Pretraining Recurrent Networks without Recurrence

Supervised Memory Training (SMT)

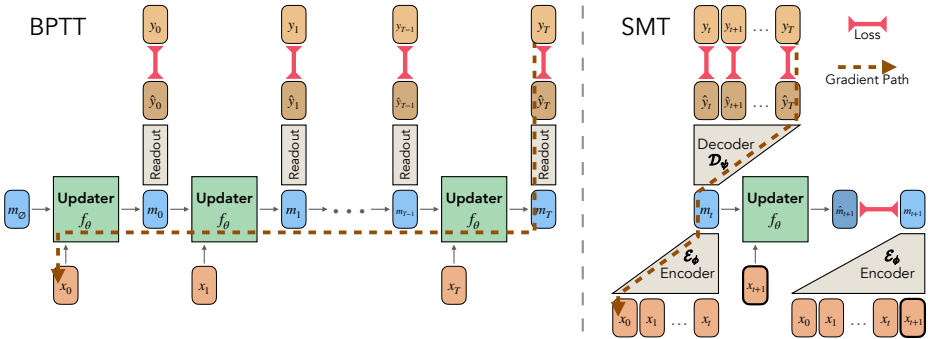
Paper authors: Akarsh Kumar and Phillip Isola, MIT
arXiv:2606.06479v2, revised 24 July 2026

Central idea

Learn useful memory states in parallel, then supervise an RNN's one-step updates between those states.

Research discussion | 29 October 2026
Publication not found as of 2026-10-29; author page: Preprint 2026.
Official code: SMT and DMT in PyTorch.

Memory labels shorten the credit assignment path



BPTT

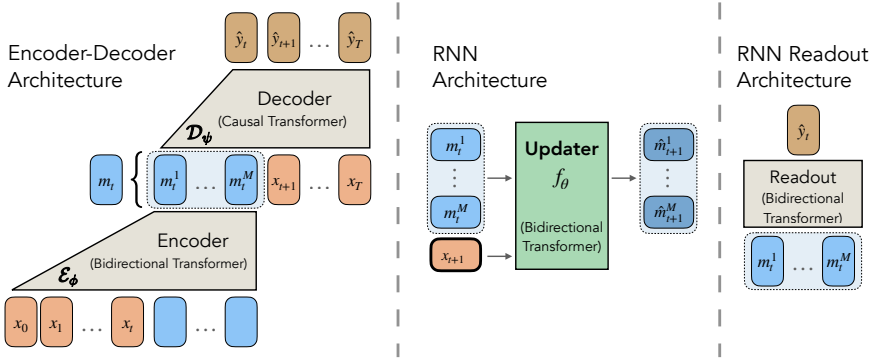
Learn representation and dynamics through a recurrent chain.

SMT

Predict the future from an encoded memory; fit one-step transitions.

Paper, Fig. 1, PDF p. 1; Secs. 1–2, pp. 2–4. Path length is relative to sequence length.

The deployed model has a fixed memory



- The teacher encodes a past window into memory tokens.
- The RNN attends only to old memory tokens and one new input.
- Default state: 16 tokens $\times$ 256 dimensions $=$ 4096 scalars.

Paper, Fig. 15 and Appendix B.1, PDF pp. 20–21; defaults, pp. 22–23.

Future prediction gives memory its meaning

$$m_t = \mathcal{E}_\phi(\mathbf{x}_t^{\mathrm{ctx}}), \qquad \mathbf{x}_t^{\mathrm{ctx}} = [x_0, \ldots, x_t]$$

$$p(\mathbf{y}_t^{\mathrm{fut}} \mid \mathbf{x}_t^{\mathrm{ctx}}, \mathbf{x}_t^{\mathrm{fut}}) = \prod_{\tau=t}^{T} p_\psi(y_\tau \mid m_t, \mathbf{x}_{t+1:\tau})$$

Predictive-state objective

$$\mathcal{L}_t^{\mathrm{dec}} = \mathrm{CE}(\mathbf{y}_t^{\mathrm{fut}}, \mathcal{D}_\psi(m_t, \mathbf{x}_t^{\mathrm{fut}}))$$

- Bidirectional encoding stays within the past context.
- Causal decoding uses teacher-forced future inputs.
- To predict y_τ, the decoder sees inputs only through τ.

