Towards a Mathematical Understanding of Deep Convolutional Neural Networks

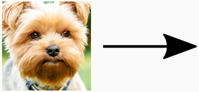
Florentin Guth

Learning from data

Image classification

“cat”

“dog”

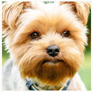


“dog”

Learning from data

Image classification

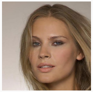
“cat” “dog”



→ “dog”

Image generation

→

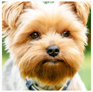


Learning from data

Image classification

“cat”

“dog”



→ “dog”

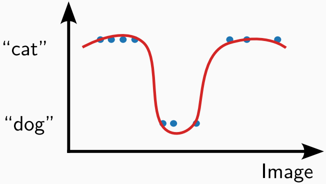
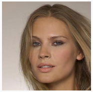
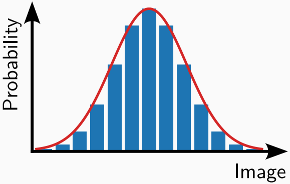


Image generation

→





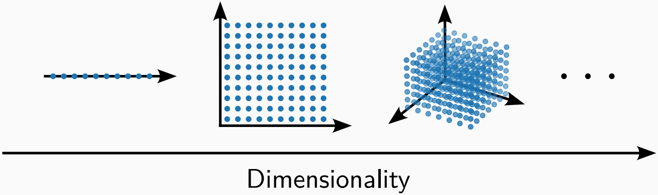
The curse of dimensionality

Images are high-dimensional: millions of degrees of freedom

The curse of dimensionality

Images are high-dimensional: millions of degrees of freedom

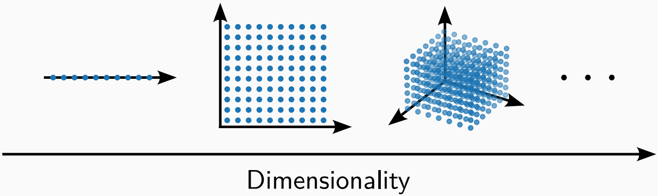
Curse of dimensionality: exponential number of possibilities



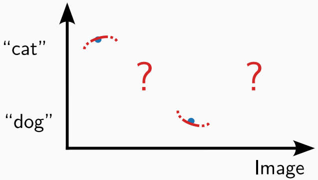
The curse of dimensionality

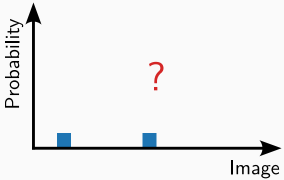
Images are high-dimensional: millions of degrees of freedom

Curse of dimensionality: exponential number of possibilities

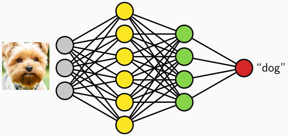


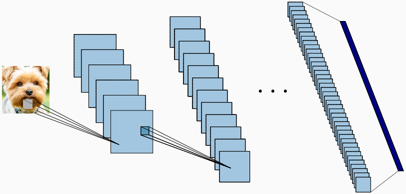
How to learn in high dimensions?



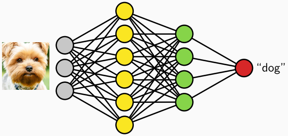


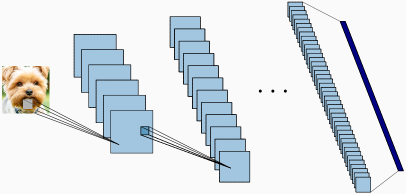
Enter deep learning





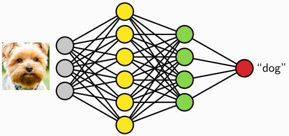
Enter deep learning

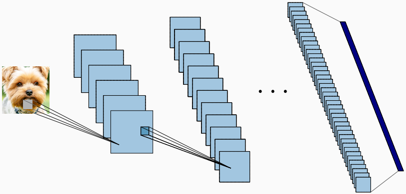




Training: initialize connections randomly, iterate over the examples, and adjust connections iteratively when making a mistake.

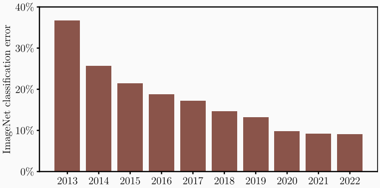
Enter deep learning





Training: initialize connections randomly, iterate over the examples, and adjust connections iteratively when making a mistake.

It works!

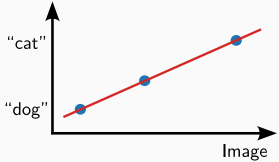


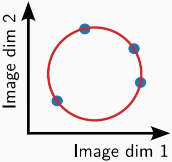
Searching for simplicity

The curse of dimensionality is a worst-case observation, for arbitrarily complicated data. The success of deep learning shows that our data is simple.

Searching for simplicity

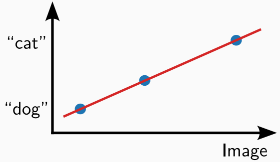
The curse of dimensionality is a worst-case observation, for arbitrarily complicated data. The success of deep learning shows that our data is simple.

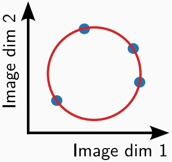




Searching for simplicity

The curse of dimensionality is a worst-case observation, for arbitrarily complicated data. The success of deep learning shows that our data is simple.

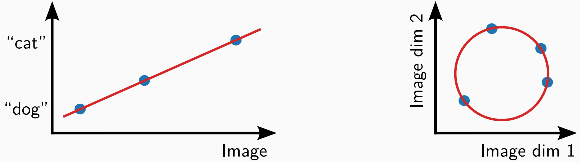




What do we mean by simple?

Searching for simplicity

The curse of dimensionality is a worst-case observation, for arbitrarily complicated data. The success of deep learning shows that our data is simple.



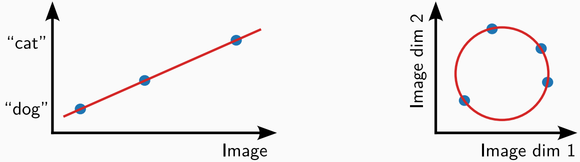
What do we mean by simple?

In this talk: search for mathematical structure

- In the data distribution: what are its properties?

Searching for simplicity

The curse of dimensionality is a worst-case observation, for arbitrarily complicated data. The success of deep learning shows that our data is simple.



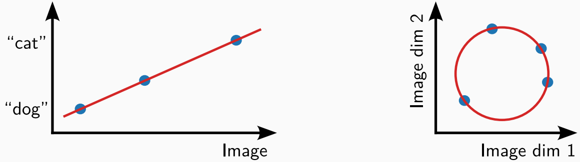
What do we mean by simple?

In this talk: search for mathematical structure

- In the data distribution: what are its properties?
- In the network computations: what are its functional blocks?

