Learning in Transformers with Associative Memories

Alberto Bietti

Flatiron Institute, Simons Foundation

JSM, Boston, August 2026

Learning in Transformers with Associative Memories

Alberto Bietti

Flatiron Institute, Simons Foundation

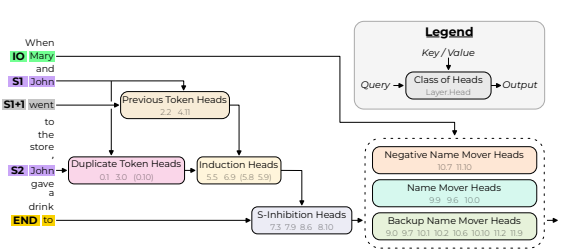
JSM, Boston, August 2026

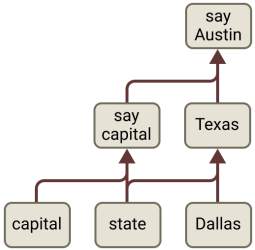
w/ M. Vural (UofT → Berkeley), D. Wu (NYU / Flatiron → Oxford), M. Soltanolkotabi (USC)
Also: V. Cabannes, E. Dohmatob, D. Bouchacourt, H. Jégou, L. Bottou, E. Nichani, J. Lee.

What are Transformer LLMs doing?

Reasoning over context

- A "biology" of circuits (Elhage et al., 2021; Wang et al., 2022; Lindsey et al., 2025)
- Many results on expressivity (e.g., circuits, formal languages, graph algorithms)
  - e.g., (Merrill et al., 2022; Liu et al., 2023; Sanford et al., 2023)





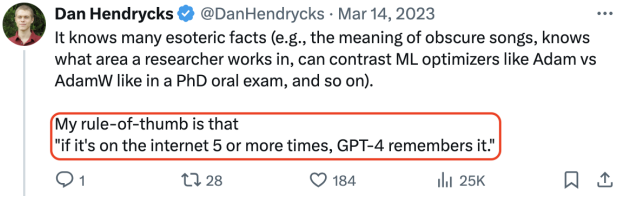
What are Transformer LLMs doing?

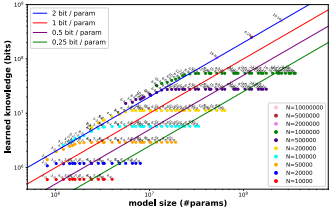
Reasoning over context

- A "biology" of circuits (Elhage et al., 2021; Wang et al., 2022; Lindsey et al., 2025)
- Many results on expressivity (e.g., circuits, formal languages, graph algorithms)
  - e.g., (Merrill et al., 2022; Liu et al., 2023; Sanford et al., 2023)

Knowledge storage

- Memorization, factual recall, parameter scaling
  - e.g., (Geva et al., 2020; Allen-Zhu and Li, 2024)
- Allows higher-level reasoning





What are Transformer LLMs doing?

Reasoning over context

- A "biology" of circuits (Elhage et al., 2021; Wang et al., 2022; Lindsey et al., 2025)
- Many results on expressivity (e.g., circuits, formal languages, graph algorithms)
  - e.g., (Merrill et al., 2022; Liu et al., 2023; Sanford et al., 2023)

Knowledge storage

- Memorization, factual recall, parameter scaling
  - e.g., (Geva et al., 2020; Allen-Zhu and Li, 2024)
- Allows higher-level reasoning

Goal: tractable model for both + training dynamics?

Transformer language models

Input: sequence of discrete tokens $(z_1, \ldots, z_T) \in [N]^T$

Transformer language models

Input: sequence of discrete tokens $(z_1, \ldots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Transformer language models

Input: sequence of discrete tokens $(z_1, \ldots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Residual streams (Elhage et al., 2021)

- embed each token $z_t \in [N]$ as $x_t := e_{z_t} + p_t$

Transformer language models

Input: sequence of discrete tokens $(z_1, \dots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Residual streams (Elhage et al., 2021)

- embed each token $z_t \in [N]$ as $x_t := e_{z_t} + p_t$
- (causal) self-attention $x_t := x_t + \mathrm{MHSA}(x_t, x_{1:t})$

$$\mathrm{MHSA}(x_t, x_{1:t}) = \sum_{h=1}^{H} \sum_{s=1}^{t} \beta_s^h W_O^{h\top} W_V^h x_s, \qquad \text{with } \beta_s^h = \frac{\exp({x_s}^\top W_K^{h\top} W_Q^h x_t)}{\sum_{s=1}^{t} \exp({x_s}^\top W_K^{h\top} W_Q^h x_t)}$$

where $W_K, W_Q, W_V, W_O \in \mathbb{R}^{d_h \times d}$ (key/query/value/output matrices)

Transformer language models

Input: sequence of discrete tokens $(z_1, \dots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Residual streams (Elhage et al., 2021)

- embed each token $z_t \in [N]$ as $x_t := e_{z_t} + p_t$
- (causal) self-attention $x_t := x_t + \mathrm{MHSA}(x_t, x_{1:t})$

$$\mathrm{MHSA}(x_t, x_{1:t}) = \sum_{s=1}^{t} \beta_s^h W_{OV} x_s, \qquad \text{with } \beta_s^h = \frac{\exp({x_s}^\top W_{KQ} x_t)}{\sum_{s=1}^{t} \exp({x_s}^\top W_{KQ} x_t)}$$

where $W_{KQ}, W_{OV} \in \mathbb{R}^{d \times d}$ (key-query and value-output matrices)

Transformer language models

Input: sequence of discrete tokens $(z_1, \dots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Residual streams (Elhage et al., 2021)

- embed each token $z_t \in [N]$ as $x_t := e_{z_t} + p_t$
- (causal) self-attention $x_t := x_t + \mathrm{MHSA}(x_t, x_{1:t})$
- feed-forward $x_t := x_t + \mathrm{MLP}(x_t)$

$$\mathrm{MLP}(x_t) = V^\top \sigma(U x_t)$$

where $U, V \in \mathbb{R}^{m \times d}$, often $m = 4d$

Transformer language models

Input: sequence of discrete tokens $(z_1, \ldots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Residual streams (Elhage et al., 2021)

- embed each token $z_t \in [N]$ as $x_t := e_{z_t} + p_t$
- (causal) self-attention $x_t := x_t + \mathrm{MHSA}(x_t, x_{1:t})$
- feed-forward $x_t := x_t + \mathrm{MLP}(x_t)$
- residual stream x_t is a sum of embeddings/"features"

Transformer language models

Input: sequence of discrete tokens $(z_1, \dots, z_T) \in [N]^T$

Embeddings

- input e_z, positional p_t, output u_y, in $\mathbb{R}^d$
- this talk: **fixed** to **random** init $\mathcal{N}(0, 1/d)$

Residual streams (Elhage et al., 2021)

- embed each token $z_t \in [N]$ as $x_t := e_{z_t} + p_t$
- (causal) self-attention $x_t := x_t + \mathrm{MHSA}(x_t, x_{1:t})$
- feed-forward $x_t := x_t + \mathrm{MLP}(x_t)$
- residual stream x_t is a sum of embeddings/"features"

Next-token prediction

- cross-entropy loss

$$\sum_{t<T} \ell(z_{t+1}; ({u_j}^\top x_t)_j)$$

Outline

Transformer weights as associative memories

- Consider sets of **nearly orthonormal embeddings** $\{e_z\}_{z\in\mathcal{Z}}$ and $\{u_y\}_{y\in\mathcal{Y}}$:

$$\|e_z\| \approx 1 \quad \text{and} \quad {e_z}^\top e_{z'} \approx 0$$
$$\|u_y\| \approx 1 \quad \text{and} \quad {u_y}^\top u_{y'} \approx 0$$

Transformer weights as associative memories

- Consider sets of **nearly orthonormal embeddings** $\{e_z\}_{z\in\mathcal{Z}}$ and $\{u_y\}_{y\in\mathcal{Y}}$:

$$\|e_z\| \approx 1 \quad \text{and} \quad {e_z}^\top e_{z'} \approx 0$$
$$\|u_y\| \approx 1 \quad \text{and} \quad {u_y}^\top u_{y'} \approx 0$$

- For **pairwise associations** $z \mapsto f_*(z)$ with $z \in \mathcal{M}$, define:

$$W = \sum_{z\in\mathcal{M}} u_{f_*(z)} {e_z}^\top$$

Transformer weights as associative memories

- Consider sets of **nearly orthonormal embeddings** $\{e_z\}_{z\in\mathcal{Z}}$ and $\{u_y\}_{y\in\mathcal{Y}}$:

$$\|e_z\| \approx 1 \quad \text{and} \quad {e_z}^\top e_{z'} \approx 0$$
$$\|u_y\| \approx 1 \quad \text{and} \quad {u_y}^\top u_{y'} \approx 0$$

- For **pairwise associations** $z \mapsto f_*(z)$ with $z \in \mathcal{M}$, define:

$$W = \sum_{z\in\mathcal{M}} u_{f_*(z)} {e_z}^\top \quad \Longrightarrow \quad {u_y}^\top W e_z \approx \mathbb{1}\{y = f_*(z)\}$$

Transformer weights as associative memories

- Consider sets of **nearly orthonormal embeddings** $\{e_z\}_{z\in\mathcal{Z}}$ and $\{u_y\}_{y\in\mathcal{Y}}$:

$$\|e_z\| \approx 1 \quad \text{and} \quad e_z{}^\top e_{z'} \approx 0$$
$$\|u_y\| \approx 1 \quad \text{and} \quad u_y{}^\top u_{y'} \approx 0$$

