Recovering Selectivity with LTI State Space Operators for Portable Long-Context Inference

Minseon Gwak [1 2] **N. Benjamin Erichson** [1 2] **PooGyeon Park** [3]

Abstract

Mamba has established selective state space models as powerful building blocks for long-context foundation models by replacing linear time-invariant (LTI) recurrences with input-dependent selective recurrences. However, efficient training of such recurrences often relies on specialized fused kernels, limiting portability across diverse hardware. We propose SelectLTI, a simple but effective block that augments an LTI recurrence with input-dependent modulation to recover selective behavior. The proposed block equips any LTI state-space layer with sigmoid-gated modulators, making it trainable and deployable without specialized fused kernels. We evaluate SelectLTI on Selective Copying, a task designed to isolate selective-memory behavior in sequence models. SelectLTI matches or exceeds S6, the selective state-space operator of Mamba, showing that selective behavior can be recovered from an LTI recurrence augmented with lightweight modulation using only standard primitives. We position SelectLTI as a portable building block for efficient long-context foundation models deployed across diverse compute environments.

1. Introduction

Long-horizon sequence modeling requires memory systems that are expressive, stable, and computationally efficient. This requirement is becoming more important as foundation models move from a single cloud setting to heterogeneous inference environments (Bommasani et al., 2021): large accelerators in datacenters, smaller workstations in laboratories, and edge devices attached to sensors. For resource-adaptive inference, a practical sequence backbone should therefore provide not only strong accuracy but also an efficient and portable deployment path under varying hardware constraints.

State space models (SSMs) have emerged as promising long-context backbones since they can process sequences in linear time while maintaining compact recurrent states (Gu et al., 2021; 2022; Smith et al., 2022; Gu & Dao, 2023; Dao & Gu, 2024). However, once trained, classical linear time-invariant (LTI) SSMs apply the same dynamics across timesteps and inputs; as a result, their memory update lacks a selective mechanism for deciding which tokens to write, suppress, or retain. Mamba addresses this limitation by using an input-dependent S6 operator that generates selective state-space parameters at each step (Gu & Dao, 2023). This content-aware recurrence is a central reason why selective SSMs are attractive for foundation models: the current token can control how information enters and exits the memory.

At the same time, this mechanism often introduces an implementation bottleneck. Efficient training and inference of selective recurrences typically rely on specialized kernels, such as custom CUDA kernels, to optimize computation and memory movement on specific hardware. (Gu & Dao, 2023; Dao et al., 2022; Dao, 2023; Yang et al., 2023). While such kernels can be highly effective on the accelerator for which they are written, they complicate deployment on the other environments. In other words, a selective backbone may be algorithmically linear time, yet still difficult to use as a general resource-adaptive foundation model component.

To this end, we revisit selective SSM design through the lens of classical physical modeling. Hammerstein, Wiener, and Hammerstein-Wiener systems represent nonlinear dynamical processes by placing memoryless nonlinearities around a linear dynamical core (Khalil & Grizzle, 2002; Giri & Bai, 2010; Giri et al., 2009). This pattern appears in physical systems where the memory dynamics are structured by transport, diffusion, oscillation, or filtering, while nonlinearities enter through saturation, clipping, dead zones, or readout effects. The analogy suggests a practical alternative to generating a new recurrent system at every timestep: keep an LTI memory operator and place lightweight input-dependent modulators around it.

[1]Lawrence Berkeley National Laboratory [2]International Computer Science Institute [3]Pohang University of Science and Technology. Correspondence to: PooGyeon Park <ppg@postech.ac.kr>.

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

We instantiate this idea as SelectLTI, a portable selective SSM block. SelectLTI augments any LTI state-space operator, such as S4, S4D, or S5, with small sigmoid-gated bottleneck modulators at the input and optionally at the output. This design preserves the standard implementation of the LTI backbone while recovering the selective behavior needed for long-context foundation models.

On Selective Copying, SelectLTI consistently upgrades LTI backbones; with S5, validation accuracy improves from 48.88% to 94.90% while retaining standard-primitive throughput above 10M tokens/s. The resulting block matches or exceeds S6 accuracy, whereas an unfused S6 implementation is substantially slower. These results support our goal of building a selective sequence modeling block that is easier to deploy across resource budgets and hardware backends.

