A Modern Self-Referential Weight Matrix That Learns to Modify Itself

Learning a program that changes its own update controller

Paper by Kazuki Irie, Imanol Schlag,
Róbert Csordás, and Jürgen Schmidhuber

ICML 2022 | arXiv:2202.05780v2
Research discussion prepared 16 October 2026

A shared initial matrix becomes a task-dependent computation.

Can the update controller adapt with experience?

The research question

Learn a matrix that changes both its predictions and the parameters that generate its next weight update.

Model	Runtime state	Update controller
Fixed-weight RNN	Hidden activations	Fixed parameters
DeltaNet	Fast weight matrix	Fixed slow network
SRWM	Its own weight matrix	Self-modifying parameters

- The algebraic update rule is fixed; its generated instructions adapt.
- Outer products replace the explicit weight addressing of early SRWMs.

Paper, Secs. 1–3 and 6, PDF pp. 1–3, 8.

DeltaNet separates the programmer from the memory

$$
\begin{aligned}
(\mathbf{k}_t, \mathbf{v}_t, \mathbf{q}_t, \beta_t) &= \mathbf{W}_{\text{slow}} \mathbf{x}_t \\
\bar{\mathbf{v}}_t &= \mathbf{W}_{t-1} \phi(\mathbf{k}_t) \\
\mathbf{W}_t &= \mathbf{W}_{t-1} + \sigma(\beta_t)(\mathbf{v}_t - \bar{\mathbf{v}}_t)\phi(\mathbf{k}_t)^\top \\
\mathbf{y}_t &= \mathbf{W}_t \phi(\mathbf{q}_t)
\end{aligned}
$$

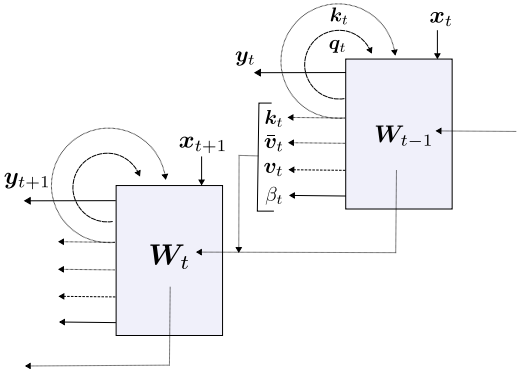
- $\mathbf{W}_{\text{slow}}$ generates the key, target value, query, and gate.
- The fast matrix $\mathbf{W}_t$ stores associations; it usually starts at zero.
- ϕ normalizes addresses; $\sigma(\beta_t)$ controls the write.

The remaining fixed component

The network that decides how to update the fast matrix stays fixed.

Paper, Sec. 2, Eqs. (1)–(4), PDF p. 2.

SRWM closes the loop through the same matrix



Original Fig. 1; Sec. 3, PDF pp. 2–3.

1. Produce the output and update instructions.
2. Read current and target contents from the same matrix.
3. Write a delta correction back to every output block.

The key, query, and gate generators are themselves modified.

Four equations define a self-modifying layer

$$\mathbf{W}_{t-1} \in \mathbb{R}^{m \times d_{\text{in}}}, \qquad m = d_{\text{out}} + 2d_{\text{in}} + 1$$

$$\begin{aligned}
(\mathbf{y}_t, \mathbf{k}_t, \mathbf{q}_t, \beta_t) &= \mathbf{W}_{t-1}\phi(\mathbf{x}_t) && \text{generate} \\
\bar{\mathbf{v}}_t &= \mathbf{W}_{t-1}\phi(\mathbf{k}_t) && \text{read old} \\
\mathbf{v}_t &= \mathbf{W}_{t-1}\phi(\mathbf{q}_t) && \text{read target} \\
\mathbf{W}_t &= \mathbf{W}_{t-1} + \sigma(\beta_t)(\mathbf{v}_t - \bar{\mathbf{v}}_t)\phi(\mathbf{k}_t)^\top && \text{write}
\end{aligned}$$

- $\mathbf{k}_t, \mathbf{q}_t \in \mathbb{R}^{d_{\text{in}}}$; both values lie in $\mathbb{R}^m$.
- The target includes the output and the update-control blocks.
- $\mathbf{y}_t$ uses the pre-update matrix; DeltaNet reads after updating.

Paper, Eqs. (5)–(8), PDF p. 3. Input activation differs by task; see slide 7.

