Learning to Memorize at Test Time

Ali Behrouz

alibehrouz@cs.cornell.edu
alibehrouz@google.com

@behrouz_ali

Google Research

CORNELL UNIVERSITY FOUNDED A.D. 1865

It's All Connected! Sometimes Perspective Matters for Improvement!

Memory Perspective

- Long and short-term memory
- Forget Gate
- Associative Memory
- ...

Matmul and Transformers era

Dynamical System

- Linear approximation can be surprisingly effective!
- Discretization (extension of forget gate)
- Local and global convolutions
- ...

Modern Memory Perspective

- Data-dependent gating
- Expressive memory management
- Expressive memory architectures
- Online learning and test time training

What is Learning? What is Memorization?

- Can we learn without memory?
- Can we memorize without learning?

Memory

Memory is a neural update caused by an input.

Learning

Learning is the process of acquiring effective and useful memory.

What is Learning? What is Memorization?

- Can we learn without memory? No!
- Can we memorize without learning? Yes!

Memory

Memory is a neural update caused by an input.

Learning

Learning is the process of acquiring effective and useful memory.

What is Learning? What is Memorization?

- Can we learn without memory? No!
- Can we memorize without learning? Yes!

Memory

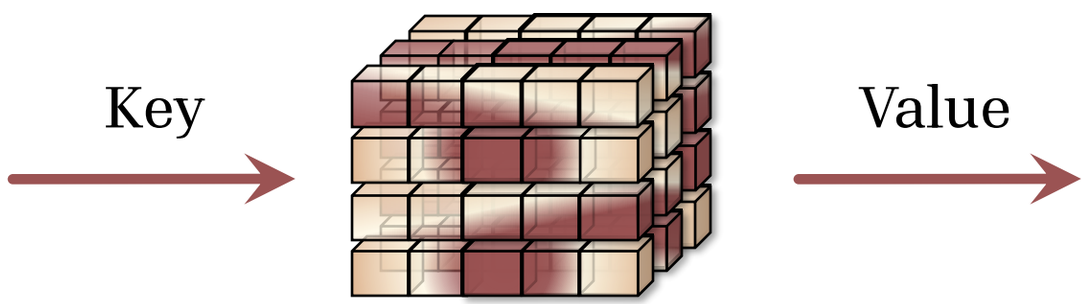
Memory is a neural update caused by an input.

Learning

Learning is the process of acquiring effective and useful memory.

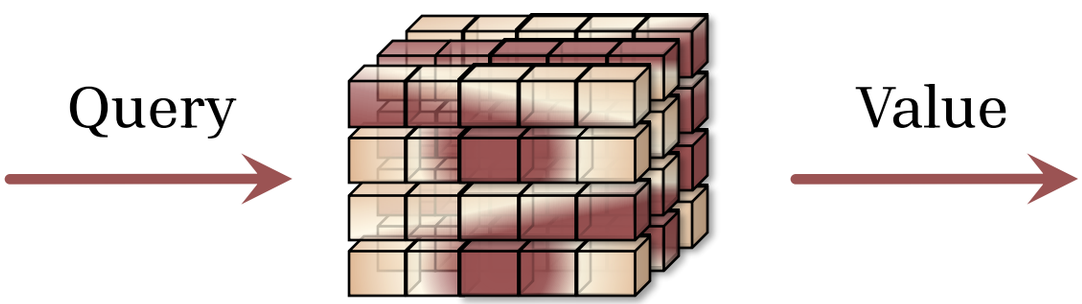
Associative Memory

- Is the ability to remember the relationship between (~~unrelated~~) events/items/...
- Think of it as a dictionary:
  - We have keys and values
  - Learning is the process of acquiring the mapping!
- **Training**:



Associative Memory

- Is the ability to remember the relationship between (~~unrelated~~) events/items/...
- Think of it as a dictionary:
  - We have keys and values
  - Learning is the process of acquiring the mapping!
- **Inference**:

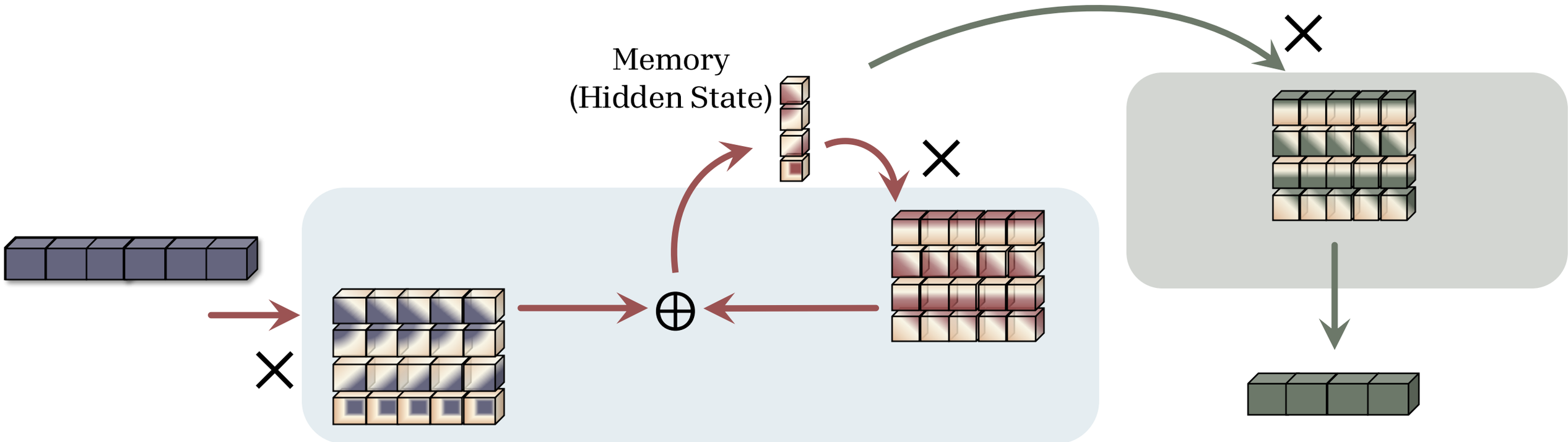


What are RNNs in This Perspective?