2. Selective SSMs and Portability

LTI SSM backbones. An LTI SSM maps an input sequence $u_k \in \mathbb{R}^H$ to an output sequence $y_k \in \mathbb{R}^H$ through a recurrent state $x_k \in \mathbb{R}^N$ with fixed system matrices,

$$x_{k+1} = Ax_k + Bu_k, \qquad y_k = Cx_k + Du_k. \tag{1}$$

Since A, B, C, D do not depend on the token or absolute timestep, the same memory operator can be implemented by recurrent scans, convolutional kernels, or transfer-function-based formulations (Gu et al., 2021; 2022; Smith et al., 2022; Parnichkun et al., 2024; Poli et al., 2023). This is attractive for efficient foundation models, but fixed LTI memory can struggle when a task requires content-aware writing and forgetting.

Selective recurrences. Mamba-style SSMs address this gap by allowing the token to control the recurrent coefficients. A simplified view of S6 is

$$x_{k+1} = A_k x_k + B_k u_k, \qquad y_k = C_k x_k, \tag{2}$$

where the coefficients are generated from the current input. This construction makes the memory selective: two tokens appearing at the same location can induce different write and read behavior. However, the input-dependent recurrence is difficult to realize efficiently with a naive implementation. In practice, feasible training is achieved by specialized fused kernels that reduce memory movement and exploit hardware-specific parallelism. The resulting implementation is powerful, but it also ties the selective backbone to a narrower software-hardware path.

A resource-adaptive view. The relevant question is not only whether a selective backbone can be fast on a single accelerator, but whether it provides a useful quality-speed trade-off across deployment budgets. A fused CUDA path may be the best choice when the target hardware and software stack are fixed. A portable standard-primitive path becomes more attractive when the same foundation model must run on different accelerators or hardware where custom kernels are unavailable.

3. SelectLTI for Portable Long-Context Inference

Selective recurrences obtain content awareness by allowing each token to induce a different effective state-space system. In S6, for example, input-conditioned system coefficients make the effective transition and input/output maps vary with the time index, so the model no longer reduces to the standard LTI convolution or fixed-coefficient scan used by classical SSMs. However, this flexibility is expressive, requiring efficient implementations that carefully control memory movement. For portable long-context inference, we therefore seek a middle ground: retain the LTI state-space propagation so that standard implementations remain usable, while adding enough input dependence to recover selective behavior.

SelectLTI realizes this middle ground by keeping the LTI memory propagation in Equation (1) and adding lightweight input-dependent modulation around it. Given an input token u_k, the block computes

$$\begin{aligned} z_k &= u_k \odot g_\phi^{\text{in}}(u_k), \\ x_{k+1} &= Ax_k + Bz_k, \\ \hat{y}_k &= Cx_k + Dz_k, \\ y_k &= \hat{y}_k \odot g_\psi^{\text{out}}(\hat{y}_k), \end{aligned} \tag{3}$$

where (A, B, C, D) are supplied by any LTI SSM core and g_ϕ^{in}, g_ψ^{out} are memoryless modulators. The output modulator is optional; in our experiments it is used when the backbone lacks a comparable nonlinear operation.For parameter efficiency, each modulator is designed as a two-layer bottleneck module,

$$g_\phi(u) = W_2\,\sigma(W_1 u + b_1) + b_2, \qquad R = \text{rows}(W_1) \ll H, \tag{4}$$

followed by elementwise multiplication denoted by $\odot$. Here $W_1 \in \mathbb{R}^{R\times H}$ and $W_2 \in \mathbb{R}^{H\times R}$ are learnable projection matrices, and $b_1 \in \mathbb{R}^R$ and $b_2 \in \mathbb{R}^H$ are learnable biases. The function $\sigma(\cdot)$ denotes a sigmoid nonlinearity, and R is the bottleneck rank.

