Parallelizing Linear Transformers with the Delta Rule over Sequence Length

Massachusetts Institute of Technology

Songlin Yang[1] Bailin Wang[1] Yu Zhang [2] Yikang Shen[3] Yoon Kim[1]

[1]MIT CSAIL [2]Soochow University [3]MIT-IBM Watson AI Lab

MIT-IBM Watson AI Lab

Summary

DeltaNet: a variant of linear Transformer whose update is given by the Delta Rule

	Softmax Attention	Linear Attention	DeltaNet
Training (Parallel)	$\mathbf{O} = \mathrm{softmax}\left((\mathbf{Q}\mathbf{K}^\top) \odot \mathbf{M}\right)\mathbf{V}$	$\mathbf{S}_{[t+1]} = \mathbf{S}_{[t]} + \mathbf{V}_{[t]}^\top \mathbf{K}_{[t]}$ $\mathbf{O}_{[t]} = \underbrace{\mathbf{Q}_{[t]}\mathbf{S}_{[t]}^\top}_{\text{inter-chunk}} + \underbrace{\left((\mathbf{Q}_{[t]}\mathbf{K}_{[t]}^\top) \odot \mathbf{M}\right)\mathbf{V}_{[t]}}_{\text{intra-chunk}}$	??? (This work)
Inference (Iterative)	$\boldsymbol{o}_t = \frac{\sum_{i=1}^t \exp(\boldsymbol{q}_t\boldsymbol{k}_i^\top)\boldsymbol{v}_i}{\sum_{i=1}^t \exp(\boldsymbol{q}_t\boldsymbol{k}_i^\top)}$	$\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{v}_t\boldsymbol{k}_t^\top$ $\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$	$\mathbf{S}_t = \mathbf{S}_{t-1}(\mathbf{I} - \beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top) + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\top$ $\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$

Our contributions

- Revisit and demonstrate DeltaNet's effectiveness on in-context retrieval tasks.
- Develop a new parallel training algorithm to scale up DeltaNet.
- Conduct billion-scale experiments for DeltaNet and two hybrid models, showing strong language modeling and recall performance

DeltaNet is a better RNN in-context learner

DeltaNet [Schlag, Irie and Schmidhuber, '21]: Use vector representations to retrieve and update memory ("Fast Weight Programmers")

$\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t = \mathbf{W}_Q\boldsymbol{x}_t, \mathbf{W}_K\boldsymbol{x}_t, \mathbf{W}_V\boldsymbol{x}_t$	Query, key and value vectors are computed
$\boldsymbol{v}_t^{\text{old}} = \mathbf{S}_{t-1}\boldsymbol{k}_t$	Old memory is retrieved using key
$\boldsymbol{v}_t^{\text{new}} = \beta_t\boldsymbol{v}_t + (1-\beta_t)\boldsymbol{v}_t^{\text{old}}$	New memory combines current value and old memory
$\mathbf{S}_t = \mathbf{S}_{t-1} - \underbrace{\boldsymbol{v}_t^{\text{old}}\boldsymbol{k}_t^\top}_{\text{remove old}} + \underbrace{\boldsymbol{v}_t^{\text{new}}\boldsymbol{k}_t^\top}_{\text{write new}}$	State matrix is updated
$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$	Final output is computed using query

