Associative Memory with Heavy-Tailed Data

Alberto Bietti
Flatiron CCM

Vivien Cabannes
Meta AI

Elvis Dohmatob
Meta AI

Leon Bottou
Meta AI

Hervé Jegou
Kyutai

Meta AI

Context: renewed interest for associative memories in LLM settings

Augmenting Self-attention with Persistent Memory

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, Armand Joulin
Facebook AI Research
sainbar,egrave,guismay,rvj,ajoulin@fb.com

HOPFIELD NETWORKS IS ALL YOU NEED

Hubert Ramsauer[*] **Bernhard Schäfl**[*] **Johannes Lehner**[*] **Philipp Seidl**[*]
Michael Widrich[*] **Thomas Adler**[*] **Lukas Gruber**[*] **Markus Holzleitner**[*]
Milena Pavlović[‡,§] **Geir Kjetil Sandve**[§] **Victor Greiff**[‡] **David Kreil**[†]
Michael Kopp[†] **Günter Klambauer**[*] **Johannes Brandstetter**[*] **Sepp Hochreiter**[*,†]

Energy Transformer

Benjamin Hoover[*]
IBM Research
Georgia Tech
benjamin.hoover@ibm.com

Yuchen Liang[*]
Department of CS
RPI
liangy7@rpi.edu

Bao Pham[*]
Department of CS
RPI
phamb@rpi.edu

Rameswar Panda
MIT-IBM Watson AI Lab
IBM Research
rpanda@ibm.com

Hendrik Strobelt
MIT-IBM Watson AI Lab
IBM Research
hendrik.strobelt@ibm.com

Duen Horng Chau
College of Computing
Georgia Tech
polo@gatech.edu

Mohammed J. Zaki
Department of CS
RPI
zaki@cs.rpi.edu

Dmitry Krotov
MIT-IBM Watson AI Lab
IBM Research
krotov@ibm.com

Why it makes sense to think of learning in terms of memory?

Arguably, learning is about discovery and memorization of abstract rules

I.e., find the right hierarchical patterns, and memorize them for future pattern matching

Contribution: A throughout study of a simple associative memory model

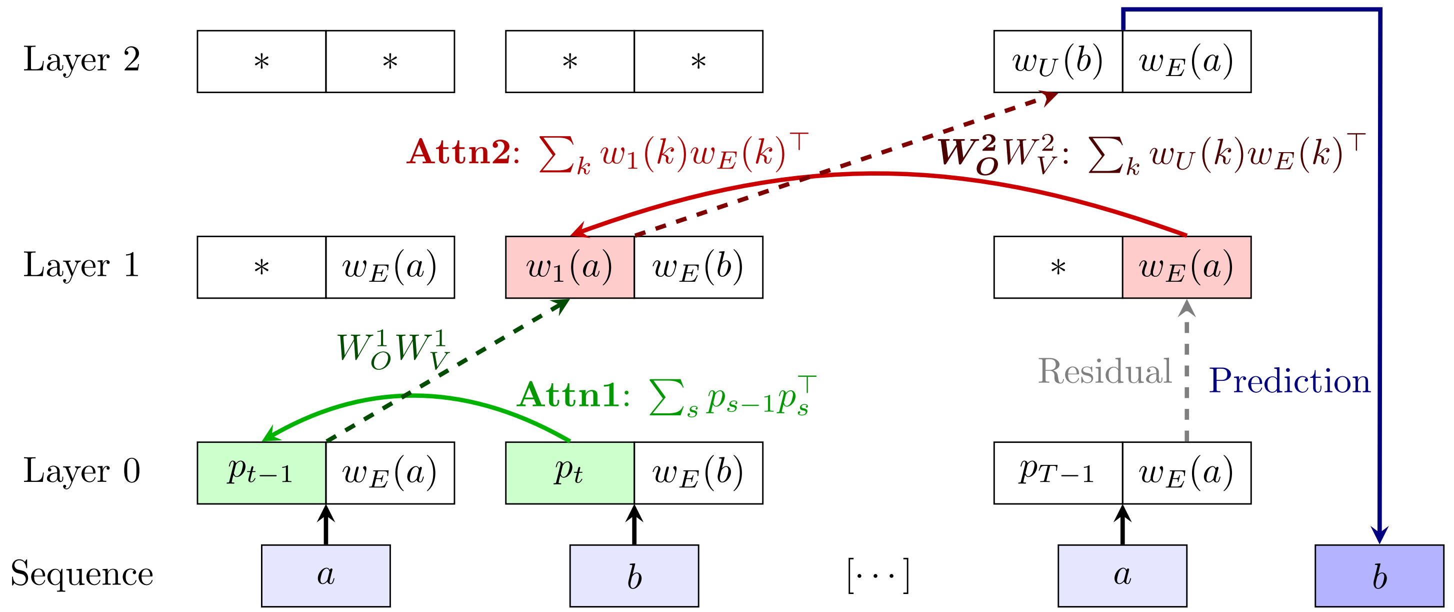
This models stems from our paper *“Birth of a Transformer” (NeurIPS 2023 Spotlight)*

How is it related to transformers?

Those memory blocks can describe induction heads,

which are the foundations of circuits,

believed to explain transformers



Setup

Data

Discrete input $x \in \mathbb{N}$, discrete output $y \in \mathbb{N}$ with Zipf law

$$p(x) \propto x^{-\alpha}, \qquad p(y \mid x) = \delta_{f_*(x)}(y)$$

Model

Embeddings	Latent transformation	Probability score	Input/output rule
$e_x, u_y \sim \mathcal{N}(0, I) \in \mathbb{R}^d$	$W \in \mathbb{R}^{d \times d}$	$p_W(y \mid x) \propto \exp(u_y^\top W e_x)$	$f_W(x) = \arg\max_y p_W(y \mid x)$

Associative memory parametrization

$$W = \sum_{x,y} q(x, y) u_y e_x^\top \qquad (\mathrm{span}(u_y \otimes e_x)_{x,y} = \mathrm{span}(1_i \otimes 1_j)_{ij})$$

Measure of success

$$\mathscr{E}(W) = \mathbb{E}_{(x,y)\sim p}[\ell(f_W(x), y)] \qquad \text{with} \qquad \ell(f(x), y) = 1_{f(x)\neq y}$$