Searching for simplicity

The curse of dimensionality is a worst-case observation, for arbitrarily complicated data. The success of deep learning shows that our data is simple.



What do we mean by simple?

In this talk: search for mathematical structure

- In the data distribution: what are its properties?
- In the network computations: what are its functional blocks?
- In the network weights: what has been learned?

Outline

Exploiting Structure in Image Probability Distributions

Enforcing Structure in Convolutional Network Architectures

Discovering Structure in Learned Network Weights

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$
- Fit the parameters θ to the training samples $x_1, \ldots, x_n$

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$
- Fit the parameters θ to the training samples $x_1, \ldots, x_n$
- Generate samples from the model $p_\theta(x)$

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$
- Fit the parameters θ to the training samples $x_1, \ldots, x_n$
- Generate samples from the model $p_\theta(x)$

Each step introduces errors! What kind of assumptions allow controlling them in high-dimensions?

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$
- Fit the parameters θ to the training samples $x_1, \ldots, x_n$
- Generate samples from the model $p_\theta(x)$

Each step introduces errors! What kind of assumptions allow controlling them in high-dimensions?

- Small number of parameters

Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$
- Fit the parameters θ to the training samples $x_1, \ldots, x_n$
- Generate samples from the model $p_\theta(x)$

Each step introduces errors! What kind of assumptions allow controlling them in high-dimensions?

- Small number of parameters
- Log-concavity

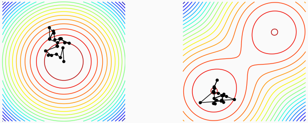
Generative modeling

We have samples $x_1, \ldots, x_n$ drawn independently from a probability distribution $p(x)$. Goal: generate new samples from $p(x)$

- Choose a parameterized family $\{p_\theta(x)\}$
- Fit the parameters θ to the training samples $x_1, \ldots, x_n$
- Generate samples from the model $p_\theta(x)$

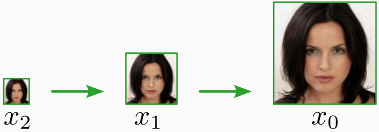
Each step introduces errors! What kind of assumptions allow controlling them in high-dimensions?

- Small number of parameters
- Log-concavity



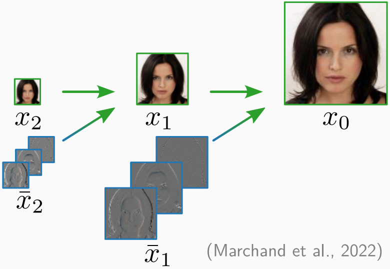
Multiscale generation

But we don't have to generate the image all at once! We can perform iterative generation:



Multiscale generation

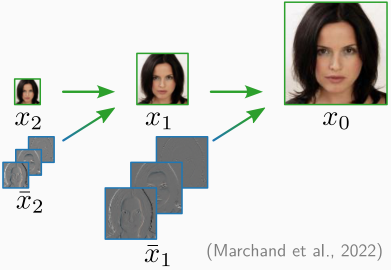
But we don't have to generate the image all at once! We can perform iterative generation:



(Marchand et al., 2022)

Multiscale generation

But we don't have to generate the image all at once! We can perform iterative generation:



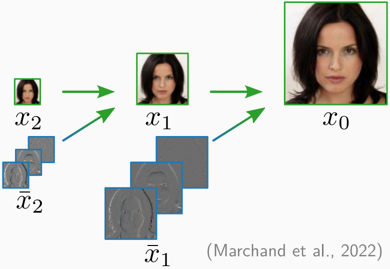
(Marchand et al., 2022)

Corresponds to a factorization of the probability distribution:

$$p(x_0) = p(x_J) \prod_{j=1}^{J} p(\bar{x}_j | x_j)$$

Multiscale generation

But we don't have to generate the image all at once! We can perform iterative generation:



(Marchand et al., 2022)

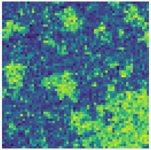
Corresponds to a factorization of the probability distribution:

$$p(x_0) = p(x_J) \prod_{j=1}^{J} p(\bar{x}_j | x_j)$$

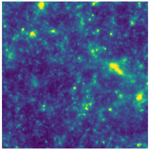
What are the properties of these *conditional* distributions?

Conditional locality

A “simpler” class of image distributions: physical fields



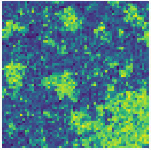
φ^4



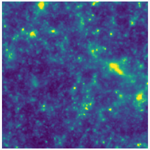
Weak lensing

Conditional locality

A "simpler" class of image distributions: physical fields



φ^4

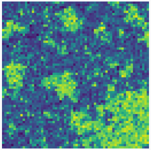


Weak lensing

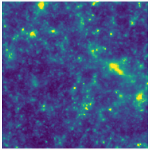
- The distribution can be written $p(x) = \frac{1}{Z} \mathrm{e}^{-E(x)}$ where $E(x)$ is an "energy" function

Conditional locality

A "simpler" class of image distributions: physical fields



φ^4

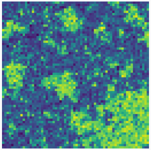


Weak lensing

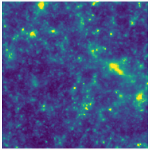
- The distribution can be written $p(x) = \frac{1}{Z}\mathrm{e}^{-E(x)}$ where $E(x)$ is an "energy" function
- If interactions are local, $E(x)$ decomposes as a sum of local potentials (Markov random field)

Conditional locality

A "simpler" class of image distributions: physical fields



φ^4

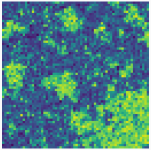


Weak lensing

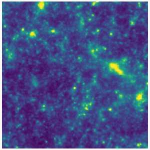
- The distribution can be written $p(x) = \frac{1}{Z} e^{-E(x)}$ where $E(x)$ is an "energy" function
- If interactions are local, $E(x)$ decomposes as a sum of local potentials (Markov random field)
- More generally, it is sufficient to have local *conditional* interactions *at each scale* (Marchand et al., 2022)

Conditional locality

A “simpler” class of image distributions: physical fields



φ^4



Weak lensing

- The distribution can be written $p(x) = \frac{1}{Z}\mathrm{e}^{-E(x)}$ where $E(x)$ is an “energy” function
- If interactions are local, $E(x)$ decomposes as a sum of local potentials (Markov random field)
- More generally, it is sufficient to have local *conditional* interactions *at each scale* (Marchand et al., 2022)
- $E(\bar{x}_j | x_j)$ then decomposes as a sum of local potentials (conditional Markov random field)

Conditional log-concavity

In addition, we show that $p(\bar{x}_j|x_j)$ is log-concave.