Thus, $g_\phi(u) \in \mathbb{R}^H$ acts as a feature-wise input-dependent gate applied by elementwise multiplication. The input and output modulators share this bottleneck form but use separate parameters, denoted by ϕ and ψ. The sigmoid nonlinearity provides a simple smooth modulating mechanism, allowing the block to suppress, amplify, or reweight sequence

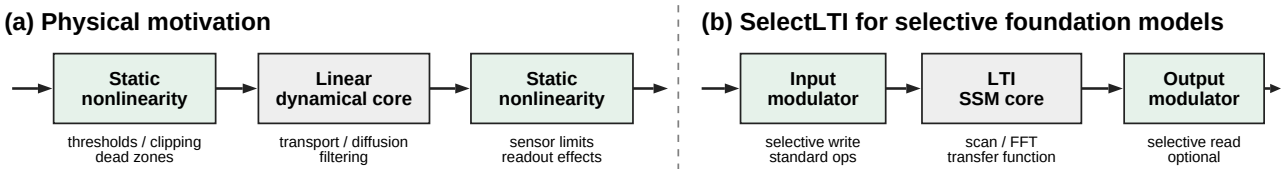

Figure 1. Physical motivation and architecture. Hammerstein-Wiener systems use static nonlinearities around a linear dynamical core. SelectLTI adopts this pattern for selective foundation model backbones by adding lightweight modulators around a fixed LTI SSM core, retaining standard tensor primitives rather than a specialized selective-scan kernel.

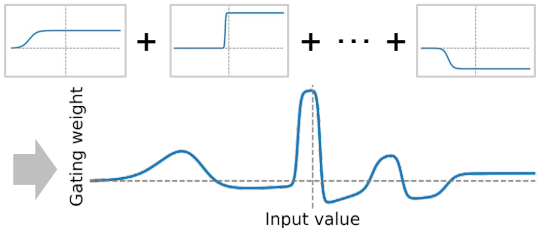

Figure 2. Sigmoid-gated bottleneck modulator. The one-dimensional view illustrates that a rank-R modulator can form a smooth input-dependent gain by combining shifted sigmoid functions.

content before it enters the memory core, and optionally before the state-space output is passed to subsequent layers.

Figure 2 visualizes the modulator in the one-dimensional case. In the case, Equation (4) reduces to a linear combination of sigmoid basis functions,

$$g(u) = \sum_{i=1}^{R} a_i \, \sigma(\alpha_i u + \beta_i) + \gamma, \tag{5}$$

where each hidden unit contributes a soft transition and the bottleneck rank R controls the number of such transitions. This gives a compact way to produce input-dependent gains: small values of R provide a smooth gate, while larger values of R allow more flexible patterns.

In the full vector-valued block, the same computation is applied through two dense projections and a pointwise sigmoid nonlinearity. The resulting gain vector is multiplied elementwise with the signal entering or leaving the LTI state-space core. Importantly, the modulator is *memoryless*: it depends only on the current token or current pre-output, and does not introduce an input-dependent recurrent transition. Thus, all temporal memory remains in the LTI backbone, while selectivity is supplied by a small standard-primitive path consisting of dense projections, sigmoid activations, and elementwise products.

The input and output modulators correspond to two intuitive forms of selectivity. The input modulator acts as a selective write mechanism: it determines which parts of the current token should enter the long-range memory. The optional output modulator acts as a selective read mechanism: it reshapes the signal exposed to later layers or a decoder. Crucially, this selectivity is introduced without changing the recurrent coefficients of the LTI core. Existing scan, convolution, or transfer-function implementations of the backbone can therefore remain intact.

This design gives simple resource knobs. A practitioner can choose the LTI backbone, use only the input modulator for a cheaper block, add the output modulator for richer selectivity, or lower the bottleneck rank R to reduce adaptive capacity and compute. These knobs are useful for resource-adaptive foundation model inference since they expose a continuum between a pure LTI backbone and a more expressive selective block. They are also easy to combine with standard compression, quantization, or pruning methods, since the modulators are small dense modules rather than custom recurrent kernels.

We emphasize the scope of the contribution. SelectLTI is not meant to replace every optimized selective-kernel implementation. On a GPU with a mature fused path, S6 can still provide excellent throughput. Instead, SelectLTI offers a standard-primitive alternative when portability, extensibility, or heterogeneous deployment is important.

4. Related Work

Long-context foundation model backbones. Efficient long-context modeling has developed along several complementary directions. Structured SSMs such as S4, S4D, and S5 exploited fixed linear dynamics to obtain long-range memory with scan or convolutional implementations (Gu et al., 2021; 2022; Smith et al., 2022). Hyena and transfer-function formulations similarly used long filters or frequency-domain views to obtain efficient sequence processing (Poli et al., 2023; Parnichkun et al., 2024). Mamba showed that selective SSMs can serve as strong foundation model backbones by combining recurrent efficiency with input-dependent memory updates (Gu & Dao, 2023; Dao & Gu, 2024). Our work keeps the long-context motivation of this line, but asks whether part of the selective behavior can be recovered without making the recurrent coefficients themselves input-dependent.