Surrogate training objective: the cross-entropy loss

$$\mathscr{L}(W) = \mathbb{E}_{(x,y)\sim p}[\ell_S(W; x, y)] \qquad \text{with} \qquad \ell_S(W; x, y) = -\log p_W(y\,|\,x)$$

Training data

$$(x_t, y_t) \sim p \qquad \text{for} \qquad t = 1,\ldots, T$$

Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \ldots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x)\neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Statistical Study

Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Approximation guarantees

When $W = \sum q(x) u_{f_*(x)} e_x^\top$, $\qquad \mathbb{E}_{e,u}[\mathscr{E}(W)] = p\left(\{x \mid q(x) d < c\|q\|_2^2\}\right)$

Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Approximation guarantees

When $W = \sum q(x) u_{f_*(x)} e_x^\top$, $\quad \mathbb{E}_{e,u}[\mathscr{E}(W)] = p\left(\{x \,|\, q(x) d < c \|q\|_2^2\}\right)$

Proof: Develop the model

$$f_W(r) = \arg\max \langle u_y, q(r) \|e_r\|^2 u_{f_*(r)} + \sum_{x \neq r} q(x) e_x^\top e_r \cdot u_{f_*(x)} \rangle$$

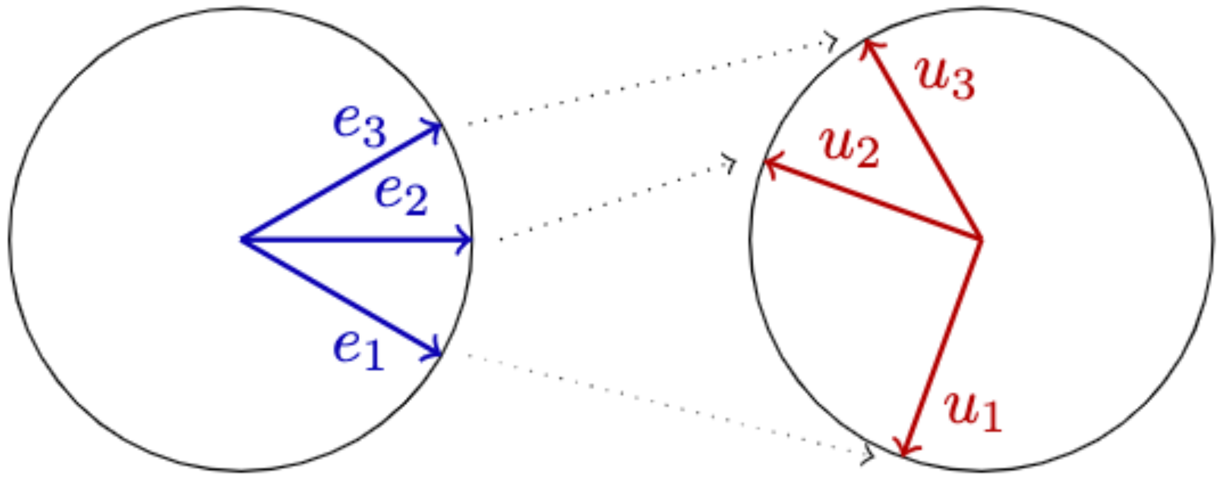
Interference between memories

$$f_W(x) \neq y \quad \Longleftrightarrow \quad q(r) \|e_r\|^2 \|u_{f_*(r)}\|^2 < \max_y \sum_x q(x) e_x^\top e_r \, u_{f_*(x)}^\top u_y$$

Need to ensure the right score maximizer

With permutation of expectations,

the problem reduces to max of Gaussian deviation



Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Approximation guarantees

When $W = \sum q(x) u_{f_*(x)} e_x^\top$, $\quad \mathbb{E}_{e,u}[\mathcal{E}(W)] = p\left(\{x \mid q(x)d < c\|q\|_2^2\}\right)$

Finite-sample complexity

When $q_T = F((\{t \mid x_t = x\})_x)$, $\quad \mathcal{E}(q_T) - \mathcal{E}(q_\infty) = \int p(x) e^{-Tx} \mathrm{d}x$

Proof: Binning samples output according to their empirical frequencies (Csiszár's type method)

Worse case deviation for $q(x)$ is controlled by $e^{-Tp(x)}$

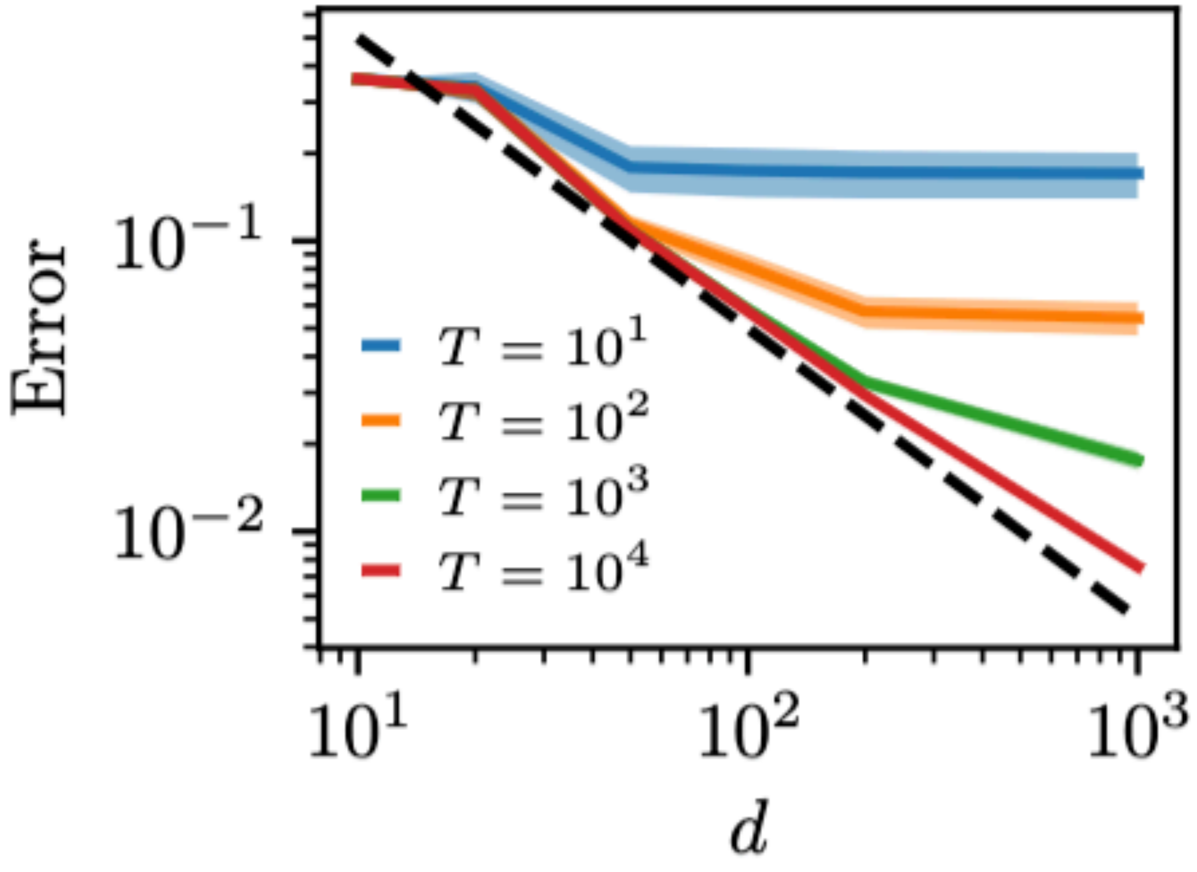
Linearity of expectation leads to the summation

