu, M., Ao, T., and Shi, G. Understanding transformer from the perspective of associative memory. *CoRR*, abs/2505.19488, 2025. doi: 10.48550/ARXIV.2505.19488. URL https://doi.org/10.48550/arXiv.2505.19488.

A. Related Works

Standard attention (Vaswani et al., 2017) has suffered from the quadratic time complexity to deal with the long context. Linear attention (Bisk et al., 2020) replaces the softmax operation with a positive feature map, enabling the attention computation to be reformulated as two associative matrix multiplications. This reformulation reduces the memory cost from sequence-length dependent to fixed-size, and lowers the computational complexity of similarity evaluation from $\mathcal{O}(T^2)$ to $\mathcal{O}(T)$. Recent linear-attention-based works construct a *Diagonal-Plus-Low-Rank* (DPLR) structure (Gu et al., 2022; Yang et al., 2025a; Zhang et al., 2025; Zhong et al., 2025; Yang et al., 2025b; Peng et al., 2025), defined as $D - ab^T$.During computation, this structured matrix is typically diagonalized in the complex domain via joint diagonalization, enabling efficient evaluation of matrix powers or exponentials. As such structure do calculation with recurrent, making the model hard to parallel. To solve this problem, Flash Linear Attention (Yang et al., 2024) introduces chunk-wise parallelism recursive inversion. Following this transformation, the DPLR-based models can be further reformulated into a chunk-wise parallel computation framework. FlahsQLA (Zhang et al., 2026) applies reasonable operator fusion and performance optimization to the forward and backward passes of GDN (Yang et al., 2025a) Chunked Prefill. However, these approaches do not address the bottleneck associated with matrix inversion. They are all use forward substitution to calculate the matrix inversion. As the matrix size grows, block-wise matrix inversion is employed to facilitate parallel computation, as described in Appendix F. DeltaFormer (Zhong et al., 2025) employs an algebraic factorization of the Neumann series to approximate matrix inversion, but overlooks the numerical stability issues under low-bit quantization. In contrast, GDN Tri-Inverse (Huawei CSL, 2026) provides a more robust implementation by explicitly accounting for such numerical constraints. Specifically, it restricts the maximum size of a single matrix inversion to 16×16 and leverages block-wise inversion to enable parallel computation.

In this work, we achieve numerically stable matrix inversion for matrices up to 64×64 and further extend the computation to INT16 precision. Compared to prior approaches, our method significantly enlarges the feasible inversion size while maintaining numerical robustness under low-bit quantization. This enables more efficient large-scale matrix operations, reduces the reliance on fine-grained block partitioning, and improves hardware utilization by aligning with integer-friendly accelerators. Consequently, our approach provides a practical foundation for scalable and low-power deployment of linear-attention-based models on edge devices.

B. Details of Gated DeltaNet

Gated DeltaNet maintains a matrix-valued recurrent state $S_t \in \mathbb{R}^{d_k \times d_v}$ and updates it through a gated delta rule. For token t, the state transition and output computation are defined as

$$S_t = \alpha_t S_{t-1} - \beta_t S_{t-1} k_t k_t^\top + \beta_t v_t k_t^\top, \tag{14}$$

$$y_t = S_t q_t, \tag{15}$$

where q_t, k_t, and v_t denote the query, key, and value vectors, respectively. The scalar gate α_t controls global state decay, while β_t modulates the strength of the delta-rule update induced by the current key-value pair. The second term in Eq. (14) removes the component of the previous memory associated with k_t, whereas the third term writes the new value information into the same key direction. Therefore, the update can be interpreted as a gated memory rewrite mechanism that combines exponential forgetting with associative state correction.

