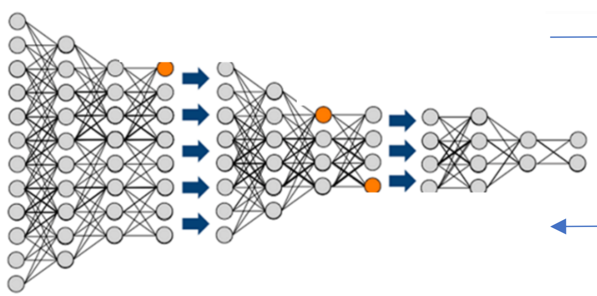
Loss landscape, over-parametrization and curse of dimensionality in deep learning

Matthieu Wyart

Collaborators *G. Biroli, F. Cagnetta, S. d'Ascoli, A. Favero, M. Geiger, C. Hongler, A. Jacot, J. Paccolat, L. Petrini , S. Spigler, L.Sagun, K. Tyloo, E. Vanden-Eijden*



EPFL SIMONS FOUNDATION FNSNF

Classifying data in large dimension

- Pre-requisite for Artificial Intelligence:

build algorithm that can make sense, *classify*, data in large dimension

- Example: computer vision. Is it a cat or a dog?

- Learn from examples (supervised learning)

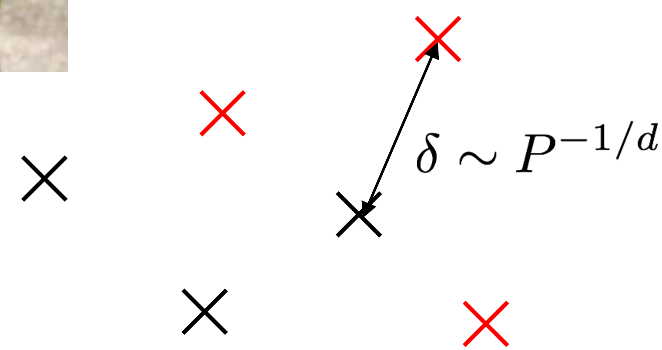
10^6 

10^6 

- Problem: ML algorithms should

not work, curse of dimensionality

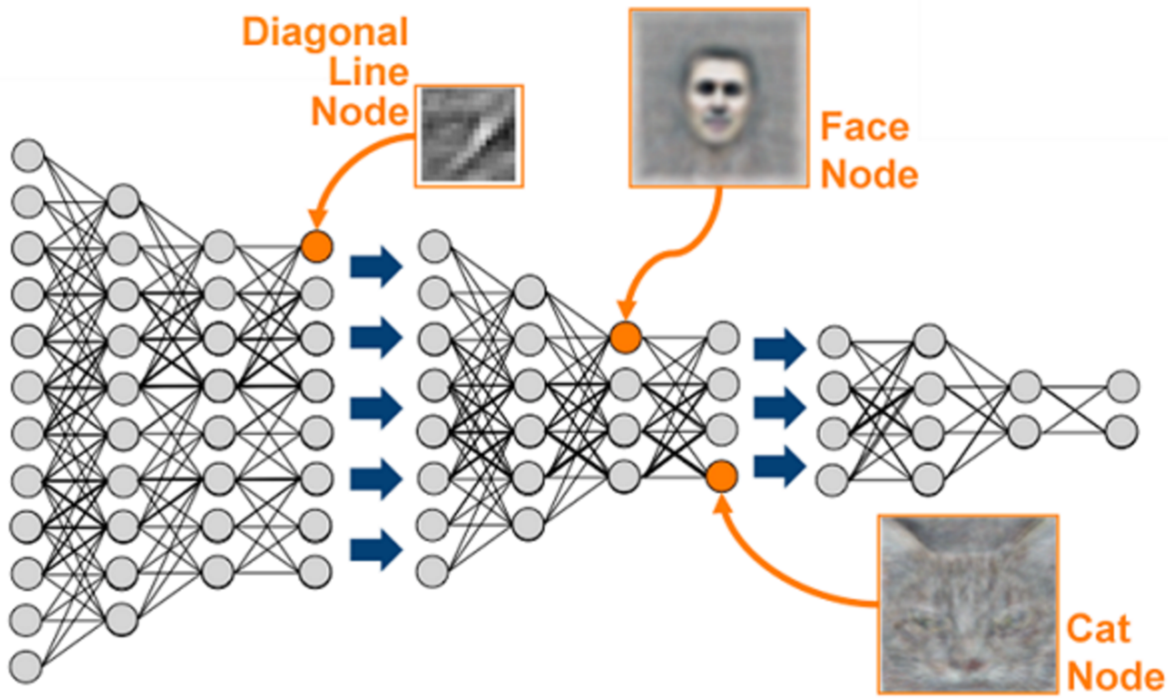
$\boxed{\epsilon_{test} \sim P^{-\beta}}$ $\boxed{\beta \sim \frac{1}{P}}$