Gating and selective memory. Input-dependent gating has a long history in recurrent and convolutional sequence models, including LSTM- and GRU-style mechanisms, and gated convolutions (Dauphin et al., 2017). Recent foundation model architectures also use gating mechanisms in linear attention, RWKV-style recurrence, gated linear attention, gated delta networks, and gated attention variants (Katharopoulos et al., 2020; Peng et al., 2023; Yang et al., 2023; 2024; Qiu et al., 2025). SelectLTI follows a similar principle: it uses input-dependent multiplicative modulation to regulate information flow. The difference is architectural placement and modulation form: SelectLTI wraps the modulator around an LTI SSM core, rather than making the recurrent transition itself input-dependent, so the memory operator remains compatible with standard LTI implementation paths. Moreover, although the bottleneck uses sigmoid activations, the final modulation vector is an affine combination of sigmoid features and is not constrained to the interval $[0, 1]$. It should therefore be interpreted as a learned modulation signal rather than a purely attenuating gate.

Hardware-aware efficiency and portability. Many efficient foundation model systems obtain speed by matching algorithms to hardware memory hierarchies. FlashAttention is a canonical example for attention, and selective SSM implementations use a similar philosophy by fusing operations to reduce memory traffic (Dao et al., 2022; Dao, 2023; Gu & Dao, 2023). Such optimization is valuable; however, broad adoption requires models that can generalize across diverse resource constraints. SelectLTI therefore complements hardware-aware kernels: it provides a portable selective operator that remains usable with standard primitives.

5. Empirical Validation

Task. We evaluate SelectLTI on Selective Copying (Jing et al., 2019), a task used by Mamba to isolate the need for selectivity (Gu & Dao, 2023). Each input contains long stretches of noise, 16 target tokens inserted at random positions within a length-4096 prefix, and marker tokens appended at the end. Since the model must recover the targets, performance requires both *long-range memory* and *content-aware filtering*. Mamba attributes the failure of LTI SSMs on this benchmark to their lack of selectivity, identifying the selective property of S6 as the key driver of its success.

Although this task is synthetic, it captures a minimal structure that appears in many long-context foundation model workloads: a long sequence contains mostly irrelevant content, a few task-relevant events must be retained, and a later query asks the model to recall those events. Document-level reasoning has this form when key facts are embedded in long text, and scientific foundation models often face the same pattern when rare events appear in long sensor streams or simulation trajectories. The benchmark therefore isolates the selective memory operation that a long-context backbone must provide.

Baselines. All models use a token embedding layer, two SSM layers with channel dimension 64, and a linear decoder. We train for 400K steps with batch size 64 and learning rate 10^{-3} across three random seeds. S4 and S4D use their standard gated linear unit (GLU)-based channel mixing (Dauphin et al., 2017); for these backbones we add only the input modulator. S5 uses SiLU activation (Ramachandran et al., 2017); for S5 we add both input and output modulators.

For S6, we isolate the selective state-space core used inside a Mamba block. That is, the S6 baseline does not include the surrounding Mamba components such as input expansion, convolution, or output projection. This keeps the comparison focused on the state-space memory operator. Throughput is measured for the full model with sequence length 4128 and batch size 64 on a single NVIDIA A6000.

Results. Table 1 shows that lightweight modulation substantially improves LTI SSM backbones on the benchmark. The largest gain appears with S5: accuracy rises from 48.88% to 94.90%, slightly exceeding the S6 baseline at 93.44%. The same model maintains 12M tokens/s with standard primitives, whereas unfused S6 reaches only 404K tokens/s. S4 and S4D also improve by 12.41 and 9.90 percentage points, respectively, even though their baselines already contain GLU-based channel mixing. These results indicate that input-dependent modulation around the memory core provides a useful and efficient form of selectivity.

Notably, our S4 and S4D baselines exhibit significantly more robustness than the 'very weak' LTI baseline reported in Gu & Dao (2023). We hypothesize that this discrepancy arises from architectural configurations. This makes the comparison more conservative: our baseline already contains a nonlinearity, yet an explicit input modulator still improves accuracy. The S5 result is more revealing since the baseline has no comparable gate. There, adding the two small modulators changes the model from a poor selective-copying solution into one that matches S6 accuracy. This suggests that the gain is not simply from adding parameters, but from placing a small input-conditioned gain exactly where the memory write and read decisions occur.