Although the token-wise formulation in Eq. (14) is memory efficient, it imposes a sequential dependency over tokens. To expose parallelism, the recurrence can be reformulated in a chunk-wise manner. Let the input sequence be partitioned into chunks of length C, and denote by $S^{[i]}$ the recurrent state before processing the i-th chunk. The composition of token-wise delta updates within a chunk can be expressed as a block-level transition

$$S^{[i+1]} = S^{[i]} + U^{[i]} T^{[i]} - W^{[i]} T^{[i]} S^{[i]} K^{[i]}, \tag{16}$$

where $K^{[i]}$, $U^{[i]}$, and $W^{[i]}$ are chunk-level matrices constructed from the keys, values, and gates inside the i-th chunk. The matrix $T^{[i]}$ captures the cumulative causal interaction among token-level updates within the chunk and is given by

$$T^{[i]} = \left(I - A^{[i]}\right)^{-1}, \tag{17}$$

where

$$A^{[i]} = \operatorname{tril}\left(-\operatorname{diag}(\beta^{[i]}) K^{[i]} {K^{[i]}}^\top, -1\right). \tag{18}$$

Here, $\mathrm{tril}(\cdot, -1)$ extracts the strictly lower-triangular part, ensuring that the chunk-level operator respects causal ordering.

This chunk-wise formulation preserves the exact semantics of the original token-wise recurrence while converting the intra-chunk computation into structured matrix operations. In particular, all quantities depending only on tokens within the same chunk can be computed locally, while cross-chunk dependence is carried solely by the recurrent state $S^{[i]}$. Consequently, the sequential depth is reduced from token-level recurrence over the full sequence to state propagation across chunks. For a sequence of length L and chunk size C, the number of inter-chunk recurrent steps is reduced to approximately L/C, making the formulation substantially more amenable to parallel prefill computation.

Overall, the gated delta rule provides a recurrent memory update with selective forgetting and rewriting, while the chunk-wise formulation lifts the recurrence into a block-structured computation. This representation is particularly useful for efficient linear-attention implementations, since it retains causal recurrent behavior while exposing parallel matrix operations within each chunk.

C. Truncation Error of Finite Neumann Expansion

Proof. Since A is strictly lower triangular, it is nilpotent and satisfies $A^k = 0$, hence

$$(I - A)^{-1} = \sum_{n=0}^{k-1} A^n. \tag{19}$$

Therefore,

$$E_m = \sum_{n=m+1}^{k-1} A^n. \tag{20}$$

Applying the triangle inequality and submultiplicativity yields

$$\|E_m\| \le \sum_{n=m+1}^{k-1} \|A^n\| \le \sum_{n=m+1}^{k-1} \|A\|^n. \tag{21}$$

If $\|A\| < 1$, the finite sum is further bounded by the tail of a convergent geometric series:

$$\sum_{n=m+1}^{k-1} \|A\|^n \le \sum_{n=m+1}^{\infty} \|A\|^n = \frac{\|A\|^{m+1}}{1 - \|A\|}. \tag{22}$$

This completes the proof. □

Empirical validation (chunk size $k = 64$). We empirically validate the error with chunk size $k = 64$. We construct a strictly lower-triangular matrix $A \in \mathbb{R}^{64\times 64}$ whose entries decay with the distance to the main diagonal (to mimic diagonal-concentrated structure), and rescale it to satisfy $\|A\|_2 = 0.55 < 1$. We compute the exact inverse via the finite expansion $T = \sum_{n=0}^{63} A^n$ and the truncated approximation $T_m = \sum_{n=0}^{m} A^n$ for $m = 0, 1, \ldots, 63$. We report truncation errors in Frobenius and spectral norms, $\|E_m\|_F$ and $\|E_m\|_2$, and compare them with the geometric upper bound $\|A\|_2^{m+1}/(1-\|A\|_2)$ from Lemma 3.2.

Results. Figure 5 shows that truncation errors decrease rapidly with m and enter a near-flat regime far before the full expansion order $k - 1$. Using a simple captured-structure proxy $r_m = 1 - \|E_m\|_F/\|T\|_F$, we observe that $m = 3$ already captures $\approx 95.5\%$ of the inverse structure, $m = 5$ captures $\approx 98.8\%$, and $m = 8$ captures $\approx 99.8\%$ in this $k = 64$ study.