Conditional log-concavity

In addition, we show that $p(\bar{x}_j | x_j)$ is log-concave. Motivation:

$$E(x) = \underbrace{\frac{1}{2} x^{\mathrm{T}} K x}_{\text{kinetic}} + \underbrace{U(x)}_{\text{potential}}$$

Conditional log-concavity

In addition, we show that $p(\bar{x}_j | x_j)$ is log-concave. Motivation:

$$E(x) = \underbrace{\frac{1}{2} x^{\mathrm{T}} K x}_{\text{kinetic}} + \underbrace{U(x)}_{\text{potential}}$$

$$\nabla^2 E(x) = K + \nabla^2 U(x)$$

Conditional log-concavity

In addition, we show that $p(\bar{x}_j|x_j)$ is log-concave. Motivation:

$$E(x) = \underbrace{\frac{1}{2}x^{\mathrm{T}}Kx}_{\text{kinetic}} + \underbrace{U(x)}_{\text{potential}}$$

$$\nabla^2 E(x) = K + \nabla^2 U(x)$$

- Largest eigenvalues of K correspond to directions that are "more" log-concave

Conditional log-concavity

In addition, we show that $p(\bar{x}_j | x_j)$ is log-concave. Motivation:

$$E(x) = \underbrace{\frac{1}{2} x^{\mathrm{T}} K x}_{\text{kinetic}} + \underbrace{U(x)}_{\text{potential}}$$

$$\nabla^2 E(x) = K + \nabla^2 U(x)$$

- Largest eigenvalues of K correspond to directions that are "more" log-concave
- For multiscale stationary fields, these correspond to high-frequency details

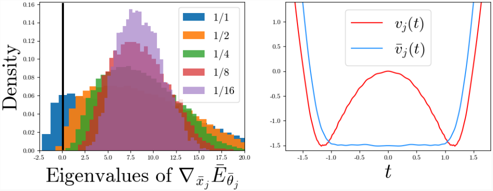
Conditional log-concavity

In addition, we show that $p(\bar{x}_j|x_j)$ is log-concave. Motivation:

$$E(x) = \underbrace{\frac{1}{2}x^{\mathrm{T}}Kx}_{\text{kinetic}} + \underbrace{U(x)}_{\text{potential}}$$

$$\nabla^2 E(x) = K + \nabla^2 U(x)$$

- Largest eigenvalues of K correspond to directions that are "more" log-concave
- For multiscale stationary fields, these correspond to high-frequency details



Conditionally log-concave models

For physical fields,

G*, Lempereur*, Bruna, and Mallat. Conditionally strongly log-concave generative models. *ICML*, 2023.

Conditionally log-concave models

For physical fields,

- the conditional distributions $p(\bar{x}_j|x_j)$ are local and log-concave

G*, Lempereur*, Bruna, and Mallat. Conditionally strongly log-concave generative models. *ICML*, 2023.

Conditionally log-concave models

For physical fields,

- the conditional distributions $p(\bar{x}_j|x_j)$ are local and log-concave
- theoretical bounds on estimation and generation errors

G*, Lempereur*, Bruna, and Mallat. Conditionally strongly log-concave generative models. *ICML*, 2023.

Conditionally log-concave models

For physical fields,

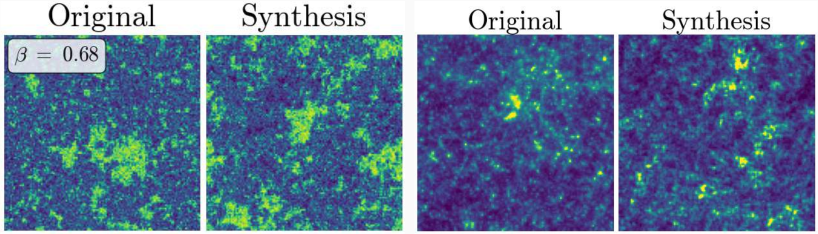
- the conditional distributions $p(\bar{x}_j|x_j)$ are local and log-concave
- theoretical bounds on estimation and generation errors
- log-concavity enables efficient parameter estimation with score matching

G*, Lempereur*, Bruna, and Mallat. Conditionally strongly log-concave generative models. *ICML*, 2023.

Conditionally log-concave models

For physical fields,

- the conditional distributions $p(\bar{x}_j|x_j)$ are local and log-concave
- theoretical bounds on estimation and generation errors
- log-concavity enables efficient parameter estimation with score matching



G*, Lempereur*, Bruna, and Mallat. Conditionally strongly log-concave generative models. *ICML*, 2023.

Score-based diffusion models

What about more complex image distributions? Not expected to be conditionally log-concave.

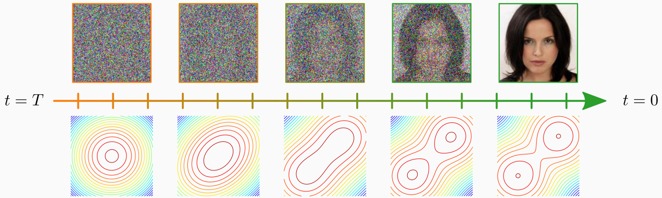
Score-based diffusion models

What about more complex image distributions? Not expected to be conditionally log-concave.

Enter diffusion models:

$$x_{t+\mathrm{d}t} \,|\, x_t \sim \mathcal{N}(x_t, \mathrm{d}t\,\mathrm{Id})$$

$$x_{t-\mathrm{d}t} \,|\, x_t \sim \mathcal{N}(x_t + \mathrm{d}t\,\nabla \log p_t(x_t), \mathrm{d}t\,\mathrm{Id})$$

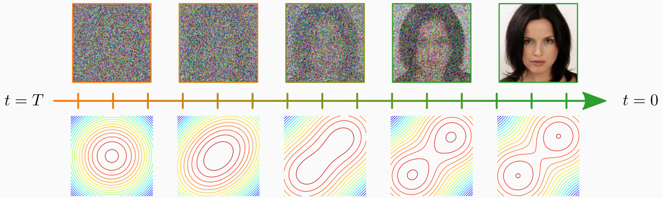


Score-based diffusion models

What about more complex image distributions? Not expected to be conditionally log-concave.

Enter diffusion models:

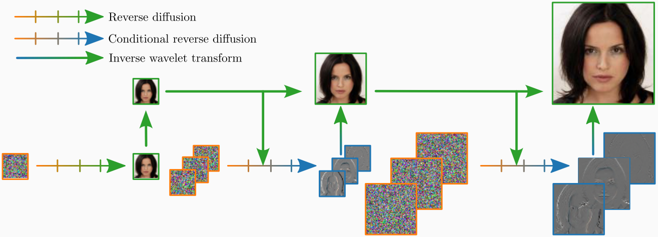
$$x_{t+\mathrm{d}t} \,|\, x_t \sim \mathcal{N}(x_t, \mathrm{d}t\,\mathrm{Id})$$
$$x_{t-\mathrm{d}t} \,|\, x_t \sim \mathcal{N}(x_t + \mathrm{d}t\,\nabla \log p_t(x_t), \mathrm{d}t\,\mathrm{Id})$$



Diffusion models solve the issues associated with non-log-concavity (Song et al., 2021; Chen et al., 2022). Remaining burning question: how do deep networks learn the score?