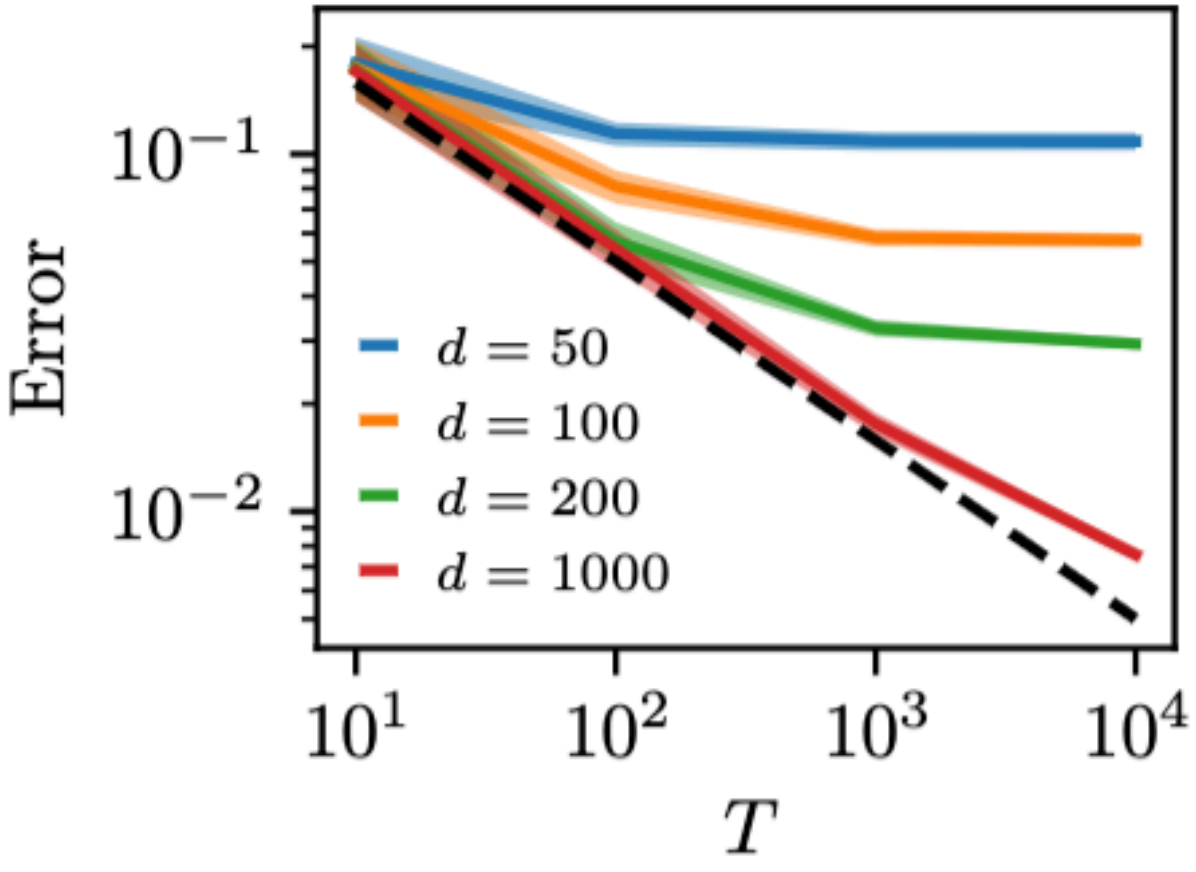
Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Instantiation for heavy tailed data

$$\mathbb{E}_{e,u}[\mathscr{E}(W)] = p\left(\{x \,|\, q(x) d > c\|q\|_2^2\}\right) + \int p(x) e^{-Tx} \mathrm{d}x \qquad\qquad p(x) \propto x^{-\alpha}$$

Model	Error scaling	Comment
$q(x) = p(x)$	$d^{-(\alpha-1)/2\alpha} + T^{-1+1/\alpha}$	Found with large batches in one step
$q(x) = \mathbf{1}_{x \leq d}$	$d^{-\alpha+1} + T^{-1+1/\alpha}$	Optimal scaling with random embeddings