Real value experiments. Based on Lemma 3.1, the truncation error can be efficiently estimated using the accumulated power ratio along the diagonal dimension. We empirically measure this accumulated power ratio on the Qwen3-NextA3B-80B (Team, 2025) model across different linear attention layers. As shown in Fig. 3, the accumulated power ratio exhibits a consistent trend across layers: it increases rapidly with the order nnn and approaches saturation. In most layers, more than 98% of the power is captured within a relatively small number of steps ($n \approx 10$–20), while achieving a higher threshold (e.g., 0.99) typically requires significantly larger orders ($n \approx 30$–50). Notably, there exists substantial layer-wise variation,

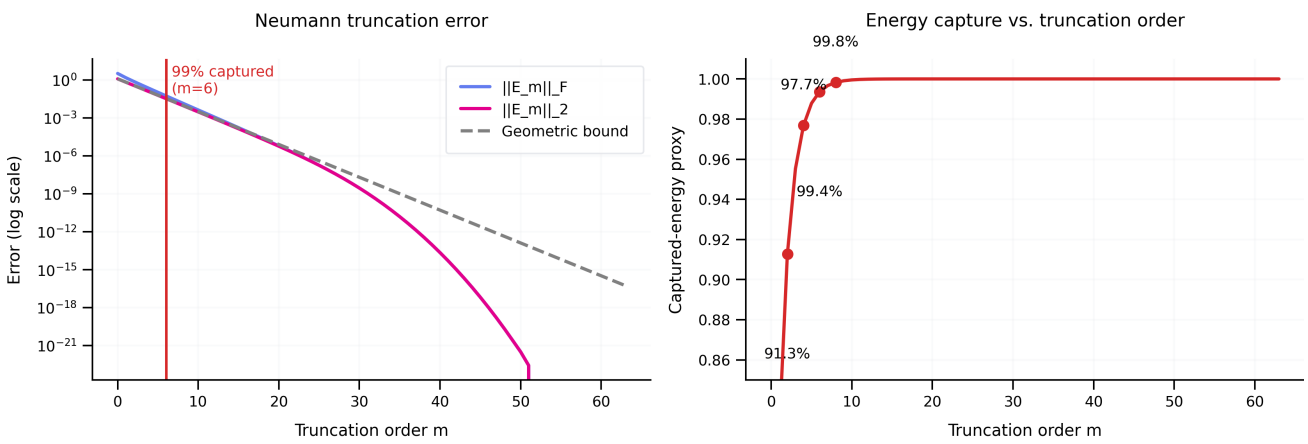

Figure 5. **Low-order truncation is sufficient for a strictly lower-triangular 64×64 example.** Left: truncation errors $\|E_m\|_F$ and $\|E_m\|_2$ versus truncation order m, together with the geometric-series upper bound $\|A\|_2^{m+1}/(1-\|A\|_2)$. Right: a captured-structure proxy $1 - \|E_m\|_F/\|T\|_F$. Errors drop sharply within the first few orders, indicating that most inverse structure is captured by $m \ll k$, making full expansion unnecessary in practice.

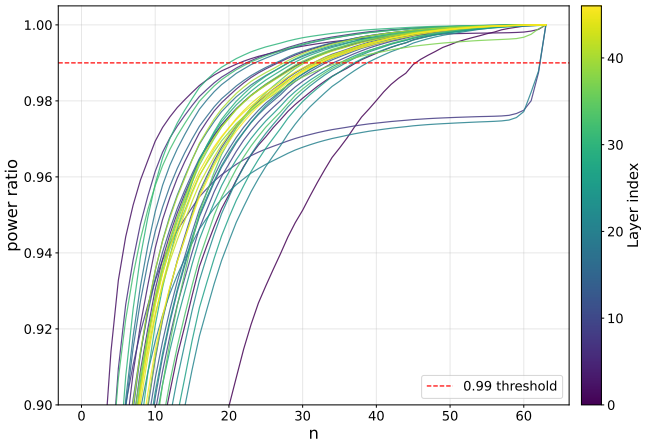

Figure 6. Accumulated diagonal power ratio across layers. The ratio saturates quickly with increasing n; while ¿98% is typically captured at small n, achieving 0.99 requires substantially larger orders, revealing layer-wise variation and the cost–accuracy trade-off.