Resource Efficiency and Adaptability. The improvement also comes without increasing the state dimension of the LTI backbone. For S5, the parameter count increases from 10K to 15K, while throughput remains in the same standard-kernel regime. This is important for foundation model scaling since the state dimension and memory traffic of sequence layers often dominate long-context cost. A

Table 1. Selective Copying results. SelectLTI adds a small number of parameters through sigmoid-gated modulators and retains high throughput using standard primitives. S6 is shown with and without the specialized fused path. Overall, SelectLTI achieves a favorable accuracy–throughput trade-off, recovering selective behavior while avoiding the kernel dependency of S6.

Backbone	Selective mechanism	Use custom kernel?	Params.	Accuracy / Throughput
S6	Input-dependent dynamics	No	9K	93.44% / 404K tok/s
S6	Input-dependent dynacmis	Yes	9K	93.44% / 24M tok/s
S4	None	No	21K	81.78% / 11M tok/s
S4	SelectLTI (Input modulator)	No	23K	**94.19%** / 10M tok/s
S4D	None	No	21K	77.68% / 13M tok/s
S4D	SelectLTI (Input modulator)	No	23K	**87.58%** / 11M tok/s
S5	None	No	10K	48.88% / 16M tok/s
S5	SelectLTI (Input+output modulators)	No	15K	**94.90%** / 12M tok/s

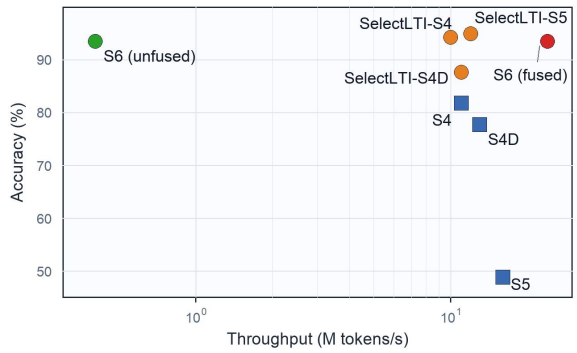

Figure 3. Quality-throughput trade-off on Selective Copying. Square markers indicate LTI models, and circle markers represent selective models. The result of unfused S6 suggests that fused kernels are necessary for practical S6 training. In contrast, SelectLTI recovers selective behavior with standard primitives while keeping LTI backbones above 10M tokens/s.

small bottleneck modulator is much easier to slim, quantize, or remove under a resource budget than a redesigned recurrent kernel. Thus, SelectLTI provides a practical intermediate point between a pure LTI backbone and a fully input-dependent selective recurrence.

Figure 3 summarizes the quality-speed trade-off. Among the models that avoid custom fused kernels, SelectLTI shifts the operating point upward in accuracy while remaining in the same throughput regime as the LTI backbones. The fused S6 point is faster on this specific GPU setup, but it depends on the specialized path that our method is designed to avoid. The key message for resource-adaptive foundation model inference is therefore not that SelectLTI dominates all hand-optimized kernels, but that it recovers selective behavior while keeping a portable implementation path open.

6. Discussion

Our empirical evidence is obtained on Selective Copying, which isolates the central behavior studied in this work: whether a model can perform selective memory updates beyond what is typically expected from an LTI memory mechanism. The results show that an input-dependent modulation can recover much of the selective behavior associated with S6 while avoiding dependence on specialized fused kernels. This supports the view that selectivity need not be implemented only through a Mamba-style operator; it can also be induced by placing lightweight input-dependent nonlinearities around an LTI state space operator.

This observation is useful for resource-adaptive foundation model inference. Custom fused kernels can make selective SSMs efficient on supported accelerators, but they also make the inference path less portable across heterogeneous hardware. In contrast, SelectLTI keeps the recurrent state-space computation tied to an LTI backbone and implements the selective part using small sigmoid-gated modulators, whose optimized implementations are widely available in common backends. This makes SelectLTI easier to adapt across deployment settings: the same block can be deployed on different hardware backends, without redesigning the algorithm of computation.