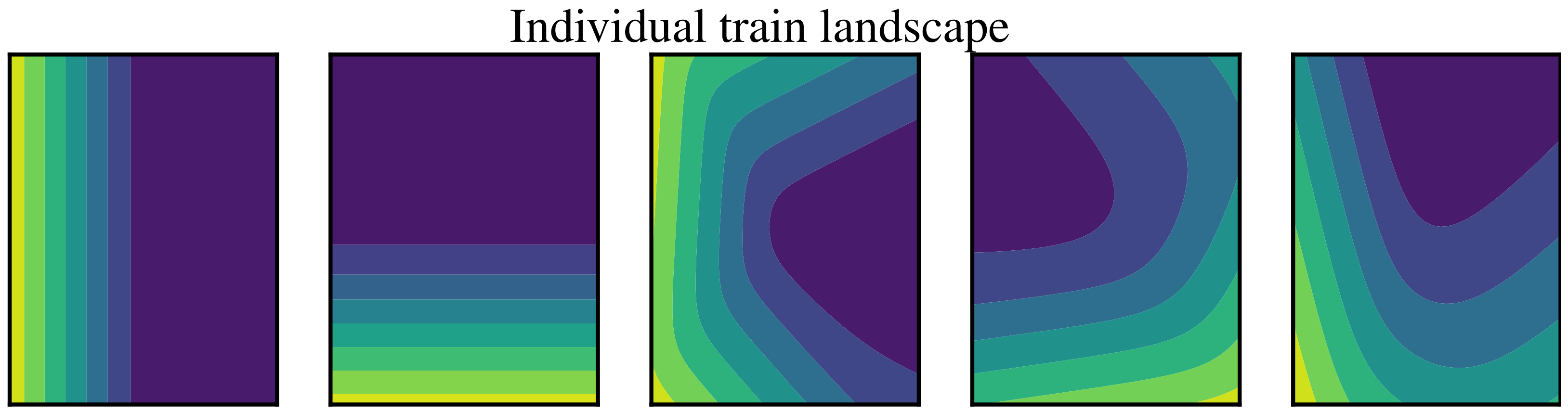
Optimization Study

Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Training dynamics

$$\mathscr{L}(W) = \mathbb{E}_{(x,y)\sim p}[\ell_S(W; x, y)] \qquad \text{with} \qquad \ell_S(W; x, y) = -\log p_W(y \mid x)$$

Each association (x, y) creates a landscape $W \mapsto \ell_S(W; x, y)$ that pushes towards $u_y e_x^\top \in \mathbb{R}^{d\times d}$



Level lines examples with $n = 5$ tokens x, in dimension $d = 2$ on $\mathrm{Span}(e_i e_i^\top)$

Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Training dynamics

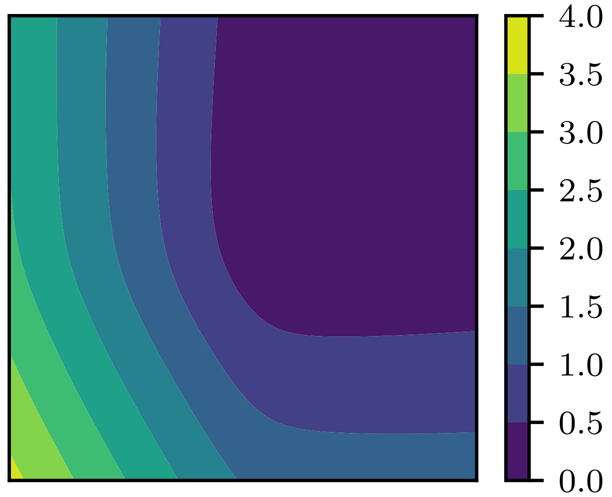
$$\mathscr{L}(W) = \mathbb{E}_{(x,y)\sim p}[\ell_S(W; x, y)] \quad \text{with} \quad \ell_S(W; x, y) = -\log p_W(y \mid x)$$

Each association (x, y) creates a landscape $W \mapsto \ell_S(W; x, y)$ that pushes towards $u_y e_x^\top \in \mathbb{R}^{d\times d}$

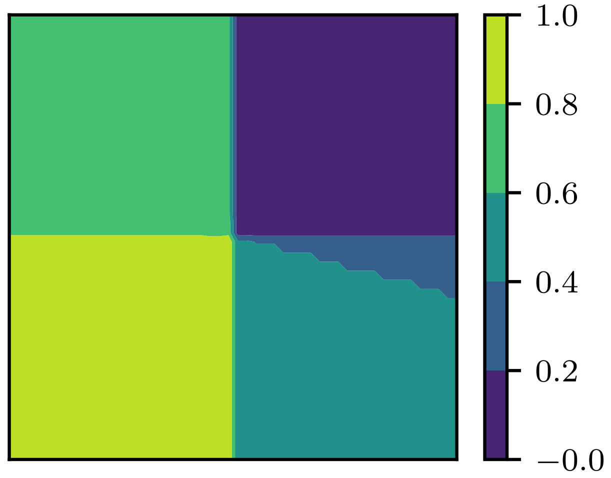
Individual train landscape

Level lines examples with $n = 5$ tokens x, in dimension $d = 2$ on $\mathrm{Span}(e_i e_i^\top)$

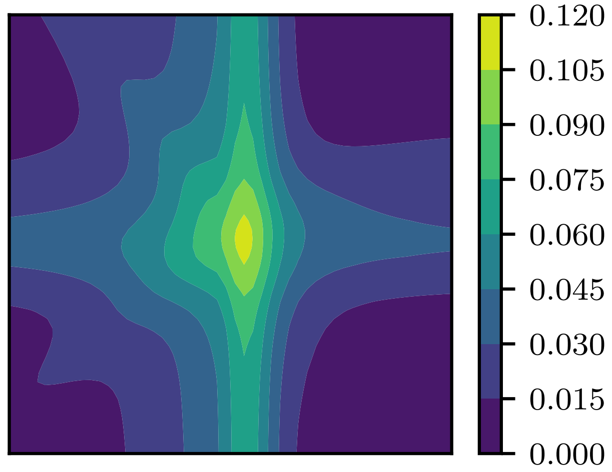
Population train landscape



Population test landscape



Sharpness



Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

Training dynamics

$$\mathscr{L}(W) = \mathbb{E}_{(x,y)\sim p}[\ell_S(W; x, y)] \qquad \text{with} \qquad \ell_S(W; x, y) = -\log p_W(y \mid x)$$

Each association (x, y) creates a landscape $W \mapsto \ell_S(W; x, y)$ that pushes towards $u_y e_x^\top \in \mathbb{R}^{d\times d}$

$$W_{t+1} = W_t - \gamma_t \sum_{x \in B_t} \nabla_W \ell(W; x, f_*(x))$$

$p(x) \propto x^{-\alpha}$

If d is large enough compared to the number of frequent tokens (or α the vanishing rates of p), SGD updates take place in quasi-orthogonal direction (negligible memory interference).