P points in d dimensions

- Learnable data must be highly structured. *Structure?*

Benefits of learning a data representation?



- Neurons respond to features that are more and more abstract
- Hierarchy similar to our brain

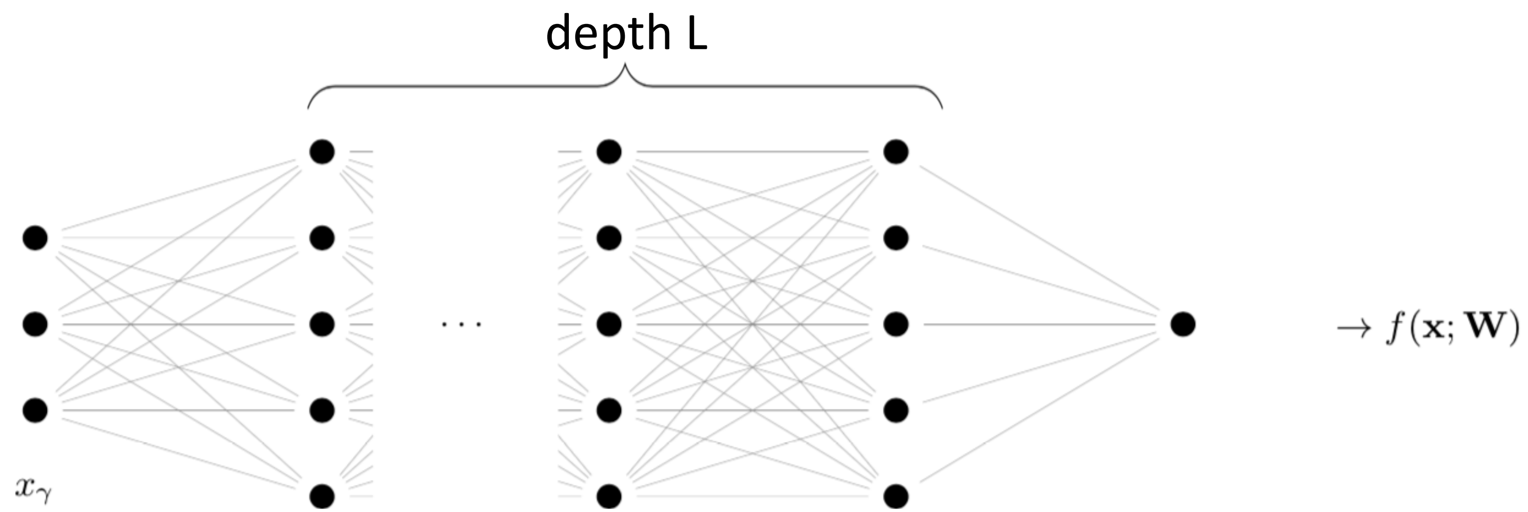
1/ Is it always learnt? Is it always beneficial?

2/ idea beneficial: reduces the dimension of the problem *Ansuini et al. 19'*

What information is lost in this representation?

