Pseudo-likelihood produces associative memories able to generalize, even for asymmetric couplings

Francesco D'Amico

SAPIENZA
UNIVERSITÀ DI ROMA

Dipartimento di Fisica

4th Workshop of UMI Group, Januray 23, 2026

What is pseudo-likelihood? *Generative framework*

- Energy-based generative models:

$$p_{\beta,\theta}(x) = \frac{e^{-\beta \mathscr{H}_\theta(x)}}{Z(\beta,\theta)}$$

What is pseudo-likelihood? *Generative framework*

- Energy-based generative models:

$$p_{\beta,\theta}(x) = \frac{e^{-\beta \mathscr{H}_\theta(x)}}{Z(\beta,\theta)}$$

- Goal: dataset distribution $q(x)$, find θ that minimizes $D_{KL}(q||p_{\beta,\theta})$

What is pseudo-likelihood? *Generative framework*

- Energy-based generative models:

$$p_{\beta,\theta}(x) = \frac{e^{-\beta \mathscr{H}_\theta(x)}}{Z(\beta,\theta)}$$

- Goal: dataset distribution $q(x)$, find θ that minimizes $D_{KL}(q||p_{\beta,\theta})$
- Loss: given P training data $\{\xi^\mu\}_{\mu=1,\dots,P}$, *(negative) likelihood*

$$\mathscr{L} = -\sum_\mu \log p_{\beta,\theta}(\xi^\mu)$$

Problem: cost of exact computation of $Z(\beta,\theta)$ is $\mathscr{O}(e^N)$

What is pseudo-likelihood? *Generative framework*

- Energy-based generative models:

$$p_{\beta,\theta}(x) = \frac{e^{-\beta \mathscr{H}_\theta(x)}}{Z(\beta,\theta)}$$

- Goal: dataset distribution $q(x)$, find θ that minimizes $D_{KL}(q||p_{\beta,\theta})$
- Loss: given P training data $\{\xi^\mu\}_{\mu=1,\dots,P}$, *(negative) likelihood*

$$\mathscr{L} = -\sum_\mu \log p_{\beta,\theta}(\xi^\mu)$$

  Problem: cost of exact computation of $Z(\beta,\theta)$ is $\mathscr{O}(e^N)$

- Pseudo-likelihood approximation: given $x = \{x_i\}_{i=1,\dots,N}$

$$p_{\lambda,\theta}(x) = \prod_{i=1}^{N} p_{\lambda,\theta,i}(x_i|x_{/i})$$

What is pseudo-likelihood? *Generative framework*

- Energy-based generative models:

$$p_{\beta,\theta}(x) = \frac{e^{-\beta \mathscr{H}_\theta(x)}}{Z(\beta,\theta)}$$

- Goal: dataset distribution $q(x)$, find θ that minimizes $D_{KL}(q||p_{\beta,\theta})$
- Loss: given P training data $\{\xi^\mu\}_{\mu=1,\dots,P}$, *(negative) likelihood*

$$\mathscr{L} = -\sum_\mu \log p_{\beta,\theta}(\xi^\mu)$$

  Problem: cost of exact computation of $Z(\beta,\theta)$ is $\mathscr{O}(e^N)$

- Pseudo-likelihood approximation: given $x = \{x_i\}_{i=1,\dots,N}$

$$p_{\lambda,\theta}(x) = \prod_{i=1}^{N} p_{\lambda,\theta,i}(x_i|x_{/i})$$

- Pseudo-likelihood loss $\mathscr{O}(N)$ to compute:

$$\mathscr{PL} = -\sum_\mu \sum_i \log p_{\lambda,i}(\xi_i^\mu|\xi_{/i}^\mu)$$

Setting

- Two-bodies interaction: parameters $\theta = \{J_{ij}\}_{i,j=1,\dots N}$

$$\mathscr{H}(x) = -\sum_{i<j} x_i J_{ij} x_j$$

Setting

- Two-bodies interaction: parameters $\theta = \{J_{ij}\}_{i,j=1,\dots N}$

$$\mathscr{H}(x) = -\sum_{i<j} x_i J_{ij} x_j$$

- Binary variables: $x \in \{\pm 1\}^N$

Setting

- Two-bodies interaction: parameters $\theta = \{J_{ij}\}_{i,j=1,\dots N}$

$$\mathscr{H}(x) = -\sum_{i<j} x_i J_{ij} x_j$$

- Binary variables: $x \in \{\pm 1\}^N$

- Pseudo-likelihood loss coincides with cross-entropy:

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}\left[\left(\lambda \xi_i^\mu \sum_{j(\neq i)} J_{ij}\xi_j^\mu\right) - \log\cosh\left(\lambda \xi_i^\mu \sum_{j(\neq i)} J_{ij}\xi_j^\mu\right)\right]$$

Setting

- Two-bodies interaction: parameters $\theta = \{J_{ij}\}_{i,j=1,\dots N}$

$$\mathscr{H}(x) = -\sum_{i<j} x_i J_{ij} x_j$$

- Binary variables: $x \in \{\pm 1\}^N$

- Pseudo-likelihood loss coincides with cross-entropy:

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}\left[\left(\lambda \xi_i^{\mu} \sum_{j(\neq i)} J_{ij}\xi_j^{\mu}\right) - \log\cosh\left(\lambda \xi_i^{\mu} \sum_{j(\neq i)} J_{ij}\xi_j^{\mu}\right)\right]$$

