Locality defeats the curse of dimensionality in convolutional teacher-student scenarios

Alessandro Favero, Francesco Cagnetta, Matthieu Wyart

EPFL

NEURAL INFORMATION PROCESSING SYSTEMS

Learning in high dimensions

- **Supervised learning**: learn a target function $f^*(\boldsymbol{x})$ from P observations

$$\{(\boldsymbol{x}^\mu, y^\mu)\}_{\mu=1}^P$$

$$\boldsymbol{x}^\mu \in \mathbb{R}^d, \quad y^\mu = f^*(\boldsymbol{x}^\mu)$$

- **How many observations?** If one only assumes f^* is Lipschitz continuous, one needs $\mathcal{O}(\epsilon^{-d})$ observations to learn f^* up to error ϵ: **curse of dimensionality**

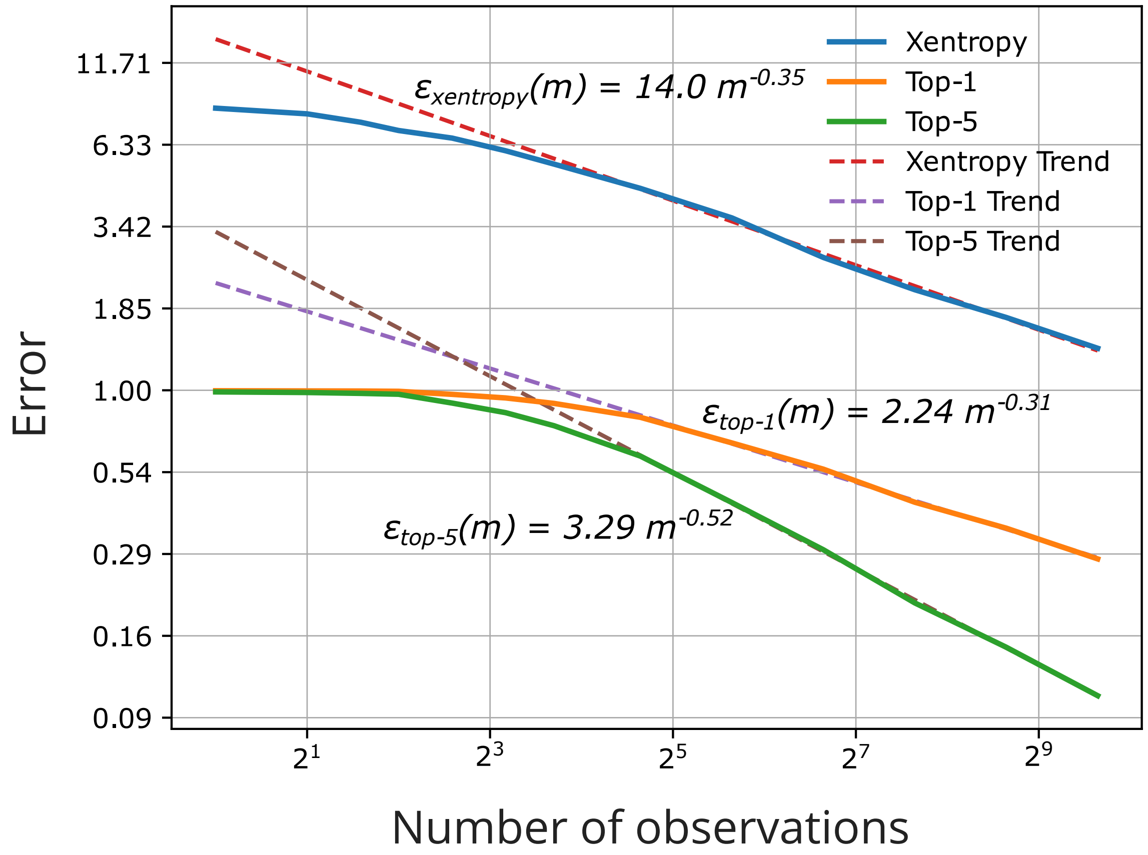
$$\epsilon = \mathcal{O}(P^{-1/d})$$

Learning seems impossibile!