- For **pairwise associations** $z \mapsto f_*(z)$ with $z \in \mathcal{M}$, define:

$$W = \sum_{z\in\mathcal{M}} u_{f_*(z)} e_z{}^\top \quad \Longrightarrow \quad u_y{}^\top W e_z \approx \mathbb{1}\{y = f_*(z)\}$$

Examples in Transformers

- e_z, u_y are input/output/positional embeddings, or intermediate representations
- Logits in attention heads: $x_k^\top W_{KQ} x_q$
- Logits in next-token prediction: $u_y^\top U\sigma(Vx_t)$ or $u_y^\top W_{OV} x_k$

Transformer weights as associative memories

- Consider sets of **nearly orthonormal embeddings** $\{e_z\}_{z\in\mathcal{Z}}$ and $\{u_y\}_{y\in\mathcal{Y}}$:

$$\|e_z\| \approx 1 \quad \text{and} \quad e_z{}^\top e_{z'} \approx 0$$
$$\|u_y\| \approx 1 \quad \text{and} \quad u_y{}^\top u_{y'} \approx 0$$

- For **pairwise associations** $z \mapsto f_*(z)$ with $z \in \mathcal{M}$, define:

$$W = \sum_{z\in\mathcal{M}} u_{f_*(z)} e_z{}^\top \quad \Longrightarrow \quad u_y{}^\top W e_z \approx \mathbb{1}\{y = f_*(z)\}$$

Examples in Transformers

- e_z, u_y are input/output/positional embeddings, or intermediate representations
- Logits in attention heads: $x_k^\top W_{KQ} x_q$
- Logits in next-token prediction: $u_y^\top U\sigma(Vx_t)$ or $u_y^\top W_{OV} x_k$
- Related to Hopfield (1982); Kohonen (1972); Willshaw et al. (1969); Iscen et al. (2017)

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = u_k{}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings.*

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = {u_k}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings. Then,*

$$\nabla L(W) = \sum_{k=1}^{K} \mathbb{E}_z[(\hat{p}_W(y = k|z) - p(y = k|z)) u_k {e_z}^\top]$$

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = {u_k}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings. Then,*

$$\nabla L(W) = \sum_{k=1}^{K} \mathbb{E}_z[(\hat{p}_W(y = k|z) - p(y = k|z)) u_k {e_z}^\top]$$

- **Example**: $z \sim \mathrm{Unif}([N])$, $y = f_*(z)$

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = {u_k}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings. Then,*

$$\nabla L(W) = \sum_{k=1}^{K} \mathbb{E}_z[(\hat{p}_W(y = k|z) - p(y = k|z)) u_k {e_z}^\top]$$

- **Example**: $z \sim \text{Unif}([N])$, $y = f_*(z)$
  - After **one gradient step** on the population loss, assuming near-orthonormal embeddings

$$W_1 = \frac{\eta}{N} \sum_{z,k} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right) u_k {e_z}^\top \quad \Longrightarrow \quad {u_k}^\top W_1 e_z \approx \frac{\eta}{N} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right)$$

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = u_k{}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings. Then,*

$$\nabla L(W) = \sum_{k=1}^{K} \mathbb{E}_z[(\hat{p}_W(y = k|z) - p(y = k|z)) u_k e_z{}^\top]$$

- **Example**: $z \sim \mathrm{Unif}([N])$, $y = f_*(z)$
  - After **one gradient step** on the population loss, assuming near-orthonormal embeddings

$$W_1 = \frac{\eta}{N} \sum_{z,k} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right) u_k e_z{}^\top \quad \Longrightarrow \quad u_k{}^\top W_1 e_z \approx \frac{\eta}{N} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right)$$

  - **Corollary**: $\hat{f}(z) = \arg\max_k u_k{}^\top W_1 e_z$ has near-perfect accuracy

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = u_k{}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings. Then,*

$$\nabla L(W) = \sum_{k=1}^{K} \mathbb{E}_z[(\hat{p}_W(y = k|z) - p(y = k|z)) u_k e_z{}^\top]$$

- **Example**: $z \sim \mathrm{Unif}([N])$, $y = f_*(z)$
  - After **one gradient step** on the population loss, assuming near-orthonormal embeddings

$$W_1 = \frac{\eta}{N} \sum_{z,k} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right) u_k e_z{}^\top \quad \Longrightarrow \quad u_k{}^\top W_1 e_z \approx \frac{\eta}{N} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right)$$

  - **Corollary**: $\hat{f}(z) = \arg\max_k u_k{}^\top W_1 e_z$ has near-perfect accuracy
- More generally, replace u_k by "backward" vector

Gradient associative memories

Lemma (Gradients as memories, B. et al., 2023)

Let p be a data distribution over $(z, y) \in [N]^2$, and consider the loss

$$L(W) = \mathbb{E}_{(z,y)\sim p}[\ell(y, F_W(z))], \quad F_W(z)_k = u_k{}^\top W e_z,$$

*with ℓ the **cross-entropy loss** and e_z, u_k input/output embeddings. Then,*

$$\nabla L(W) = \sum_{k=1}^{K} \mathbb{E}_z[(\hat{p}_W(y = k|z) - p(y = k|z)) u_k e_z{}^\top]$$

- **Example**: $z \sim \mathrm{Unif}([N])$, $y = f_*(z)$
  - After **one gradient step** on the population loss, assuming near-orthonormal embeddings

$$W_1 = \frac{\eta}{N} \sum_{z,k} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right) u_k e_z{}^\top \quad \Longrightarrow \quad u_k{}^\top W_1 e_z \approx \frac{\eta}{N} \left(\mathbb{1}\{f_*(z) = k\} - \frac{1}{N} \right)$$

  - **Corollary**: $\hat{f}(z) = \arg\max_k u_k{}^\top W_1 e_z$ has near-perfect accuracy
- More generally, replace u_k by "backward" vector

Note: related to (Ba et al., 2022; Damian et al., 2022; Dandi et al., 2023; Yang and Hu, 2021)

Capacity: Intuition

- Random embeddings $e_z, u_y \sim \mathcal{N}(0, \frac{1}{d} I)$

Capacity: Intuition

- Random embeddings $e_z, u_y \sim \mathcal{N}(0, \frac{1}{d} I)$
- For some $f^* : [N] \to [M]$, consider "one gradient step" solution

$$W = \sum_{z=1}^{N} u_{f^*(z)} e_z^\top \in \mathbb{R}^{d \times d}$$

Capacity: Intuition

- Random embeddings $e_z, u_y \sim \mathcal{N}(0, \frac{1}{d} I)$
- For some $f^* : [N] \to [M]$, consider "one gradient step" solution

$$W = \sum_{z=1}^{N} u_{f^*(z)} e_z^\top \in \mathbb{R}^{d \times d}$$

- When can we recover $\arg\max_y \gamma_{z,y} = f^*(z)$ for all z?

$$\gamma_{z,y} := u_y^\top W e_z = \sum_{z'} u_y^\top u_{f^*(z')} e_z^\top e_{z'}$$

Capacity: Intuition

- Random embeddings $e_z, u_y \sim \mathcal{N}(0, \frac{1}{d} I)$
- For some $f^* : [N] \to [M]$, consider "one gradient step" solution

$$W = \sum_{z=1}^{N} u_{f^*(z)} e_z^\top \in \mathbb{R}^{d \times d}$$

- When can we recover $\arg\max_y \gamma_{z,y} = f^*(z)$ for all z?

$$\gamma_{z,y} := u_y^\top W e_z = \sum_{z'} u_y^\top u_{f^*(z')} e_z^\top e_{z'}$$

$$\mathbb{E}[\gamma_{z,y}] = \begin{cases} 1, & \text{if } y = f^*(z) \\ 0, & \text{otherwise.} \end{cases} \qquad \mathrm{Var}[\gamma_{z,y}] \lesssim \frac{|\{f^*(z') = y\}|}{d} + \frac{|\{f^*(z') \neq y\}|}{d^2} \overset{?}{\lesssim} 1$$

Capacity: Intuition

- Random embeddings $e_z, u_y \sim \mathcal{N}(0, \frac{1}{d} I)$
- For some $f^* : [N] \to [M]$, consider "one gradient step" solution

$$W = \sum_{z=1}^{N} u_{f^*(z)} e_z^\top \in \mathbb{R}^{d \times d}$$

- When can we recover $\arg\max_y \gamma_{z,y} = f^*(z)$ for all z?

$$\gamma_{z,y} := u_y^\top W e_z = \sum_{z'} u_y^\top u_{f^*(z')} e_z^\top e_{z'}$$

$$\mathbb{E}[\gamma_{z,y}] = \begin{cases} 1, & \text{if } y = f^*(z) \\ 0, & \text{otherwise.} \end{cases} \qquad \mathrm{Var}[\gamma_{z,y}] \lesssim \frac{|\{f^*(z') = y\}|}{d} + \frac{|\{f^*(z') \neq y\}|}{d^2} \overset{?}{\lesssim} 1$$

- **Examples**: (Cabannes, Dohmatob, and B., 2024a; Nichani, Lee, and B., 2025)
  - f^* one-to-one: can store up to $N \approx d^2$ associations (much better than one hot!)