- There is a fixed-size memory (called hidden state):
  - We aim to compress the past data into this memory
  - Write: Given an incoming data, we compress and add it to the memory!
  - Read: Send a query and receive the corresponding information from the memory!

What are RNNs in This Perspective?

- Write: Given an incoming data, we compress and add it to the memory!
- Read: Send a query and receive the corresponding information from the memory!



Let’s Get Inspired from Our Brain!

- Human brain is effective and efficient!
- Our brain constantly memorizes!
- We have short-term and long-term memory!

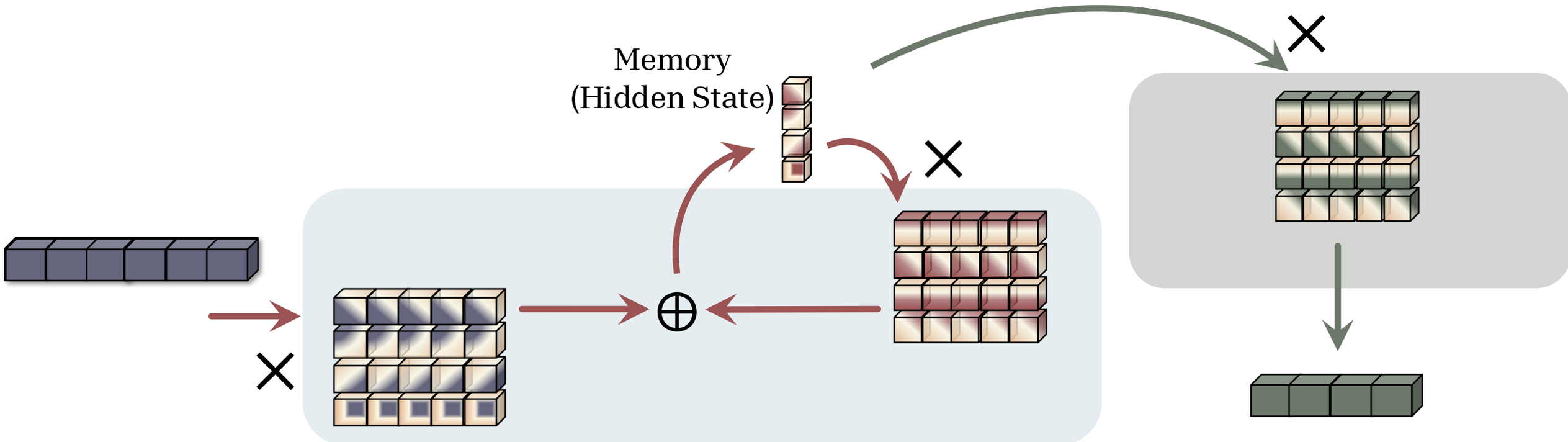
- Our short-term memory is accurate (like attention) but for long-term memories we might hallucinate (like fading memories)!
- Our memory is a neural architecture, not a simple set of neurons.

Let’s Get Inspired from Our Brain!

- Human brain is effective and efficient!
- Our brain constantly memorizes!
- We have short-term and long-term memory!

- Our short-term memory is accurate (like attention) but for long-term memories we might hallucinate (like fading memories)!
- **Our memory is a neural architecture, not a simple set of neurons.**