A delta write moves one association toward another

For this step, set $k = \phi(\mathbf{k}_t)$, $q = \phi(\mathbf{q}_t)$, $\eta = \sigma(\beta_t)$, and $W = \mathbf{W}_{t-1}$.

$$W_t = W + \eta(Wq - Wk)k^\top$$

Reading back the same address gives

$$\boxed{W_t k = (1 - \eta\|k\|^2)Wk + \eta\|k\|^2 Wq.}$$

- The effective interpolation rate depends on the key concentration.
- Another address z changes in proportion to its overlap $k^\top z$.

A one-row teaching example

With $w = [1, 3]$, $k = e_1$, $q = e_2$, and $\eta = 0.5$, the update gives $w' = [2, 3]$.

Presentation derivation from Eq. (8), PDF p. 3. One-hot addresses illustrate the limiting case of concentrated softmax.

The practical layer uses four gates and multiple heads

For each row block $s \in \{y, k, q, \beta\}$:

$$\mathbf{W}_t^s = \mathbf{W}_{t-1}^s + \sigma(\beta_{s,t})\left[\mathbf{W}_{t-1}^s \phi(\mathbf{q}_t) - \mathbf{W}_{t-1}^s \phi(\mathbf{k}_t)\right]\phi(\mathbf{k}_t)^\top.$$

- Four gates give $d_{\text{out}} + 2d_{\text{in}} + 4$ output rows.
- Split inputs into H heads and run independent SRWM updates.
- Per-head key projections have shape (H, d, d), versus a dense (Hd, Hd) slow projection in DeltaNet.

Experiment	Input to first projection	Key/query addresses
Supervised	$\mathbf{x}_t$	Softmax
Reinforcement learning	softmax($\mathbf{x}_t$)	Softmax

Paper, Appendix A, Eqs. (9)–(11), PDF p. 13; Appendix B.2, PDF p. 14.

Gradient descent learns the starting program

Outer training

- Unroll the task and its weight updates.
- Backpropagate task losses to $\mathbf{W}_0$.
- Also train the vision and feedforward modules.

Runtime adaptation

- Start each new episode from $\mathbf{W}_0$.
- Generate keys, queries, and gates from experience.
- Apply the learned delta updates in the forward pass.

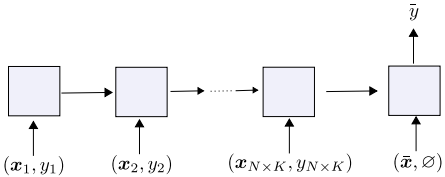
$$\mathbf{W}_0 \xrightarrow{\text{input 1}} \mathbf{W}_1 \xrightarrow{\text{input 2}} \mathbf{W}_2 \xrightarrow{\text{input stream}} \mathbf{W}_T$$

- Within the SRWM layer, $\mathbf{W}_0$ is the gradient-trained parameter.
- CNNs and surrounding Transformer blocks remain ordinary modules.

Paper, Secs. 3–4, PDF pp. 3–6; Appendix B.2, PDF p. 14.

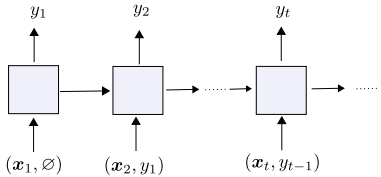
Two label protocols test different adaptation behavior

Synchronous labels



- Read NK image/label pairs.
- Predict only the final query.
- Standard N-way K-shot learning.

Delayed labels



- Read image t and label $t-1$.
- Predict at every step.
- Switch datasets during one input stream.

Labels are reassigned within each few-shot episode.

Original Figs. 2–3, PDF p. 3; Secs. 4.1–4.2, PDF pp. 3–5.

Standard few-shot performance is close to DeltaNet

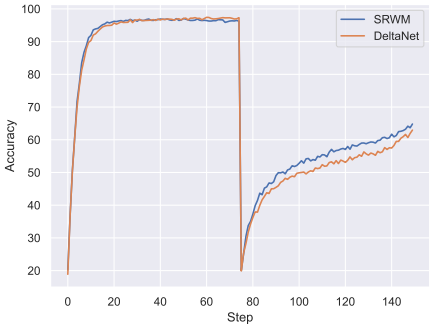
Model	Omniglot 1-shot	Mini-ImageNet 1-shot	Mini-ImageNet 5-shot
SNAIL*	99.1 ± 0.2	45.1	55.2
DeltaNet	97.2 ± 0.0	47.0 ± 0.1	62.7 ± 0.1
SRWM	97.4 ± 0.0	47.0 ± 0.2	61.4 ± 0.1
HyperTransformer*	96.2	53.8	67.1

- 5-way classification accuracy (%); higher is better.
- Mini-ImageNet uses Conv-4-32 for these rows.
- SRWM matches 1-shot DeltaNet; it is 1.3 pp lower in 5-shot.
- Self-modification works, but task-specific methods can score higher.