Conditionally local diffusion models

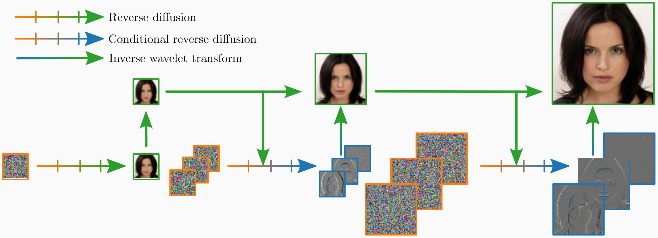
Benefits of combining diffusion models with multiscale approaches?



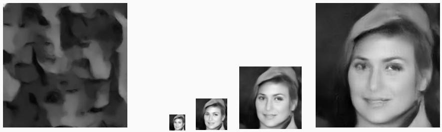
G, Coste, De Bortoli, and Mallat. Wavelet score-based generative modeling. *NeurIPS*, 2022.
Kadkhodaie, G, Mallat, and Simoncelli. Learning multi-scale local conditional probability models of images. *ICLR*, 2023.

Conditionally local diffusion models

Benefits of combining diffusion models with multiscale approaches?



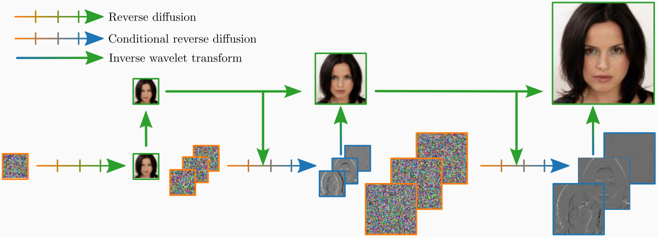
Locality:



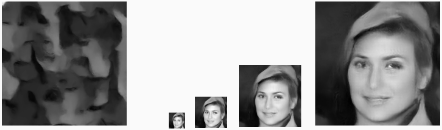
G, Coste, De Bortoli, and Mallat. Wavelet score-based generative modeling. *NeurIPS*, 2022.
Kadkhodaie, G, Mallat, and Simoncelli. Learning multi-scale local conditional probability models of images. *ICLR*, 2023.

Conditionally local diffusion models

Benefits of combining diffusion models with multiscale approaches?



Locality:

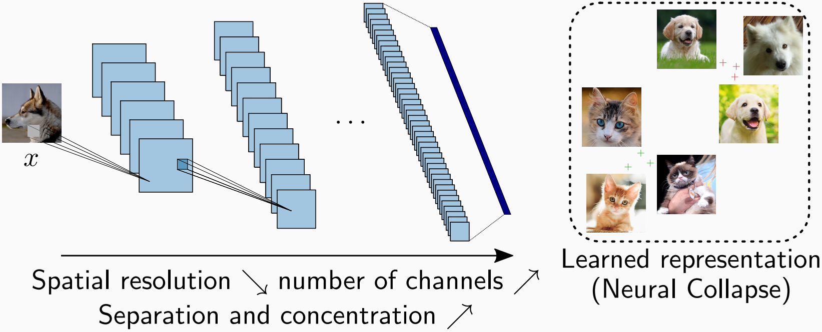


Sampling efficiency:

G, Coste, De Bortoli, and Mallat. Wavelet score-based generative modeling. *NeurIPS*, 2022.
Kadkhodaie, G, Mallat, and Simoncelli. Learning multi-scale local conditional probability models of images. *ICLR*, 2023.

Outline

Neural collapse

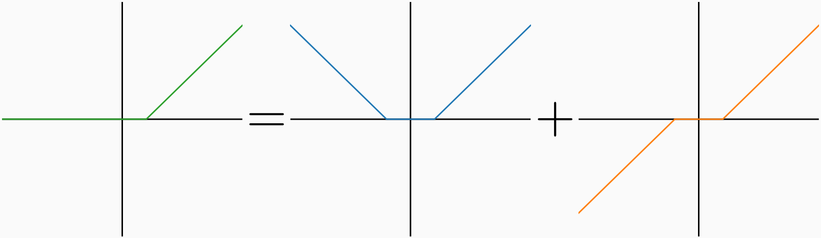


CNN classifiers simultaneously move spatial information into channels and increase linear separation

Can we define a non-linear operator with these properties?

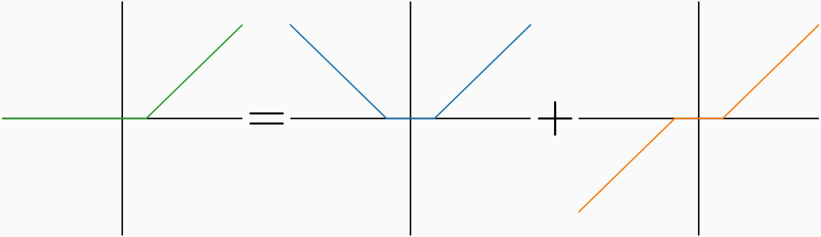
Decomposition of ReLU

ReLUs can can be separated in two opposite non-linearities with an even-odd decomposition:



Decomposition of ReLU

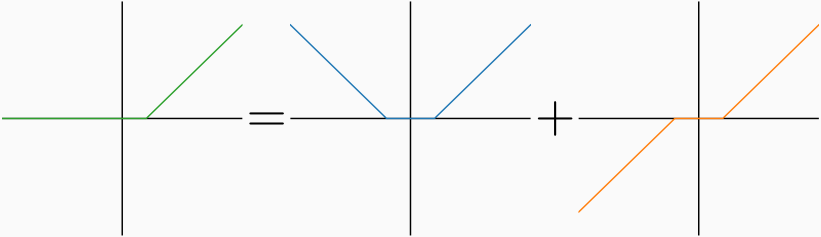
ReLUs can can be separated in two opposite non-linearities with an even-odd decomposition:



- Absolute value: collapses the sign, preserves the amplitude

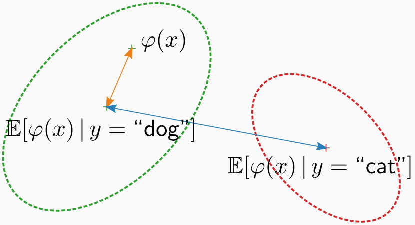
Decomposition of ReLU

ReLUs can can be separated in two opposite non-linearities with an even-odd decomposition:

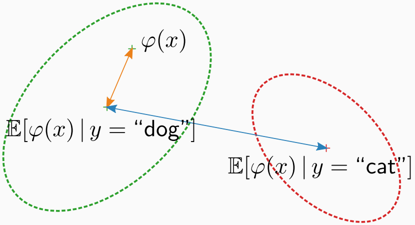


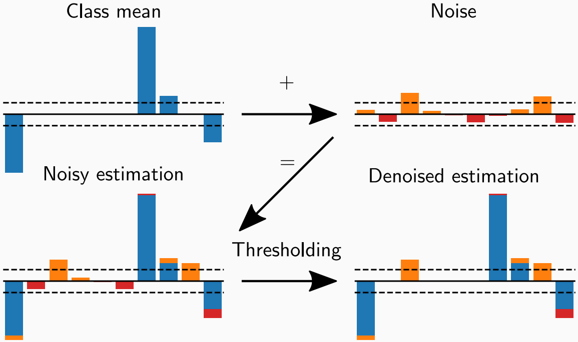
- Absolute value: collapses the sign, preserves the amplitude
- Soft-thresholding: preserves the sign, thresholds the amplitude

Concentration with soft-thresholding



Concentration with soft-thresholding





Separation with phase collapse

- Images have group variability: x and $g \cdot x$ have the same class

Separation with phase collapse

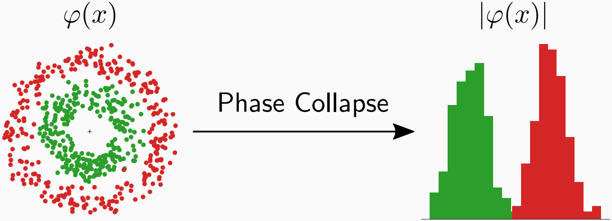
- Images have group variability: x and $g \cdot x$ have the same class
- Diagonalization of the group action: $\varphi(g \cdot x) = \mathrm{e}^{i\alpha(g)}\varphi(x)$

Separation with phase collapse

- Images have group variability: x and $g \cdot x$ have the same class
- Diagonalization of the group action: $\varphi(g \cdot x) = \mathrm{e}^{i\alpha(g)}\varphi(x)$
- The group within-class variability is a variability in the phases of the representation

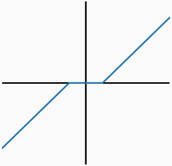
Separation with phase collapse

- Images have group variability: x and $g \cdot x$ have the same class
- Diagonalization of the group action: $\varphi(g \cdot x) = \mathrm{e}^{i\alpha(g)} \varphi(x)$
- The group within-class variability is a variability in the phases of the representation

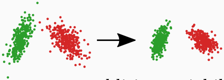


Comparison between sparsity and phase collapse

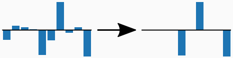
Concentration with soft-thresholding



Odd part of ReLU
Collapses small amplitudes

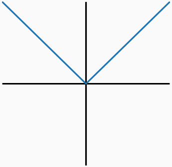


Concentrates additive variability
Does not separate class means

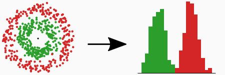


Performs denoising
Cannot be further sparsified

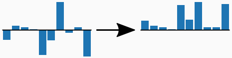
Separation with complex modulus



Even part of ReLU
Collapses complex phases

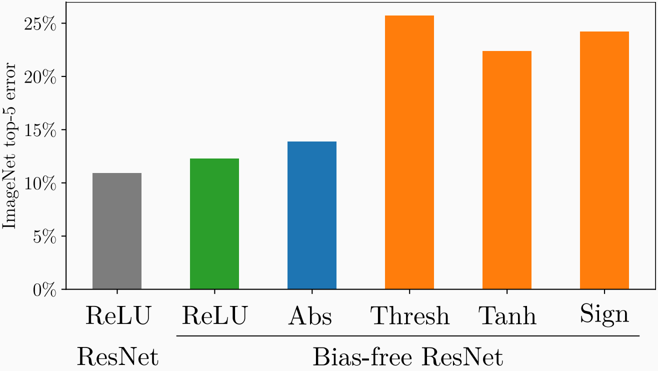


Concentrates multiplicative variability
Separates class means

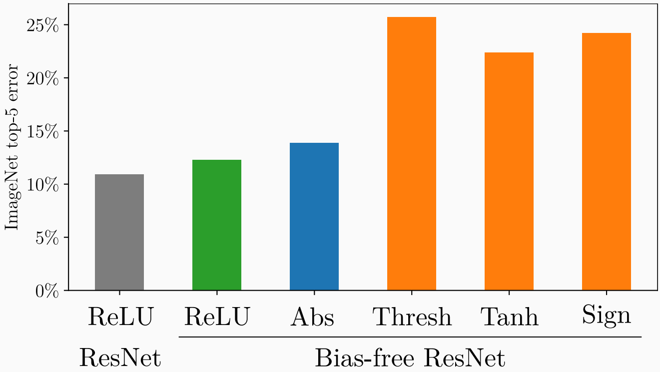


Computes support
Can be further sparsified

Phase collapse versus sparsity: numerical results



Phase collapse versus sparsity: numerical results



Phase collapse is sufficient to achieve good performance, while any non-linearity which preserves the phase is not. Phase collapse is thus also necessary.

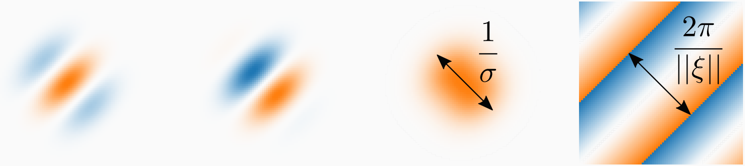
How far can we further constrain the network?

Diagonalizing local translations

Known source of within-class variability: local translations

Diagonalizing local translations

Known source of within-class variability: local translations



$\mathrm{Re}(\psi)$ $\mathrm{Im}(\psi)$ $|\psi|$ $\alpha(\psi)$

Small translations τ of an image x become phase shifts:

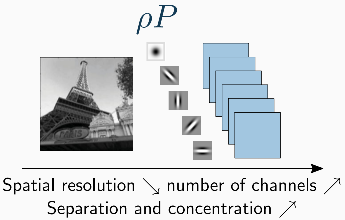
$$(\tau \cdot x) * \psi \approx e^{-i\xi \cdot \tau}(x * \psi)$$

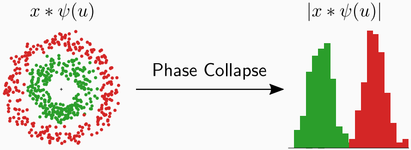
with a relative error bounded by $\sigma|\tau|$: approximate diagonalization!

The phase collapse operator

Constrain the spatial filters with the phase collapse operator:

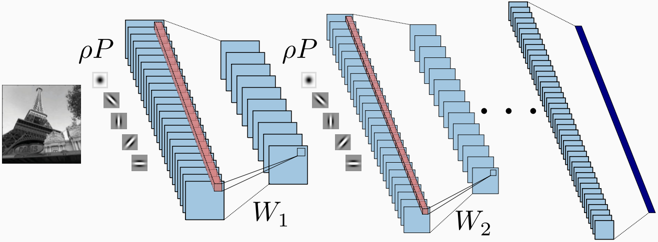
$$\rho P x(u) = \Big(x * \phi(2u), \left(|x * \psi_\theta(2u)|\right)_\theta \Big)$$





- Mathematical definition: no learning
- Combines linear and non-linear invariants to local translations
- All the desired properties!
- What accuracy can we achieve with this?