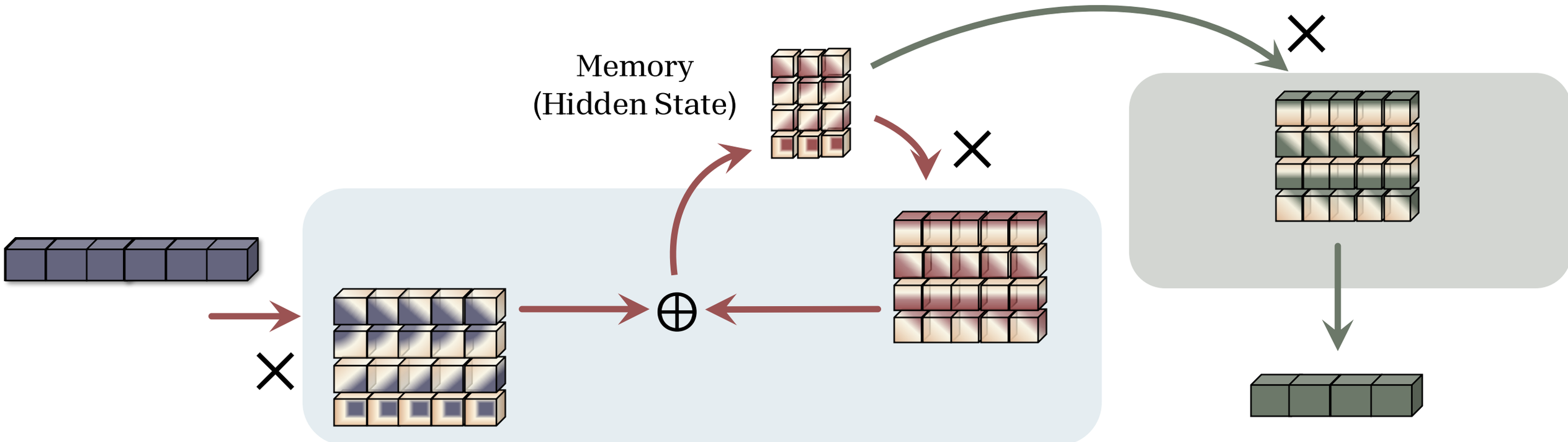
First Generation of RNNs



$$h_t = Ah_{t-1} + B\,x_t$$

$$y_t = Ch_t$$

Second Generation of RNNs

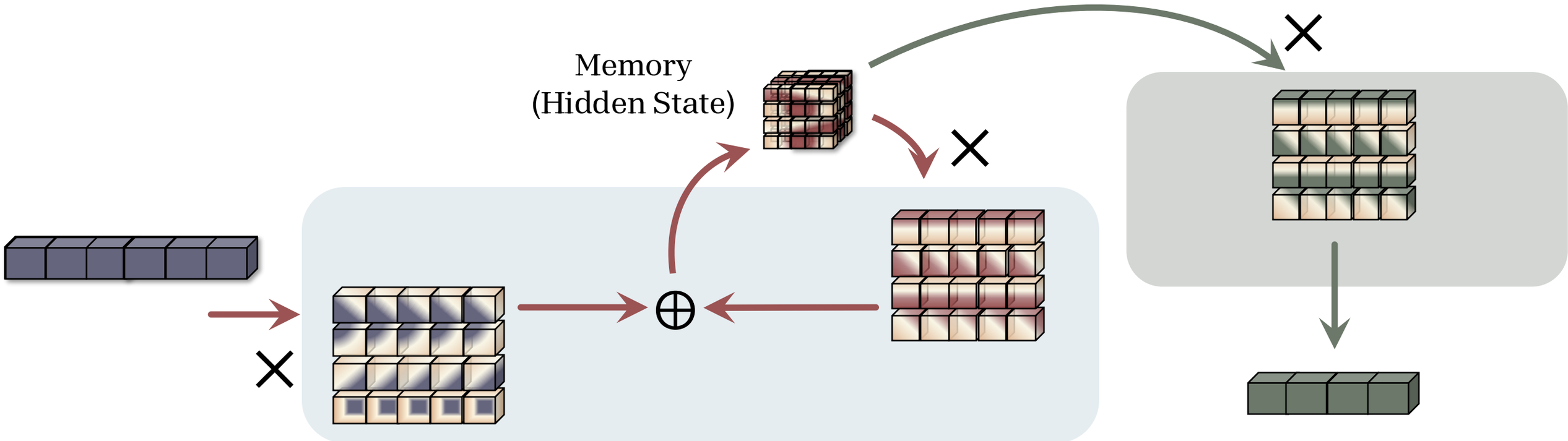


$$h_t = Ah_{t-1} + B\,x_t$$

$$y_t = Ch_t$$

Linear Transformers, DeltaNet, RWKV, ...

Let's Generalize this Process!



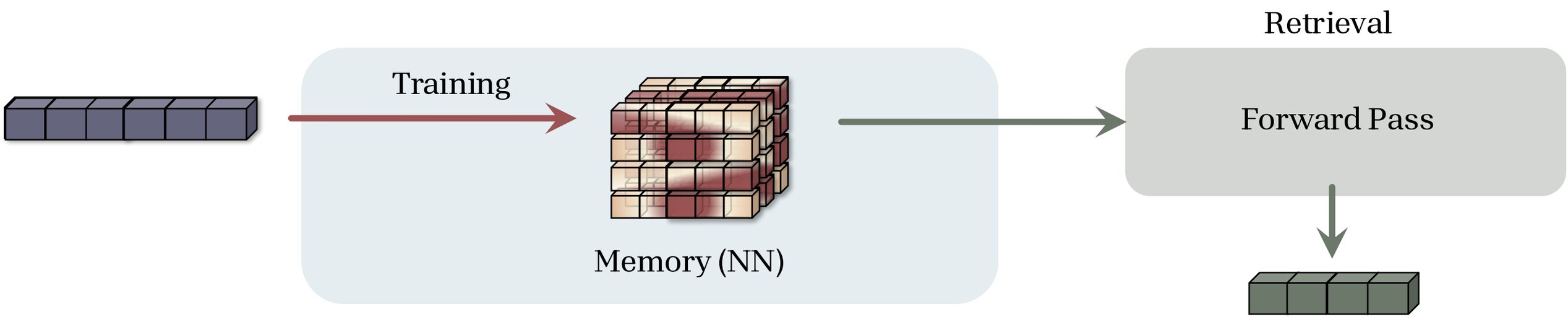
$$h_t = Ah_{t-1} + B\,x_t$$

$$y_t = Ch_t$$

Several papers with different perspectives ...

How to Generalize the Process?

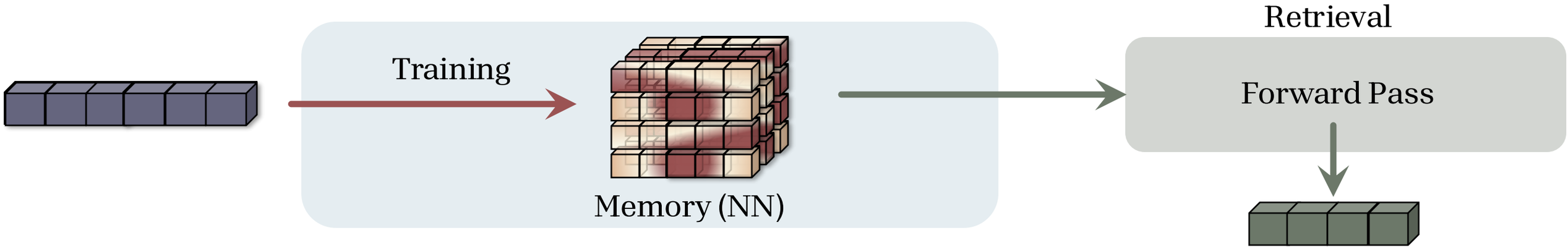
- Neural networks are great memorizers of their training data!
- We train a neural network so it learns how to compress the data into its parameters!