In this setting the dynamic can be decoupled on the different $q(x)$ in $W_t = \sum_x q_t(x) u_{f_*(x)} e_x^\top$

$$q_T(x) = F(\#\{t \mid x_t = x\}) \qquad \text{with} \qquad F(n) = \underbrace{f \circ f \circ \cdots \circ f}_{\times n}(0) \qquad \text{and} \qquad f : x \mapsto x + \frac{\gamma}{1 + M^{-1}\exp(x)}$$

Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \ldots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

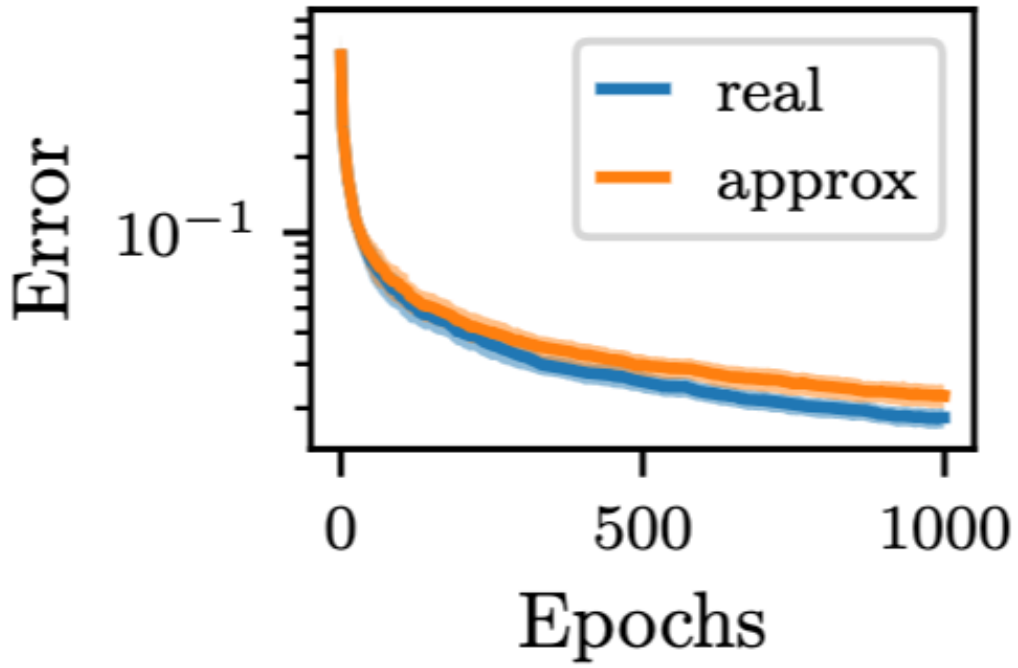
Training dynamic

Model	Error scaling	Comment
$q(x) = p(x)$	$d^{-(\alpha-1)/2\alpha} + T^{-1+1/\alpha}$	Found with large batches in one step
$q(x) = \mathbf{1}_{x \leq d}$	$d^{-\alpha+1} + T^{-1+1/\alpha}$	Optimal scaling with random embeddings

$$W_{t+1} = W_t - \gamma_t \sum_{x \in B_t} \nabla_W \ell(W; x, f_*(x)) \qquad\qquad W_t = \sum_x q_t(x) u_{f_*(x)} e_x^\top$$

$$q_T(x) = F(\{t \mid x_t = x\}) \quad \text{with} \quad F(n) = \underbrace{f \circ f \circ \cdots \circ f}_{\times n}(0) \quad \text{and} \quad f : x \mapsto x + \frac{\gamma}{1 + M^{-1}\exp(x)}$$

Approximation matches practice, it can be used to predict the $q_T(x)$ for different learning rates



Tokens	Embeddings	Model	Scaling
$y_{i_t} = f_*(x_{i_t})$ $t \in \{1, 2, \dots, T\}$	$e_x, u_y \in \mathbb{R}^d$ $e_x \sim \mathcal{N}(0, I)$	$W = \sum_x q(x) u_{f_*(x)} e_x^\top$ $f_q(x) = \arg\max_y u_y W e_x$	$\mathcal{E}(q) = \mathbb{E}[\mathbf{1}_{f_q(x) \neq f_*(x)}]$ $\mathcal{E}(q) = F(d, T; q)$