Paper, Sec. 2.2, PDF pp. 3–4, Eq. (2); Appendix B.1, p. 20.

Three losses shape useful, updatable memories

$$\hat{m}_{t+1} = f_\theta(m_t, x_{t+1}), \qquad \mathcal{L}_t^{\mathrm{dyn}} = \mathrm{MSE}(\hat{m}_{t+1}, m_{t+1})$$

$$\mathcal{L}^{\mathrm{unif}} = \log \mathbb{E}_{a,b} \exp\left(-2\|m_a - m_b\|_2^2\right)$$

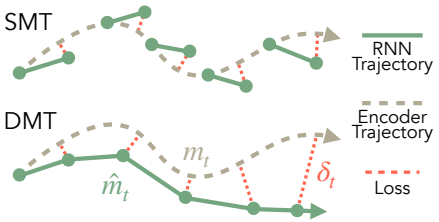
$$\boxed{\mathcal{L}^{\mathrm{smt}} = \lambda_{\mathrm{dec}} \mathbb{E}_t \mathcal{L}_t^{\mathrm{dec}} + \lambda_{\mathrm{dyn}} \mathbb{E}_t \mathcal{L}_t^{\mathrm{dyn}} + \lambda_{\mathrm{unif}} \mathcal{L}^{\mathrm{unif}}}$$

- Prediction: retain information useful for future outputs.
- Dynamics: train the updater *and* encourage Markovian states.
- Uniformity: discourage collapse of distinct memories.

Default SMT jointly trains the teacher and the RNN.

Paper, Sec. 2.2, PDF p. 4, Eqs. (3)–(5).

DMT corrects drift on the RNN's own states



Paper, Fig. 2, PDF p. 4.

$$\underbrace{(m_t, x_{t+1})}_{\text{SMT}} \longrightarrow \underbrace{(\hat{m}_t, x_{t+1})}_{\text{DMT}}$$

$$\mathcal{L}^{\text{dmt}} = \mathbb{E}_t[\text{MSE}(\hat{m}_t, m_t)]$$

- Freeze the teacher; roll out the RNN.
- Inputs remain teacher-forced.
- Rollout is sequential; temporal gradients are optional.

Readout adaptation

Copy decoder weights to the RNN readout, then fine-tune the readout with next-token loss. This loss does not update RNN dynamics.

Paper, Sec. 2.3, PDF pp. 4–5, Eq. (6); Appendix B.3, p. 22.

Parallel pretraining leads to recurrent deployment

	BPTT	SMT	DMT
RNN rollout in training	Yes	No	Yes
Serial operations vs. T	$O(T)$	$O(1)$	$O(T)$
Supervisory signal	Task targets	Teacher states	Teacher states

- SMT's $O(1)$ path is relative to time horizon, at fixed depth.
- Deployment uses fixed memory and constant per-token work *with respect to history length*.
- The time-parallel teacher still has expressivity limits.

The full evaluated method is SMT→DMT.

Paper, Table 1, PDF p. 5; Discussion, p. 11; Appendices A and E, pp. 20, 25.

The experiments isolate training and memory effects

Comparison

- BPTT RNN.
- SMT→DMT RNN.
- SMT Encoder*: teacher with the same memory bottleneck.

Updater backbones
Transformer, MLP, GRU.