Capacity: Intuition

- Random embeddings $e_z, u_y \sim \mathcal{N}(0, \frac{1}{d} I)$
- For some $f^* : [N] \to [M]$, consider "one gradient step" solution

$$W = \sum_{z=1}^{N} u_{f^*(z)} e_z^\top \in \mathbb{R}^{d \times d}$$

- When can we recover $\arg\max_y \gamma_{z,y} = f^*(z)$ for all z?

$$\gamma_{z,y} := u_y^\top W e_z = \sum_{z'} u_y^\top u_{f^*(z')} e_z^\top e_{z'}$$

$$\mathbb{E}[\gamma_{z,y}] = \begin{cases} 1, & \text{if } y = f^*(z) \\ 0, & \text{otherwise.} \end{cases} \qquad \mathrm{Var}[\gamma_{z,y}] \lesssim \frac{|\{f^*(z') = y\}|}{d} + \frac{|\{f^*(z') \neq y\}|}{d^2} \overset{?}{\lesssim} 1$$

- **Examples**: (Cabannes, Dohmatob, and B., 2024a; Nichani, Lee, and B., 2025)
  - f^* one-to-one: can store up to $N \approx d^2$ associations (much better than one hot!)
  - $f^*(z) \in \{0, 1\}$: can store up to $N \approx d$ associations

Capacity $\approx$ number of parameters

Low-rank

- $W = W_1^\top W_2$, with $W_1, W_2 \in \mathbb{R}^{m \times d}$ (e.g., key-query or output-value matrices)
- can store $N \approx md$ associations when $m \leq d$
- construction: random W_1, one step on W_2

(Nichani, Lee, and B., 2025), related to Krotov and Hopfield (2016); Demircigil et al. (2017)

Capacity $\approx$ number of parameters

Low-rank

- $W = W_1^\top W_2$, with $W_1, W_2 \in \mathbb{R}^{m \times d}$ (e.g., key-query or output-value matrices)
- can store $N \approx md$ associations when $m \leq d$
- construction: random W_1, one step on W_2

Non-linear MLP

- $\hat{f}(z) = \arg\max_y u_y{}^\top W_1 \sigma(W_2^\top e_z)$, $W_1, W_2 \in \mathbb{R}^{d \times m}$
- can store $N \approx md$ associations for any width m
- construction: using Hermite polynomials of degree $\approx \log N / \log d$ in kernel regime

(Nichani, Lee, and B., 2025), related to Krotov and Hopfield (2016); Demircigil et al. (2017)

Capacity $\approx$ number of parameters

Low-rank

- $W = W_1^\top W_2$, with $W_1, W_2 \in \mathbb{R}^{m \times d}$ (e.g., key-query or output-value matrices)
- can store $N \approx md$ associations when $m \leq d$
- construction: random W_1, one step on W_2

Non-linear MLP

- $\hat{f}(z) = \arg\max_y u_y{}^\top W_1 \sigma(W_2^\top e_z)$, $W_1, W_2 \in \mathbb{R}^{d \times m}$
- can store $N \approx md$ associations for any width m
- construction: using Hermite polynomials of degree $\approx \log N / \log d$ in kernel regime

Multi-input

- $\hat{f}(z_1, z_2) = \arg\max_y u_y{}^\top W_1 \sigma(W_2^\top (e_{z_1} + \tilde{e}_{z_2}))$
- also $N \approx md$ capacity

(Nichani, Lee, and B., 2025), related to Krotov and Hopfield (2016); Demircigil et al. (2017)

Capacity $\approx$ number of parameters

Low-rank

- $W = W_1^\top W_2$, with $W_1, W_2 \in \mathbb{R}^{m \times d}$ (e.g., key-query or output-value matrices)
- can store $N \approx md$ associations when $m \leq d$
- construction: random W_1, one step on W_2

Non-linear MLP

- $\hat{f}(z) = \arg\max_y u_y{}^\top W_1 \sigma(W_2^\top e_z)$, $W_1, W_2 \in \mathbb{R}^{d \times m}$
- can store $N \approx md$ associations for any width m
- construction: using Hermite polynomials of degree $\approx \log N / \log d$ in kernel regime

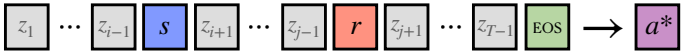
Multi-input

- $\hat{f}(z_1, z_2) = \arg\max_y u_y{}^\top W_1 \sigma(W_2^\top (e_{z_1} + \tilde{e}_{z_2}))$
- also $N \approx md$ capacity

Note: matches information-theoretic lower bounds

(Nichani, Lee, and B., 2025), related to Krotov and Hopfield (2016); Demircigil et al. (2017)

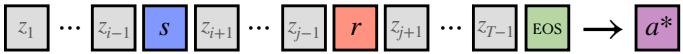
Application to Transformers I: factual recall (Nichani, Lee, and B., 2025)



The capital of France is Paris

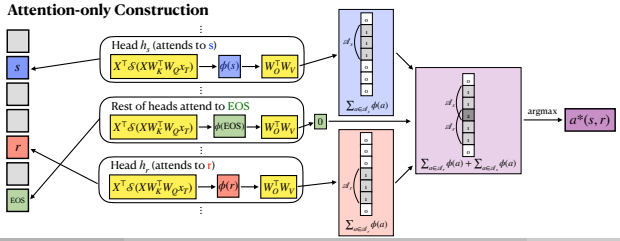
- $s \in \mathcal{S}, r \in \mathcal{R}$: **subject** and **relation** tokens
- $a^*(s, r) \in \mathcal{A}_r$: **attribute**/fact to be stored
- $z_i \in \mathcal{N}$: **noise** tokens

Application to Transformers I: factual recall (Nichani, Lee, and B., 2025)

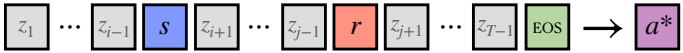


The capital of France is Paris

- $s \in \mathcal{S}, r \in \mathcal{R}$: **subject** and **relation** tokens
- $a^*(s, r) \in \mathcal{A}_r$: **attribute**/fact to be stored
- $z_i \in \mathcal{N}$: **noise** tokens
- Nichani, Lee, and B. (2025): Transformers with associative memory parameters can achieve perfect factual recall with $\tilde{O}(SR)$ parameters.

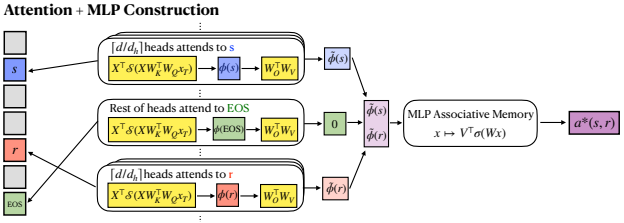


Application to Transformers I: factual recall (Nichani, Lee, and B., 2025)

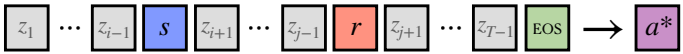


The capital of France is Paris

- $s \in \mathcal{S}, r \in \mathcal{R}$: **subject** and **relation** tokens
- $a^*(s, r) \in \mathcal{A}_r$: **attribute**/fact to be stored
- $z_i \in \mathcal{N}$: **noise** tokens
- Nichani, Lee, and B. (2025): Transformers with associative memory parameters can achieve perfect factual recall with $\tilde{O}(SR)$ parameters.

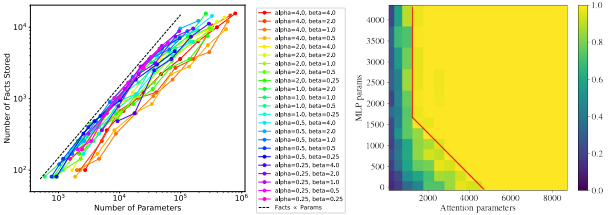


Application to Transformers I: factual recall (Nichani, Lee, and B., 2025)

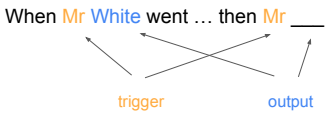


The capital of France is Paris

- $s \in \mathcal{S}, r \in \mathcal{R}$: **subject** and **relation** tokens
- $a^*(s, r) \in \mathcal{A}_r$: **attribute**/fact to be stored
- $z_i \in \mathcal{N}$: **noise** tokens
- Nichani, Lee, and B. (2025): Transformers with associative memory parameters can achieve perfect factual recall with $\tilde{O}(SR)$ parameters.

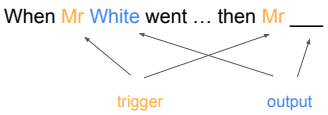


Application to Transformers II: in-context recall (B. et al., 2023)



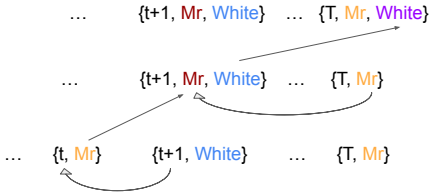
When Mr White went to the mall, it started raining, then Mr White witnessed an odd occurrence. While walking around the mall with his family, Mr White heard the sound of a helicopter landing in the parking lot. Curious, he made his way over to see what was going on.

Application to Transformers II: in-context recall (B. et al., 2023)



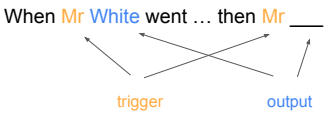
When Mr White went to the mall, it started raining, then Mr White witnessed an odd occurrence. While walking around the mall with his family, Mr White heard the sound of a helicopter landing in the parking lot. Curious, he made his way over to see what was going on.