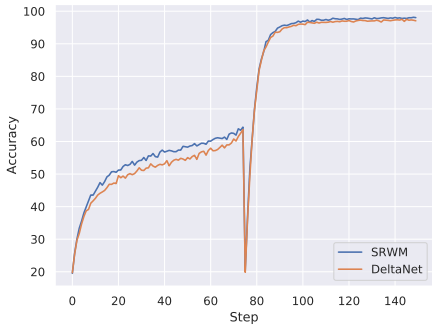
Paper, Table 1, PDF p. 4. *Results cited by the paper. Own-model CIs use 5 sets of 16K episodes, not training-seed variance (App. B.3, p. 14).

Both models recover after a dataset switch

Omniglot → Mini-ImageNet



Mini-ImageNet → Omniglot



Blue: SRWM. Orange: DeltaNet. Accuracy drops at the task switch and rises as new label feedback arrives.

Original Figs. 6–7, PDF p. 15. Horizontal axis: input step; vertical axis: mean test accuracy (%). Switch marked at step 74 in the paper.

Mini-ImageNet gains persist in either task order

Task order	Evaluated segment	DeltaNet	SRWM	Change
Omniglot first	Omniglot	92.2	92.3	+0.1 pp
	Mini-ImageNet	50.4	53.3	+2.9 pp
Mini-ImageNet first	Mini-ImageNet	50.7	53.5	+2.8 pp
	Omniglot	92.2	92.9	+0.7 pp

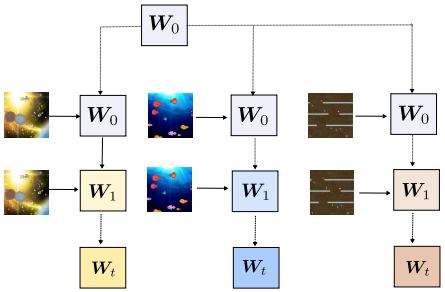
- Delayed-label 5-way classification; up to 15 examples per class.
- After Omniglot, Mini-ImageNet's 10th-instance accuracy improves from 54.8% to 58.0%.
- First-instance accuracy is almost unchanged: 20.8% vs. 20.9%.

Interpretation

The improvement appears mainly after feedback has been received.

Paper, Table 2, PDF p. 5; Table 7, PDF p. 16. Values in the table are segment-level accuracy (%).

Multi-task RL shares an initial program across games



- Six ProcGen **easy** environments; 300M total steps.
- IMPALA, 48 actors, shared 15-layer residual vision model.
- DeltaNet/SRWM: two layers, width 128.
- Backpropagate 50 steps; retain state until the episode ends.

Generalization protocol

200 training levels per game. Three fixed test sets of 200 levels. Three independent training runs produce nine run/test-set measurements.

Original Fig. 5; Sec. 4.3, PDF pp. 5–7. Games: Bigfish, Fruitbot, Maze, Leaper, Plunder, Starpilot.

Self-modification helps the multi-task RL agent

Model	Parameters	Train	Test
Feedforward	626K	22.5 (2.6)	16.4 (1.6)
LSTM	1.2M	28.3 (1.4)	15.7 (1.6)
Fake SR	959K	27.0 (1.8)	15.3 (1.9)
DeltaNet	1.05M	35.0 (1.6)	18.6 (1.7)
SRWM	968K	34.6 (1.8)	**20.0 (1.8)**

- Normalized aggregate score $\times 100$, mean (SD); higher is better.
- Fake SR removes self-modification: the test gap is 4.7 points.
- The test gap to DeltaNet is smaller: 1.4 points.
- Reported variability is substantial; no significance test is provided.

Paper, Table 3, PDF p. 7. Three training runs; nine test measurements. These are score units, not accuracy percentages.

Bigfish reveals a benefit specific to joint training

Game	Training	Fake SR	SRWM
Bigfish	Six games	4.7 (2.4)	9.0 (2.0)
	Expert	15.8 (1.7)	14.2 (2.0)
Starpilot	Six games	49.6 (2.1)	54.6 (2.4)
	Expert	52.9 (1.2)	57.3 (1.6)

- Raw **test** returns, mean (SD).
- Experts receive 50M steps; joint agents receive about 50M per game.
- On Bigfish, SRWM reduces the loss from training jointly.
- It still falls short of the single-game expert.