- Each neuron has its own independent (*local*) cost to minimize

$$\mathscr{PL} = -\sum_{\mu}\sum_{i} e_i(\xi^{\mu})$$

Associative memory framework

- We select T=0 dynamical rule:

$$x_i(t+\Delta t) = \text{sign}\left[\sum_{j(\neq i)} J_{ij} x_j(t)\right]$$

Associative memory framework

- We select T=0 dynamical rule:

$$x_i(t+\Delta t) = \text{sign}\left[\sum_{j(\neq i)} J_{ij} x_j(t)\right]$$

- *Almost* equivalent to Hopfield network.
  Difference: *global learning rule*

$$J_{ij}^{\text{Hopfield}} := \frac{1}{N}\sum_{\mu}^{P} \xi_i^{\mu}\xi_j^{\mu} \quad \leftarrow \textbf{Hebb's rule}$$

Associative memory framework

- We select T=0 dynamical rule:

$$x_i(t+\Delta t) = \text{sign}\left[\sum_{j(\neq i)} J_{ij} x_j(t)\right]$$

- *Almost* equivalent to Hopfield network.
  Difference: *global learning rule*

$$J_{ij}^{\text{Hopfield}} := \frac{1}{N}\sum_{\mu}^{P} \xi_i^\mu \xi_j^\mu \quad \leftarrow \textbf{Hebb's rule}$$