What are We Learning?

- Associative Memory: Learning the mapping!
  - What is the objective?

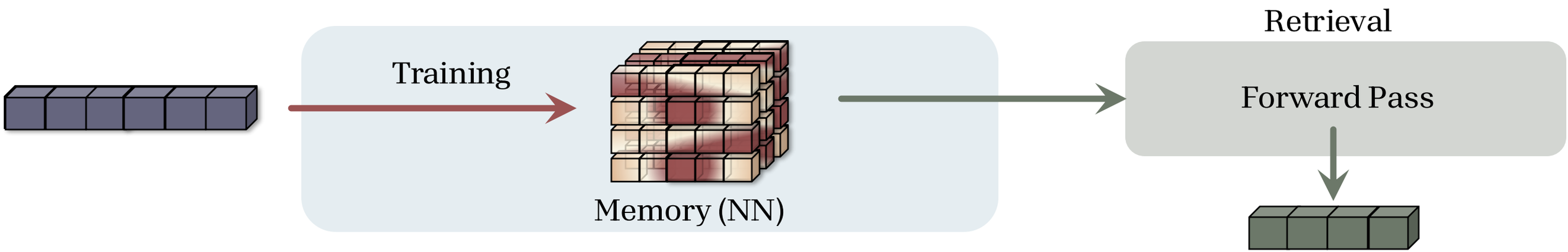
$$\underset{m \in \mathcal{M}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{T} \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2$$



What are We Learning?

- Associative Memory: Learning the mapping!
  - What tokens should be remembered?

$$M_t = M_{t-1} - \theta_t \underbrace{\nabla \ell(M_{t-1}; x_t)}_{\text{Surprise}}$$



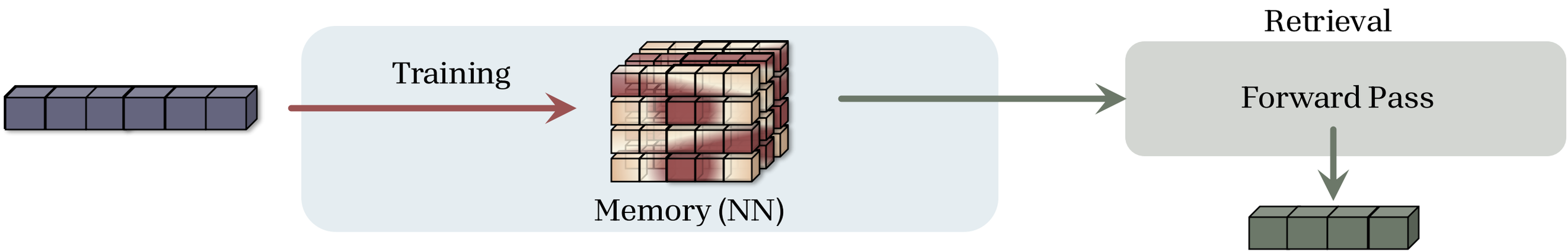
What are We Learning?

- Associative Memory: Learning the mapping!
  - What tokens should be remembered?

~~$M_t = M_{t-1} - \theta_t \underbrace{\nabla \ell(M_{t-1}; x_t)}_{\text{Surprise}}$~~

$$M_t = M_{t-1} + S_t,$$

$$S_t = \eta_t \underbrace{S_{t-1}}_{\text{Past Surprise}} - \theta_t \underbrace{\nabla \ell\left(M_{t-1}; x_t\right)}_{\text{Momentary Surprise}}$$

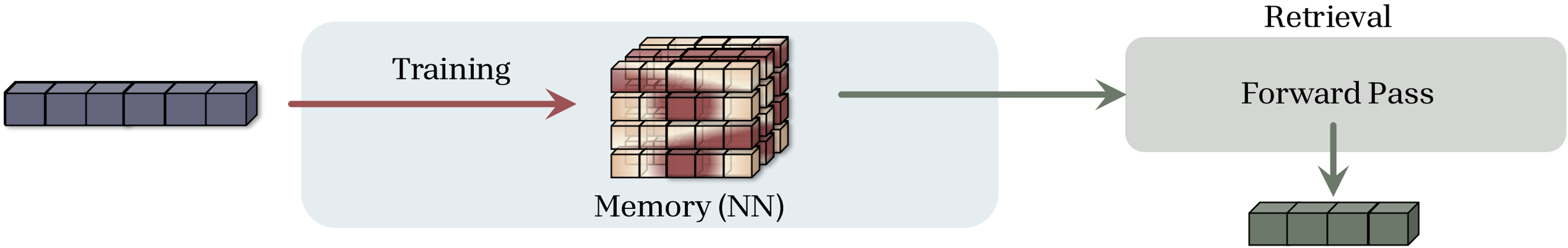


What are We Learning?