Performance on Synthetic In-context Tasks

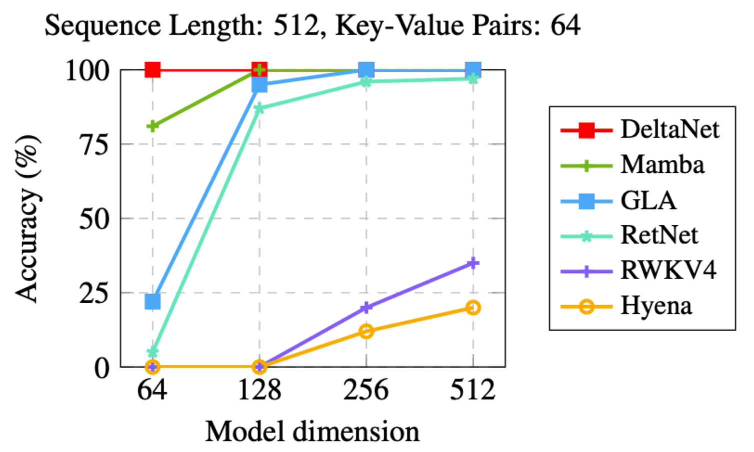


Figure 1. MQAR (Arora et al, 2023)

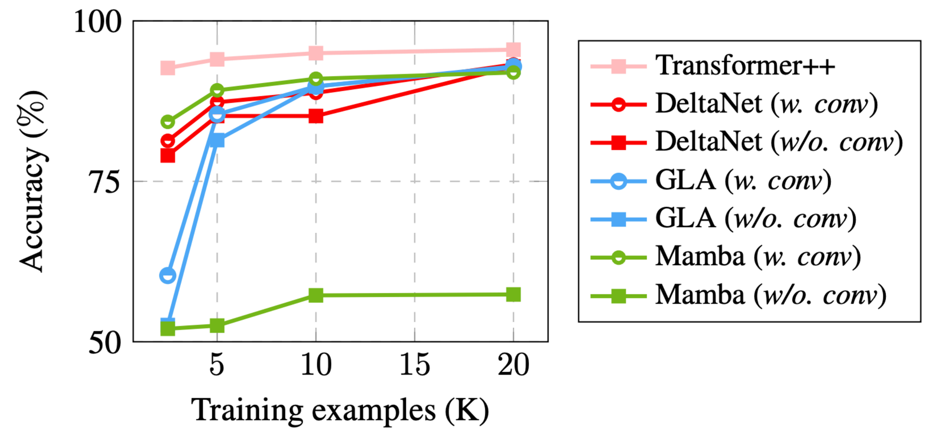


Figure 2. Regbench (Akyürek et al, 2024)

Model	Compress	Fuzzy Recall	In-Context Recall	Memorize	Noisy Recall	Selective Copy	Average
Transformer	51.6	29.8	94.1	85.2	86.8	99.6	74.5
Hyena	45.2	7.9	81.7	89.5	78.8	93.1	66.0
Multihead Hyena	44.8	14.4	99.0	89.4	98.6	93.0	73.2
Mamba	52.7	6.7	90.4	89.5	90.1	86.3	69.3
GLA	38.8	6.9	80.8	63.3	81.6	88.6	60.0
DeltaNet	42.2	35.7	100	52.8	100	100	71.8