with some layers converging more slowly and requiring much larger nnn to reach the same threshold. These results indicate that, although the Neumann series converges in principle, achieving high-accuracy approximation uniformly across layers demands a large truncation order. This further reinforces the practical limitation of using high-order truncation, as it not only increases computational cost but also exacerbates numerical instability and quantization issues discussed earlier.

Although $(I - A)^{-1}$ admits an exact finite expansion for strictly lower-triangular A, both our analysis (Lemma 3.2) and empirical validation (Figure 5) show that a much lower truncation order already captures most of the inverse structure. We therefore use a low-order truncated expansion as an initializer and recover the remaining tail error via residual correction.

D. Proof of Lemma 3.3

Proof. By expanding the matrix product,

$$(A^k)_{ij} = \sum_{j<t_1<\cdots<t_{k-1}<i} A_{i,t_{k-1}} A_{t_{k-1},t_{k-2}} \cdots A_{t_1,j},$$

where strict lower-triangularity enforces strictly increasing intermediate indices. Since each factor satisfies $0 \le A_{pq} \le 1$, every product term is bounded above by 1. The number of admissible index sequences equals the number of ways to choose $k-1$ indices from the $d-1 = i-j-1$ integers between j and i, yielding $\binom{d-1}{k-1}$.

Tightness follows by choosing $A_{pq} = 1$ for all $p > q$, in which case every product term equals 1 and the bound is attained with equality. □

Corollary D.1 (Worst-Case Characterization)**.** *Under the assumptions of Lemma 3.3,*

$$\max_{0 \le A_{ij} \le 1} (A^k)_{ij} = \binom{d-1}{k-1}.$$

Thus, the binomial coefficient gives the exact worst-case growth of individual entries of A^k, depending only on the diagonal distance $d = i - j$ and the power k.

E. Proof of reformulation for Residual correction

Proof. By definition, $(I - A)T^{(0)} = I - E$. Since M is nonsingular and $\rho(E) < 1$, the matrix $(I - E)$ is invertible and

$$T^{(0)} = (I - A)^{-1}(I - E) \quad \Rightarrow \quad (I - A)^{-1} = T^{(0)}(I - E)^{-1}. \tag{23}$$

When $\rho(E) < 1$, the Neumann series converges:

$$(I - E)^{-1} = \sum_{s=0}^{\infty} E^s. \tag{24}$$

Thus $M^{-1} = T^{(0)} \sum_{s=0}^{\infty} E^s$. For truncation at S, the remainder is

$$M^{-1} - T^{(S)} = T^{(0)} \sum_{s=S+1}^{\infty} E^s = T^{(0)} E^{S+1} \sum_{t=0}^{\infty} E^t. \tag{25}$$

Taking norms and using submultiplicativity gives

$$\|M^{-1} - T^{(S)}\| \le \|T^{(0)}\| \cdot \|E\|^{S+1} \sum_{t=0}^{\infty} \|E\|^t = \|T^{(0)}\| \cdot \frac{\|E\|^{S+1}}{1 - \|E\|}, \tag{26}$$

which completes the proof. □

F. Block-wise Matrix Inversion for Chunk-wise Processing

Let $T \in \mathbb{R}^{n \times n}$ be a strictly lower triangular matrix arising from chunk-wise Gated DeltaNet,

$$T = I - A, \qquad A_{ij} = 0 \ \ \forall\, i \le j. \tag{27}$$

We partition T into K blocks of size $b \times b$:

$$T = \begin{bmatrix} T_{11} & 0 & \cdots & 0 \\ T_{21} & T_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ T_{K1} & T_{K2} & \cdots & T_{KK} \end{bmatrix}. \tag{28}$$