Learned scattering network



- Simplified architecture with phase collapses and minimal learning
- No learned spatial filters nor biases
- Only one learned component: channel matrices at every layer
- Reaches ResNet-18 accuracy with only 11 layers

Zarka, G, and Mallat. Separation and concentration in deep networks. *ICLR*, 2021.

G, Zarka, and Mallat. Phase collapse in neural networks. *ICLR*, 2022.

Outline

What has the network learned?

What has the network learned?

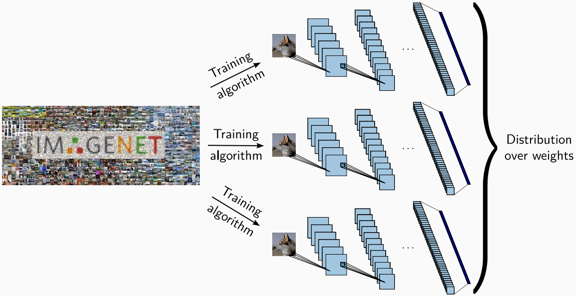
- No unique parameterization of a network due to internal symmetries

What has the network learned?

- No unique parameterization of a network due to internal symmetries
- Breaking this symmetry requires randomness

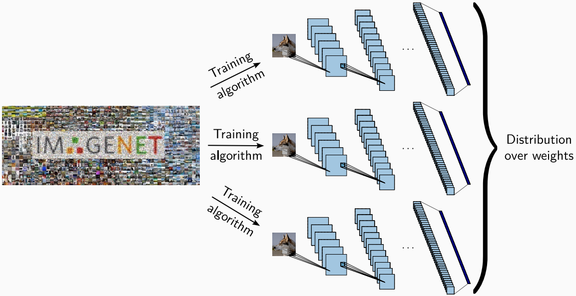
What has the network learned?

- No unique parameterization of a network due to internal symmetries
- Breaking this symmetry requires randomness



What has the network learned?

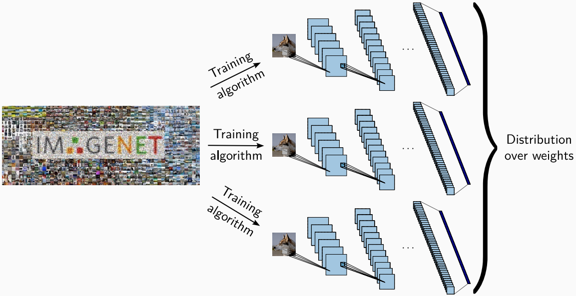
- No unique parameterization of a network due to internal symmetries
- Breaking this symmetry requires randomness



What is the distribution of trained network weights?

What has the network learned?

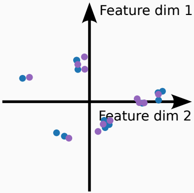
- No unique parameterization of a network due to internal symmetries
- Breaking this symmetry requires randomness



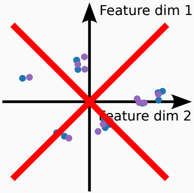
What is the distribution of trained network weights?

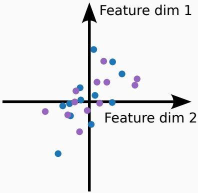
- Many parameters: laws of large numbers

Law of large numbers 1: weight statistics

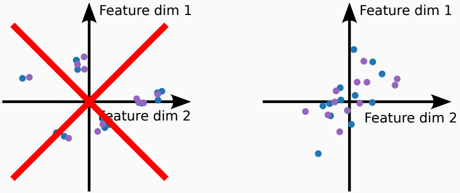


Law of large numbers 1: weight statistics

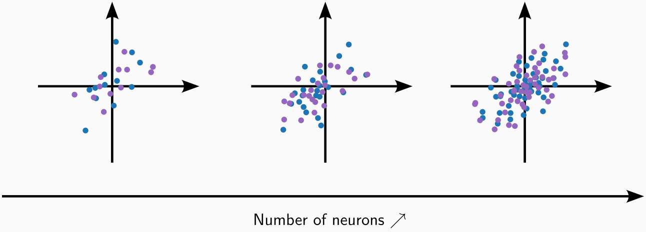




Law of large numbers 1: weight statistics



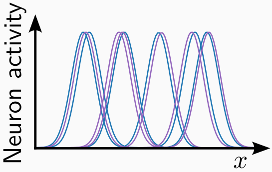
First law of large numbers: statistics of the neuron weights



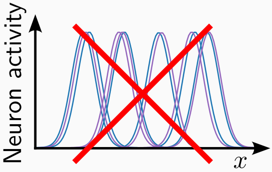
Mean-field (infinite-width) limit of neural networks

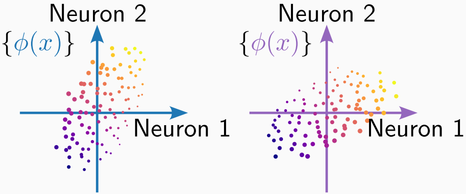
(Chizat and Bach, 2018; Mei et al., 2018; Rotskoff and Vanden-Eijnden, 2018; Sirignano and Spiliopoulos, 2020)

Law of large numbers 2: representation geometry

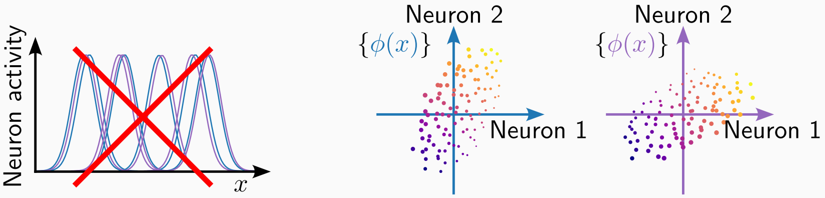


Law of large numbers 2: representation geometry





Law of large numbers 2: representation geometry

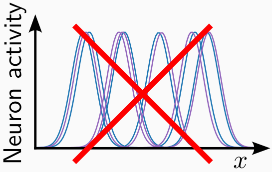


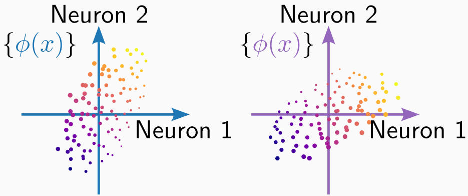
Second law of large numbers: geometry of the representation