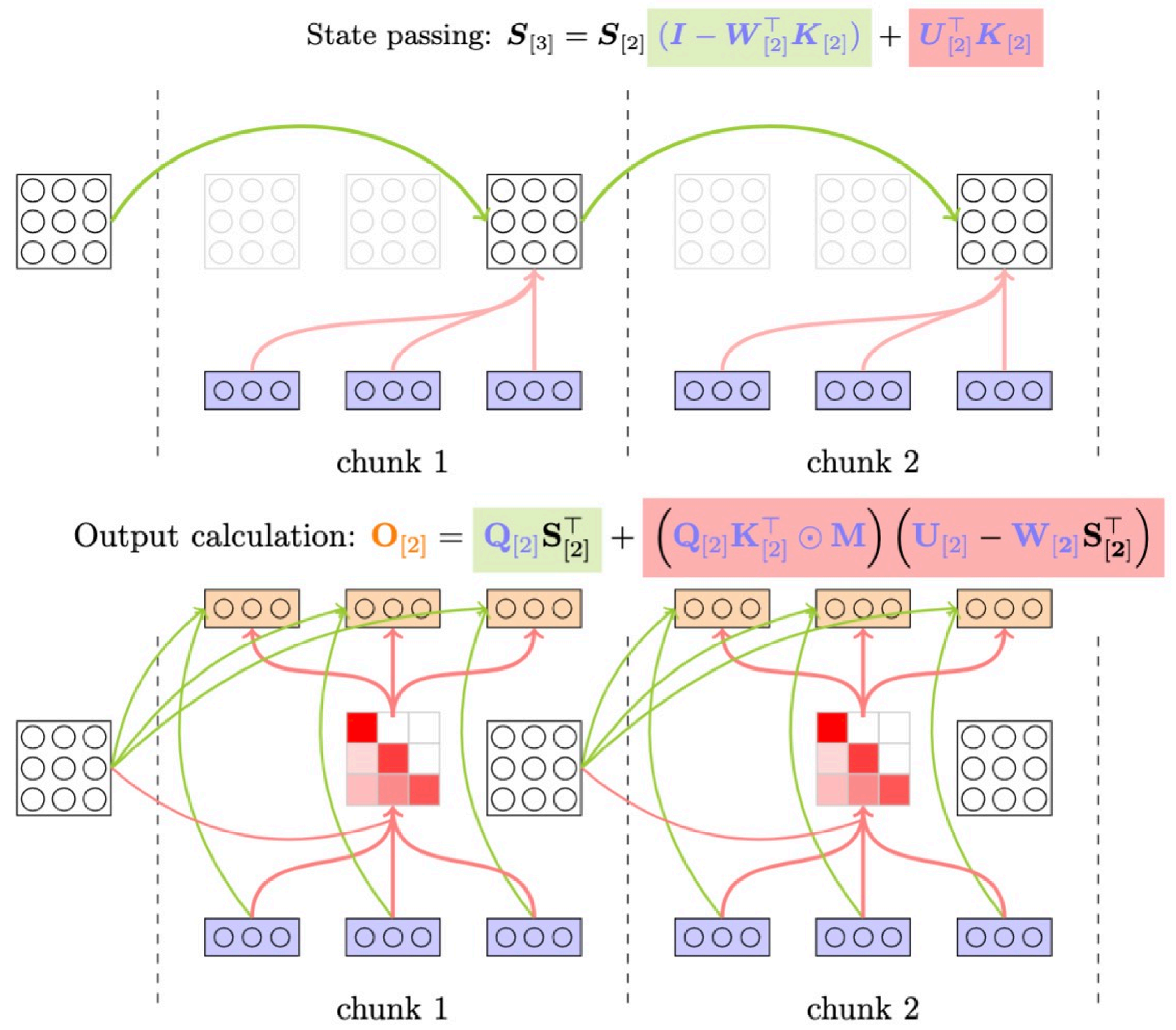
Table 1. MAD (Poli et al, 2024).

Parallelizing DeltaNet across sequence dimension

$$\mathbf{S}_t = \mathbf{S}_{t-1} - \beta_t(\mathbf{S}_{t-1}\boldsymbol{k}_t)\boldsymbol{k}_t^\top + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\top = \mathbf{S}_{t-1}(\mathbf{I} - \beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top) + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\top = \sum_{i=1}^{t}(\beta_i\boldsymbol{v}_i\boldsymbol{k}_i^\top \underbrace{\prod_{j=i+1}^{t}(\mathbf{I} - \beta_j\boldsymbol{k}_j\boldsymbol{k}_j^\top)}_{\text{defined as: } \mathbf{P}_i^t})$$

where $\mathbf{S}_t$ and $\mathbf{P}_t := P_0^t$ allow for compact WY representation.