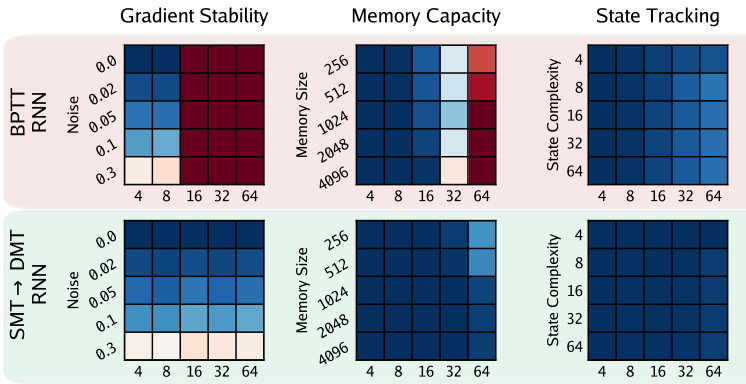
Tasks

- Five controlled synthetic tasks.
- TinyStories: character-level language modeling.
- MNIST: 784 raw-pixel tokens per image.
- Sketchy: 1024 binary-patch tokens per image.

Test loss ↓; compare serial compute and processed tokens.

Paper, Sec. 3, PDF pp. 5–7; Appendix B, pp. 20–23.

SMT improves long-range retrieval and state use



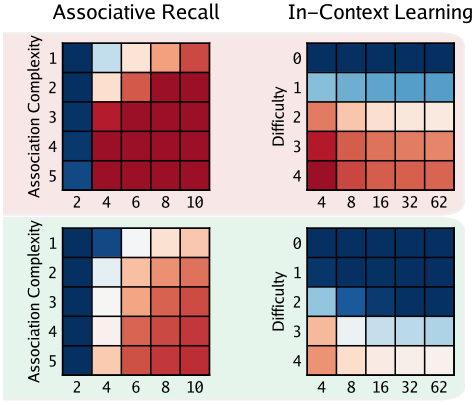
- Retrieval resists longer delays.
- Copying uses larger memory.
- Stack tracking remains more accurate.

Top: BPTT. Bottom: SMT→DMT.

Horizontal axes: sequence length, 4–64. Test loss: blue 0, red 3.

Paper, Fig. 3, PDF p. 6. First three task columns and two RNN rows cropped; teacher row omitted.

Associative recall remains difficult



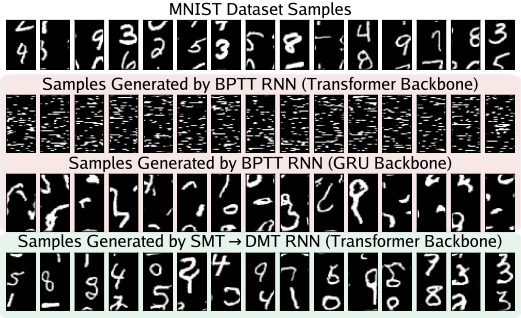
- More pairs and more complex strings still hurt recall.
- In-context arithmetic improves, with failures at high difficulty.
- The full figure also shows limitations in the teacher.

Top: BPTT. Bottom: SMT→DMT. Test loss: blue 0, red 3.

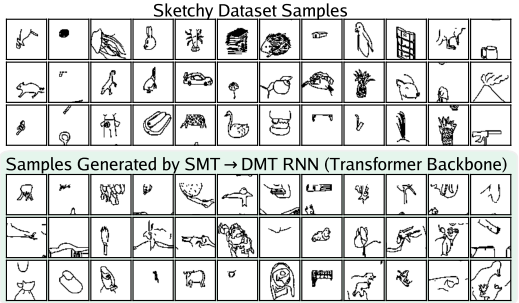
Horizontal axes: number of key–value pairs / in-context examples.

Paper, Fig. 3, PDF p. 6. Last two task columns, two RNN rows; task definitions, pp. 21–22.

Pixel streams reveal the value of long-range memory



MNIST: data, BPTT Transformer, BPTT GRU, SMT→DMT Transformer.