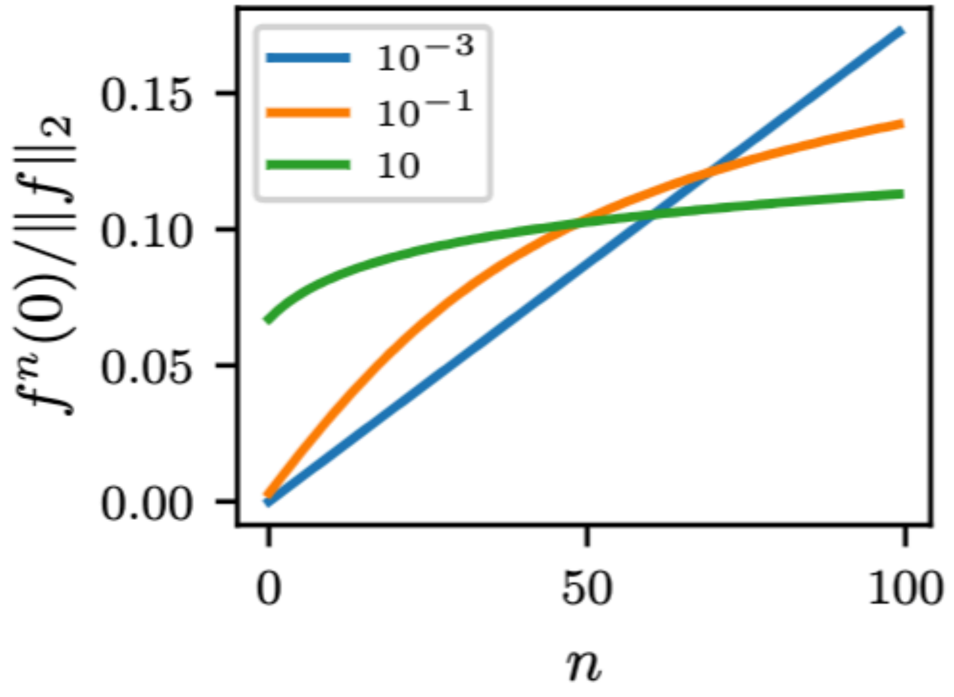
Training dynamic

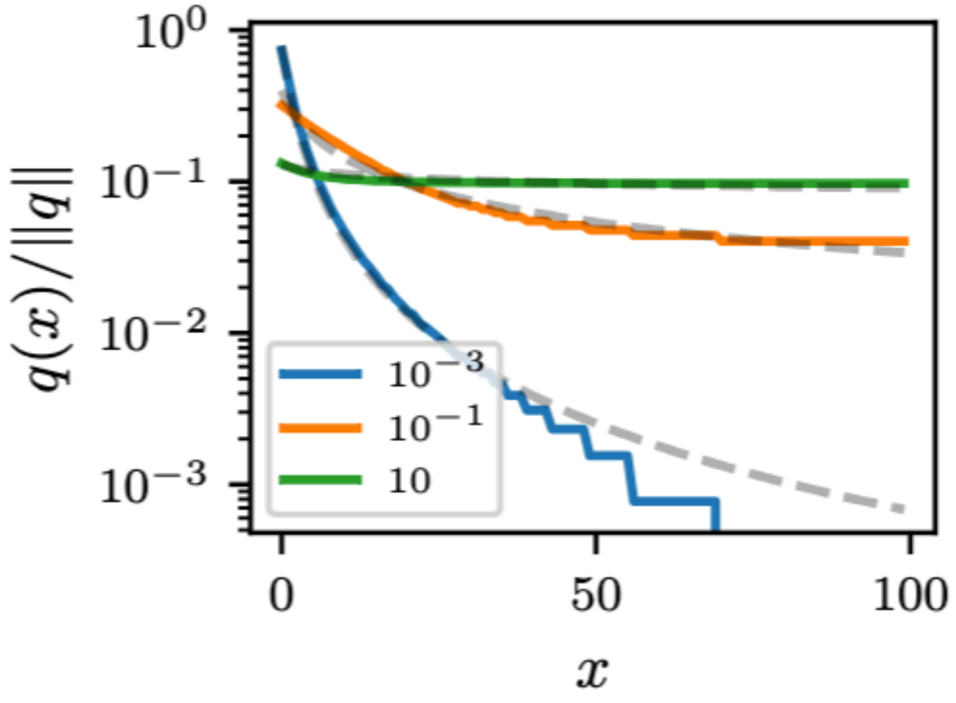
Model	Error scaling	Comment
$q(x) = p(x)$	$d^{-(\alpha-1)/2\alpha} + T^{-1+1/\alpha}$	Found with large batches in one step
$q(x) = \mathbf{1}_{x \leq d}$	$d^{-\alpha+1} + T^{-1+1/\alpha}$	Optimal scaling with random embeddings

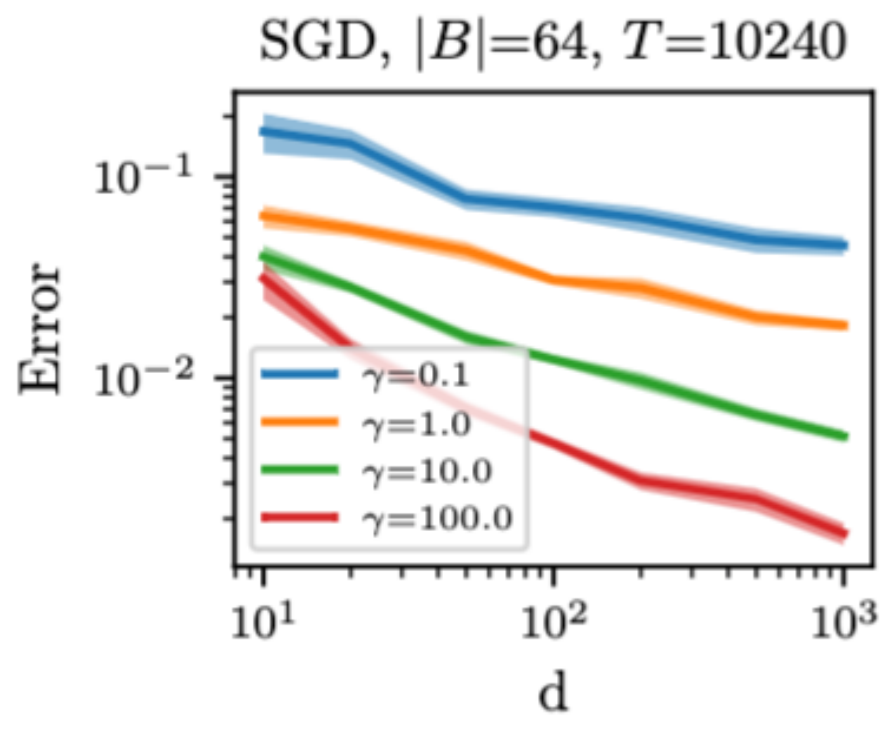
$$W_{t+1} = W_t - \gamma_t \sum_{x \in B_t} \nabla_W \ell(W; x, f_*(x)) \qquad W_t = \sum_x q_t(x) u_{f_*(x)} e_x^\top$$

$$q_T(x) = F(\{t \mid x_t = x\}) \quad \text{with} \quad F(n) = \underbrace{f \circ f \circ \cdots \circ f}_{\times n}(0) \quad \text{and} \quad f : x \mapsto x + \frac{\gamma}{1 + M^{-1}\exp(x)}$$

Approximation matches practice, it can be used to predict the $q_T(x)$ for different learning rates