$$\mathbf{P}_t = \mathbf{I} - \sum_{i=1}^{t}\boldsymbol{w}_i\boldsymbol{k}_i^\top, \boldsymbol{w}_t = \beta_t\left(\boldsymbol{k}_t - \sum_{i=1}^{t-1}\boldsymbol{w}_i(\boldsymbol{k}_i^\top\boldsymbol{k}_i)\right), \qquad \mathbf{S}_t = \sum_{i=1}^{t}\boldsymbol{u}_i\boldsymbol{k}_i^\top, \boldsymbol{u}_t = \beta_t\left(\boldsymbol{v}_t - \sum_{i=1}^{t-1}\boldsymbol{u}_i(\boldsymbol{k}_i^\top\boldsymbol{k}_i^r)\right)$$



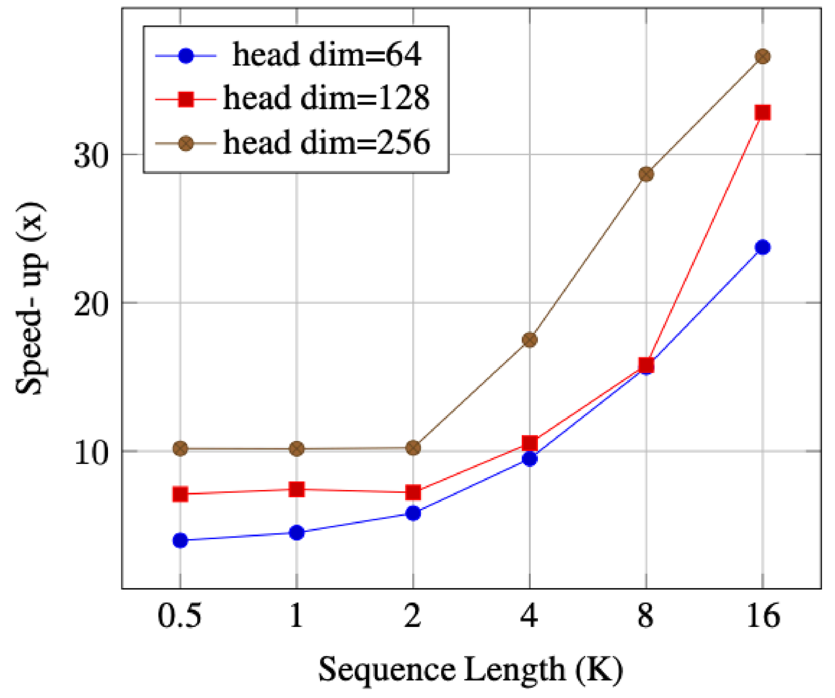


Figure 3. Speedup of chunkwise vs. recurrent implementations.

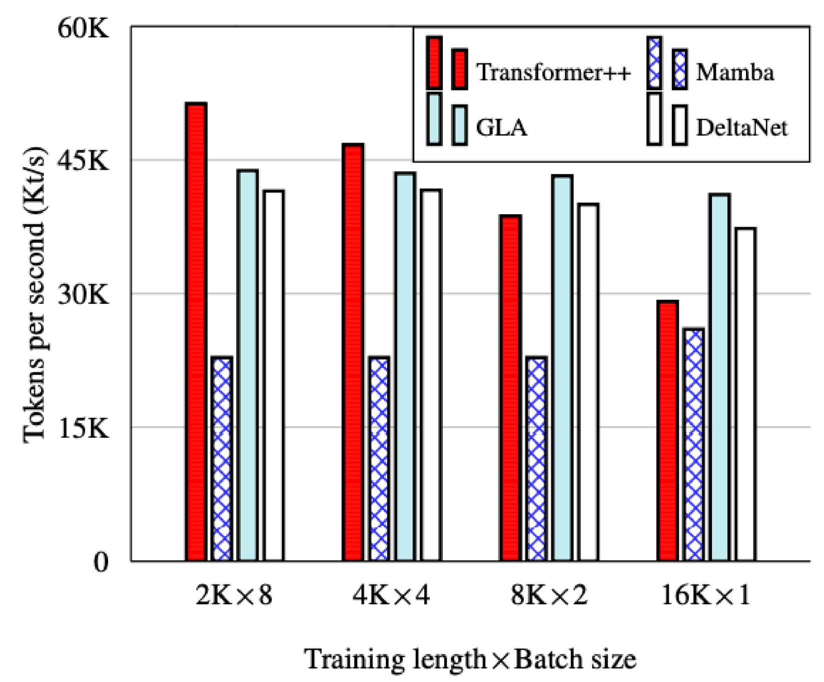


Figure 4. Training throughputs on a single H100.