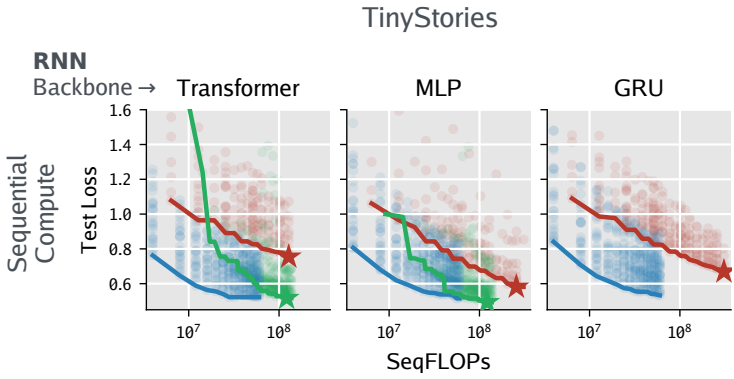


Sketchy: data above, SMT→DMT generations below.

SMT→DMT captures coherent strokes; some generated sketches remain hard to identify.

Paper, Figs. 4–5, PDF p. 6; additional samples and discussion, pp. 23–24, 27.

TinyStories gains are clearest in serial compute

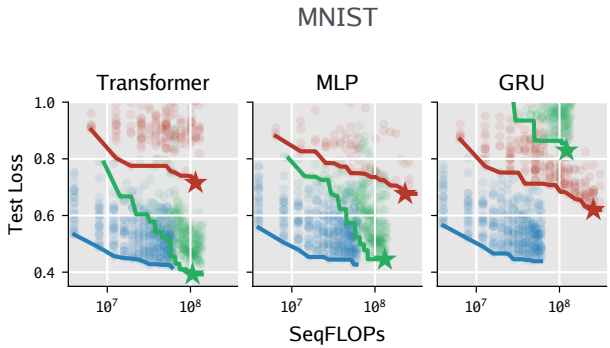


BPTT RNN SMT→DMT RNN SMT Encoder (teacher)

- Transformer/MLP RNNs reach lower loss more efficiently in SeqFLOPs.
- Token efficiency is roughly similar to BPTT in the paper's sweep.

Paper, Fig. 6, PDF p. 7 (TinyStories). Stars: best RNN loss. SeqFLOPs measure serial work, not elapsed time.

MNIST gains coexist with a GRU failure

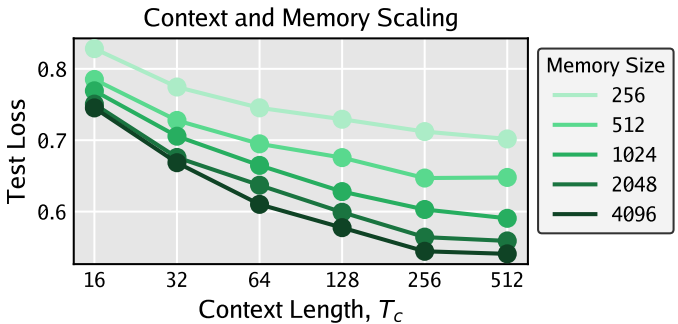


BPTT RNN SMT→DMT RNN SMT Encoder (teacher)

- Transformer/MLP improve in both serial compute and data efficiency.
- GRU training suffers memory collapse under this SMT recipe.

Paper, Fig. 6, PDF p. 7 (MNIST). Stars: best RNN loss. Each run in this sweep is capped at one H200-day.

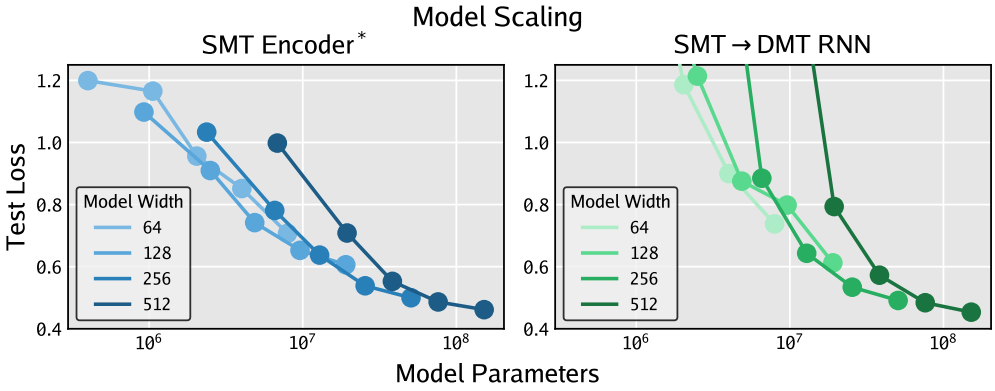
Context and state capacity provide complementary gains