- Associative Memory: Learning the mapping!
  - What tokens should be **forgotten**?

$$\mathcal{M}_t = \mathcal{M}_{t-1} + S_t,$$
$$S_t = \eta_t \underbrace{S_{t-1}}_{\text{Past Surprise}} - \theta_t \underbrace{\nabla \ell\left(M_{t-1}; x_t\right)}_{\text{Momentary Surprise}}$$

$$\mathcal{M}_t = (1 - \alpha_t)\mathcal{M}_{t-1} + S_t,$$
$$S_t = \eta_t S_{t-1} - \theta_t \nabla \ell\left(M_{t-1}; x_t\right)$$



How to Train This?

✶ The recurrence in this formulation is non-linear and so is not parallelizable!

$$\mathcal{M}_t = (1 - \alpha_t)\mathcal{M}_{t-1} + S_t,$$
$$S_t = \eta_t S_{t-1} - \theta_t \nabla \ell\left(M_{t-1}; x_t\right)$$

✶ Divide the sequence into subsequences:

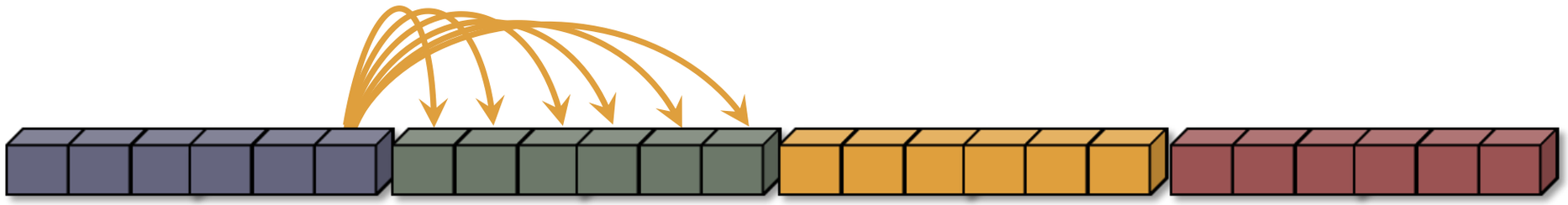
$$S_t = \eta_t S_{t-1} - \theta_t u_t$$
$$u_t = \nabla \ell\left(M_{t'}; x_t\right)$$

How to Train This?

✶ The recurrence in this formulation is non-linear and so is not parallelizable!

$$\mathcal{M}_t = (1 - \alpha_t)\mathcal{M}_{t-1} + S_t,$$
$$S_t = \eta_t S_{t-1} - \theta_t \cancel{\nabla \ell\left(M_{t-1}; x_t\right)} \nabla \ell\left(M_{t'}; x_t\right)$$

✶ Divide the sequence into subsequences:



$$S_t = \eta_t S_{t-1} - \theta_t u_t$$
$$u_t = \nabla \ell\left(M_{t'}; x_t\right)$$

How to Train This?

- This is still a recurrent model!
  - We need to write it as matrix multiplications!

$$\mathcal{M}_t = (1-\alpha_t)\mathcal{M}_{t-1} - \theta_t \nabla \ell(\mathcal{M}_{t-1}; x_t) = \beta_t \mathcal{M}_0 - \sum_{i=1}^{t} \theta_i \frac{\beta_t}{\beta_i} \nabla \ell(\mathcal{M}_{t'}; x_i),$$

- We can reformulate the above as: ($\nabla \ell(W_0; x_t) = (W_0 k_t - v_t) x_t^\top$)

$$\Rightarrow \sum_{i=1}^{b} \theta_i \frac{\beta_b}{\beta_i} \nabla \ell(W_0; x_i) = \Theta_b \mathbf{B}_b (W_0 K - V) X^\top,$$

How to Train This?

- This is still a recurrent model!
  - We need to write it as matrix multiplications!

$$\beta_i = \prod_{j=1}^{i}(1-\alpha_j)$$

$$\mathcal{M}_t = (1-\alpha_t)\mathcal{M}_{t-1} - \theta_t \nabla \ell(\mathcal{M}_{t-1}; x_t) = \beta_t \mathcal{M}_0 - \sum_{i=1}^{t} \theta_i \frac{\beta_t}{\beta_i} \nabla \ell(\mathcal{M}_{t'}; x_i),$$

- We can reformulate the above as: ($\nabla \ell(W_0; x_t) = (W_0 k_t - v_t) x_t^\top$)

$$\Rightarrow \sum_{i=1}^{b} \theta_i \frac{\beta_b}{\beta_i} \nabla \ell(W_0; x_i) = \Theta_b \mathbf{B}_b (W_0 K - V) X^\top,$$

How to Train This?

- This is still a recurrent model!
  - We need to write it as matrix multiplications!

$$\mathcal{M}_t = (1-\alpha_t)\mathcal{M}_{t-1} - \theta_t \nabla \ell(\mathcal{M}_{t-1}; x_t) = \beta_t \mathcal{M}_0 - \sum_{i=1}^{t} \theta_i \frac{\beta_t}{\beta_i} \nabla \ell(\mathcal{M}_{t'}; x_i),$$

$$\beta_i = \prod_{j=1}^{i} (1-\alpha_j)$$

- We can reformulate the above as: ($\nabla \ell(W_0; x_t) = (W_0 k_t - v_t) x_t^\top$)

$$\Rightarrow \sum_{i=1}^{b} \theta_i \frac{\beta_b}{\beta_i} \nabla \ell(W_0; x_i) = \Theta_b \mathbf{B}_b (W_0 K - V) X^\top,$$

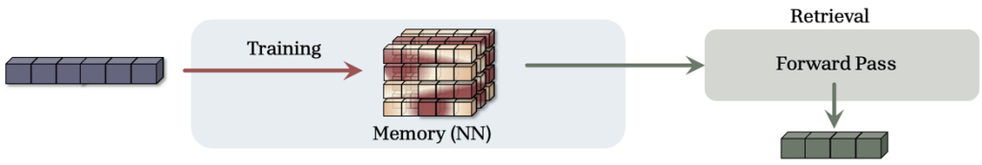
Recap!