Induction heads (Elhage et al., 2021; Olsson et al., 2022)



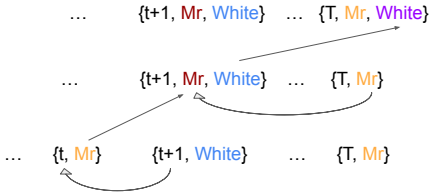
- 1st layer: **previous-token head**
  - attends to previous token and copies it to residual stream

Application to Transformers II: in-context recall (B. et al., 2023)



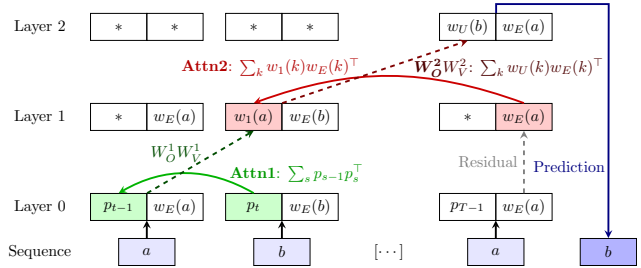
When Mr White went to the mall, it started raining, then Mr White witnessed an odd occurrence. While walking around the mall with his family, Mr White heard the sound of a helicopter landing in the parking lot. Curious, he made his way over to see what was going on.

Induction heads (Elhage et al., 2021; Olsson et al., 2022)



- 1st layer: **previous-token head**
  - attends to previous token and copies it to residual stream
- 2nd layer: **induction head**
  - attends to output of previous token head, copies attended token

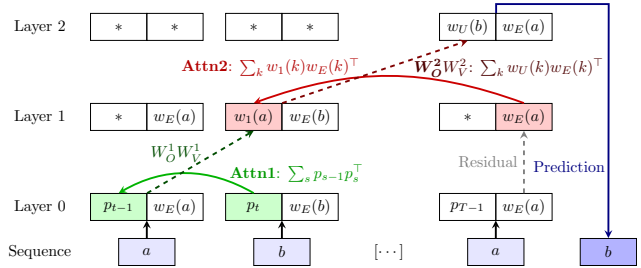
Induction head with associative memory weights



$$W_{KQ}^1 = \beta \sum_{t=2}^{T} p_t p_{t-1}^\top, \quad W_{KQ}^2 = \beta \sum_{k \in Q} e_k \tilde{e}_k^\top, \quad W_{OV}^2 = \sum_{k=1}^{N} u_k e_k^\top,$$

- Random embeddings e_k, u_k, random matrix W_{OV}^1 (frozen at init)
- **Remapped** previous tokens: $\tilde{e}_k := W_{OV}^1 e_k$
- $\beta \to \infty$ for one-hot attention

Induction head with associative memory weights



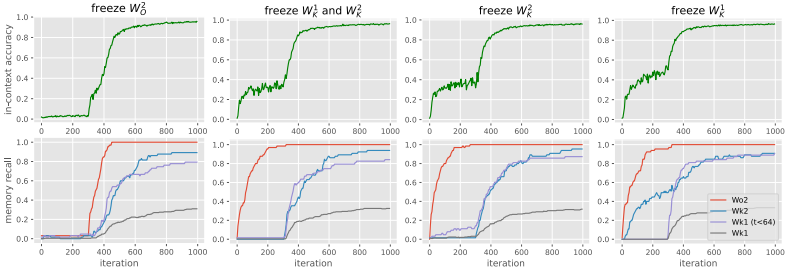
$$W_{KQ}^1 = \beta \sum_{t=2}^{T} p_t p_{t-1}^\top, \quad W_{KQ}^2 = \beta \sum_{k \in Q} e_k \tilde{e}_k^\top, \quad W_{OV}^2 = \sum_{k=1}^{N} u_k e_k^\top,$$

- Random embeddings e_k, u_k, random matrix W_{OV}^1 (frozen at init)
- **Remapped** previous tokens: $\tilde{e}_k := W_{OV}^1 e_k$
- $\beta \to \infty$ for one-hot attention

Q: Does this match practice?

Empirically probing the dynamics

Train only W^1_{KQ}, W^2_{KQ}, W^2_{OV}, loss on predictable tokens after trigger

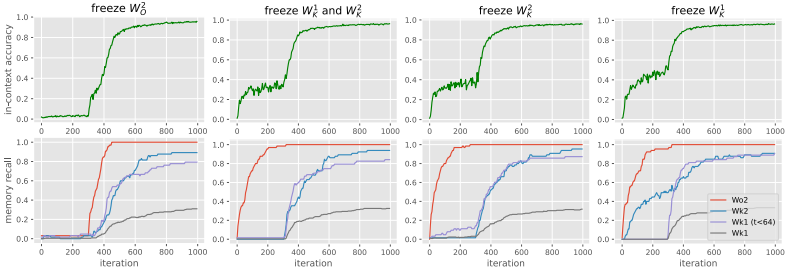


- "Memory recall **probes**": for target memory $W_* = \sum_{i=1}^{M} u_i e_i^\top$, compute

$$R(\hat{W}, W_*) = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}\{i = \arg\max_j u_j^\top \hat{W} e_i\}$$

Empirically probing the dynamics

Train only W^1_{KQ}, W^2_{KQ}, W^2_{OV}, loss on predictable tokens after trigger



- "Memory recall **probes**": for target memory $W_* = \sum_{i=1}^M u_i e_i^\top$, compute

$$R(\hat{W}, W_*) = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}\{i = \arg\max_j u_j^\top \hat{W} e_i\}$$

- Natural learning "**order**": W^2_{OV} first, W^2_{KQ} next, W^1_{KQ} last
  - B. et al. (2023): Three successive gradient steps can learn the three associative memories.

Outline

Setup: in-context retrieval + factual recall

"The capital of $\underset{t^\star}{\text{Italy}}$ $\underset{\text{EOS}}{\text{is}}$ " $\longrightarrow$ $\underset{f^\star(z_{t^\star})}{\text{Rome}}$

"What is the opposite of $\underset{t^\star}{\text{big}}$ in English $\underset{\text{EOS}}{?}$ " $\longrightarrow$ $\underset{f^\star(z_{t^\star})}{\text{small}}$

Setup: in-context retrieval + factual recall

"The capital of $\underset{t^\star}{\text{Italy}}$ $\underset{\text{EOS}}{\text{is}}$ " $\longrightarrow$ $\underset{f^\star(z_{t^\star})}{\text{Rome}}$ "What is the opposite of $\underset{t^\star}{\text{big}}$ in English $\underset{\text{EOS}}{?}$ " $\longrightarrow$ $\underset{f^\star(z_{t^\star})}{\text{small}}$

Data:

- $z_1, \ldots, z_T \sim \text{Unif}([N])$, $t^* \sim \text{Unif}([T])$
- $y = f^*(z_{t^*})$, with $f^* : [N] \to [N]$ one-to-one

Setup: in-context retrieval + factual recall

"The capital of $\underbrace{\text{Italy}}_{t^\star}$ $\underbrace{\text{is}}_{\text{EOS}}$ " $\longrightarrow$ $\underbrace{\text{Rome}}_{f^\star(z_{t^\star})}$ "What is the opposite of $\underbrace{\text{big}}_{t^\star}$ in English $\underbrace{?}_{\text{EOS}}$ " $\longrightarrow$ $\underbrace{\text{small}}_{f^\star(z_{t^\star})}$

Data:

- $z_1, \dots, z_T \sim \text{Unif}([N])$, $t^* \sim \text{Unif}([T])$
- $y = f^*(z_{t^*})$, with $f^* : [N] \to [N]$ one-to-one

One-layer Transformer model: with random embeddings + a "trigger" feature

- $F_k(X; W_{OV}, W_{KQ}) = u_k W_{OV} X \mathcal{S}(X^\top W_{KQ} e_{EOS})$, with
  - $\mathcal{S}$ the attention softmax
  - $X := [x_1, \dots, x_T] \in \mathbb{R}^{d \times T}$
  - $x_t := e_{z_t} + \mathbb{1}\{t = t^*\} e_\star$, where $e_\star$ is a "trigger" feature marking z_{t^*}

Setup: in-context retrieval + factual recall

"The capital of $\underbrace{\text{Italy}}_{t^\star}$ $\underbrace{\text{is}}_{\text{EOS}}$ " $\longrightarrow$ $\underbrace{\text{Rome}}_{f^\star(z_{t^\star})}$ "What is the opposite of $\underbrace{\text{big}}_{t^\star}$ in English $\underbrace{?}_{\text{EOS}}$ " $\longrightarrow$ $\underbrace{\text{small}}_{f^\star(z_{t^\star})}$

Data:

- $z_1, \dots, z_T \sim \text{Unif}([N])$, $t^* \sim \text{Unif}([T])$
- $y = f^*(z_{t^*})$, with $f^* : [N] \to [N]$ one-to-one

One-layer Transformer model: with random embeddings + a "trigger" feature

- $F_k(X; W_{OV}, W_{KQ}) = u_k W_{OV} X \mathcal{S}(X^\top W_{KQ} e_{EOS})$, with
  - $\mathcal{S}$ the attention softmax
  - $X := [x_1, \dots, x_T] \in \mathbb{R}^{d \times T}$
  - $x_t := e_{z_t} + \mathbb{1}\{t = t^*\} e_\star$, where $e_\star$ is a "trigger" feature marking z_{t^*}
- **Goal**: perfect accuracy, i.e. $\arg\max_k F_k(X) = f^*(z_{t^*})$ w.h.p. on test data

Setup: in-context retrieval + factual recall

"The capital of $\underset{t^\star}{\text{Italy}}$ $\underset{\text{EOS}}{\text{is}}$ " $\longrightarrow$ $\underset{f^\star(z_{t^\star})}{\text{Rome}}$ "What is the opposite of $\underset{t^\star}{\text{big}}$ in English $\underset{\text{EOS}}{?}$ " $\longrightarrow$ $\underset{f^\star(z_{t^\star})}{\text{small}}$