(Rahimi and Recht, 2007)

$$\langle \phi(x), \phi(x') \rangle = \frac{1}{n} \sum_{i=1}^{n} \rho(\langle w_i, x \rangle)\, \rho(\langle w_i, x' \rangle) \to \mathbb{E}_{w \sim \pi}\Big[\rho(\langle w, x \rangle)\, \rho(\langle w, x' \rangle)\Big]$$

Law of large numbers 2: representation geometry





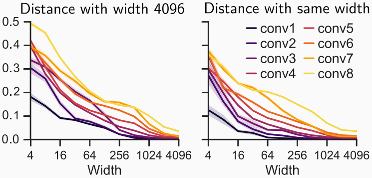
Second law of large numbers: geometry of the representation

(Rahimi and Recht, 2007)

$$\langle \phi(x), \phi(x') \rangle = \frac{1}{n} \sum_{i=1}^{n} \rho(\langle w_i, x \rangle)\, \rho(\langle w_i, x' \rangle) \rightarrow \mathbb{E}_{w \sim \pi}\Big[\rho(\langle w, x \rangle)\, \rho(\langle w, x' \rangle)\Big]$$

$$\mathbb{E}_{x,x'}\Big[\big(\langle \phi(x), \phi(x') \rangle - \langle \phi(x), \phi(x') \rangle\big)^2\Big]$$

(Kornblith et al., 2019)

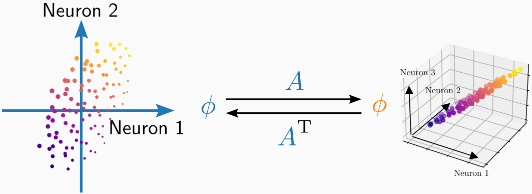


Network alignment

$$\langle \phi(x), \phi(x') \rangle$$
$$\to \langle \phi(x), \phi(x') \rangle$$

Network alignment

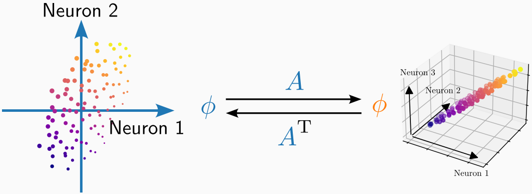
$$\langle \phi(x), \phi(x') \rangle$$
$$\to \langle \phi(x), \phi(x') \rangle$$



Network alignment

$\langle \phi(x), \phi(x') \rangle$

$\to \langle \phi(x), \phi(x') \rangle$

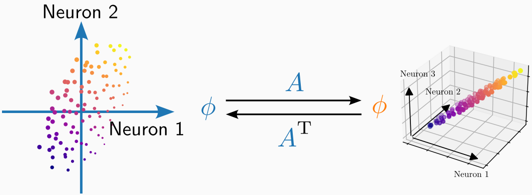


Define the alignment A with $\min_{A^{\mathrm{T}}A=\mathrm{Id}} \; \mathbb{E}_x\left[\|A\,\phi(x) - \phi(x)\|^2\right]$

Network alignment

$\langle \phi(x), \phi(x') \rangle$

$\to \langle \phi(x), \phi(x') \rangle$

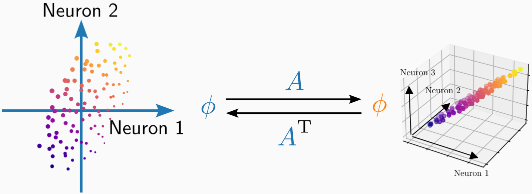


Define the alignment A with $\min_{A^{\mathrm{T}}A=\mathrm{Id}} \mathbb{E}_x\left[\|A\,\phi(x) - \phi(x)\|^2\right]$

We have $A = UV^{\mathrm{T}}$ from the SVD of $\mathbb{E}_x\left[\phi(x)\,\phi(x)^{\mathrm{T}}\right] = USV^{\mathrm{T}}$.

Network alignment

$$\langle \phi(x), \phi(x') \rangle \to \langle \phi(x), \phi(x') \rangle$$



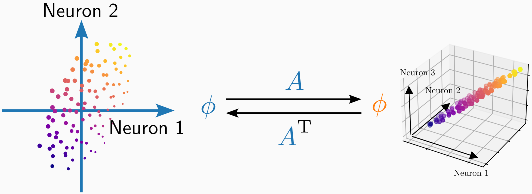
Define the alignment A with $\min_{A^{\mathrm{T}}A=\mathrm{Id}} \mathbb{E}_x\left[\|A\,\phi(x) - \phi(x)\|^2\right]$

We have $A = UV^{\mathrm{T}}$ from the SVD of $\mathbb{E}_x\left[\phi(x)\,\phi(x)^{\mathrm{T}}\right] = USV^{\mathrm{T}}$.

Theorem: *If neuron weights are i.i.d. samples from π, then $A\,\phi \to \phi$ in mean square, polynomially in the width, and independently of the dimension.*

Network alignment

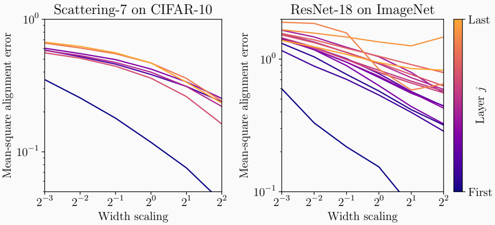
$\langle \phi(x), \phi(x') \rangle$
$\to \langle \phi(x), \phi(x') \rangle$



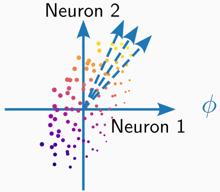
Define the alignment A with $\min_{A^{\mathrm{T}}A=\mathrm{Id}} \ \mathbb{E}_x\left[\left\| A\,\phi(x) - \phi(x) \right\|^2\right]$

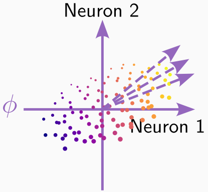
We have $A = UV^{\mathrm{T}}$ from the SVD of $\mathbb{E}_x\left[\phi(x)\,\phi(x)^{\mathrm{T}}\right] = USV^{\mathrm{T}}$.