- P random training data $\xi^\mu \in \{\pm 1\}^N$ i.i.d. are the *memories*

- $\alpha = \frac{P}{N}$ is the control parameter

Hopfield associative memory

Memories are fixed points of dynamics

Hopfield associative memory

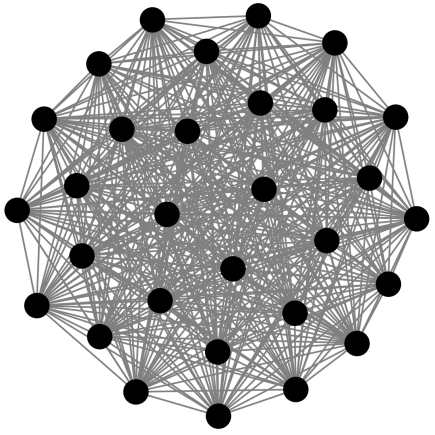
Memories are fixed points of dynamics

Hopfield associative memory

Memories are fixed points of dynamics

Input

$\Rightarrow$

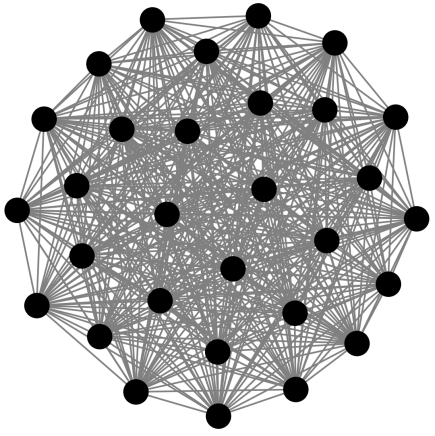


Hopfield associative memory

Memories are fixed points of dynamics

Input **Retrieval**

$\Rightarrow$

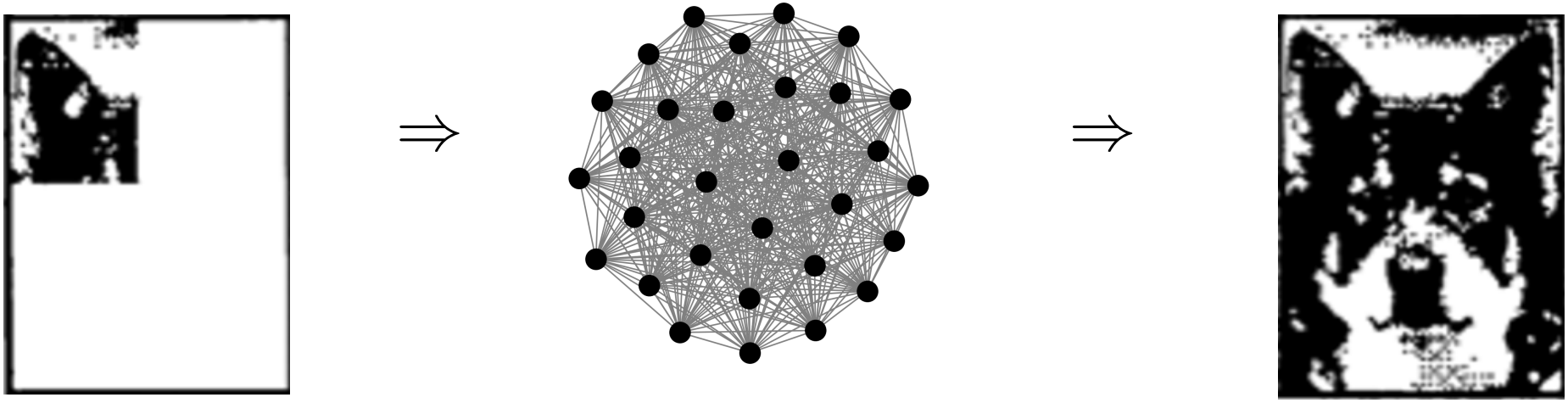


$\Rightarrow$

Hopfield associative memory

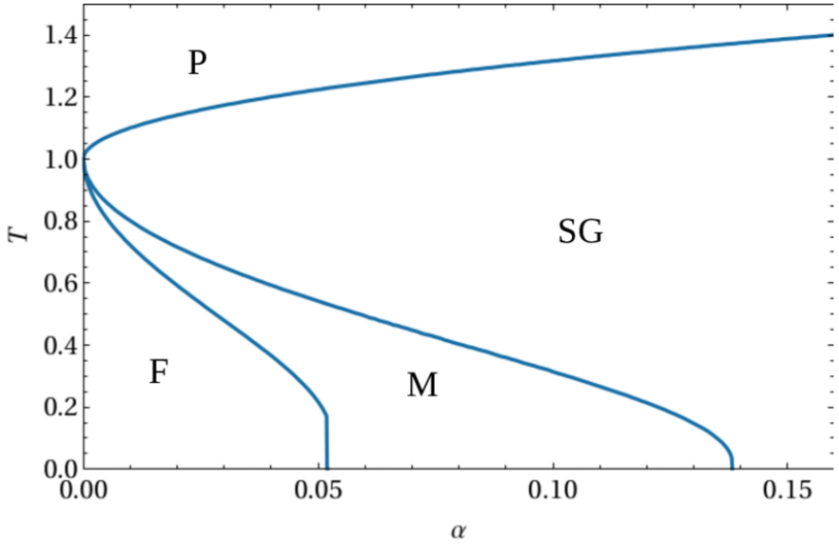
Memories are fixed points of dynamics

Input **Retrieval**



Phase diagram of retrieval

Amit et al. (1987)
Statistical Mechanics of Neural Networks Near Saturation



Stabilities

- Pseudo-likelihood loss:

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}\left[\left(\lambda\xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}\right) - \log\cosh\left(\lambda\xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}\right)\right]$$

Stabilities

- Pseudo-likelihood loss:

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}\left[\left(\lambda\xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}\right) - \log\cosh\left(\lambda\xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}\right)\right]$$

- Each neuron has its own independent (*local*) cost to minimize

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}e_i(\xi^{\mu})$$

J_{ij} not symmetric $\Rightarrow$ no global energy

Stabilities

- Pseudo-likelihood loss:

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}\left[\left(\lambda\xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}\right) - \log\cosh\left(\lambda\xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}\right)\right]$$

- Each neuron has its own independent (*local*) cost to minimize

$$\mathscr{PL} = -\sum_{\mu}\sum_{i}e_i(\xi^{\mu})$$

J_{ij} not symmetric $\Rightarrow$ no global energy

- Stabilities

$$\Delta_i^{\mu} := \xi_i^{\mu}\sum_{j(\neq i)}J_{ij}\xi_j^{\mu}$$

If $\Delta_i^{\mu} > 0 \quad \forall i \quad \Rightarrow \quad \xi^{\mu}$ is a fixed point of dynamics

Technical contribution (1) - fixed $||J||$

- Equivalent to N independent perceptrons, each one with loss

$$\mathscr{P}\mathscr{L} = -\sum_{\mu} \left[\lambda \Delta^{\mu} - \log\cosh\left(\lambda \Delta^{\mu}\right)\right]$$

Technical contribution (1) - fixed $||J||$

- Equivalent to N independent perceptrons, each one with loss

$$\mathscr{P}\mathscr{L} = -\sum_{\mu} [\lambda \Delta^{\mu} - \log \cosh (\lambda \Delta^{\mu})]$$

- Gardner replica computation for each perceptron

Technical contribution (1) - fixed $||J||$

- Equivalent to N independent perceptrons, each one with loss

$$\mathscr{P}\mathscr{L} = -\sum_{\mu} \left[\lambda \Delta^{\mu} - \log\cosh\left(\lambda \Delta^{\mu}\right)\right]$$

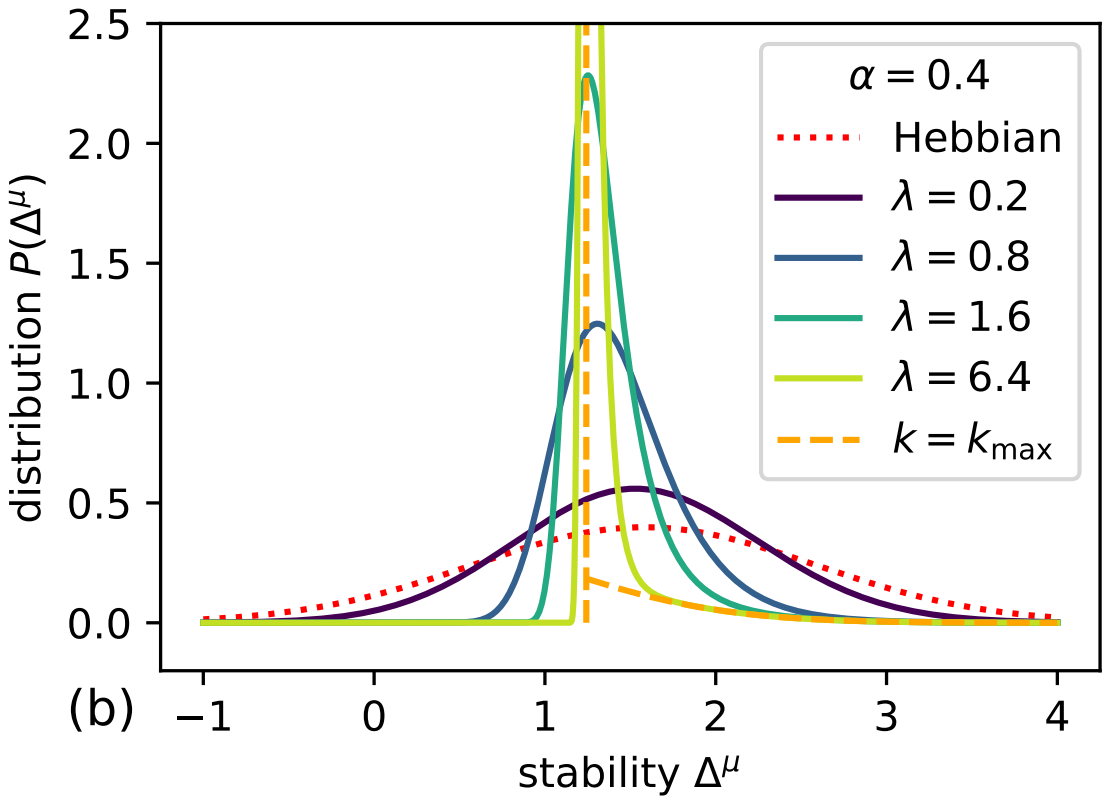
- Gardner replica computation for each perceptron
- Result: probability distribution of stabilities $P_{\lambda}\left(\Delta\right)$