Learning in high dimensions

- **How many observations in practice?** For ResNets on ImageNet ($d = 6.2 \times 10^4$)

  $\epsilon \sim P^{-0.3}$ [Hestness 1712.00409]



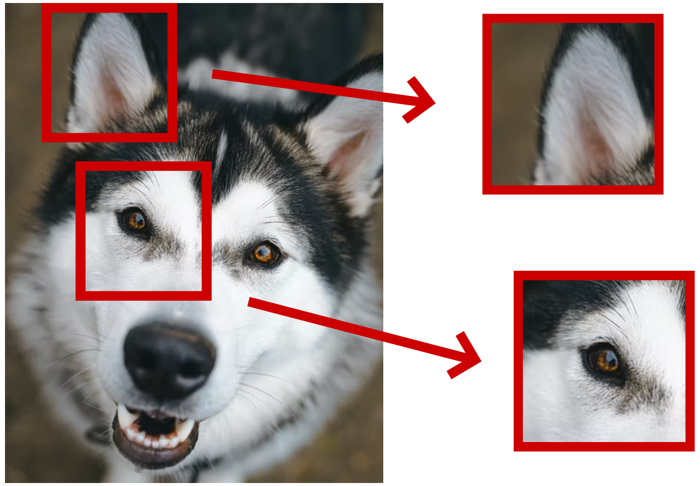
Images are physically structured

- If deep learning works in high dimensions, **data must be very structured**

- Several ideas:
  - Data live on a **manifold** $\mathcal{M}$ of lower dimensionality $d_{\mathcal{M}} \ll d$
  - Presence of **invariants**, as shift-invariance or deformation stability
  - The task is **local** and **compositional**

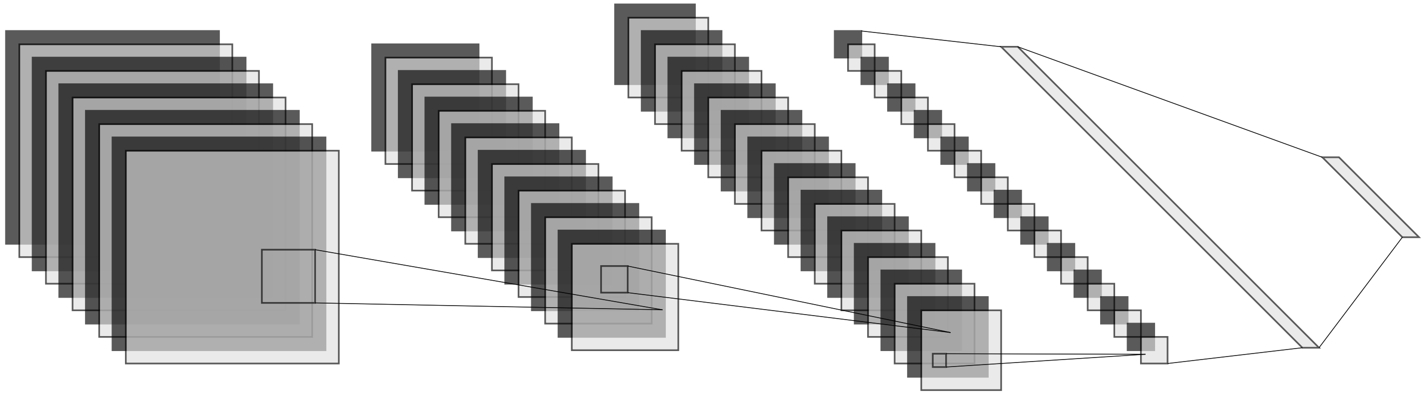
    [Poggio 1611.00740, 2006.13915] [Bietti 2102.10032]

Does a local compositional structure affect the learning curve?



Good architectures have good priors

- **Convolutional neural networks** have shared filter weights with local support



- Numerical experiments suggest that **local connectivity is key to performance** [Neyshabur 2007.13657]

Can we quantify the respective advantages of weight sharing and local connectivity?

Learning scenario: the teacher

- **Inputs** are d-dimensional random sequences

$$\boldsymbol{x} = (x_1, ..., \underbrace{x_i, ..., x_{i+t-1}}_{\boldsymbol{x}_i}, ..., x_d)$$

t-dimensional **patch**

- The **target function** is either
  - **local** $f^{*LC} = \sum_{i=1}^{d} g_i(\boldsymbol{x}_i)$, e.g. $f^{*LC}(x_1, x_2, x_3) = g_1(x_1, x_2) + g_2(x_2, x_3) + g_3(x_3, x_1)$
  - or **convolutional** $f^{*CN} = \sum_{i=1}^{d} g(\boldsymbol{x}_i)$

$g_i : \mathbb{R}^t \to \mathbb{R}$ is a Gaussian random function with controlled smoothness α_t