- TinyStories, SMT→DMT.
- Longer context helps at fixed state size.
- Larger states retain more predictive information.
- Context gains diminish near $T_c = 256$–512.

Paper, Fig. 7, PDF p. 7. Memory-size labels count scalars, not memory tokens.

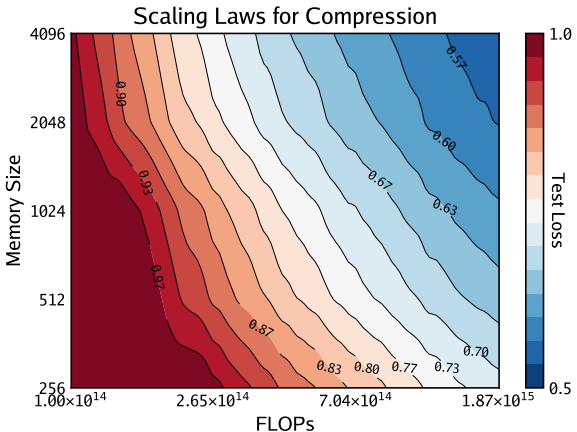
Larger RNNs approach their teacher's performance



- Width and depth are varied jointly across a model-size sweep.
- Both improve with scale; the student gap narrows at larger sizes.

Paper, Fig. 8, PDF p. 7. TinyStories; teacher left, RNN right. No universal fitted exponent is reported.

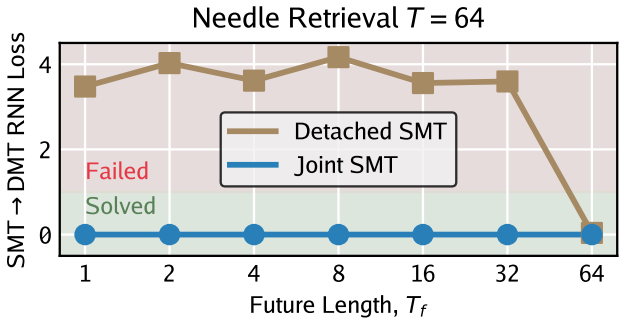
More training compute can buy a smaller memory



- Black curves are equal-loss contours.
- Moving right permits moving down: more compute, less state.
- These measurements are for the SMT encoder.
- RNN rollout quality still depends on fitting its dynamics.

Paper, Fig. 9, PDF p. 8. Horizontal axis: total training FLOPs; vertical axis: memory scalars.

Joint training compensates for short future windows



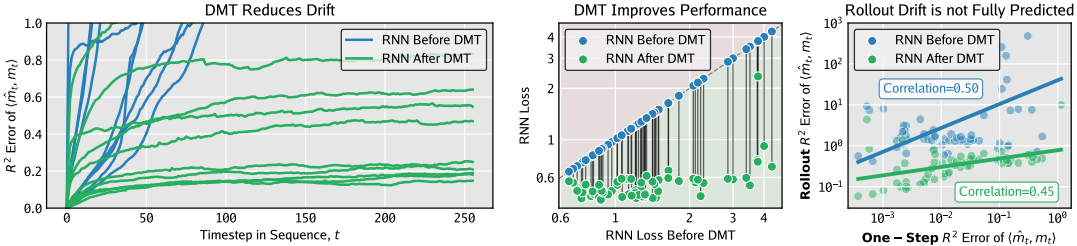
- Retrieval horizon: $T = 64$.
- Detached dynamics need $T_f = 64$ to capture the distant target.
- Joint training succeeds even at $T_f = 1$.