Technical contribution (1) - fixed $||J||$

- Equivalent to N independent perceptrons, each one with loss

$$\mathscr{P}\mathscr{L} = -\sum_{\mu} \left[\lambda \Delta^{\mu} - \log\cosh\left(\lambda \Delta^{\mu}\right)\right]$$

- Gardner replica computation for each perceptron
- Result: probability distribution of stabilities $P_{\lambda}(\Delta)$



Technical contribution (2) - free $||J||$

- GD on cross-entropy in this setting[1] monotonically increases $||J||$

[1] D. Soudry et al., *The implicit bias of gradient descent on separable data*, 2018.

Technical contribution (2) - free $||J||$

- GD on cross-entropy in this setting[1] monotonically increases $||J||$
- We rescale at training step t: $\lambda(t) = ||J||(t)$ and we fix $||J|| = 1$

$$\mathscr{P}\mathscr{L} = -\sum_{\mu} [\lambda(t)\Delta^{\mu} - \log\cosh(\lambda(t)\Delta^{\mu})]$$

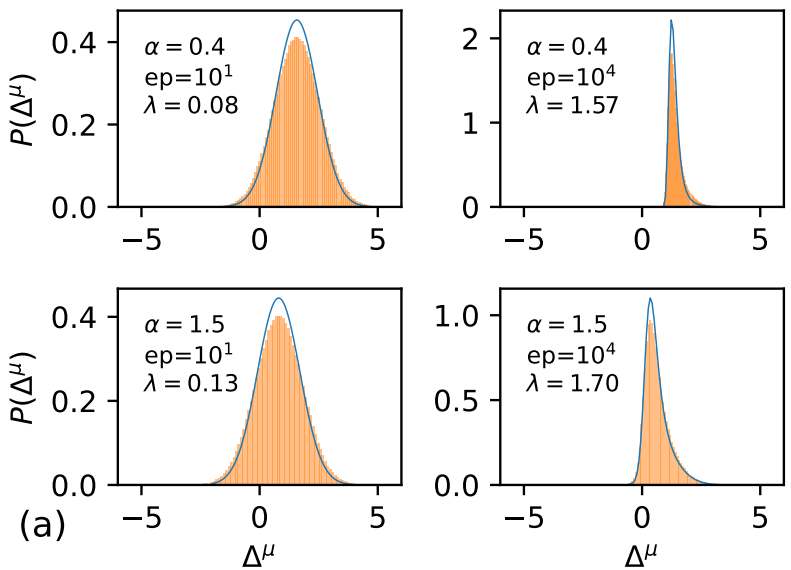
[1] D. Soudry et al., *The implicit bias of gradient descent on separable data*, 2018.

Technical contribution (2) - free $||J||$

- GD on cross-entropy in this setting[1] monotonically increases $||J||$
- We rescale at training step t: $\lambda(t) = ||J||(t)$ and we fix $||J|| = 1$

$$\mathscr{P}\mathscr{L} = -\sum_{\mu} [\lambda(t)\Delta^{\mu} - \log\cosh(\lambda(t)\Delta^{\mu})]$$

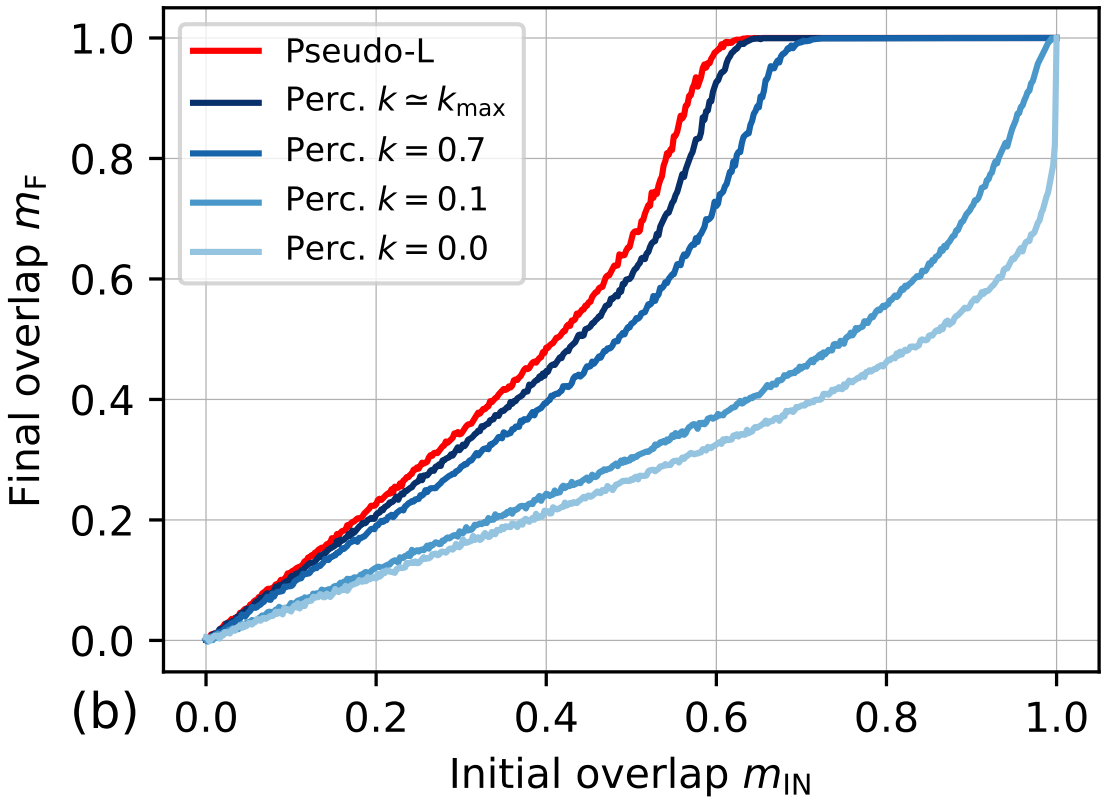
- Result: $P_{\lambda}(\Delta)$ at fixed λ good approximation of $P_{\lambda(t)}(\Delta)$