- How to design an effective long-term memory!
- How to train long-term memory in a paralleliza
- How to incorporate short-term and long-term r
- How does it work in practice?

What are We Learning?

- Associative Memory: Learning the mapping!
  - What tokens should be **forgotten**?

$$\mathcal{M}_t = \mathcal{M}_{t-1} + S_t, \quad S_t = \eta_t \underbrace{S_{t-1}}_{\text{Past Surprise}} - \theta_t \underbrace{\nabla \ell\left(M_{t-1}; x_t\right)}_{\text{Momentary Surprise}} \longrightarrow \mathcal{M}_t = (1-\alpha_t)\mathcal{M}_{t-1} + S_t, \quad S_t = \eta_t S_{t-1} - \theta_t \nabla \ell\left(M_{t-1}; x_t\right)$$



Recap!

- How to design an effective long-term memory!
- How to train long-term memory in a parallelizable manner!
- How to incorporate short-term and long-term memory?
- H

How to Train This?

- This is still a recurrent model!
  - We need to write it as matrix multiplications!

$$\mathcal{M}_t = (1-\alpha_t)\mathcal{M}_{t-1} - \theta_t \nabla \ell(\mathcal{M}_{t-1}; x_t) = \beta_t \mathcal{M}_0 - \sum_{i=1}^{t} \theta_i \frac{\beta_t}{\beta_i} \nabla \ell(\mathcal{M}_{t'}; x_i), \qquad \beta_i = \prod_{j=1}^{i}(1-\alpha_j)$$

- We can reformulate the above as: ($\nabla \ell(W_0; x_t) = (W_0 k_t - v_t) x_t^\top$)

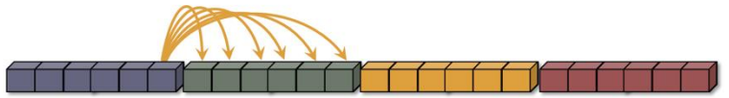
$$\Rightarrow \sum_{i=1}^{b} \theta_i \frac{\beta_b}{\beta_i} \nabla \ell(W_0; x_i) = \Theta_b \mathbf{B}_b (W_0 K - V) X^\top,$$

How to Train This?

- The recurrence in this formulation is non-linear and so is not parallelizable!

$$\mathcal{M}_t = (1-\alpha_t)\mathcal{M}_{t-1} + S_t,$$
$$S_t = \eta_t S_{t-1} - \theta_t \cancel{\nabla \ell(\mathcal{M}_{t-1}; x_t)} \nabla \ell(M_{t'}; x_t)$$

- Divide the sequence into subsequences:



$$S_t = \eta_t S_{t-1} - \theta_t u_t$$
$$u_t = \nabla \ell(M_{t'}; x_t)$$

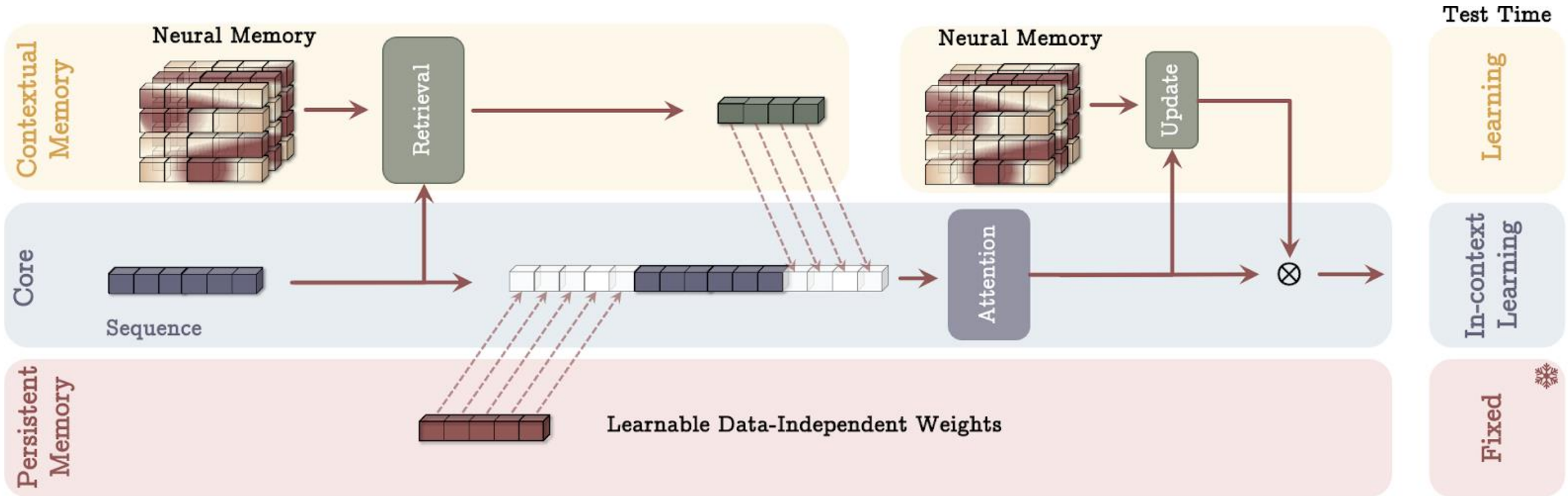
Recap!

- How to design a long-term memory!
- How to train long-term memory in a parallelizable manner!
- How to incorporate short-term and long-term memory?
- How does it work in practice?