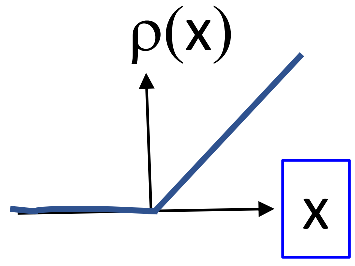
Set-up

- binary classification task, P training data $\{\mathbf{x}_i, y_i = \pm 1\}$
- Deep net $f_{\mathbf{W}}(\mathbf{x}_i)$ with N parameters, width h (N~h^{2}L)



$$\boxed{a_\beta = \rho\left(\sum_{\alpha \in \text{ previous layer}} W_{\alpha,\beta} a_\beta - B_\beta\right)}$$

ρ: non-linear function



Training

- Training: gradient descent in loss function

$$\boxed{\mathcal{L} = \frac{1}{P}\sum_{i=1}^{P} \ell(y_i f_{\mathbf{w}}(x_i))}$$

- Here: quadratic (or linear) hinge Loss:

$$l_i(f_{\mathbf{w}}(x_i)) = 0 \quad \text{if} \quad f_{\mathbf{w}}(x_i) y_i > 1$$

$$l_i(f_{\mathbf{w}}(x_i)) = (f_{\mathbf{w}}(x_i) y_i - 1)^2 \ \text{ if } \quad f_{\mathbf{w}}(x_i) y_i < 1$$

- *satisfability problem* $\mathcal{L} = 0 \Leftrightarrow f_{\mathbf{w}}(x_i) y_i > 1 \forall i$

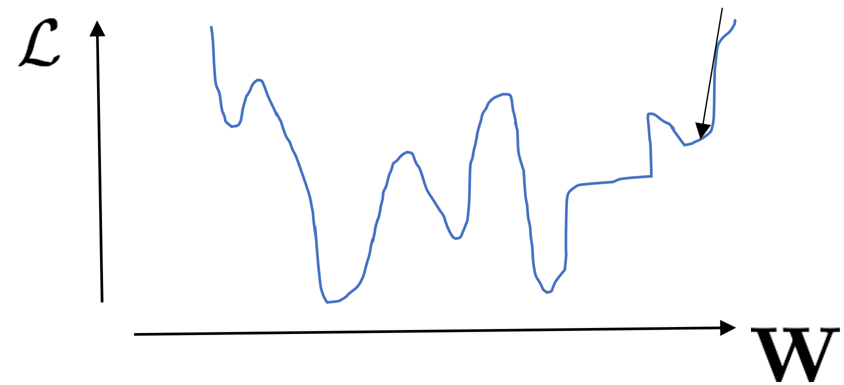
- train up to $\mathcal{L} = 0$ (no arbitrary stopping time)
- Essentially same performance as cross-entropy, most results holds in both cases

Geometry of Loss Landscape?

Biroli's lecture

- High dimensional, not convex

landscape.



<u>Questions:</u>

-why not stuck in bad local minima?

-Landscape geometry?

- Glassy landscape? Possibly when under-parametrized

Baity-Jesy et al. 18'

- Many flat directions if over-parametrized *Soudry, Hoffer 17' Sagun et al. 17' Cooper 18' Baity-Jesy et al. 18'*

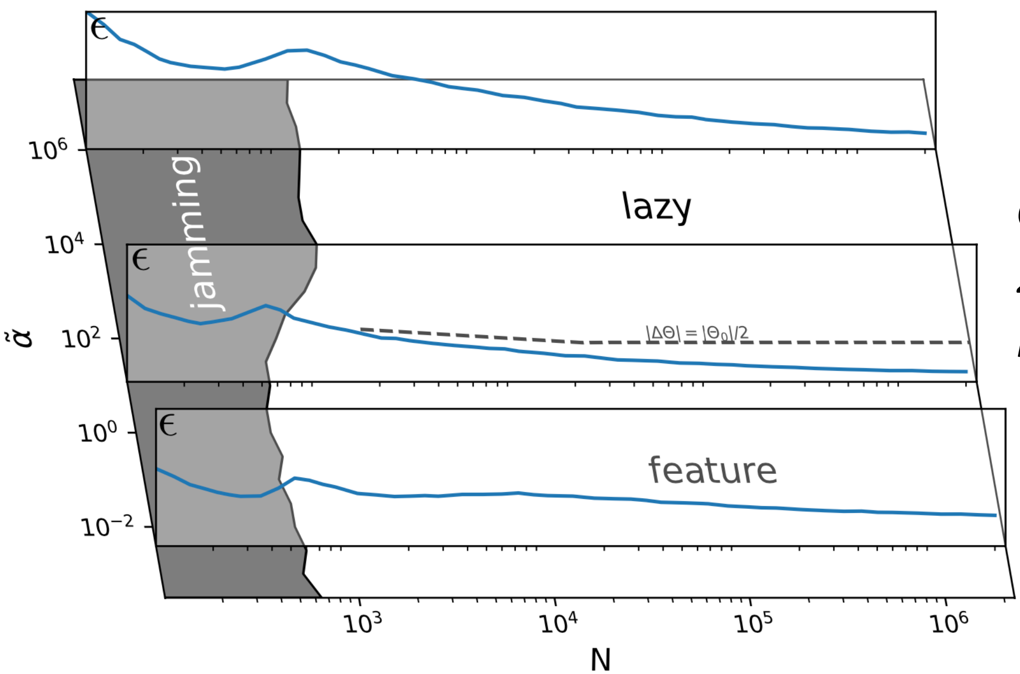
Transition in the landscape as N increases?