Paper, Table 4, PDF p. 7. Starpilot also benefits in the expert setting.

Truncating gradients must not erase the adaptive state

State policy	Train score	Test score
Retain until episode boundary	34.6 (1.8)	20.0 (1.8)
Reset every 50 steps	28.5 (1.2)	16.1 (2.2)
No self-modification (Fake SR)	27.0 (1.8)	15.3 (1.9)

Two different operations

Detach: retain the current weights, truncate the gradient history.

Reset: discard the accumulated weight changes.

- The standard model uses a 50-step gradient horizon.
- The reset ablation changes both training and evaluation.

Paper, Sec. 4.3, PDF p. 7; Table 9, PDF p. 17. Normalized aggregate score $\times 100$, mean (SD).

The memory experiment adds SRWM to a DeltaNet

SR-DeltaNet is a different architecture

Replace DeltaNet's slow projection with an SRWM, while retaining its separate fast weight memory.

- Four partially observable ProcGen **memory** environments.
- 800M steps; 500 training levels; 100-step backpropagation.
- Scores use the ProcGen **hard** normalization constants.

	DeltaNet	SR-DeltaNet
Train	51.8 (2.6)	59.0 (2.1)
Test	38.0 (4.1)	38.5 (3.2)

Better training fit; the test improvement is only 0.5 score points.

Paper, Appendix C.1, Table 8, PDF pp. 16–17. Normalized score $\times 100$, mean (SD). Authors describe this as preliminary.

A matrix state has capacity and a practical cost

State and computation

- A width-d head carries $O(d^2)$ changing weights.
- Matrix-vector products and a rank-one write per step.
- Linear in sequence length at fixed width.
- Implemented with custom CUDA kernels.

RL throughput on one P100

Model	Steps/s
FF / LSTM	$\sim$ 3500
DeltaNet	$\sim$ 2300
SRWM	$\sim$ 1700

Actor/learner state copying adds overhead.

Workload matters

On 1-shot Omniglot with Conv-4 and batch 128, LSTM, DeltaNet, and SRWM all process about 8000 images/s on one P100.

Paper, Sec. 5, PDF pp. 7–8. Complexity interpretation follows the layer equations.

The evidence establishes useful runtime adaptation

- **Learnability:** competitive few-shot behavior can emerge from a single self-referential matrix.
- **Adaptation:** feedback helps after task switches; persistent self-modification improves multi-task RL.
- **Optimization:** some runs learn label-allocation shortcuts instead of effective feedback use.
- **Scope:** direct tests of recursive self-improvement and long-run objective preservation remain open.

A useful next experiment

Ablate the k, q, and gate-generating updates separately, then test repeated task switches and returns to earlier tasks.

Paper, Secs. 4–6, PDF pp. 5–9; Appendix B.5, p. 16. The proposed next experiment is presentation analysis.

Three ideas to take away

1. **Let the controller change.**

   The same matrix computes outputs and generates its own updates.

2. **Use associations as a programming interface.**

   Keys, queries, and gates turn a delta rule into a learned task-dependent modification.

3. **Evaluate the complete adaptive process.**

   Task order, state persistence, training success, and runtime cost all affect the value of self-modification.

Synthesis of Secs. 3–6 and Appendices A–C.

Paper and reusable materials

Kazuki Irie, Imanol Schlag, Róbert Csordás, Jürgen Schmidhuber
A Modern Self-Referential Weight Matrix That Learns to Modify Itself.
ICML 2022. arXiv:2202.05780v2, revised 17 June 2022.

- Paper: arxiv.org/abs/2202.05780v2
- Author implementation: github.com/IDSIA/modern-srwm
- Companion notes: `reading-notes.zh.md`
- Editable presentation: `slides.en.tex`

Figures are reproduced from the authors' source files. Tables select reported values; original page and table numbers appear on the relevant slides.

Backup slides: FC100, LSTM, training dynamics, game-level results, and reproduction details.

Backup: v2 substantially revises the FC100 result

Model	Vision encoder	Accuracy (%)
ProtoNet Cosine*	Res-12-512	40.9 ± 0.6
ProtoNet Cosine Scaled*	Res-12-512	51.0 ± 0.6
TADAM*	Res-12-512	56.1 ± 0.4
DeltaNet	Conv-4-32	43.2 ± 0.1
SRWM	Conv-4-32	44.5 ± 0.1
SRWM	Res-12-256	46.3 ± 0.1

Use the corrected 5-way 5-shot result

An earlier preprint reported 57.1% for SRWM / Conv-4-32. Version 2 corrects a meta-test shuffling bug and reports 44.5%.