Data:

- $z_1, \dots, z_T \sim \text{Unif}([N])$, $t^* \sim \text{Unif}([T])$
- $y = f^*(z_{t^*})$, with $f^* : [N] \to [N]$ one-to-one

One-layer Transformer model: with random embeddings + a "trigger" feature

- $F_k(X; W_{OV}, W_{KQ}) = u_k W_{OV} X \mathcal{S}(X^\top W_{KQ} e_{EOS})$, with
  - $\mathcal{S}$ the attention softmax
  - $X := [x_1, \dots, x_T] \in \mathbb{R}^{d \times T}$
  - $x_t := e_{z_t} + \mathbb{1}\{t = t^*\} e_\star$, where $e_\star$ is a "trigger" feature marking z_{t^*}
- **Goal**: perfect accuracy, i.e. $\arg\max_k F_k(X) = f^*(z_{t^*})$ w.h.p. on test data
- Associative memory solution: $(\beta \to \infty)$

$$W_{OV} = \sum_{i=1}^{N} u_{f^*(i)} e_i^\top, \qquad W_{KQ} = \beta e_\star e_{EOS}^\top$$

Training algorithm: three gradient steps

Training loss:

- Sample n training sequences (X_i, y_i)
- Empirical loss:

$$\hat{L}_n(W_{OV}, W_{KQ}) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, F(X_i; W_{OV}, W_{KQ})),$$

- ℓ: cross-entropy loss

Training algorithm: three gradient steps

Training loss:

- Sample n training sequences (X_i, y_i)
- Empirical loss:

$$\hat{L}_n(W_{OV}, W_{KQ}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, F(X_i; W_{OV}, W_{KQ})),$$

- ℓ: cross-entropy loss

Training algorithm:

- **Initialize** $W_{OV}^{(0)} = W_{KQ}^{(0)} = 0$
- Three steps:
  - **Step 1** (learn OV): $W_{OV} := W_{OV} - \eta \nabla_{W_{OV}} \hat{L}_n(W_{OV}, W_{KQ})$
  - **Step 2** (learn KQ): $W_{KQ} := W_{KQ} - \eta \nabla_{W_{KQ}} \hat{L}_n(W_{OV}, W_{KQ})$
  - **Step 3** (refine OV): $W_{OV} := W_{OV} - \eta \nabla_{W_{OV}} \hat{L}_n(W_{OV}, W_{KQ})$

Training algorithm: three gradient steps

Training loss:

- Sample n training sequences (X_i, y_i)
- Empirical loss:

$$\hat{L}_n(W_{OV}, W_{KQ}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, F(X_i; W_{OV}, W_{KQ})),$$

- ℓ: cross-entropy loss

Training algorithm:

- **Initialize** $W_{OV}^{(0)} = W_{KQ}^{(0)} = 0$
- Three steps:
  - **Step 1** (learn OV): $W_{OV} := W_{OV} - \eta \nabla_{W_{OV}} \hat{L}_n(W_{OV}, W_{KQ})$
  - **Step 2** (learn KQ): $W_{KQ} := W_{KQ} - \eta \nabla_{W_{KQ}} \hat{L}_n(W_{OV}, W_{KQ})$
  - **Step 3** (refine OV): $W_{OV} := W_{OV} - \eta \nabla_{W_{OV}} \hat{L}_n(W_{OV}, W_{KQ})$

Q: How do N, T, d, n need to scale for perfect accuracy?

Warmup: infinite data

- **First gradient** finds relevant associative memory under uniform attention:

$$W_{OV}^{(1)} = -\eta \nabla_{W_{OV}} L(0,0) = \eta \, \mathbb{E}[(u_{f^*(z_{t^*})} - \bar{u})(\frac{1}{T}\sum_{t=1}^{T} e_{z_t})^\top]$$

$$\approx \frac{\eta}{NT} \sum_{i=1}^{N} u_{f^*(i)} e_i^\top$$

Warmup: infinite data

- **First gradient** finds relevant associative memory under uniform attention:

$$W_{OV}^{(1)} = -\eta \nabla_{W_{OV}} L(0,0) = \eta \, \mathbb{E}[(u_{f^*(z_{t^*})} - \bar{u})(\frac{1}{T}\sum_{t=1}^{T} e_{z_t})^\top]$$

$$\approx \frac{\eta}{NT} \sum_{i=1}^{N} u_{f^*(i)} e_i^\top$$

- We then have **signal on correct associations** (when $d^2 \gtrsim N$): $u_k^\top W_{OV}^{(1)} e_i \approx \delta_{k,f^*(i)} \frac{\eta}{NT}$

Warmup: infinite data

- **First gradient** finds relevant associative memory under uniform attention:

$$W_{OV}^{(1)} = -\eta \nabla_{W_{OV}} L(0,0) = \eta\, \mathbb{E}[(u_{f^*(z_{t^*})} - \bar{u})(\frac{1}{T}\sum_{t=1}^{T} e_{z_t})^\top]$$

$$\approx \frac{\eta}{NT} \sum_{i=1}^{N} u_{f^*(i)} e_i^\top$$

- We then have **signal on correct associations** (when $d^2 \gtrsim N$): $u_k^\top W_{OV}^{(1)} e_i \approx \delta_{k,f^*(i)} \frac{\eta}{NT}$
- **Second step** identifies trigger:

$$-\nabla_{W_{KQ}} L(W_{OV}^{(1)}, 0) \approx \mathbb{E}[(u_{f^*(z_{t^*})} - \bar{u})^\top (\frac{1}{T}\sum_{t=1}^{T} W_{OV}^{(1)} x_t \cdot x_t e_{EOS}^\top)]$$

$$\approx \frac{1}{T}\, \mathbb{E}[u_{f^*(z_{t^*})}^\top W_{OV}^{(1)} e_{z_{t^*}} \cdot (e_{z^{t^*}} + e_\star) e_{EOS}^\top)]$$

$$\approx \frac{\eta}{NT^2} e_\star e_{EOS}^\top + \frac{\eta}{N^2 T^2} \sum_{i=1}^{N} e_i e_{EOS}^\top,$$

Warmup: infinite data

- **First gradient** finds relevant associative memory under uniform attention:

$$W_{OV}^{(1)} = -\eta \nabla_{W_{OV}} L(0,0) = \eta\, \mathbb{E}[(u_{f^*(z_{t^*})} - \bar{u})(\frac{1}{T}\sum_{t=1}^{T} e_{z_t})^\top]$$

$$\approx \frac{\eta}{NT} \sum_{i=1}^{N} u_{f^*(i)} e_i^\top$$

- We then have **signal on correct associations** (when $d^2 \gtrsim N$): $u_k^\top W_{OV}^{(1)} e_i \approx \delta_{k,f^*(i)} \frac{\eta}{NT}$
- **Second step** identifies trigger:

$$-\nabla_{W_{KQ}} L(W_{OV}^{(1)}, 0) \approx \mathbb{E}[(u_{f^*(z_{t^*})} - \bar{u})^\top (\frac{1}{T}\sum_{t=1}^{T} W_{OV}^{(1)} x_t \cdot x_t e_{EOS}^\top)]$$

$$\approx \frac{1}{T}\, \mathbb{E}[u_{f^*(z_{t^*})}^\top W_{OV}^{(1)} e_{z_{t^*}} \cdot (e_{z^{t^*}} + e_\star) e_{EOS}^\top)]$$

$$\approx \frac{\eta}{NT^2} e_\star e_{EOS}^\top + \frac{\eta}{N^2 T^2} \sum_{i=1}^{N} e_i e_{EOS}^\top,$$

- Second term is negligible $\longrightarrow$ **attention focuses on trigger**

What about finite samples?

Long context and random embeddings induce additional noise:

Theorem (Conditions for perfect recall, Vural et al., 2026)

With n samples, the model achieves perfect accuracy w.h.p. if $d^2 \gtrsim N$ *and* ***alignment dominates noise terms****:*

$$\underbrace{\frac{1}{NT^2}}_{\text{Alignment}} \gtrsim \underbrace{\frac{1}{n\sqrt{T}d(d \wedge T)}}_{\text{Gradient noise}} + \underbrace{\frac{1}{n\sqrt{Nd}(d \wedge T)}}_{\text{Mean bias}}.$$

Vural, B., Soltanolkotabi, and Wu, 2026. Learning to Recall with Transformers Beyond Orthogonal Embeddings

What about finite samples?

Long context and random embeddings induce additional noise:

Theorem (Conditions for perfect recall, Vural et al., 2026)

With n samples, the model achieves perfect accuracy w.h.p. if $d^2 \gtrsim N$ *and* ***alignment dominates noise terms****:*

$$\underbrace{\frac{1}{NT^2}}_{\text{Alignment}} \gtrsim \underbrace{\frac{1}{n\sqrt{T}d(d \wedge T)}}_{\text{Gradient noise}} + \underbrace{\frac{1}{n\sqrt{Nd}(d \wedge T)}}_{\text{Mean bias}}.$$

- *Alignment:* alignment between learned KQ weights and trigger direction $e_\star$.

Vural, B., Soltanolkotabi, and Wu, 2026. Learning to Recall with Transformers Beyond Orthogonal Embeddings

What about finite samples?