A phase diagram for deep learning

Geiger, Petrini, MW, Phys. Report (2021)

predictor $\tilde{\alpha}[f_{\mathbf{w}}(x) - f_{\mathbf{w_o}}(x)]$ *Chizat, Bach 19'*
Srebro, Montanari's lectures

Scale of initialization $\tilde{\alpha}$

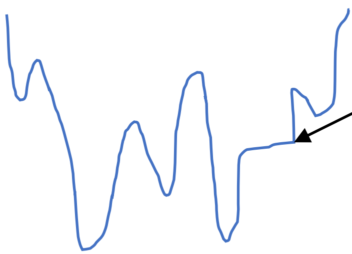


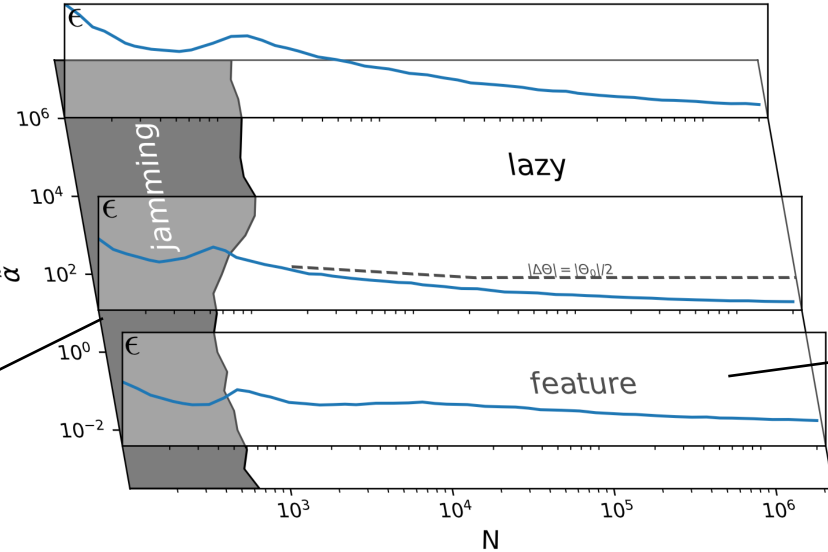
GD in Fully Connected, 2 hidden layers, MNIST