Performance

Table 2. Performance comparison under the same training settings. +SWA indicates interleaving DeltaNet layers and Sliding Window Attention layers as in Samba (Ren et al, 2024). +GlobalAttn means inserting two global attention layers at layer 2 and $N/2-1$ as in H3 (Fu et al, 2023).

Scale	Model	Wiki. ppl ↓	LM Eval. acc. ↑	Recall Tasks FDA	SWD	SQD	State expansion
340M Params 15B Tokens	Transformer++	28.39	41.2	21.4	42.2	22.1	N/A
	RetNet	32.33	41.0	2.9	13.3	27.6	512x
	Mamba	28.39	41.8	2.1	12.4	23.0	64x
	GLA	28.65	41.5	8.1	18.6	27.2	128x
	DeltaNet	28.24	42.1	12.8	26.4	28.9	128x
	+ SWA	27.06	42.1	18.8	39.3	32.5	≈1000x
	+ GlobalAttn	27.51	42.1	23.1	42.9	32.1	N/A
1.3B Params 100B Tokens	Transformer++	16.85	50.9	21.4	42.2	22.1	N/A
	RetNet	18.64	48.9	14.3	42.8	34.7	512x
	Mamba	17.06	50.0	6.2	41.4	35.2	64x
	GLA	17.22	51.0	19.9	50.6	42.6	256x
	DeltaNet	16.87	51.6	17.2	49.5	37.4	128x
	+ SWA	16.56	52.1	22.3	53.3	43.3	≈1000x
	+ GlobalAttn	16.55	51.8	29.8	71.0	43.0	N/A

Model	ARC	HellaSwag	OBQA	PIQA	WinoGrande	MMLU	Average
Llama-3.2-3B	59.1	73.6	43.4	77.5	69.2	54.1	62.8
PowerLM-3B	60.5	74.6	43.6	79.9	70.0	45.0	62.3
DeltaNet-3B	60.4	72.8	41.0	78.5	65.7	40.7	59.8
RecurrentGemma-2B	57.0	71.1	42.0	78.2	67.6	31.8	57.9
RWKV-6-3B	49.5	68.6	40.6	76.8	65.4	28.4	54.9
Mamba-2.7B	50.3	65.3	39.4	75.8	63.1	26.1	53.3

Towards a Unifying Framework for Efficient Recurrent Models

Autoregressive transformations $\boldsymbol{x}_1 \dots \boldsymbol{x}_T \mapsto \boldsymbol{o}_1 \dots \boldsymbol{o}_T$ (typically) given by:

$$\mathbf{S}_t = \mathbf{S}_{t-1} \bullet \mathbf{M}_t + \boldsymbol{v}_t\boldsymbol{k}_t^\top, \qquad \boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t,$$

where $\bullet$ is an associate operator and $\mathbf{M}_t, \boldsymbol{v}_t, \boldsymbol{k}_t, \boldsymbol{q}_t$ are (potentially nonlinear) functions of $\boldsymbol{x}_t$.