Long context and random embeddings induce additional noise:

Theorem (Conditions for perfect recall, Vural et al., 2026)

With n samples, the model achieves perfect accuracy w.h.p. if $d^2 \gtrsim N$ *and* ***alignment dominates noise terms****:*

$$\underbrace{\frac{1}{NT^2}}_{\text{Alignment}} \gtrsim \underbrace{\frac{1}{n\sqrt{T}d(d \wedge T)}}_{\text{Gradient noise}} + \underbrace{\frac{1}{n\sqrt{Nd}(d \wedge T)}}_{\text{Mean bias}}.$$

- *Alignment:* alignment between learned KQ weights and trigger direction $e_\star$.
- *Gradient noise:* finite-sample noise in updating W_{KQ}.

Vural, B., Soltanolkotabi, and Wu, 2026. Learning to Recall with Transformers Beyond Orthogonal Embeddings

What about finite samples?

Long context and random embeddings induce additional noise:

Theorem (Conditions for perfect recall, Vural et al., 2026)

With n samples, the model achieves perfect accuracy w.h.p. if $d^2 \gtrsim N$ *and* ***alignment dominates noise terms****:*

$$\underbrace{\frac{1}{NT^2}}_{\text{Alignment}} \gtrsim \underbrace{\frac{1}{n\sqrt{T}d(d \wedge T)}}_{\text{Gradient noise}} + \underbrace{\frac{1}{n\sqrt{Nd}(d \wedge T)}}_{\text{Mean bias}}.$$

- *Alignment:* alignment between learned KQ weights and trigger direction $e_\star$.
- *Gradient noise:* finite-sample noise in updating W_{KQ}.
- *Mean bias:* due to nonzero mean of token embeddings

Vural, B., Soltanolkotabi, and Wu, 2026. Learning to Recall with Transformers Beyond Orthogonal Embeddings

Capacity scaling at finite samples

- The bottleneck is the *mean bias* term
- The main theorem reduces to the parameter count ($= 2d^2$) requirement:

$$d^2 \gtrsim \begin{cases} N, & T \ll N^{\frac{1}{8}}\sqrt{n} \\ \frac{N^{\frac{2}{3}} T^{\frac{8}{3}}}{n^{\frac{4}{3}}}, & \text{o/w} \end{cases}$$

Capacity scaling at finite samples

- The bottleneck is the *mean bias* term
- The main theorem reduces to the parameter count ($= 2d^2$) requirement:

$$d^2 \gtrsim \begin{cases} N, & T \ll N^{\frac{1}{8}}\sqrt{n} \\ \frac{N^{\frac{2}{3}} T^{\frac{8}{3}}}{n^{\frac{4}{3}}}, & \text{o/w} \end{cases}$$

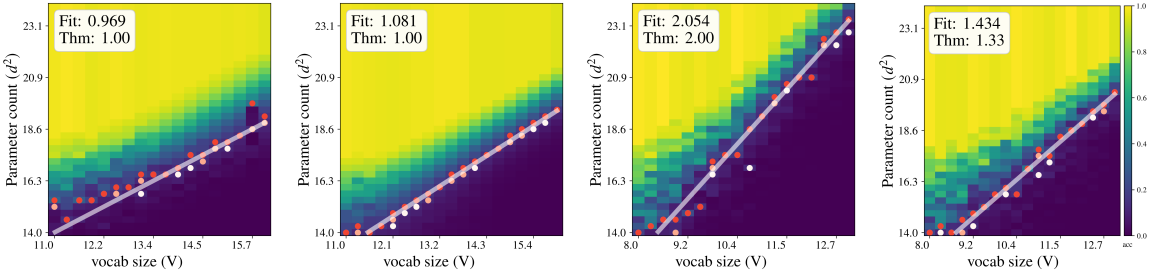
- Empirics:
  - Short vs long sequence: $T = \sqrt{N}$ vs $T = N$
  - Small vs large data: $n = N \log N$ vs $n = N\sqrt{N}$

Capacity scaling at finite samples

- The bottleneck is the *mean bias* term
- The main theorem reduces to the parameter count ($= 2d^2$) requirement:

$$d^2 \gtrsim \begin{cases} N, & T \ll N^{\frac{1}{8}}\sqrt{n} \\ \dfrac{N^{\frac{2}{3}} T^{\frac{8}{3}}}{n^{\frac{4}{3}}}, & \text{o/w} \end{cases}$$

- Empirics:
  - Short vs long sequence: $T = \sqrt{N}$ vs $T = N$
  - Small vs large data: $n = N \log N$ vs $n = N\sqrt{N}$



(a) **Short Seq / Small Data** (b) **Short Seq / Large Data** (c) **Long Seq / Small Data** (d) **Long Seq / Large Data**

Extensions to Attention + MLP

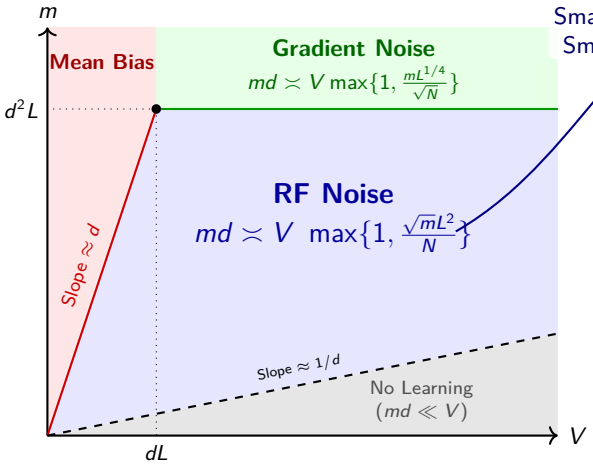
- Replace W_{OV} by random feature MLP of width m: $V, W \in \mathbb{R}^{d \times m}$ with W random

$$F_k(X; W_{OV}, W_{KQ}) = u_k V \sigma(W^\top X \mathcal{S}(X^\top W_{KQ} e_{EOS}))$$

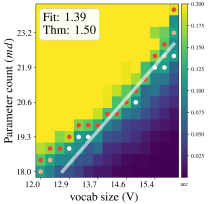
- Allows smaller embedding dimension d by using a larger width m
- Leads to additional MLP noise term that often dominates the other two

Extensions to Attention + MLP

$N \in \{V \log V, V^{1.5}\}$, $m \in \{d^2, d^3\}$, $L \asymp \sqrt{V}$

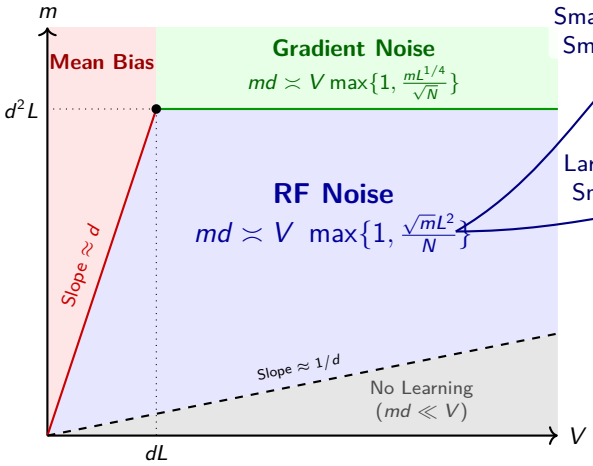


Small Sample,
Small Width



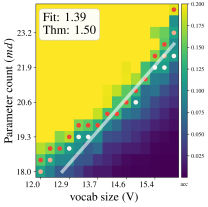
Extensions to Attention + MLP

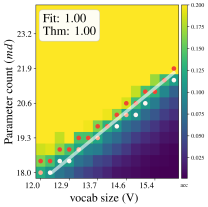
$N \in \{V \log V, V^{1.5}\}$, $m \in \{d^2, d^3\}$, $L \asymp \sqrt{V}$



Small Sample, Small Width

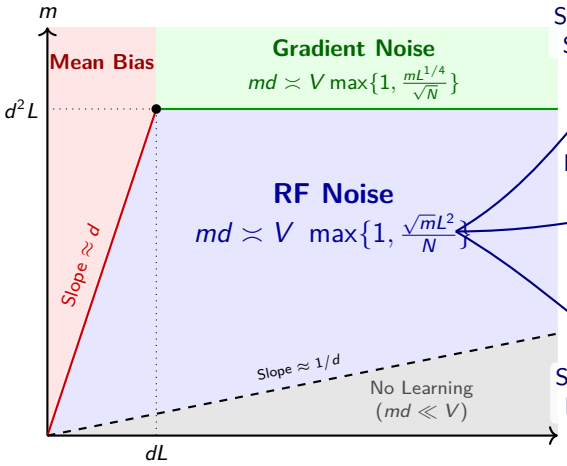
Large Sample, Small Width





Extensions to Attention + MLP

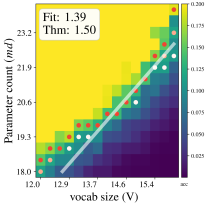
$N \in \{V \log V, V^{1.5}\}$, $m \in \{d^2, d^3\}$, $L \asymp \sqrt{V}$

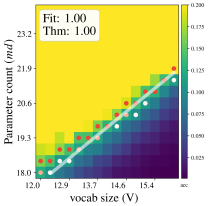


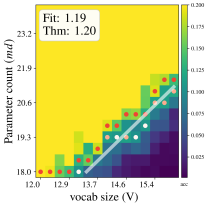
Small Sample, Small Width

Large Sample, Small Width

Small Sample, Large Width







Concluding remarks

Transformer weights as associative memories

- Storage capacity and gradient-based learning
- Toy models of reasoning and factual recall
- Finite sample results with multiplicative scaling

Concluding remarks

Transformer weights as associative memories

- Storage capacity and gradient-based learning
- Toy models of reasoning and factual recall
- Finite sample results with multiplicative scaling

Future directions

- Optimization for multiple steps
- Fine-grained study of capacity
- Learning embeddings

Concluding remarks

Transformer weights as associative memories

- Storage capacity and gradient-based learning
- Toy models of reasoning and factual recall
- Finite sample results with multiplicative scaling

Future directions

- Optimization for multiple steps
- Fine-grained study of capacity
- Learning embeddings

Thank you!