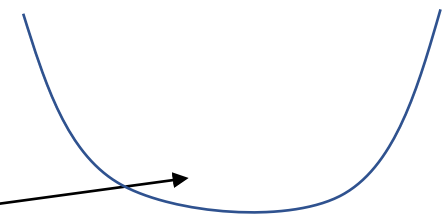
Number of parameters

A `jamming' transition as in sand

Geiger, Spigler et al., 2018

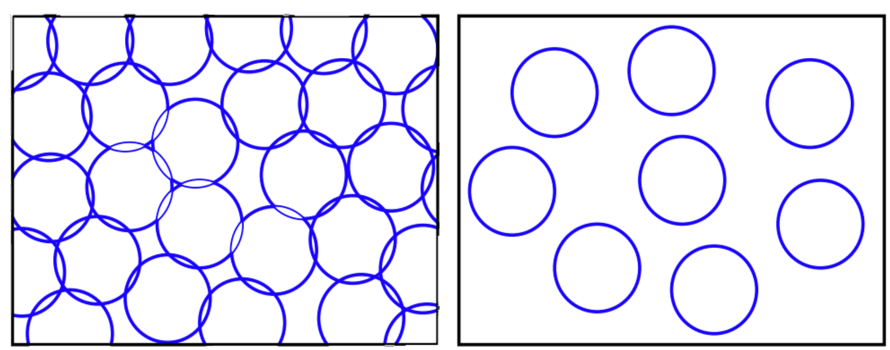






Can fit the data, reach bottom

Cannot fit the data



- Sharp phase

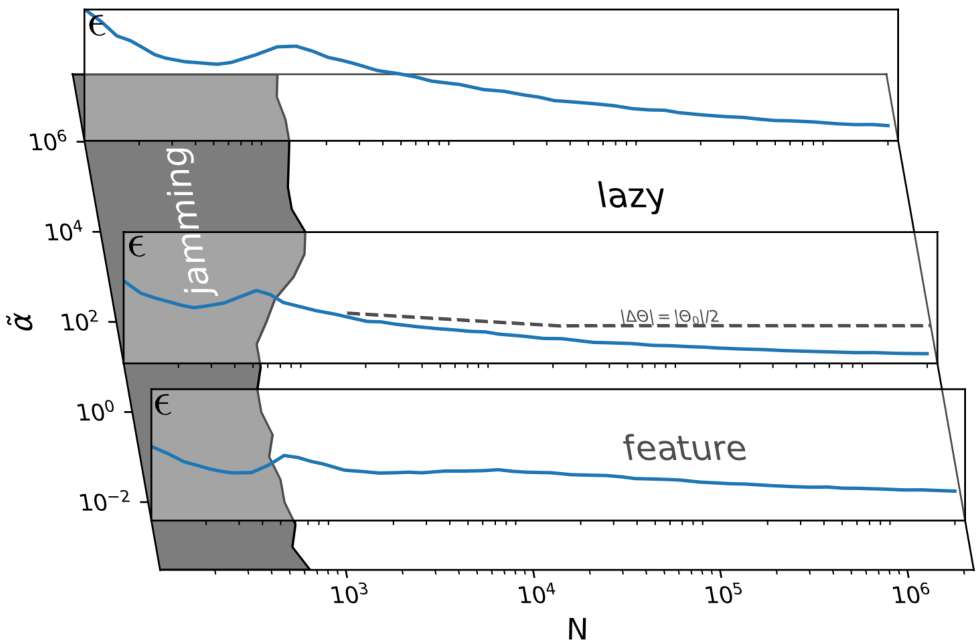
Transition, critical point (diverging predictor, critical slowing down, Hessian structure...)

- N>>P: data fitted

O'hern, Silbert, liu, nagel 03'
MW, Nagel, Witten 05'
Franz, Parisi 16'

Overfitting? Instead, a 'double descent'

Advani and Saxe 17',
Spigler et al, 18'
Belkin et al., 18'



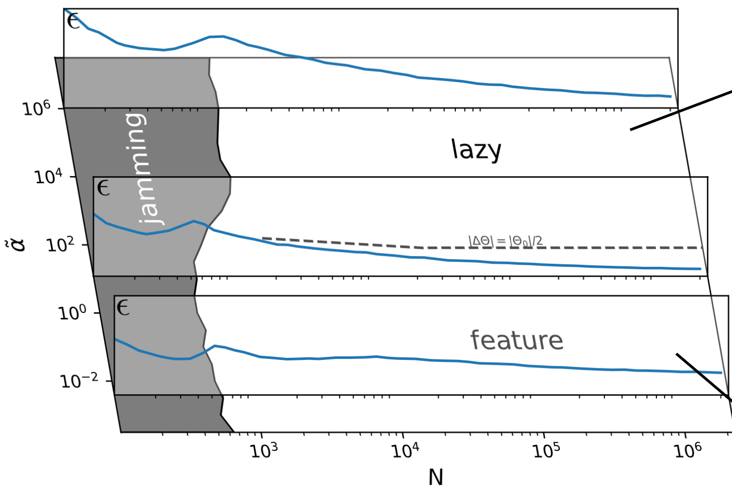
- Test error ε: probability to make a mistake
- No over-fitting as $N \to \infty$!!!
- Two interesting scaling regime: jamming and $N \to \infty$

(scaling theory in *Geiger et al, J. Stat. 19'*)