[1] D. Soudry et al., *The implicit bias of gradient descent on separable data*, 2018.

(1) *Random data*

Pseudo-likelihood produces optimal basins of attraction



(2) *Random features data*

- D random features $f^k \in \{\pm 1\}^N$

(2) *Random features data*

- D random features $f^k \in \{\pm 1\}^N$
- Training data: with random i.i.d. $c_k^\mu \in \{0, \pm 1\}$

$$\xi_i^\mu = \text{sign}\left(\frac{1}{\sqrt{D}} \sum_k c_k^\mu f_i^k\right)$$

For each μ only L non-zero c_k^μ.

(2) *Random features data*

- D random features $f^k \in \{\pm 1\}^N$
- Training data: with random i.i.d. $c_k^\mu \in \{0, \pm 1\}$

$$\xi_i^\mu = \text{sign}\left(\frac{1}{\sqrt{D}} \sum_k c_k^\mu f_i^k\right)$$

  For each μ only L non-zero c_k^μ.

- Test data: same $\{f^k\}_{k=1,\dots,D}$, new extraction of c_k^μ

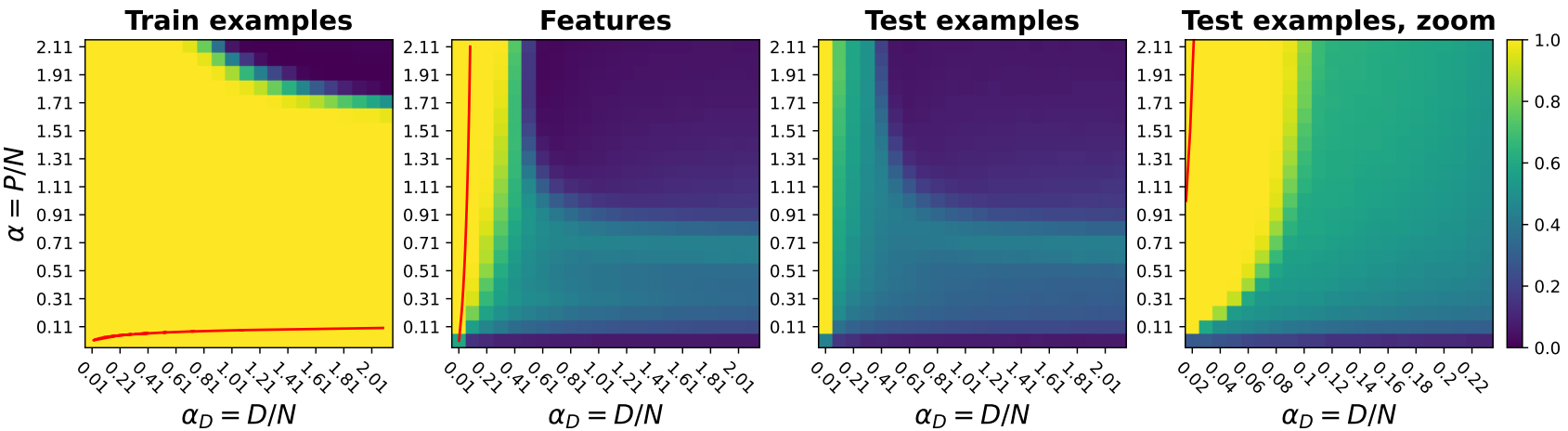
(2) *Random features data*

- D random features $f^k \in \{\pm 1\}^N$
- Training data: with random i.i.d. $c_k^\mu \in \{0, \pm 1\}$