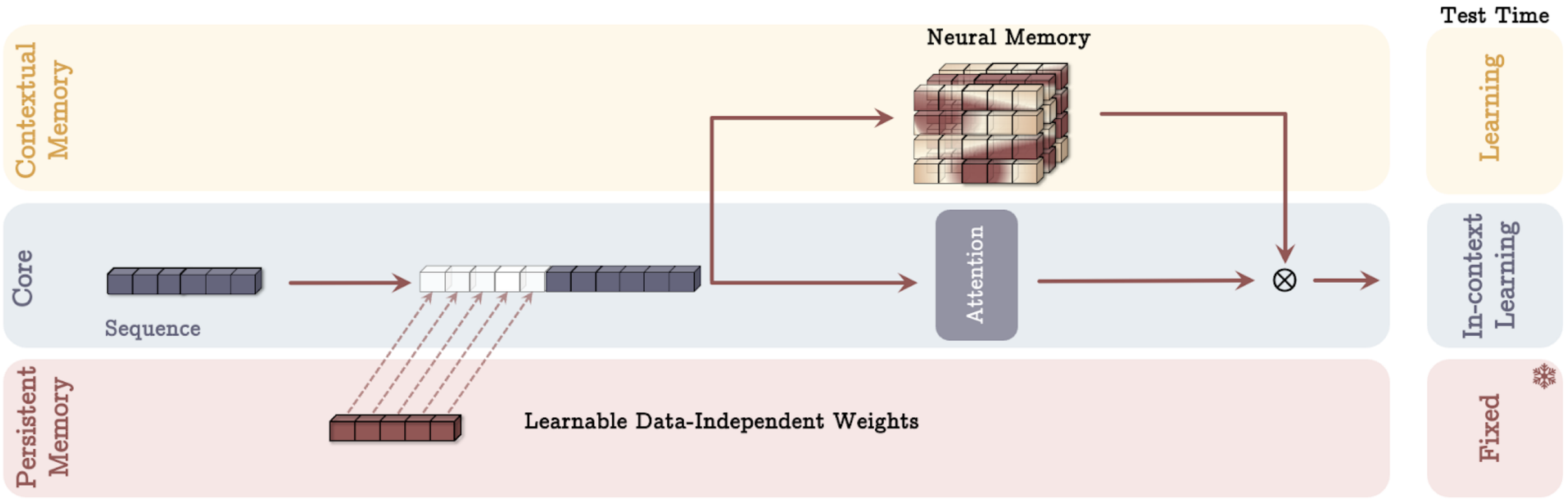
Recap!

- How to design a long-term memory!
- How to train long-term memory in a parallelizable manner!
- How to incorporate short-term and long-term memory?
- How does it work in practice?

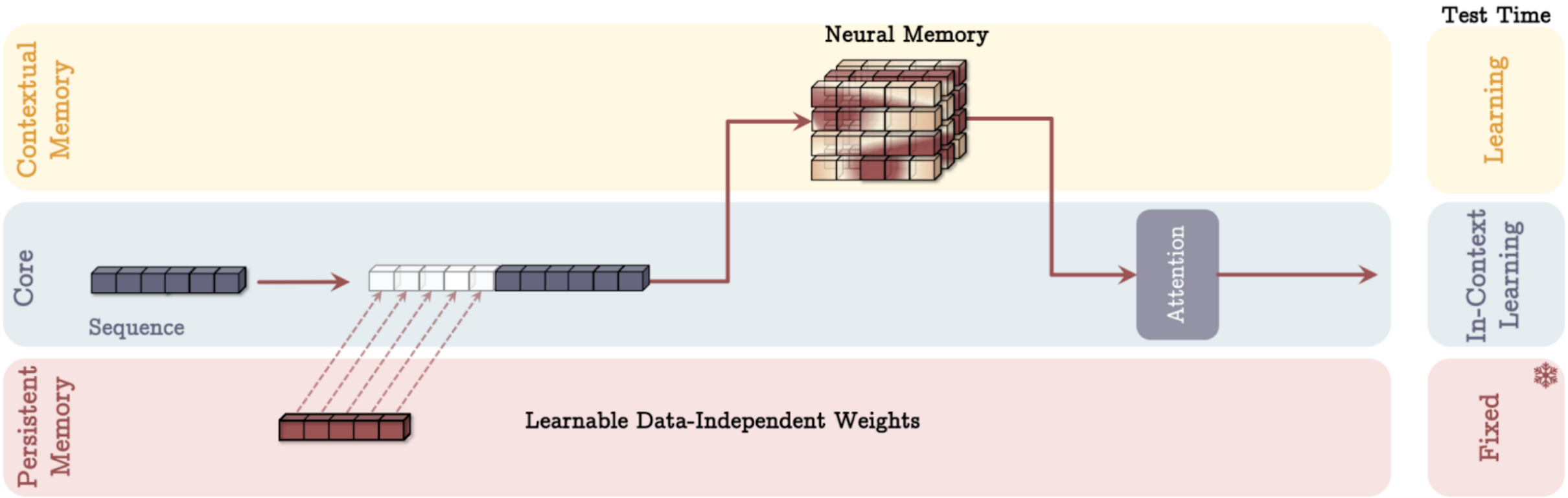
Memory As Context (MAC)



Memory As Gate (MAG)



Memory As Layer (MAL)