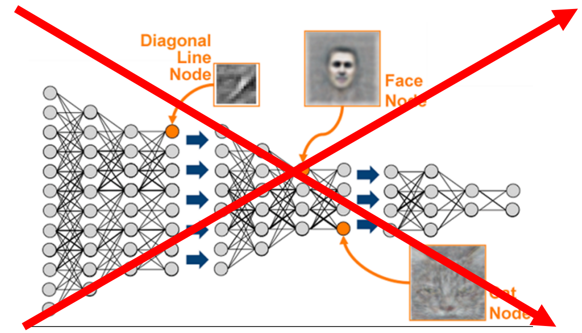
Two limiting algorithms as $N \to \infty$

See also Srebro, Montanari's lectures

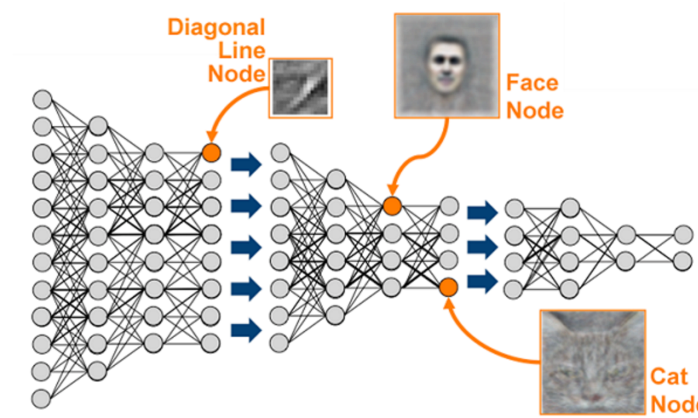
Jacot, Gabriel, Hongler 18'
Chizat, Bach 19'



Lazy regime:
No features learnt!
tiny changes of weights
Sufficient to fit data



Feature regime: data representation learnt *Chizat, Bach 18'*
Mei, Montanari, Nguyen 18'
Rotskoff, Vanden-Eijnden 18'



Second descent: noisy convergence to limiting algorithms *Geiger et al, 19', 20'*

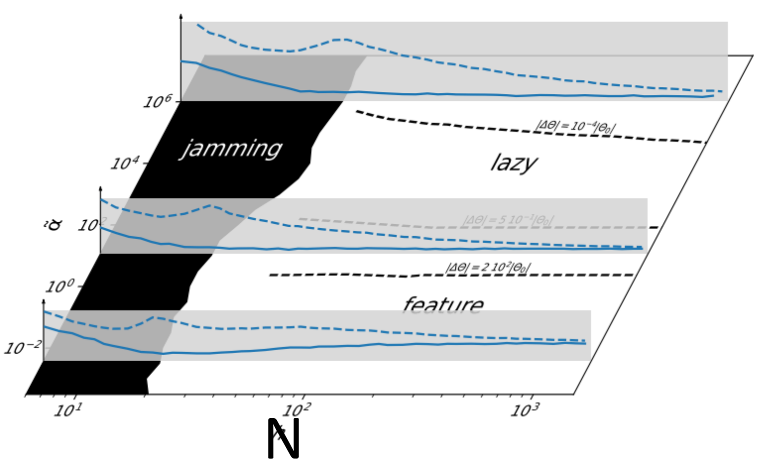
- As $N \to \infty$, randomness of initialization does not matter *Neal et al.,18'*
- At finite N, it leads to fluctuations of the predictor

$$\boxed{\delta f \sim N^{-1/4}} \qquad \boxed{\epsilon_t(N) - \epsilon_t(N=\infty) \sim N^{-1/2}}$$

- Easily evidenced by averaging the output of several networks

Geiger et al, 19

Lee et al. 20'