Learning scenario: the student

- Kernel method with a **local** or **convolutional** kernel with s-dimensional patches and smoothness α_s learns from P examples

$$K^{LC}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d} \sum_{i=1}^{d} C(\boldsymbol{x}_i, \boldsymbol{x}'_i)$$

Learning scenario: the student

- Kernel method with a **local** or **convolutional** kernel with s-dimensional patches and smoothness α_s learns from P examples

$$K^{LC}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d} \sum_{i=1}^{d} C(\boldsymbol{x}_i, \boldsymbol{x}'_i) \qquad K^{CN}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d^2} \sum_{i,j=1}^{d} C(\boldsymbol{x}_i, \boldsymbol{x}'_j)$$

Learning scenario: the student

- Kernel method with a **local** or **convolutional** kernel with s-dimensional patches and smoothness α_s learns from P examples

$$K^{LC}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d}\sum_{i=1}^{d} C(\boldsymbol{x}_i, \boldsymbol{x}'_i) \qquad K^{CN}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d^2}\sum_{i,j=1}^{d} C(\boldsymbol{x}_i, \boldsymbol{x}'_j)$$

- Including the kernels of simple CNNs as special cases! [Jacot 1806.07572]

- **Generalization error** $\epsilon = \mathbb{E}_{\boldsymbol{x}, f^*}[(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2] \sim P^{-\beta}$

Generalization in kernel regression

- **Mercer's theorem:** spectral decomposition $K(\boldsymbol{x}, \boldsymbol{x}') = \sum_\rho \lambda_\rho \phi_\rho(\boldsymbol{x}) \phi_\rho(\boldsymbol{x}')$

- We can expand f^* in the (student) kernel basis: $f^*(\boldsymbol{x}) = \sum_\rho c_\rho \phi_\rho(\boldsymbol{x})$

- From statistical physics, **kernel regression learns the first P projections**

  [Bordelon 2002.02561] [Spigler 1905.10843]

$$\epsilon(P) \sim \sum_{\rho > P} \mathbb{E}[c_\rho^{*2}]$$

Asymptotic learning curves

- K_T conv. with $\boldsymbol{t}$**-dimensional constituents** (filter size) and **smoothness** $\boldsymbol{\alpha_t}$
- K_S conv./loc. with $\boldsymbol{s}$**-dimensional constituents,** $\boldsymbol{s \geq t}$**,** and **smoothness** $\boldsymbol{\alpha_s}$ with $\alpha_s \geq \alpha_t/2 - s$

conv. student $\epsilon(P) \sim P^{-\alpha_t/s}$

loc. student $\epsilon(P) \sim \left(\frac{P}{d}\right)^{-\alpha_t/s}$

Asymptotic learning curves

- K_T conv. with $\boldsymbol{t}$**-dimensional constituents** (filter size) and **smoothness** $\boldsymbol{\alpha_t}$
- K_S conv./loc. with $\boldsymbol{s}$**-dimensional constituents,** $\boldsymbol{s \geq t}$**,** and **smoothness** $\boldsymbol{\alpha_s}$ with $\alpha_s \geq \alpha_t/2 - s$

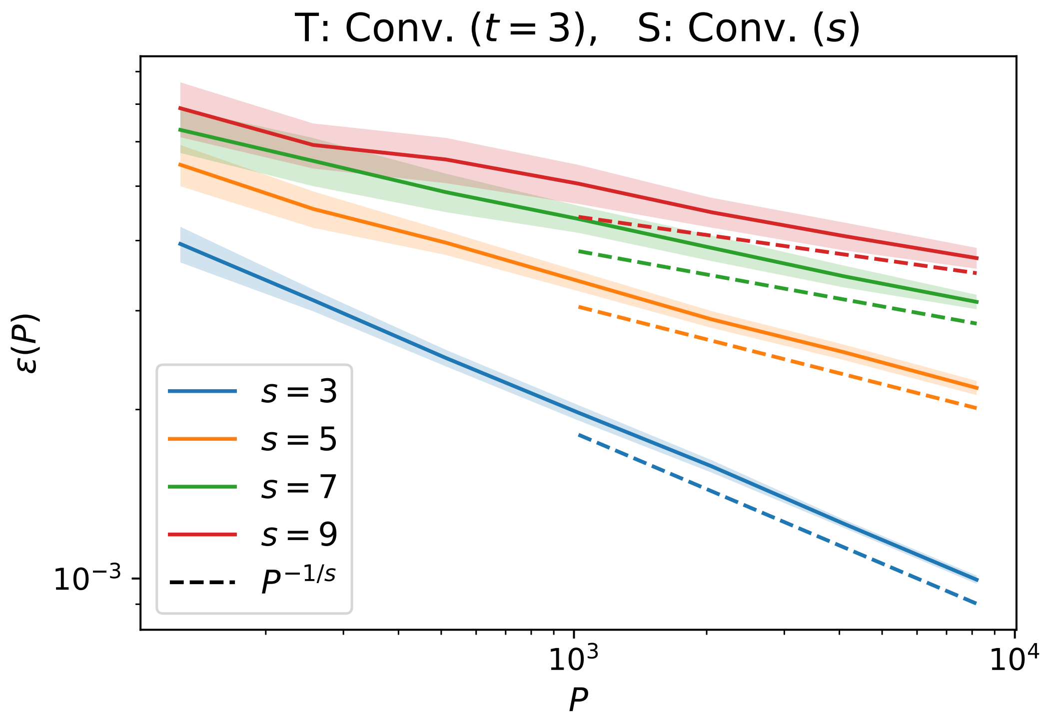
conv. student $\epsilon(P) \sim P^{-\alpha_t/s}$