Authors' mechanism interpretation

Credit can propagate across optimization steps through jointly trained adjacent memories, even without an unrolled RNN graph.

Paper, Fig. 10 and Sec. 3.6, PDF p. 8. This figure does not establish a general iteration lower bound.

DMT improves long rollouts beyond one-step fitting



- Left: post-DMT drift often plateaus at a lower level.
- Middle: prediction loss improves across many configurations.
- Right: one-step and rollout drift are only partially correlated (reported correlations: 0.50 before, 0.45 after).

Paper, Fig. 12, PDF p. 9. Plotted drift uses $1 - R^2$; the training objective uses MSE.

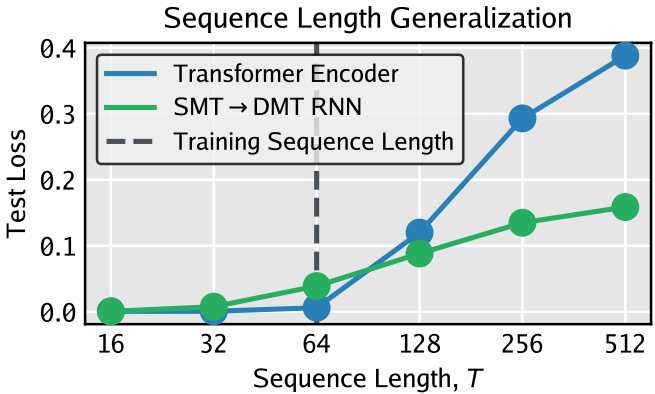
Regularization trades loss against state dispersion

Dataset	Setting	λ_{dyn}	λ_{unif}	Test loss ↓	$\mathcal{L}^{\mathrm{unif}}$
TinyStories	Default	0.1	0.001	0.567	−3.294
TinyStories	Grid minimum	0.01	0.003	**0.565**	−3.682
MNIST	Default	0.1	0.001	0.516	−3.618
MNIST	Grid minimum	0.1	0	**0.507**	−1.273
MNIST	High dynamics	10	0	0.521	−0.058

- The default balances task loss and memory geometry.
- The text's *best* setting differs from numerical grid minima.
- A nearly collapsed state can still have a competitive task loss.

Exact cells from paper, Fig. 16, PDF p. 23; compare the recommendation in Sec. 3.6, p. 8.

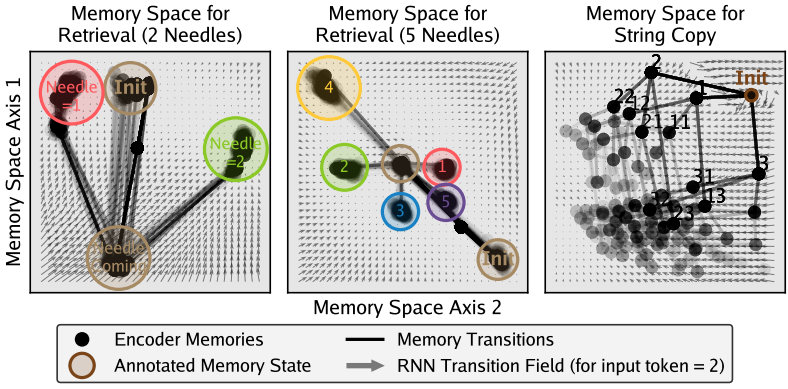
The RNN can extrapolate beyond its teacher



- Synthetic stack tracking.
- Training length: 64.
- At length 512: teacher ≈ 0.39, RNN ≈ 0.16 loss.
- Recurrent updates provide a useful inductive bias.

Paper, Fig. 13, PDF p. 9. Approximate graph readings; a synthetic length-generalization result.