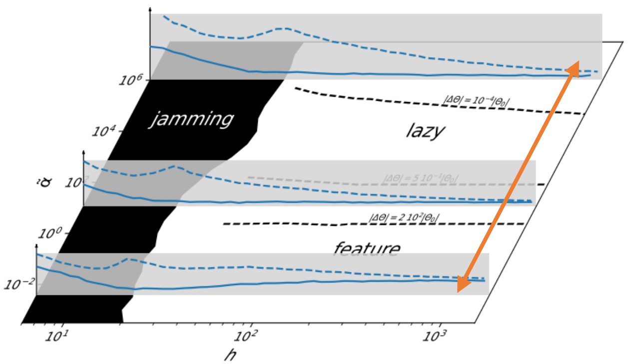


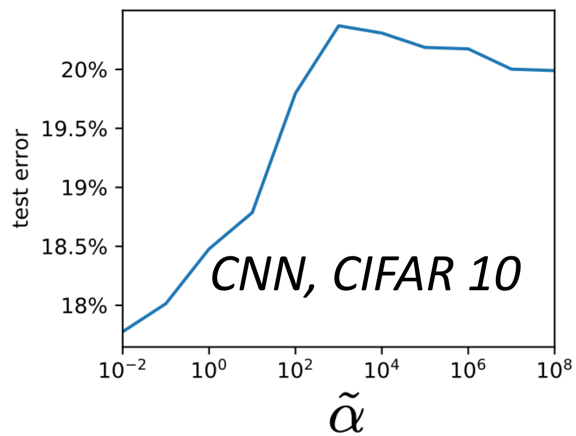
$$\boxed{\bar{f}_N = \frac{1}{M} \sum_{i=1\dots M} f_{N,i}}$$

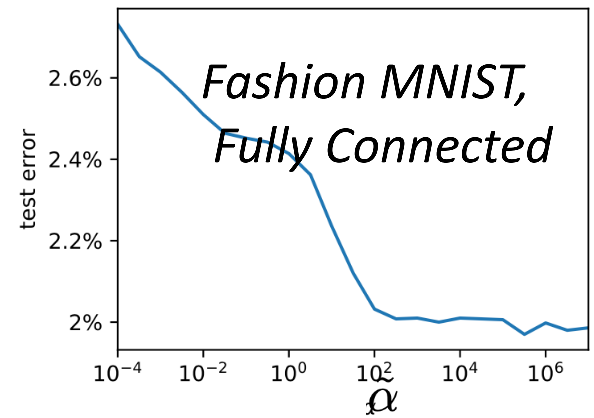
From now on, $N = \infty$

(same effect key in random feature models *d'ascoli et al. 20'*)

Image data sets: feature beats lazy in CNNs, not in fully-connected nets







- CNN performs better in feature learning regime

  Geiger et al. 19', Chizat et al. 19', Gorbani et al. 19'

- For images, Fully connected net better when lazy *Geiger et al. 19', Lee 20'*

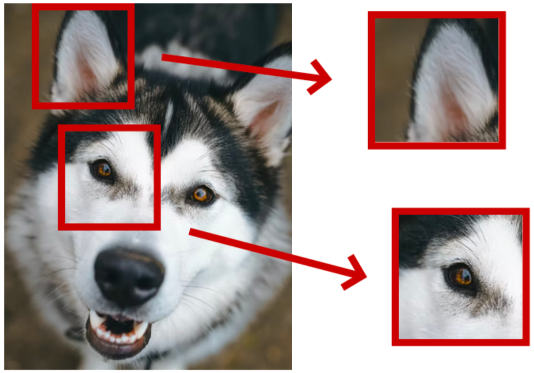
Curse of dimensionality

Which properties of the data make them learnable? Images:

1. Locality: The task depends on the presence of local features

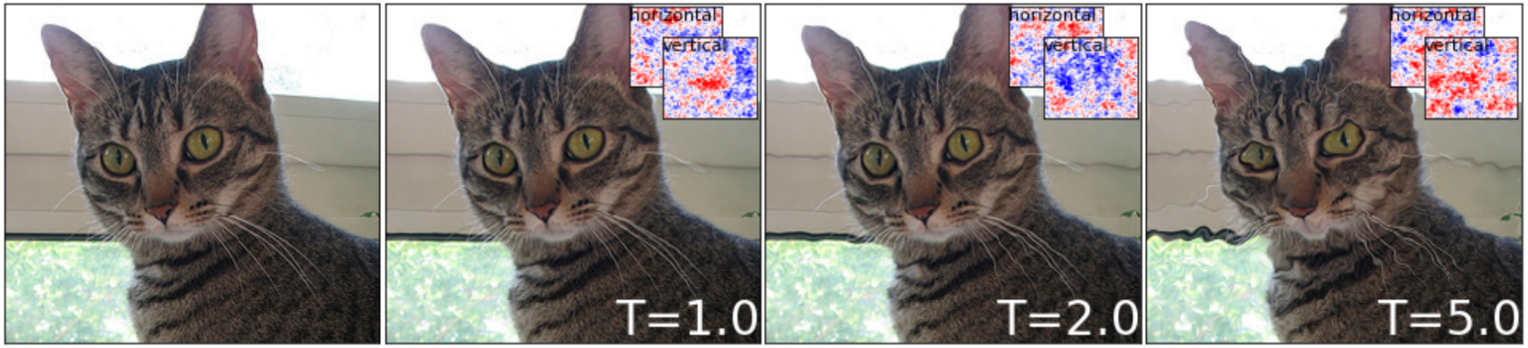
2. The task is combinatorial/hierarchical