Theorem: *If neuron weights are i.i.d. samples from π, then $A\,\phi \to \phi$ in mean square, polynomially in the width, and independently of the dimension.*

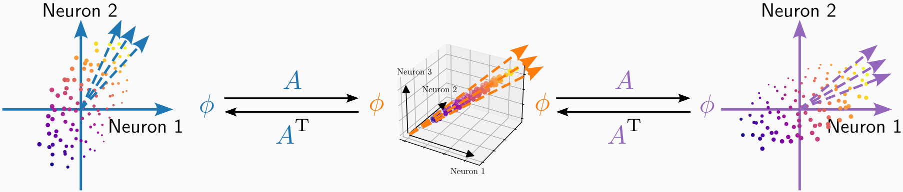


Mean-field limit in hidden layers



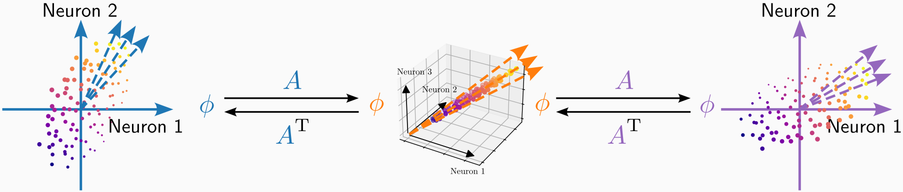


Mean-field limit in hidden layers



$$\langle w, \phi(x) \rangle = \langle Aw, A\phi(x) \rangle \approx \langle Aw, \phi(x) \rangle$$

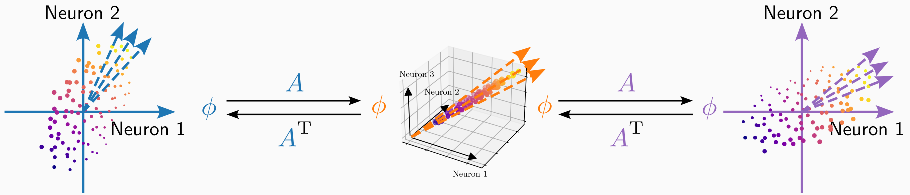
Mean-field limit in hidden layers



$$\langle w, \phi(x) \rangle = \langle Aw, A\phi(x) \rangle \approx \langle Aw, \phi(x) \rangle$$

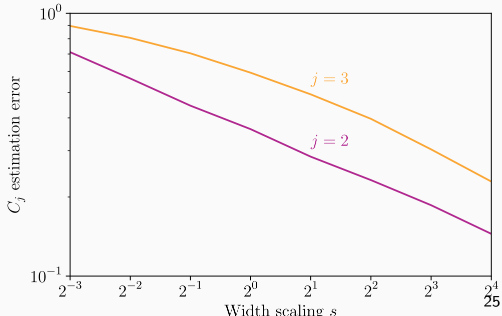
In deeper layers, we expect a mean-field limit on the **aligned** neuron weights $\{Aw\}$

Mean-field limit in hidden layers



$$\langle w, \phi(x) \rangle = \langle Aw, A\phi(x) \rangle \approx \langle Aw, \phi(x) \rangle$$

In deeper layers, we expect a mean-field limit on the **aligned** neuron weights $\{Aw\}$



A probabilistic model of network weights

"Maximum-entropy" model under the mean-field constraints at each layer

A probabilistic model of network weights

"Maximum-entropy" model under the mean-field constraints at each layer

Generative model of network weights

Neuron weights $w_{j,i} = A_{j-1}^{\mathrm{T}} w'_{j,i}$ with $w'_{j,i} \sim \pi_j$. Algorithm:

$$\text{Sample } W_1 \implies \text{Align } \phi_1 \text{ to } \phi_1 \implies \text{Sample } W_2 \implies \cdots$$

A probabilistic model of network weights

"Maximum-entropy" model under the mean-field constraints at each layer

Generative model of network weights

Neuron weights $w_{j,i} = A_{j-1}^{\mathrm{T}} w'_{j,i}$ with $w'_{j,i} \sim \pi_j$. Algorithm:

$$\text{Sample } W_1 \implies \text{Align } \phi_1 \text{ to } \phi_1 \implies \text{Sample } W_2 \implies \cdots$$

Theorem: $\forall j, A_j \, \phi_j \to \phi_j$
in mean square,
polynomially in the widths,
and independently of the dimension.

A probabilistic model of network weights

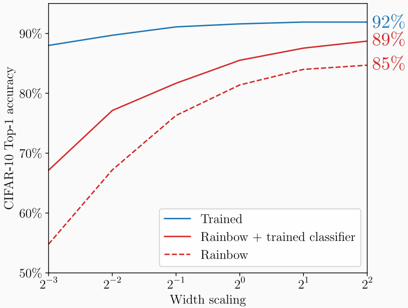
"Maximum-entropy" model under the mean-field constraints at each layer

Generative model of network weights

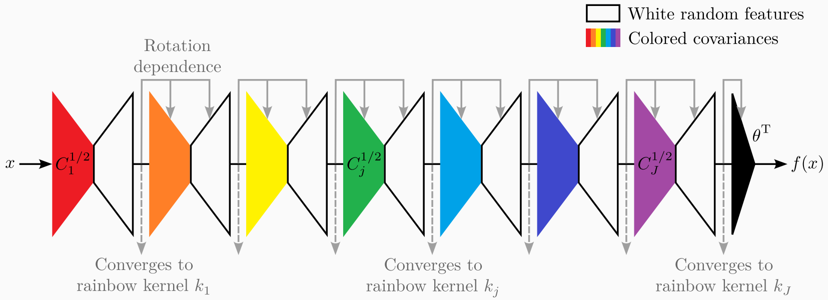
Neuron weights $w_{j,i} = A_{j-1}^{\mathrm{T}} w'_{j,i}$ with $w'_{j,i} \sim \pi_j$. Algorithm:

$$\text{Sample } W_1 \implies \text{Align } \phi_1 \text{ to } \phi_1 \implies \text{Sample } W_2 \implies \cdots$$

Theorem: $\forall j, A_j\, \phi_j \to \phi_j$ *in mean square, polynomially in the widths, and independently of the dimension.*



Covariance and dimensionality: the rainbow model



G, Ménard, Rochette, and Mallat. A rainbow in deep network black boxes. arXiv, 2023.

Conclusion

Conclusion

How can we explain the performance of deep learning?

Conclusion

How can we explain the performance of deep learning?

- A multiscale factorization of image distributions can reveal log-concavity or locality properties

Conclusion

How can we explain the performance of deep learning?

- A multiscale factorization of image distributions can reveal log-concavity or locality properties
- CNNs rely on phase collapses to separate image classes

Conclusion

How can we explain the performance of deep learning?

- A multiscale factorization of image distributions can reveal log-concavity or locality properties
- CNNs rely on phase collapses to separate image classes
- The trained weights compute colored random projections whose distribution is aligned to the input representation

Conclusion

How can we explain the performance of deep learning?

- A multiscale factorization of image distributions can reveal log-concavity or locality properties
- CNNs rely on phase collapses to separate image classes
- The trained weights compute colored random projections whose distribution is aligned to the input representation

Further research:

Conclusion

How can we explain the performance of deep learning?

- A multiscale factorization of image distributions can reveal log-concavity or locality properties
- CNNs rely on phase collapses to separate image classes
- The trained weights compute colored random projections whose distribution is aligned to the input representation

Further research:

- Why and how do score networks generalize?

Conclusion

How can we explain the performance of deep learning?

- A multiscale factorization of image distributions can reveal log-concavity or locality properties
- CNNs rely on phase collapses to separate image classes
- The trained weights compute colored random projections whose distribution is aligned to the input representation

Further research:

- Why and how do score networks generalize?
- How to understand the role of depth?

Thank you!