Paper, Appendix B.4 and footnote 4, PDF p. 14; Table 5, p. 15. *Results cited by the paper.

Backup: fixed-weight LSTMs can also learn the task

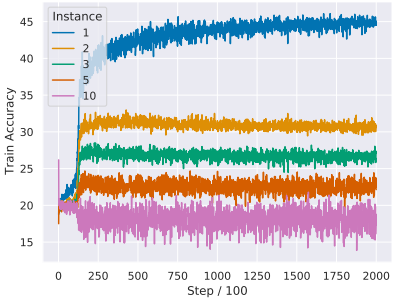
Mini-ImageNet LSTM	600K training steps	1.2M training steps
1-shot	$< 38\%$	$46.6 \pm 0.1\%$
5-shot	$< 55\%$	$59.0 \pm 0.1\%$

- One 1024-unit layer: 7.5M parameters; batch size 16.
- Some configurations or seeds remain at chance accuracy.
- The reported best SRWM / DeltaNet models use 600K steps, with 3.6M / 5.5M parameters respectively.
- Task-episode confidence intervals do not measure training reliability.

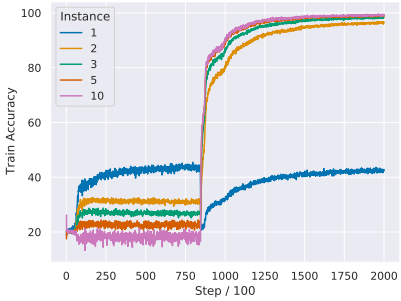
Paper, Appendix B.2–B.4, PDF pp. 14–15; Table 6, p. 15. Mini-ImageNet backends are Conv-4-32.

Backup: a training run may learn the wrong shortcut

Unsuccessful training



Successful training



- Curves measure the 1st, 2nd, 3rd, 5th, and 10th class occurrences.
- Success means later occurrences benefit strongly from label feedback.
- Batch size 32 was important in the reported delayed-label setup.

Original Figs. 8–9, Appendix B.5, PDF p. 16. Axes: training steps / 100; Omniglot training accuracy (%).

Backup: SRWM does not win on every easy game

Raw test return	Feedforward	DeltaNet	SRWM
Bigfish	4.3	7.8	9.0
Fruitbot	25.6	24.5	25.5
Leaper	3.4	3.3	3.4
Maze	1.4	1.7	1.8
Plunder	3.2	3.3	3.0
Starpilot	53.0	53.9	54.6

Gains vary by game; raw score scales differ across environments.

A source-table discrepancy

Table 9 swaps the True/Fake SR entries in its aggregate *training* row relative to Table 3. The presentation uses Table 3 for that row.

Paper, Table 9, PDF p. 17; Table 3, p. 7. Selected means shown; original table also reports SDs.

Backup: memory gains are uneven across environments

Raw test return	DeltaNet	SR-DeltaNet
Dodgeball	6.4 (0.3)	6.2 (0.6)
Heist	0.8 (0.2)	1.1 (0.3)
Maze	3.3 (0.6)	3.3 (0.4)
Miner	29.2 (1.1)	29.4 (0.7)

- SR-DeltaNet has a self-modifying programmer and a separate memory.
- Test improvement is concentrated in Heist.
- The aggregate test change is 38.0 to 38.5; training changes from 51.8 to 59.0.

Paper, Tables 8 and 10, PDF pp. 17–18. Raw returns are mean (SD); aggregate scores use hard-distribution normalization.

Backup: reproduction depends on task-specific details

- **Input activation:** no $\phi(\mathbf{x}_t)$ in supervised runs; softmax in RL. Normalize key/query addresses in both.
- **Initialization:** optimize a learned $\mathbf{W}_0$. An all-zero SRWM is a fixed point of Eqs. (5)–(8).
- **Capacity:** equal width and head count do not imply equal parameter counts for SRWM and DeltaNet.
- **Vision detail:** Sec. 4.1 says 64 channels for Omniglot, whereas Appendix B.2 describes Conv-4 with 32 channels.

These details affect a faithful comparison and should be resolved when implementing a reproduction.

Paper, Eqs. (5)–(8), p. 3; Sec. 4.1, p. 4; Appendices A–B.2, pp. 13–14. The zero-initialization observation is a direct equation-level deduction.