The inverse preserves the same block lower-triangular structure,

$$T^{-1} = \begin{bmatrix} S_{11} & 0 & \cdots & 0 \\ S_{21} & S_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ S_{K1} & S_{K2} & \cdots & S_{KK} \end{bmatrix}. \tag{29}$$

Each diagonal block is inverted independently:

$$S_{ii} = T_{ii}^{-1}, \qquad i = 1, \ldots, K. \tag{30}$$

For off-diagonal blocks ($i > j$), block-wise forward substitution satisfies

$$\sum_{k=j}^{i} T_{ik} S_{kj} = 0, \tag{31}$$

which yields the recursion

$$S_{ij} = -T_{ii}^{-1} \sum_{k=j}^{i-1} T_{ik} S_{kj}, \qquad i > j. \tag{32}$$

In particular, the first off-diagonal block is given by

$$S_{i,i-1} = -T_{ii}^{-1} T_{i,i-1} T_{i-1,i-1}^{-1}. \tag{33}$$

In chunk-wise Gated DeltaNet, each block corresponds to one chunk, and the output is computed as

$$Y = T^{-1} X, \tag{34}$$

using block-wise inversion (Eq. 4) and recursive accumulation (Eq. 6).

G. Why Algebraic Factorization Does Not Help on NPUs.

While factorizing the Neumann series reduces algebraic depth, it fundamentally alters the execution graph. The standard Neumann iteration computes powers of A via the recurrence

$$X_{k+1} = X_k A, \tag{35}$$

which forms a single linear dependency chain. Each iteration requires one matrix multiplication, maintains a single live accumulator, and admits kernel fusion across steps. This execution pattern is well aligned with NPU architectures, which are optimized for narrow, linear dependency graphs. In contrast, factorized forms compute partial sums of the form

$$G_n = \sum_{i=0}^{2^n-1} X_i, G_n = G_{n-1}(I + A^{2^{(n-1)}}), \tag{36}$$

where the term $I + A^{2^{(n-1)}}$ must be materialized prior to its multiplication with G_{n-1}. Consequently, each update introduces a two-stage dependency: first for power accumulation (to obtain A^{2^k}), and second for the subsequent multiplication. This effectively doubles the dependency width, increasing the number of live intermediate tensors and hindering fusion into a single accumulator chain.

On NPUs, such widened dependency graphs lead to higher scheduling overhead, increased memory traffic, and reduced pipeline utilization. Consequently, despite reduced algebraic depth, the factorized Neumann form incurs strictly higher execution cost than the unfactorized series.

H. Quantization Experiment

To further improve latency and power efficiency, we apply integer-only (INT) activation quantization to the chunk-wise Gated DeltaNet. We adopt a standard uniform quantization operator

$$q(x) = clip(round(\frac{x}{\Delta}), qmin, qmax) \tag{37}$$

where Δ denotes the quantization scale. Depending on the hardware implementation, asymmetric quantization maps activations to the range $[0, 2^b - 1]$, while symmetric quantization uses a signed range of $[-2^{b-1}, 2^{b-1} - 1]$, with b indicating the bit width (e.g., $b = 8$ or 16).

H.1. Quantization Settings

We summarize the quantization setting in the following table. For visual model, we do plain calibration with 25 image samples from LLaVA-COCO dataset (Liu et al., 2024). For text model, we use AdaScale from AIMET (Siddegowda et al., 2022) with 1000 samples from C4 (Raffel et al., 2020).

Table 6. W4A16 quantization configuration for Qwen3.5-4B. Decoder weights are quantized to INT4 with per-channel scaling, while all activations are retained in INT16.