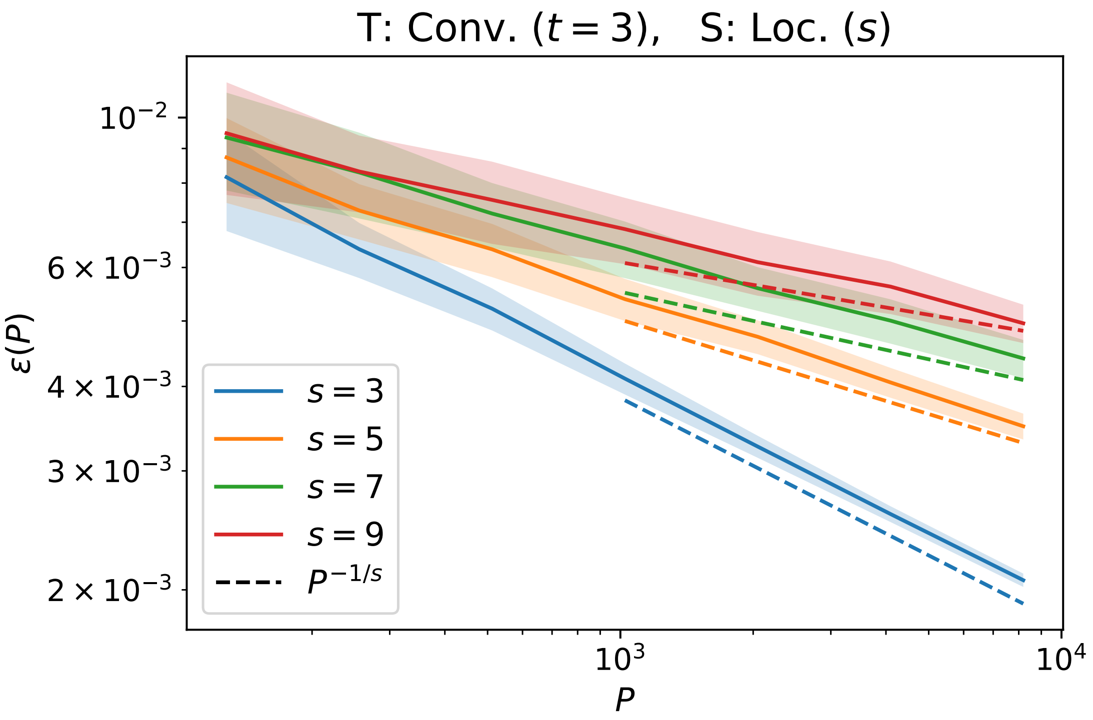
loc. student $\epsilon(P) \sim \left(\frac{P}{d}\right)^{-\alpha_t/s}$

- **The exponent is independent of d: no curse of dimensionality!**
- **Locality changes the error's decay**
- **Shift-invariance just affects the prefactor**

Asymptotic learning curves

- These predictions are **confirmed numerically** for several kernels and data distributions





Conclusions and perspectives

- **Local kernels beat the curse of dimensionality** when learning local functions

- This effect can be appreciated for **real data** also, e.g. regression on CIFAR-10

- What's missing? Exploring the **benefits of depth** by considering more complex compositional tasks as **hierarchical target** functions