Furthermore, the structure of SelectLTI is meaningful from the perspective of physical systems. Hammerstein-Wiener systems model many dynamical processes using static nonlinearities around a linear dynamical core. This pattern is useful when the dominant temporal dynamics exist, but nonlinear effects arise through sensing, control, saturation, or readout mechanisms. SelectLTI follows a similar architectural pattern: the LTI state space core carries long-range dynamical memory, while the input-dependent gates provide a compact parameterization of nonlinear effects without making the dynamic operator input-dependent. In this sense, SelectLTI offers a practical route toward resource-adaptive scientific foundation models: it retains an interpretable LTI dynamical memory, adds parameter-efficient nonlinear modulation, and avoids tying the inference path to hardware-specific fused kernels.

7. Conclusion

We proposed SelectLTI, a selective SSM block for portable long-context inference. By augmenting an LTI state space core with lightweight sigmoid-gated modulators, SelectLTI recovers selective behavior without relying on specialized fused kernels. On Selective Copying, S6 requires custom fused CUDA kernels for feasible training, whereas SelectLTI matches or exceeds S6 accuracy using only standard tensor primitives. We view SelectLTI as a practical building block for foundation models, including scientific foundation models, that must operate across changing resource budgets and diverse hardware platforms.

Limitation and future work. Although SelectLTI is designed as a candidate replacement for S6, the core selective state-space component, we have not yet evaluated how it performs when integrated into a full Mamba-style block. Future work will develop a theoretical foundation for when LTI memory with input-dependent modulation is sufficient, and explore alternative static nonlinearities.

Acknowledgements

NBE and MG would like to acknowledge support from the U.S. Department of Energy, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program and the Advancements in Artificial Intelligence for Science program, under Contract Number DE-AC02-05CH11231 at Berkeley Lab. The views, opinions, and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2025-16065922).

Impact Statement

This work advances the theoretical and architectural understanding of state space models for sequence modeling. While such advances may indirectly influence downstream applications of machine learning, we do not identify any specific societal or ethical impacts that warrant particular discussion at this stage.

References

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Dao, T. FlashAttention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Dao, T. and Gu, A. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Ré, C. FlashAttention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. Language modeling with gated convolutional networks. In *International Conference on Machine Learning*, pp. 933–941. PMLR, 2017.

Giri, F. and Bai, E.-W. *Block-oriented nonlinear system identification*, volume 1. Springer, 2010.

Giri, F., Rochdi, Y., and Chaoui, F.-Z. Hammerstein systems identification in presence of hard nonlinearities of preload and dead-zone type. *IEEE Transactions on Automatic Control*, 54(9):2174–2178, 2009.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. 2023.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

Gu, A., Goel, K., Gupta, A., and Ré, C. On the parameterization and initialization of diagonal state space models. *Advances in Neural Information Processing Systems*, 35: 35971–35983, 2022.

Jing, L., Gulcehre, C., Peurifoy, J., Shen, Y., Tegmark, M., Soljačić, M., and Bengio, Y. Gated orthogonal recurrent units: On learning to forget. *Neural Computation*, 31(4): 765–783, 2019.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pp. 5156–5165. PMLR, 2020.

Khalil, H. K. and Grizzle, J. W. *Nonlinear systems*, volume 3. Prentice Hall, 2002.

Parnichkun, R. N., Massaroli, S., Moro, A., Smith, J. T. H., Hasani, R., Lechner, M., An, Q., Ré, C., Asama, H., Ermon, S., et al. State-free inference of state-space models: The transfer function approach. *arXiv preprint arXiv:2405.06147*, 2024.

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Biderman, S., Cao, H., Cheng, X., Chung, M., Grella, M., et al. RWKV: Reinventing RNNs for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

Poli, M., Massaroli, S., Nguyen, E., Fu, D. Y., Dao, T., Baccus, S., Bengio, Y., Ermon, S., and Ré, C. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pp. 28043–28078. PMLR, 2023.

Qiu, Z., Wang, Z., Zheng, B., Huang, Z., Wen, K., Yang, S., Men, R., Yu, L., Huang, F., Huang, S., et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *arXiv preprint arXiv:2505.06708*, 2025.

Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Smith, J. T. H., Warrington, A., and Linderman, S. W. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

Yang, S., Wang, B., Shen, Y., Panda, R., and Kim, Y. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

Yang, S., Kautz, J., and Hatamizadeh, A. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024.