References I

Z. Allen-Zhu and Y. Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. *arXiv preprint arXiv:2404.05405*, 2024.

A. B., V. Cabannes, D. Bouchacourt, H. Jegou, and L. Bottou. Birth of a transformer: A memory viewpoint. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

J. Ba, M. A. Erdogdu, T. Suzuki, Z. Wang, D. Wu, and G. Yang. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

V. Cabannes, E. Dohmatob, and A. B. Scaling laws for associative memories. In *International Conference on Learning Representations (ICLR)*, 2024a.

V. Cabannes, B. Simsek, and A. B. Learning associative memories with gradient descent. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024b.

L. Chen, J. Bruna, and A. B. How truncating weights improves reasoning in language models. *arXiv preprint arXiv:2406.03068*, 2024.

A. Damian, J. Lee, and M. Soltanolkotabi. Neural networks can learn representations with gradient descent. In *Conference on Learning Theory (COLT)*, 2022.

Y. Dandi, F. Krzakala, B. Loureiro, L. Pesce, and L. Stephan. Learning two-layer neural networks, one (giant) step at a time. *arXiv preprint arXiv:2305.18270*, 2023.

References II

M. Demircigil, J. Heusel, M. Löwe, S. Upgang, and F. Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168:288–299, 2017.

N. Elhage, N. Nanda, C. Olsson, T. Henighan, N. Joseph, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, N. DasSarma, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

M. Geva, R. Schuster, J. Berant, and O. Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

M. Hutter. Learning curve theory. *arXiv preprint arXiv:2102.04074*, 2021.

A. Iscen, T. Furon, V. Gripon, M. Rabbat, and H. Jégou. Memory vectors for similarity search in high-dimensional spaces. *IEEE transactions on big data*, 4(1):65–77, 2017.

T. Kohonen. Correlation matrix memories. *IEEE Transactions on Computers*, 1972.

D. Krotov and J. J. Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

References III

J. Lindsey, W. Gurnee, E. Ameisen, B. Chen, A. Pearce, N. L. Turner, C. Citro, D. Abrahams, S. Carter, B. Hosmer, J. Marcus, M. Sklar, A. Templeton, T. Bricken, C. McDougall, H. Cunningham, T. Henighan, A. Jermyn, A. Jones, A. Persic, Z. Qi, T. B. Thompson, S. Zimmerman, K. Rivoire, T. Conerly, C. Olah, and J. Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL `https://transformer-circuits.pub/2025/attribution-graphs/biology.html`.

B. Liu, J. T. Ash, S. Goel, A. Krishnamurthy, and C. Zhang. Transformers learn shortcuts to automata. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

W. Merrill, A. Sabharwal, and N. A. Smith. Saturated transformers are constant-depth threshold circuits. *Transactions of the Association for Computational Linguistics*, 10:843–856, 2022.

E. Nichani, J. D. Lee, and A. B. Understanding factual recall in transformers via associative memories. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

C. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma, T. Henighan, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, S. Johnston, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022.

C. Sanford, D. Hsu, and M. Telgarsky. Representational strengths and limitations of transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

References IV

P. Sharma, J. T. Ash, and D. Misra. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. *arXiv preprint arXiv:2312.13558*, 2023.

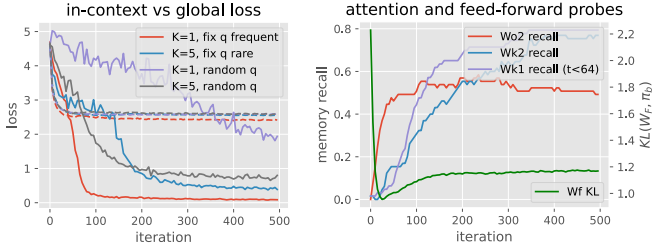
N. M. Vural, A. B., M. Soltanolkotabi, and D. Wu. Learning to recall with transformers beyond orthogonal embeddings. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026.

K. Wang, A. Variengien, A. Conmy, B. Shlegeris, and J. Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Non-holographic associative memory. *Nature*, 222(5197):960–962, 1969.

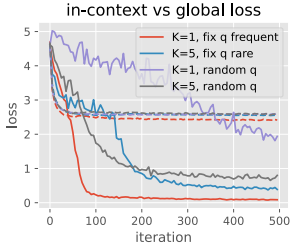
G. Yang and E. J. Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

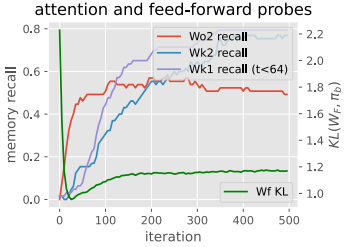
Global vs in-context associations

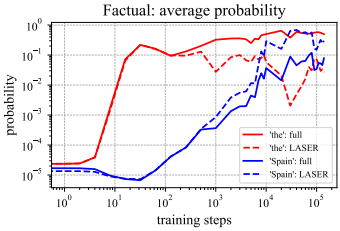


- Global bigrams are learned much faster than induction head, tend to be stored in MLPs

Global vs in-context associations





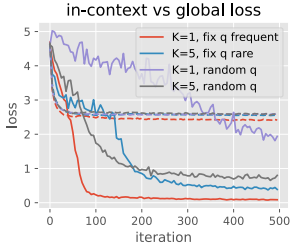


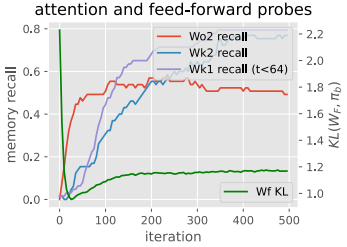
- Global bigrams are learned much faster than induction head, tend to be stored in MLPs

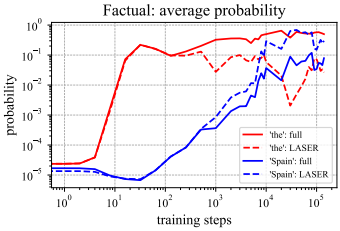
Trade-offs between global and in-context predictions (Chen, Bruna, and B., 2024)

- Trade-offs also appear in LLMs
  - "Madrid is located in" $\rightarrow$ {the, Spain} on Pythia-1B
  - Ablating late-layer MLPs (Sharma et al., 2023) changes prediction from global to in-context

Global vs in-context associations







- Global bigrams are learned much faster than induction head, tend to be stored in MLPs

Trade-offs between global and in-context predictions (Chen, Bruna, and B., 2024)

- Trade-offs also appear in LLMs
  - "Madrid is located in" $\to$ {the, Spain} on Pythia-1B
  - Ablating late-layer MLPs (Sharma et al., 2023) changes prediction from global to in-context

Theorem (Chen et al., 2024, informal)

In toy setting, feed-forward layer learns global bigram after $O(1)$ samples, attention after $O(N)$ samples due to noise.

Setup with heavy-tailed data

Setting

- $z_i \sim p(z)$, $y_i = f^*(z_i)$, n samples: $S_n = \{z_1, \dots, z_n\}$, 0/1 loss:

$$L(\hat{f}_n) = \mathbb{P}(y \neq \hat{f}_n(z))$$

Setup with heavy-tailed data

Setting

- $z_i \sim p(z)$, $y_i = f^*(z_i)$, n samples: $S_n = \{z_1, \dots, z_n\}$, 0/1 loss:

$$L(\hat{f}_n) = \mathbb{P}(y \neq \hat{f}_n(z))$$

- Heavy-tailed token frequencies: Zipf law (typical for language where N is very large)

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

Setup with heavy-tailed data

Setting

- $z_i \sim p(z)$, $y_i = f^*(z_i)$, n samples: $S_n = \{z_1, \dots, z_n\}$, 0/1 loss:

$$L(\hat{f}_n) = \mathbb{P}(y \neq \hat{f}_n(z))$$

- Heavy-tailed token frequencies: Zipf law (typical for language where N is very large)

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

- Hutter (2021): with infinite memory, we have

$$L(\hat{f}_n) \lesssim n^{-\frac{\alpha-1}{\alpha}}$$

Setup with heavy-tailed data

Setting

- $z_i \sim p(z)$, $y_i = f^*(z_i)$, n samples: $S_n = \{z_1, \dots, z_n\}$, 0/1 loss:

$$L(\hat{f}_n) = \mathbb{P}(y \neq \hat{f}_n(z))$$

- Heavy-tailed token frequencies: Zipf law (typical for language where N is very large)