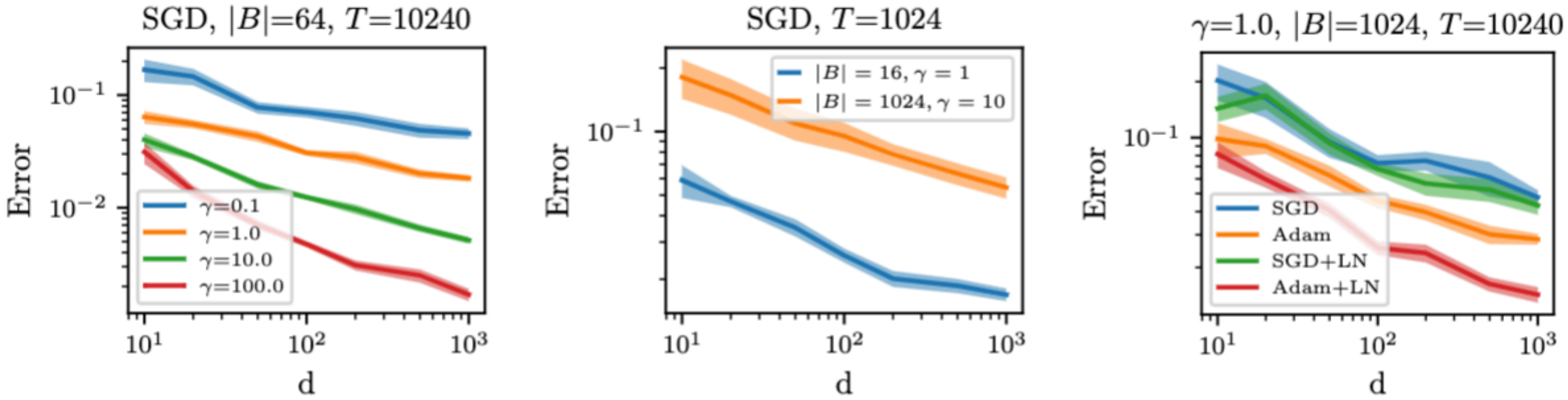




Model	Error scaling	Comment
$q(x) = p(x)$	$d^{-(\alpha-1)/2\alpha} + T^{-1+1/\alpha}$	Found with large batches in one step
$q(x) = \mathbf{1}_{x \leq d}$	$d^{-\alpha+1} + T^{-1+1/\alpha}$	Optimal scaling with random embeddings

Training dynamic

In this setting, to saturate $q_t(x)$ as fast as possible, we want small batch, large learning rates.



Find this paper on ArXiv,
"Scaling Laws for Associative Memories"

As well as its use to understand transformers,
"Birth of a Transformer: A Associative Memory Viewpoint"

Why did you used those tools?

The statistical part seems really ad-hoc, how could I generalized it?

Surrogate calibration inequality + self-consistency of logistic loss + L2-margin conditions

For the optimization part, why haven’t you used convex analysis?

We hope that our understanding in terms of memory could better scale to more complex model.

E.g., “is edge of stability related to memory overflow?”

So “memory machines” does not only apply to transformers?

While processing data, gradient descent provides signals for pattern matching

E.g., x is the image of a bike, y the label “bike” and the network factorizes as $f_\theta = f_{\theta_1} \circ f_{\theta_2}$

Imagine that $f_{\theta_2}(x) = x_2$ is the pattern of a wheel, and we are enforcing $f_{\theta_1}(x_2) \to y$

Signals are stored in the weights, with more frequent signals dominating

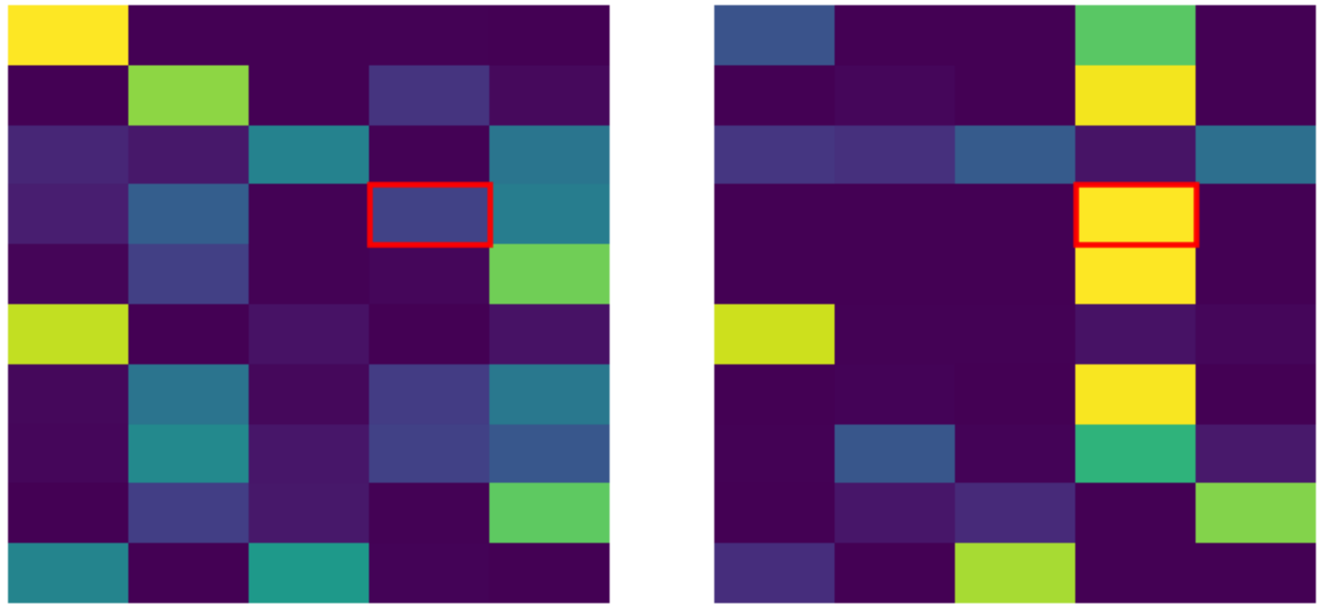
E.g., the matching “wheel” to “bike” is stored in θ_2, competing with other associations stored in θ_2