Experimental Results: Language Modeling

Model	Wiki. ppl ↓	LMB. ppl ↓	LMB. acc ↑	PIQA acc ↑	Hella. acc_n ↑	Wino. acc ↑	ARC-e acc ↑	ARC-c acc_n ↑	SIQA acc ↑	BoolQ acc ↑	Avg. ↑
760M params / 30B tokens											
Transformer++	25.21	27.64	35.78	66.92	42.19	51.95	60.38	32.46	39.51	60.37	48.69
Mamba	28.12	23.96	32.80	66.04	39.15	52.38	61.49	30.34	37.96	57.62	47.22
DeltaNet	24.37	24.60	37.06	66.93	41.98	50.65	64.87	31.39	39.88	59.02	48.97
TTT	24.17	23.51	34.74	67.25	43.92	50.99	64.53	33.81	40.16	59.58	47.32
Gated DeltaNet	21.18	22.09	35.54	68.01	44.95	50.73	66.87	33.09	39.21	59.14	49.69
Samba*	20.63	22.71	39.72	69.19	47.35	52.01	66.92	33.20	38.98	61.24	51.08
Gated DeltaNet-H2*	19.88	20.83	39.18	68.95	48.22	52.57	67.01	35.49	39.39	61.11	51.49
Titans (LMM)	20.04	21.96	37.40	69.28	48.46	52.27	66.31	35.84	40.13	62.76	51.56
Titans (MAC)	19.93	20.12	39.62	70.46	49.01	53.18	67.86	36.01	41.87	62.05	52.51
Titans (MAG)	18.61	19.86	40.98	70.25	48.94	52.89	68.23	36.19	40.38	62.11	52.50
Titans (MAL)	19.07	20.33	40.05	69.99	48.82	53.02	67.54	35.65	30.98	61.72	50.97

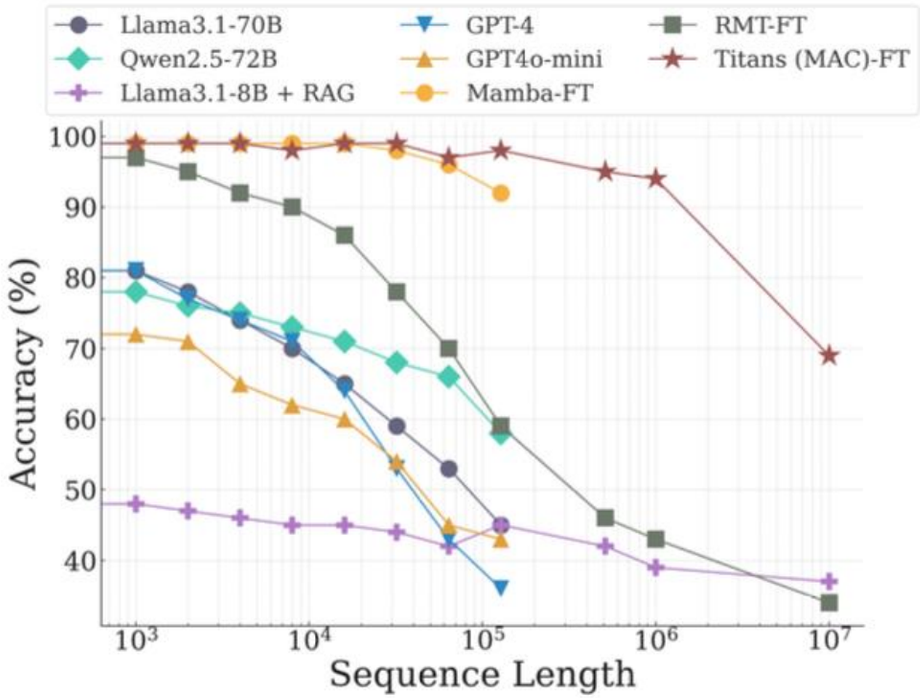
Experimental Results: Long Context (RULER)

Model	S-NIAH-PK				S-NIAH-N				S-NIAH-W		
	2K	4K	8K	16K	2K	4K	8K	16K	2K	4K	8K
TTT	98.4	98.8	98.0	88.4	60.2	36.6	10.2	4.4	78.8	28.0	4.4
Mamba2	98.6	61.4	31.0	5.4	98.4	55.8	14.2	0.0	42.2	4.2	0.0
DeltaNet	96.8	98.8	98.6	71.4	47.2	15.4	12.8	5.4	46.2	20.0	1.6
Titans (LMM)	99.8	98.4	98.2	96.2	100.0	99.8	93.4	80.2	90.4	89.4	85.8
Titans (MAC)	99.2	98.8	99.0	98.4	99.6	98.2	97.6	97.4	98.2	98.2	95.6
Titans (MAG)	99.4	98.0	97.4	97.4	99.2	98.8	97.2	98.6	98.0	98.0	90.2
Titans (MAL)	98.8	98.6	98.8	97.8	99.8	98.1	96.8	96.4	98.0	97.4	92.0

- ✶ Momentum is very effective for longer context.
- ✶ Deep memory is important.
- ✶ Architecture matters in hybrid models.

Experimental Results: Long Context

- BABILong Benchmark: Needle in haystack-style state tracking task.
- Architecture matters!
- Fine-tuning helps to understand what information should be stored in the memory for this task.



Experimental Results: Ablation Study

Model	Language Modeling ppl ↓	Reasoning acc ↑	Long Context acc ↑
LMM	27.01	47.83	92.68
+Attn (MAC)	26.67	48.65	97.95
+Attn (MAG)	25.70	48.60	96.70
+Attn (MAL)	25.91	47.87	96.91
Linear Memory	28.49	46.97	85.34
w/o Convolution	28.73	45.82	90.28
w/o Momentum	28.98	45.49	87.12
w/o Weight Decay	29.04	45.11	85.60
w/o Persistent Memory	27.63	46.35	92.49

Thank you!

alibehrouz@cs.cornell.edu
alibehrouz@google.com

@behrouz_ali

Google Research