Model	Recurrence	Memory read-out
Linear Attention	$\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{v}_t\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$
+ Kernel	$\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{v}_t\phi(\boldsymbol{k}_t)^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\phi(\boldsymbol{q}_t)$
+ Normalization	$\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{v}_t\phi(\boldsymbol{k}_t)^\top, \quad \boldsymbol{z}_t = \boldsymbol{z}_{t-1} + \phi(\boldsymbol{k}_t)$	$\boldsymbol{o}_t = \mathbf{S}_t\phi(\boldsymbol{q}_t)/(\boldsymbol{z}_t^\top\phi(\boldsymbol{q}_t))$
DeltaNet	$\mathbf{S}_t = \mathbf{S}_{t-1}(\mathbf{I} - \beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top) + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$
Gated RFA	$\mathbf{S}_t = g_t\mathbf{S}_{t-1} + (1-g_t)\boldsymbol{v}_t\boldsymbol{k}_t^\top, \quad \boldsymbol{z}_t = g_t\boldsymbol{z}_{t-1} + (1-g_t)\boldsymbol{k}_t$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t/(\boldsymbol{z}_t^\top\boldsymbol{q}_t)$
ABC	$\mathbf{S}_t^{\boldsymbol{k}} = \mathbf{S}_{t-1}^{\boldsymbol{k}} + \boldsymbol{k}_t\boldsymbol{\phi}_t^\top, \quad \mathbf{S}_t^{\boldsymbol{v}} = \mathbf{S}_{t-1}^{\boldsymbol{v}} + \boldsymbol{v}_t\boldsymbol{\phi}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^{\boldsymbol{v}}\,\mathrm{softmax}\left(\mathbf{S}_t^{\boldsymbol{k}}\boldsymbol{q}_t\right)$
RetNet	$\mathbf{S}_t = \gamma\mathbf{S}_{t-1} + \boldsymbol{v}_t\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$
Mamba	$\mathbf{S}_t = \mathbf{S}_{t-1} \odot \exp(-(\boldsymbol{\alpha}_t\mathbf{1}^\top) \odot \exp(\boldsymbol{A})) + (\boldsymbol{\alpha}_t \odot \boldsymbol{v}_t)\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t + \boldsymbol{d} \odot \boldsymbol{v}_t$
GLA	$\mathbf{S}_t = \mathbf{S}_{t-1} \odot (\mathbf{1}\boldsymbol{\alpha}_t^\top) + \boldsymbol{v}_t\boldsymbol{k}_t^\top = \mathbf{S}_{t-1}\mathrm{Diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$
RWKV-6	$\mathbf{S}_t = \mathbf{S}_{t-1}\mathrm{Diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = (\mathbf{S}_{t-1} + (\boldsymbol{d} \odot \boldsymbol{v}_t)\boldsymbol{k}_t^\top)\boldsymbol{q}_t$
HGRN-2	$\mathbf{S}_t = \mathbf{S}_{t-1}\mathrm{Diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t(\mathbf{1} - \boldsymbol{\alpha}_t)^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$
mLSTM	$\mathbf{S}_t = f_t\mathbf{S}_{t-1} + i_t\boldsymbol{v}_t\boldsymbol{k}_t^\top, \quad \boldsymbol{z}_t = f_t\boldsymbol{z}_{t-1} + i_t\boldsymbol{k}_t$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t/\max\{1, \lvert\boldsymbol{z}_t^\top\boldsymbol{q}_t\rvert\}$
Mamba-2	$\mathbf{S}_t = \gamma_t\mathbf{S}_{t-1} + \boldsymbol{v}_t\boldsymbol{k}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t\boldsymbol{q}_t$
GSA	$\mathbf{S}_t^{\boldsymbol{k}} = \mathbf{S}_{t-1}^{\boldsymbol{k}}\,\mathrm{Diag}(\boldsymbol{\alpha}_t) + \boldsymbol{k}_t\boldsymbol{\phi}_t^\top, \quad \mathbf{S}_t^{\boldsymbol{v}} = \mathbf{S}_{t-1}^{\boldsymbol{v}}\,\mathrm{Diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t\boldsymbol{\phi}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^{\boldsymbol{v}}\,\mathrm{softmax}\left(\mathbf{S}_t^{\boldsymbol{k}}\boldsymbol{q}_t\right)$