$$\xi_i^\mu = \text{sign}\left(\frac{1}{\sqrt{D}} \sum_k c_k^\mu f_i^k\right)$$

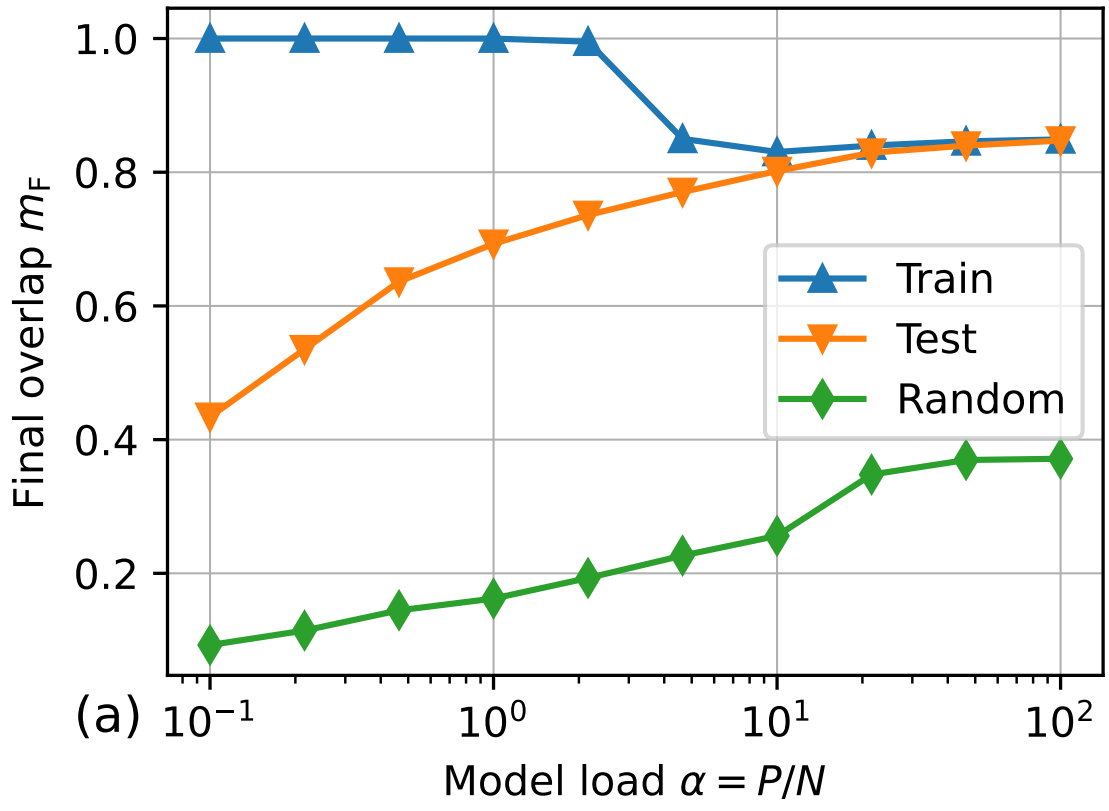
For each μ only L non-zero c_k^μ.

- Test data: same $\{f^k\}_{k=1,\dots,D}$, new extraction of c_k^μ

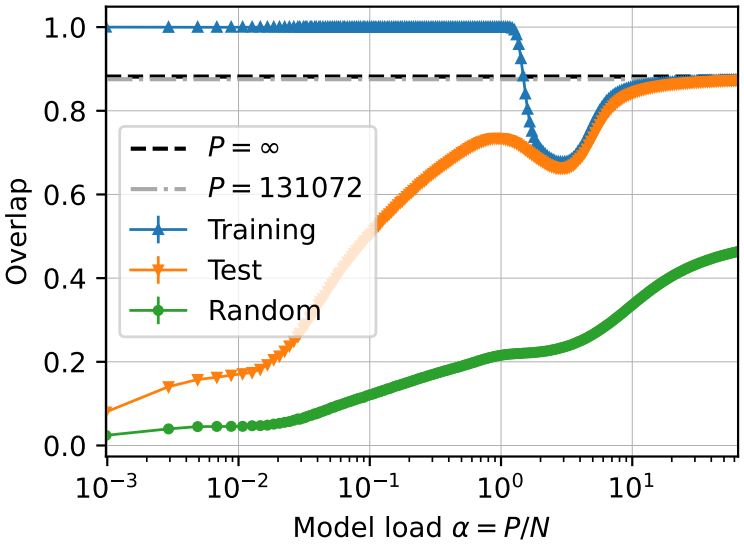


(3) binary MNIST

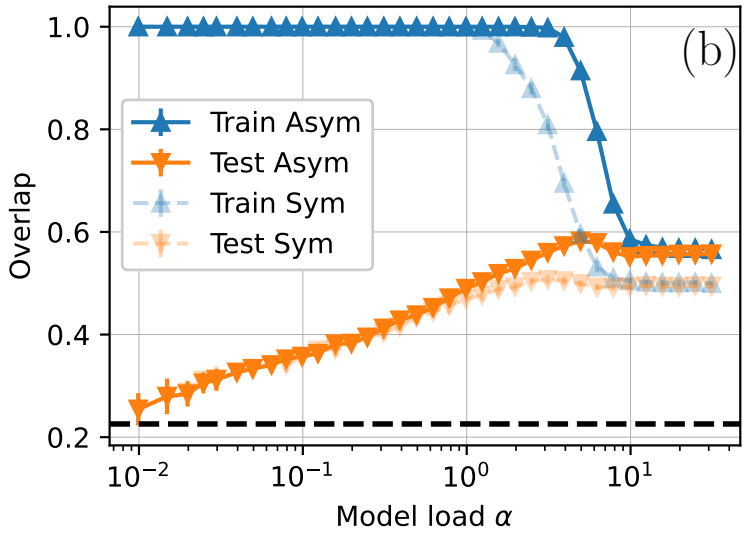
Memorization-to-generalization transition at increasing dataset size α



(3) other datasets



(a) Edwards-Anderson in two dimensions



(b) Potts model of protein family Beta Lactamase

Conclusions

- Local + max entropy learning rule:

Conclusions

- Local + max entropy learning rule:

  Associative memory

Conclusions

- Local + max entropy learning rule:

  Associative memory

- Local rules equivalent to N perceptrons:

Conclusions

- Local + max entropy learning rule:

  Associative memory

- Local rules equivalent to N perceptrons:

  $P(\Delta)$ to explain memorization in non-symmetric networks

Conclusions

- Local + max entropy learning rule:

  Associative memory

- Local rules equivalent to N perceptrons:

  $P(\Delta)$ to explain memorization in non-symmetric networks

- Structured data:

  Memorization-to-generalization transition

Thank you for your attention!