Module	**Weight**	**Activation**	**Quantization Scheme**
Decoder (LM)	INT4	INT16	Per-channel, symmetric
Matrix Inversion	–	INT16/INT8	Per-tensor
Embedding	INT16	INT16	Per-tensor, asymmetric
LM Head	INT8	INT16	Per-channel, symmetric
Normalization	INT16	INT16	Per-tensor, asymmetric
Vision Encoder (VLM)	INT8	INT16	Per-channel, symmetric
Other Ops (Mul, Add)	–	INT16	Per-tensor, asymmetric/symmetrics

H.2. Matrix inversion: INT16 vs. INT8

The proposed method effectively addresses the numerical stability issues in matrix inversion, to maintain a compact dynamic range, making it well for INT quantization. In the experiment, we further evaluate INT8 setting for matrix inversion, where keeping all others consistent with the W4A16 setting. Table 7 compares INT16 and INT8 quantization for matrix inversion. The results show that INT8 achieves comparable perplexity and downstream accuracy to INT16, demonstrating that our method remains robust even under lower-bit integer quantization.

Table 7. INT8 VS INT16 comparison in matrix inversion, CS=128

MODEL	METHOD	MATRIX INVERSE	PPL	IF-EVAL
QWEN3.5 4B	FLA-FP	FP16	8.89	0.83
	OURS-W4A16	INT16	9.71	0.78
	OURS-W4A16	INT8	9.71	0.77

I. Matrix inversion for 32×32 matrix and 128 × 128 matrix

For the 32×32 matrix, we further evaluate the impact of Neumann series order and iteration steps. As shown in Table 8, the 32×32 matrix exhibits a similar trade-off to the 64×64 case. Due to its smaller spectral magnitude, the Neumann series for the 32×32 matrix converges more rapidly. Consequently, the optimal configuration of $S = 3$ steps and $N = 4$ order in FP16 also generalizes well to the INT16 setting.

Table 8. Effect of Neumann series order n and residual steps s on numerical stability. Baseline FP16 is 8.89.

	$N = 1$	$N = 2$	$N = 3$	$N = 4$
$S = 1$	NAN	NAN	NAN	9.902
$S = 2$	NAN	NAN	9.22	8.888
$S = 3$	NAN	NAN	8.895	**8.890**
$S = 4$	NAN	8.900	**8.890**	**8.890**
$S = 5$	NAN	8.891	**8.890**	**8.890**
$S = 6$	NAN	**8.890**	**8.890**	**8.890**
$S = 7$	NAN	**8.890**	**8.890**	**8.890**
$S = 8$	8.891	**8.890**	**8.890**	**8.890**

For the 128×128 matrix, according to Lemma 3.3, even with $N = 3$, the worst-case value can grow to 341,376, which exceeds the dynamic range of both FP16 and INT16, leading to severe numerical instability. To mitigate this issue, we adopt

a block-wise matrix inversion strategy, decomposing the problem into smaller sub-matrices with a minimum size of 64×64.

J. Ablation on Different context length

We evaluate WikiText-v2 on Qwen3.5-4B across context lengths from 4k to 32k to assess the effectiveness of our method. In FP32, our approach exactly matches the FLA baseline across all context lengths, demonstrating numerical equivalence without any degradation. Under INT16 quantization, it achieves comparable performance at short contexts and consistent improvements at longer contexts. At 32k, increasing the chunk size to CS=128 reduces perplexity from 9.201 to 8.998 (-0.203), significantly narrowing the FP32–INT16 gap. Notably, the impact of chunk size is context-dependent: while CS=128 slightly degrades performance at $\leq$8k, it becomes beneficial beyond 16k, suggesting improved numerical stability in the long-context regime.

Table 9. Perplexity comparison across context lengths on WikiText-v2. FP32 results show identical performance between naïve and our method. Quantized results (INT16) demonstrate that our method consistently improves perplexity, with larger chunk sizes (CS=128) yielding further gains at long context (e.g., 32k).

Context	FP32	FP32 (Ours)		Quantized (INT16)	
		CS=64	CS=128	Naïve (CS=64)	Ours (CS=64 / 128)
4k	8.89	8.89	8.89	9.658	9.664 / 9.717
8k	8.66	8.66	8.66	9.373	9.366 / 9.405
16k	8.46	8.46	8.46	9.216	9.197 / 9.179
32k	8.41	8.41	8.41	9.201	9.188 / 8.998