Memory geometry reflects what the task must retain



- Retrieval merges irrelevant histories into a few useful states.
- String copy preserves more distinctions in a tree-like geometry.

Paper, Fig. 14, PDF p. 10. True 2D memories; arrows show transitions and the learned update field.

SMT offers a practical pretraining decomposition

What the evidence supports

Parallel predictive-state learning plus supervised dynamics makes long-range memory easier to learn in the tested nonlinear RNNs.

- **Full recipe:** SMT pretraining and sequential DMT correction.
- **Main costs:** teacher training, rollout drift, architecture sensitivity.
- **Expressivity:** the parallel teacher may miss computations that require deep recurrence.
- **Next experiment:** evaluate total cost and rollout quality jointly as state size and sequence horizon grow.

Paper, Discussion, PDF p. 11; empirical limits, pp. 7–9. Final experimental suggestion is reader analysis.

References and discussion prompts

Primary paper

Akarsh Kumar and Phillip Isola.
Pretraining Recurrent Networks without Recurrence. 2026.
arXiv:2606.06479v2 Author project page Official implementation

- Can one-step pretraining reduce the need for DMT?
- How should future horizon, state capacity and compute be allocated?
- What post-training can teach a student beyond teacher expressivity?

Experimental figures reproduced from the paper's v2 author source and checked against the PDF. Publication lookup: 2026-10-29; no formal publication found.

Backup: timestamps preserve sequence information

$$[x_0, x_1, \ldots, x_t] \quad \longleftrightarrow \quad \{(x_0, 0), (x_1, 1), \ldots, (x_t, t)\}$$

1. Sort the set by its unique timestamps.
2. Recover the original sequence.
3. Apply any recurrent memory function $\mathcal{Q}$.

What the construction establishes

A permutation-invariant set function can represent the same mapping. The construction itself still uses the original sequential computation.

A bounded-depth Transformer may need depth that grows with sequence length.

Paper, Appendix E, PDF p. 25.

Backup: Markovian memory requires ideal sufficiency

Assume m_t is an optimal minimal sufficient statistic for future prediction at every t.

$$H(\mathbf{y}_t^{\mathrm{fut}} \mid m_t, \mathbf{x}_t^{\mathrm{fut}}) = H(\mathbf{y}_t^{\mathrm{fut}} \mid \mathbf{x}_t^{\mathrm{ctx}}, \mathbf{x}_t^{\mathrm{fut}})$$

1. (m_t, x_{t+1}) remains sufficient for the remaining future.
2. Minimality makes m_{t+1} a function of that statistic.

$$m_{t+1} = f(m_t, x_{t+1}), \qquad H(m_{t+1} \mid m_t, x_{t+1}) = 0$$

Practical gap

Finite horizons and approximate optimization do not ensure minimal sufficiency. Joint dynamics explicitly encourages updatable states.

Paper, Appendix F, PDF p. 26. Cross-entropy alone does not explicitly impose minimality.

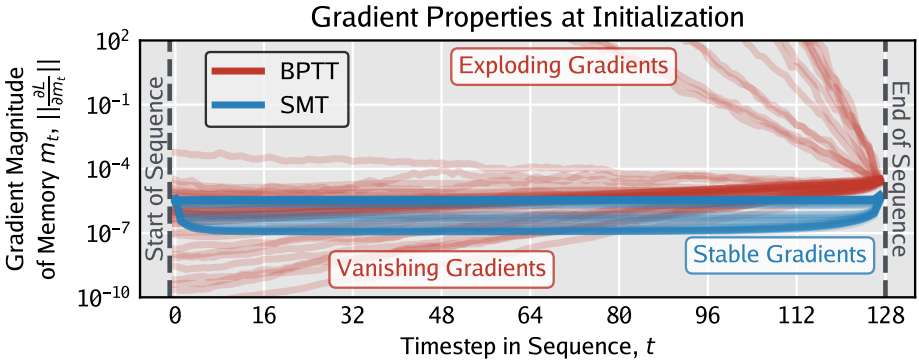
Backup: the two training protocols differ