Francesco D'Amico

SAPIENZA
UNIVERSITÀ DI ROMA

Contact: francesco.damico@uniroma1.it

Formulae

Pseudo-likelihood cost:

$$p - \mathscr{L} = -\sum_{\mu}\sum_{i}\left[\lambda \mathscr{H}(\xi_i^{\mu}|\{\xi_j^{\mu}\}_{j\neq i}) - \log\int dx_i e^{-\lambda \mathscr{H}(\xi_i^{\mu}|\{\xi_j^{\mu}\}_{j\neq i})}\right]$$

Self-Attention is pseudolikelihood optimization

- Layer of simplified Self-Attention: weights tensor $\mathbb{J} \in \mathbb{R}^{N \times N \times d \times d}$

$$\vec{X}_i^{t+1} = \sum_{j(\neq i)} \alpha_{i \leftarrow j} \mathbf{J}_{ij} \vec{X}_j^t \tag{1}$$

$$\alpha_{i \leftarrow j} = \text{softmax}_j \left[\lambda \vec{X}_i^t \cdot \mathbf{J}_{ij} \vec{X}_j^t \right] \tag{2}$$

Self-Attention is pseudolikelihood optimization

- Layer of simplified Self-Attention: weights tensor $\mathbb{J} \in \mathbb{R}^{N\times N\times d\times d}$

$$\vec{X}_i^{t+1} = \sum_{j(\neq i)} \alpha_{i\leftarrow j}\mathbf{J}_{ij}\vec{X}_j^t \tag{1}$$

$$\alpha_{i\leftarrow j} = \text{softmax}_j\left[\lambda \vec{X}_i^t \cdot \mathbf{J}_{ij}\vec{X}_j^t\right] \tag{2}$$

- t layer index $\rightarrow$ promoted to time index

Self-Attention is pseudolikelihood optimization

- Layer of simplified Self-Attention: weights tensor $\mathbb{J} \in \mathbb{R}^{N\times N\times d\times d}$

$$\vec{X}_i^{t+1} = \sum_{j(\neq i)} \alpha_{i\leftarrow j} \mathbf{J}_{ij} \vec{X}_j^t \tag{1}$$

$$\alpha_{i\leftarrow j} = \text{softmax}_j \left[\lambda \vec{X}_i^t \cdot \mathbf{J}_{ij} \vec{X}_j^t \right] \tag{2}$$

- t layer index $\rightarrow$ promoted to time index
- Eq. 1: minimization dynamics of cost

$$F(\{\vec{X}_i\}; J) = -\frac{1}{\lambda} \sum_i \log \Big[\sum_{j(\neq i)} \exp\big(\lambda \vec{X}_i \cdot \mathbf{J}_{ij} \vec{X}_j\big) \Big] = \sum_i e_i(\{\vec{X}_i\}; \mathbb{J}) \tag{3}$$

$$\vec{X}_i^{t+1} = -\nabla_{\vec{X}} F(\{\vec{X}_i\}; J) \tag{4}$$

Vector-spin associative memories

Vector spins: towards Self-Attention

Statistical mechanics of vector Hopfield network near and above saturation

Flavio Nicoletti,[1,2,*] Francesco D'Amico,[2,3,†] and Matteo Negri[2,3,‡]

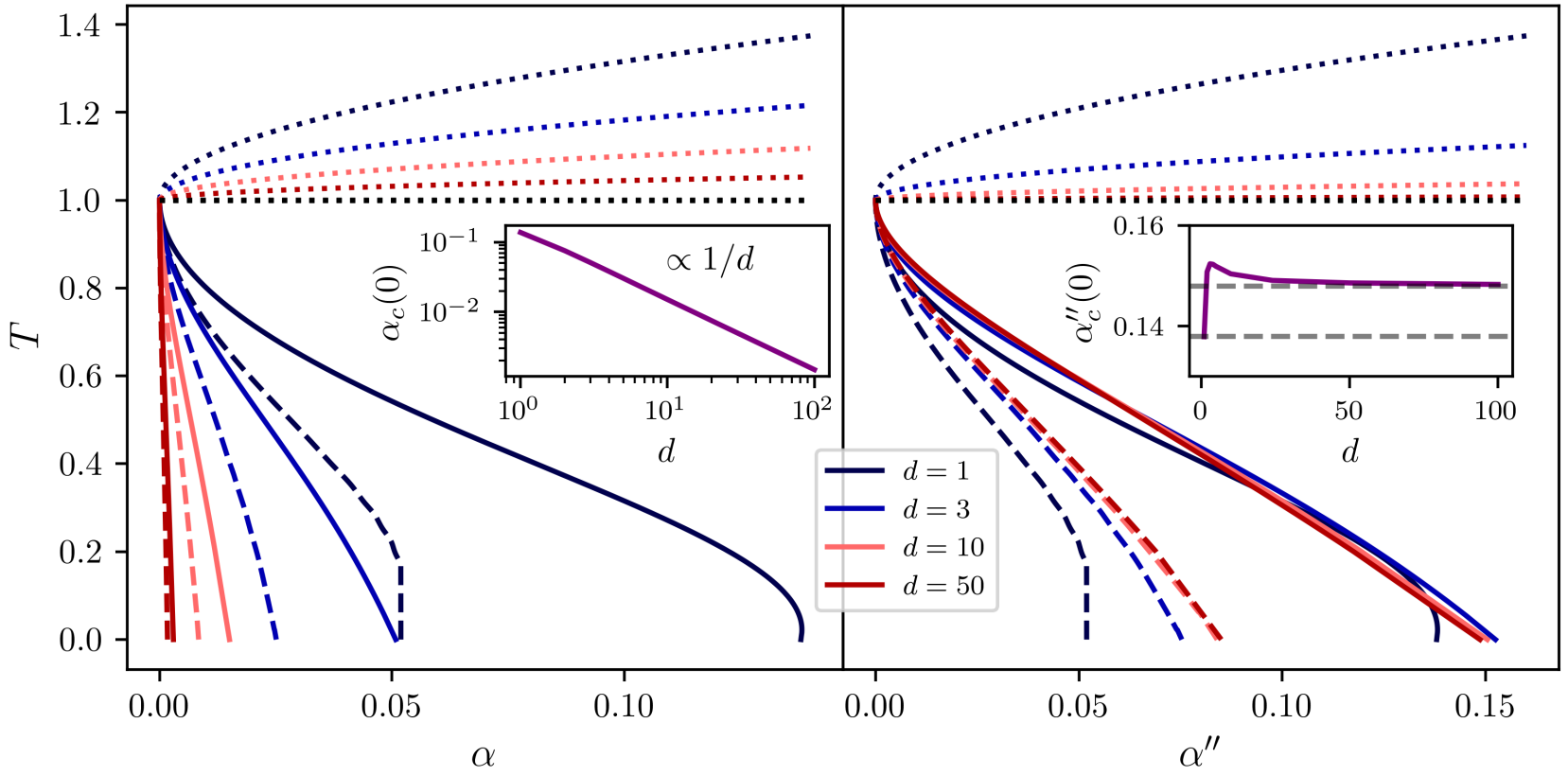
[1] *Department of Computer Science and Engineering Chalmers University of Technology and University of Gothenburg SE-41296 Gothenburg, Sweden*
[2] *Dipartimento di Fisica, Sapienza Università di Roma, 00185 Rome, Italy*
[3] *Institute of Nanotechnology, National Research Council of Italy, CNR-NANOTEC, Rome Unit*