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

- Hutter (2021): with infinite memory, we have

$$L(\hat{f}_n) \lesssim n^{-\frac{\alpha-1}{\alpha}}$$

- **Q: What about finite capacity?**

Scaling laws with finite capacity

- Random embeddings $e_z, u_y \in \mathbb{R}^d$ with $\mathcal{N}(0, 1/d)$ entries
- Estimator: $\hat{f}_{n,d}(x) = \arg\max_y {u_y}^\top W_{n,d} e_z$, with

$$W_{n,d} = \sum_{z=1}^{N} q(z) u_{f^*(z)} {e_z}^\top$$

Scaling laws with finite capacity

- Random embeddings $e_z, u_y \in \mathbb{R}^d$ with $\mathcal{N}(0, 1/d)$ entries
- Estimator: $\hat{f}_{n,d}(x) = \arg\max_y {u_y}^\top W_{n,d} e_z$, with

$$W_{n,d} = \sum_{z=1}^{N} q(z) u_{f^*(z)} {e_z}^\top$$

- Single population gradient step on cross-entropy loss: $q(z) \approx p(z)$

Scaling laws with finite capacity

- Random embeddings $e_z, u_y \in \mathbb{R}^d$ with $\mathcal{N}(0, 1/d)$ entries
- Estimator: $\hat{f}_{n,d}(x) = \arg\max_y {u_y}^\top W_{n,d} e_z$, with

$$W_{n,d} = \sum_{z=1}^{N} q(z) u_{f^*(z)} {e_z}^\top$$

- Single population gradient step on cross-entropy loss: $q(z) \approx p(z)$

Theorem (Cabannes, Dohmatob, and B., 2024a, informal)

1. *For* $q(z) = \sum_i \mathbb{1}\{z = z_i\}$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-\frac{\alpha-1}{2\alpha}}$

Scaling laws with finite capacity

- Random embeddings $e_z, u_y \in \mathbb{R}^d$ with $\mathcal{N}(0, 1/d)$ entries
- Estimator: $\hat{f}_{n,d}(x) = \arg\max_y {u_y}^\top W_{n,d} e_z$, with

$$W_{n,d} = \sum_{z=1}^{N} q(z) u_{f^*(z)} {e_z}^\top$$

- Single population gradient step on cross-entropy loss: $q(z) \approx p(z)$

Theorem (Cabannes, Dohmatob, and B., 2024a, informal)

1. *For* $q(z) = \sum_i \mathbb{1}\{z = z_i\}$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-\frac{\alpha-1}{2\alpha}}$
2. *For* $q(z) = \mathbb{1}\{z \in S_n\}$, *and* $d \gg N$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-k}$ *for any* k

Scaling laws with finite capacity

- Random embeddings $e_z, u_y \in \mathbb{R}^d$ with $\mathcal{N}(0, 1/d)$ entries
- Estimator: $\hat{f}_{n,d}(x) = \arg\max_y u_y{}^\top W_{n,d} e_z$, with

$$W_{n,d} = \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z{}^\top$$

- Single population gradient step on cross-entropy loss: $q(z) \approx p(z)$

Theorem (Cabannes, Dohmatob, and B., 2024a, informal)

1. *For* $q(z) = \sum_i \mathbb{1}\{z = z_i\}$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-\frac{\alpha-1}{2\alpha}}$
2. *For* $q(z) = \mathbb{1}\{z \in S_n\}$, *and* $d \gg N$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-k}$ *for any* k
3. *For* $q(z) = \mathbb{1}\{z$ *seen at least* s *times in* $S_n\}$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-\alpha+1}$

Scaling laws with finite capacity

- Random embeddings $e_z, u_y \in \mathbb{R}^d$ with $\mathcal{N}(0, 1/d)$ entries
- Estimator: $\hat{f}_{n,d}(x) = \arg\max_y u_y{}^\top W_{n,d} e_z$, with

$$W_{n,d} = \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z{}^\top$$

- Single population gradient step on cross-entropy loss: $q(z) \approx p(z)$

Theorem (Cabannes, Dohmatob, and B., 2024a, informal)

1. *For* $q(z) = \sum_i \mathbb{1}\{z = z_i\}$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-\frac{\alpha-1}{2\alpha}}$
2. *For* $q(z) = \mathbb{1}\{z \in S_n\}$, *and* $d \gg N$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-k}$ *for any* k
3. *For* $q(z) = \mathbb{1}\{z$ *seen at least* s *times in* $S_n\}$: $L(\hat{f}_{n,d}) \lesssim n^{-\frac{\alpha-1}{\alpha}} + d^{-\alpha+1}$

- $n^{-\frac{\alpha-1}{\alpha}}$ is the same as (Hutter, 2021)
- $q = 1$ is best if we have enough capacity
- Can store at most d memories (approximation error: $d^{-\alpha+1}$)

Scaling laws with optimization algorithms

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)] \qquad \to \qquad W_{n,d} \approx \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z^\top$$

Different algorithms lead to different memory schemes $q(z)$:

Scaling laws with optimization algorithms

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)] \qquad \to \qquad W_{n,d} \approx \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z{}^\top$$

Different algorithms lead to different memory schemes $q(z)$:

- One step of SGD with large batch: $q(z) \approx p(z)$

Scaling laws with optimization algorithms

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)] \qquad \to \qquad W_{n,d} \approx \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z{}^\top$$

Different algorithms lead to different memory schemes $q(z)$:

- One step of SGD with large batch: $q(z) \approx p(z)$
- SGD with batch size one + large step-size, $d \gg N$: $q(z) \approx 1$

Scaling laws with optimization algorithms

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)] \qquad \to \qquad W_{n,d} \approx \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z^\top$$

Different algorithms lead to different memory schemes $q(z)$:

- One step of SGD with large batch: $q(z) \approx p(z)$
- SGD with batch size one + large step-size, $d \gg N$: $q(z) \approx 1$
- For $d \leq N$, smaller step-sizes can help later in training

Scaling laws with optimization algorithms

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)] \qquad \to \qquad W_{n,d} \approx \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z^\top$$

Different algorithms lead to different memory schemes $q(z)$:

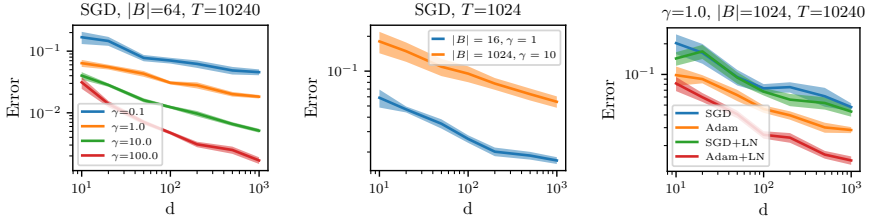
- One step of SGD with large batch: $q(z) \approx p(z)$
- SGD with batch size one + large step-size, $d \gg N$: $q(z) \approx 1$
- For $d \leq N$, smaller step-sizes can help later in training
- Adam and layer-norm help with practical settings (large batch sizes + smaller step-size)

Scaling laws with optimization algorithms

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)] \qquad \rightarrow \qquad W_{n,d} \approx \sum_{z=1}^{N} q(z) u_{f^*(z)} e_z^\top$$

Different algorithms lead to different memory schemes $q(z)$:

- One step of SGD with large batch: $q(z) \approx p(z)$
- SGD with batch size one + large step-size, $d \gg N$: $q(z) \approx 1$
- For $d \leq N$, smaller step-sizes can help later in training
- Adam and layer-norm help with practical settings (large batch sizes + smaller step-size)



Optimization with imbalance and small capacity

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)], \qquad \ell\text{: cross-entropy loss}$$

Benefits of large step-sizes + oscillations: (Cabannes, Simsek, and B., 2024b)

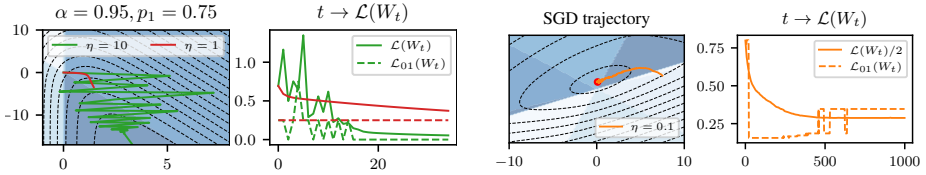
- Orthogonal embeddings $\implies$ logarithmic growth of margins for any step-size

Optimization with imbalance and small capacity

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)], \qquad \ell\text{: cross-entropy loss}$$

Benefits of large step-sizes + oscillations: (Cabannes, Simsek, and B., 2024b)

- Orthogonal embeddings $\implies$ logarithmic growth of margins for any step-size
- Correlated embeddings + imbalance $\implies$ oscillatory regimes

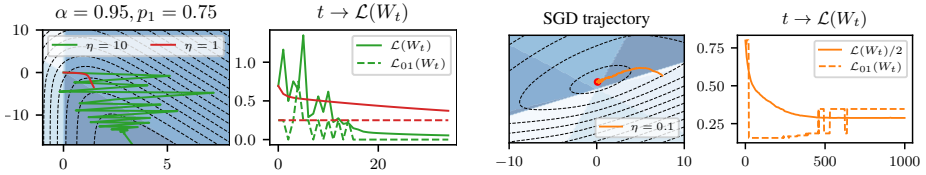


Optimization with imbalance and small capacity

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)], \qquad \ell\text{: cross-entropy loss}$$

Benefits of large step-sizes + oscillations: (Cabannes, Simsek, and B., 2024b)

- Orthogonal embeddings $\implies$ logarithmic growth of margins for any step-size
- Correlated embeddings + imbalance $\implies$ oscillatory regimes
- Large step-sizes help reach perfect accuracy faster despite oscillations (empirically)



Optimization with imbalance and small capacity

$$L(W) = \mathbb{E}_{z\sim p}[\ell(f^*(z), UWe_z)], \qquad \ell\text{: cross-entropy loss}$$

Benefits of large step-sizes + oscillations: (Cabannes, Simsek, and B., 2024b)

- Orthogonal embeddings $\implies$ logarithmic growth of margins for any step-size
- Correlated embeddings + imbalance $\implies$ oscillatory regimes
- Large step-sizes help reach perfect accuracy faster despite oscillations (empirically)
- Over-optimization can hurt in under-parameterized settings (empirically)