Associations seen often in the data will erase the other ones

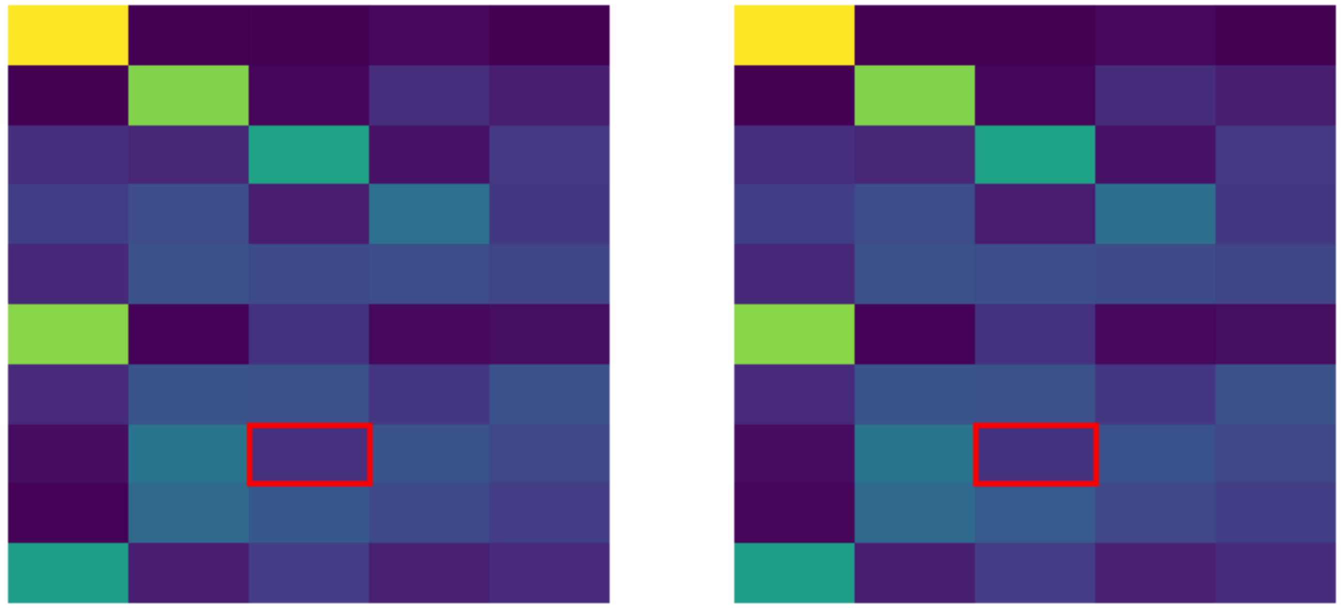
Mid-level signals that explain many high-level signals will be more frequent

If x_2 is the concatenation of “wheel” plus noise, the noise will be erased in θ_2 over time

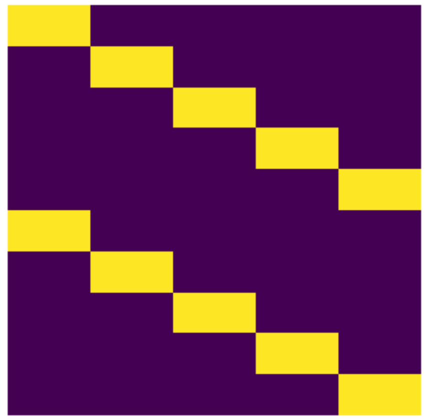
I like to mechanistically interpret a model by looking at the weights!?



Large learning rates risk erasing past memories



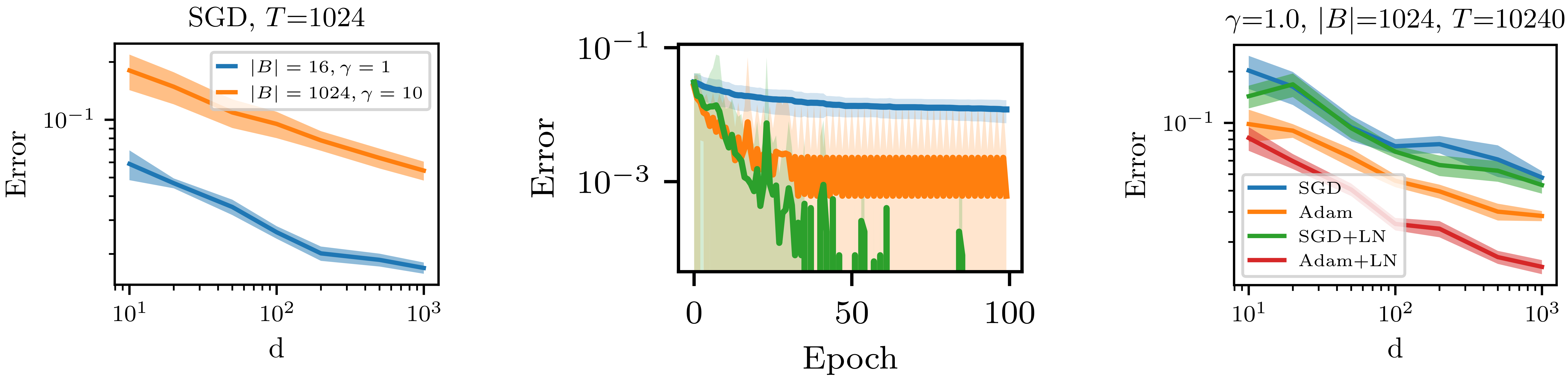
Small learning rates risk being too conservative



Target Association Matrix W
that solves for $y = x \bmod 5$

Can you remind us your training dynamics insights?

- Large learning rates is better - saturate memory faster, sporadic erasing is harmless in our model
- Small batch size is better - help saturate faster the memory to store unfrequent associations
- Adam works best - help rescale gradient updates to mimic large learning rates
- Layer norm works well - help for stability, plus add a clipping effect



It is kind of sad that a $d \times d$ matrix only store d vectors in $\mathbb{R}^d$!?

One can design a model with exponential storage capacity (with respect to the embedding space),

$$f_W(x) = \arg\max_{y \in [M]} g_W(x)_y \quad \text{with} \quad g_W(x_0)_y = u_y^\top \sum_x q(x) u_y \,\mathrm{ReLU}(e_x^\top e_0 - \eta) \quad \text{and} \quad q : \mathbb{N} \to \mathbb{R}.$$

Non linearity reduces the noise from competing associations, and improve memory capacity.

This is the basis of modern Hopfield network.

Unclear how to design exponential capacity with respect to the number of parameters?

It has probably been studied in the compression or the neural computation literature.