- N spherical vector spins $\{\vec{S}_i\}_{i=1,..,N} \in \mathbb{R}^d$, $\|\vec{S}_i\| = 1$
- P memories $\{\vec{\xi}_i\}^\mu$
- $H = -\frac{1}{2}\sum_{i\neq j}^{1,N} \vec{S}_i \cdot \mathbb{J}_{ij}\vec{S}_j$
- Hebb's couplings $\mathbb{J}_{ij} = \frac{1}{N}\sum_{\mu=1}^{P} \vec{\xi}_i^{\,\mu} \times \vec{\xi}_j^{\,\mu} \Rightarrow \mathbb{J}_{ij} \in \mathbb{R}^{d\times d}$

Vector-spin associative memories

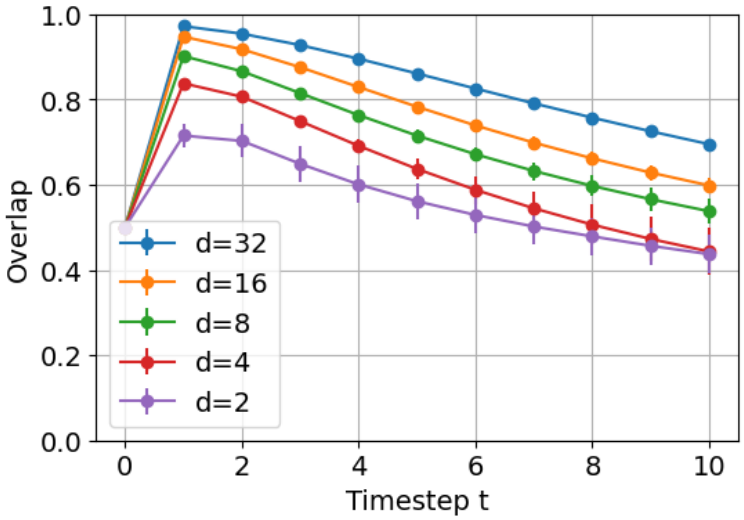
Phase diagram of retrieval at equilibrium

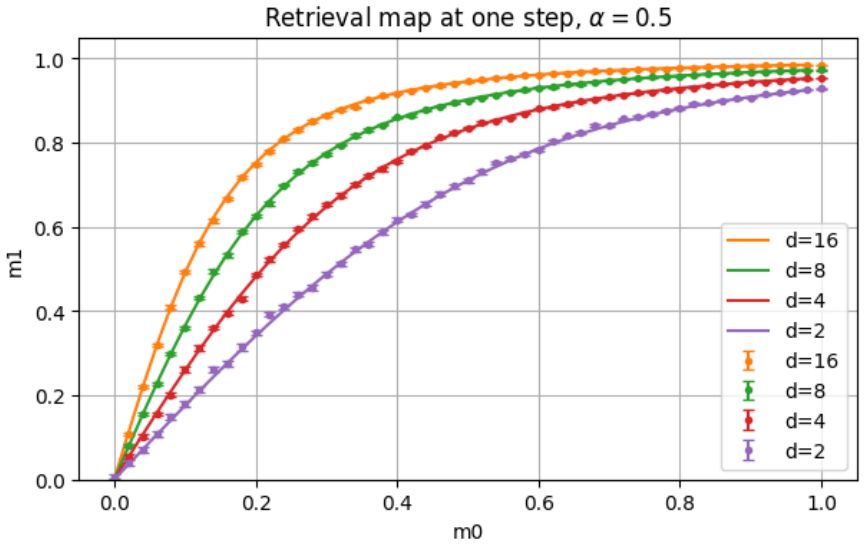
Two order parameters: $\alpha = \frac{P}{N}$, $\alpha'' = \frac{Pd}{N}$



Vector-spin associative memories

First step denoising





$$\Rightarrow P = \alpha' N d$$