	Synthetic tasks	Natural data defaults
Context / future	$T_c = T_f = T$	$T_c = 256,\ T_f = 64$
Time splits	All timesteps	One sampled timestep
Batch	32 sequences	128 sequences
Encoder contexts	$32T$	One per sampled split
Prediction loss	Defined target positions	Future target window

- AdamW; weight decay 0.01; gradient clipping at norm 1.
- Learning rate tuned separately for each method.
- Natural-data SMT: 150,000 steps; loss weights $(1, 0.1, 0.001)$.
- Encoder/updater: 8 layers; decoder/readout: 4 layers.

Paper, Appendix B.3, PDF pp. 22–23. Width 256, 16 memory tokens unless otherwise specified.

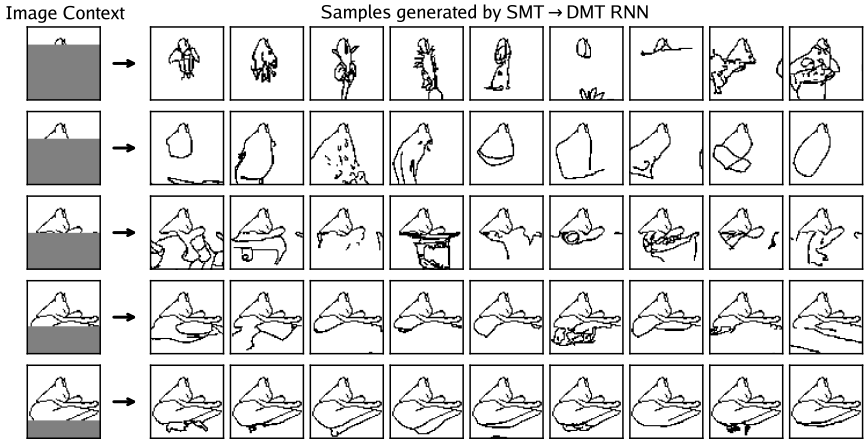
Backup: gradient probes support the short-path mechanism



- Needle retrieval; loss applied at the end of the sequence.
- Across initializations, BPTT varies with temporal distance; SMT gradients are much flatter.

Paper, Fig. 11, PDF p. 8; Sec. 3.7, p. 9. Probe at initialization, not a general convergence theorem.

Backup: more image context improves continuations



Each row supplies a larger known prefix of Attneave's cat image.

Paper, Fig. 20, PDF p. 28. Sketchy-trained RNN; samples retain errors, especially with short context.

Backup: local fitting and rollout stability differ

Define the deviation $e_t = \hat{m}_t - m_t$. If the updater is locally Lipschitz in memory,

$$\|e_{t+1}\| \le L_t \|e_t\| + \epsilon_t, \qquad \epsilon_t = \|f_\theta(m_t, x_{t+1}) - m_{t+1}\|.$$

- Small teacher-state error ϵ_t can coexist with amplification of off-trajectory errors.
- DMT trains on those off-trajectory states.
- Lower equilibrium drift may matter more than first-step error.

Interpretation, not a measured contraction bound

The paper does not establish $L_t < 1$ for DMT.

Reader's explanatory bound; empirical motivation: paper, Fig. 12, PDF p. 9.

Backup: resource estimates assume a sampled split

	BPTT	SMT	DMT
Training memory	$O(MT)$	$O(M + T_c)$	$O(MT)$
Training compute	$O(T)$	$O(T_c^2)$	$O(T_c^2 T)$
Serial operations	$O(T)$	$O(1)$	$O(T)$

- Simplified orders: model size is fixed and logarithmic factors omitted.
- SMT samples one split; the synthetic all-split setup changes work and activation memory.
- Future decoding and attention implementation affect actual costs.
- DMT has direct state supervision; optional temporal backpropagation still creates long paths.

Paper, Table 1, PDF p. 5; Secs. 2.2–2.3, pp. 4–5; Appendix B.3, p. 22.