Poggio et al. 16', 20', Bietti 21', Malach et al. 18'



3. The task is stable to smooth transformations

Mallat, Bruna 13'



Curse of dimensionality

1. Locality
2. Combinatorial/hierarchical
3. Stability diffeo

Natural guesses considering that successful architectures such as CNNs:

- Have local filters (1)
- deep so naturally express combinatorial functions (2)
- Are translational invariant (weight sharing) (3)

- *Are 1,2,3 key to beat the curse? If so, which one is most important?*

Currently:

(1,2) teacher-students models of such data structure where training curves can be computed to show that. Lazy regime.

(3) Is it true? Empirical study

1/ local tasks in lazy regime

Favero, Cagnetta, MW NEURIPS 21'

- Regression: approximating some true function f*

$$\boxed{\epsilon = \mathbb{E}_{\boldsymbol{x}, f^*}[(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2]}$$

- Inputs are d-dimensional random sequences

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

$\boldsymbol{x}_i$ t-dimensional **patch**

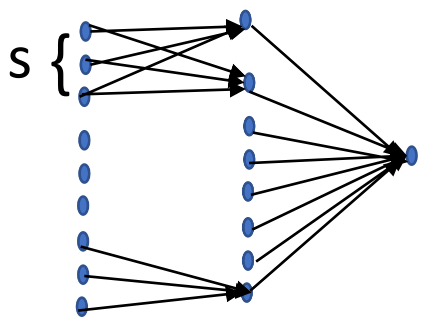
- Task is t-local:

$$\boxed{f^* = \sum_{i=1}^{d} g_i(\mathbf{x}_i)}$$

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

- Student is s-local:

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



Patches of size s

Curse of dimensionality beaten

- Calculation uses physics based-methods

Bordelon et al. 20', Spigler et al. 19'

- If $s \geq t$ $\boxed{\epsilon(P) \sim P^{-\alpha_t/s}}$

- Curse of dimensionality indeed occurs if the student has no prior on locality, i.e. s=d

- Curse beaten however when the student is local with

$\boxed{t \leq s << d}$

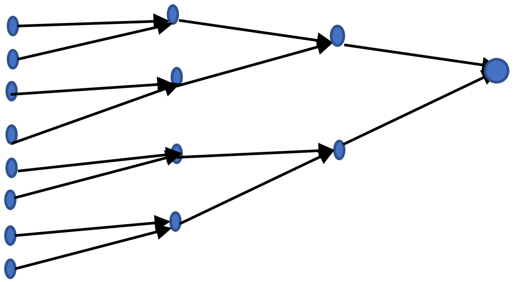
- Translation invariance has only a mild effect (multiplies P by d) (generic argument by *Bietti, Bruna 21'*). Empirically, locality appear more important *Neyshabur 20'*

- But model too simple for real data! (local 1-hidden layer does not work well). Hierarchical!

2/Hierarchical data

Cagnetta, Favero, MW submitted 22'

Hierarchical CNN. Diagonalize its NTK



- Positive results:

Hierarchichal CNN is *adaptive*
If the task only depends on t adjacent
Variable, for large t

$$\epsilon(P) \sim P^{-\beta} \text{ with } \beta \approx \alpha_t / t$$

- Interesting negative results: too complicated as a teacher!

$$\epsilon(P) \sim P^{-\beta} \text{ with } \beta \sim \frac{1}{d}$$

- Images must have stronger structure to be learnable

e.g. Poggio et al. 16', 20'

3/ Testing empirically stability to diffeo

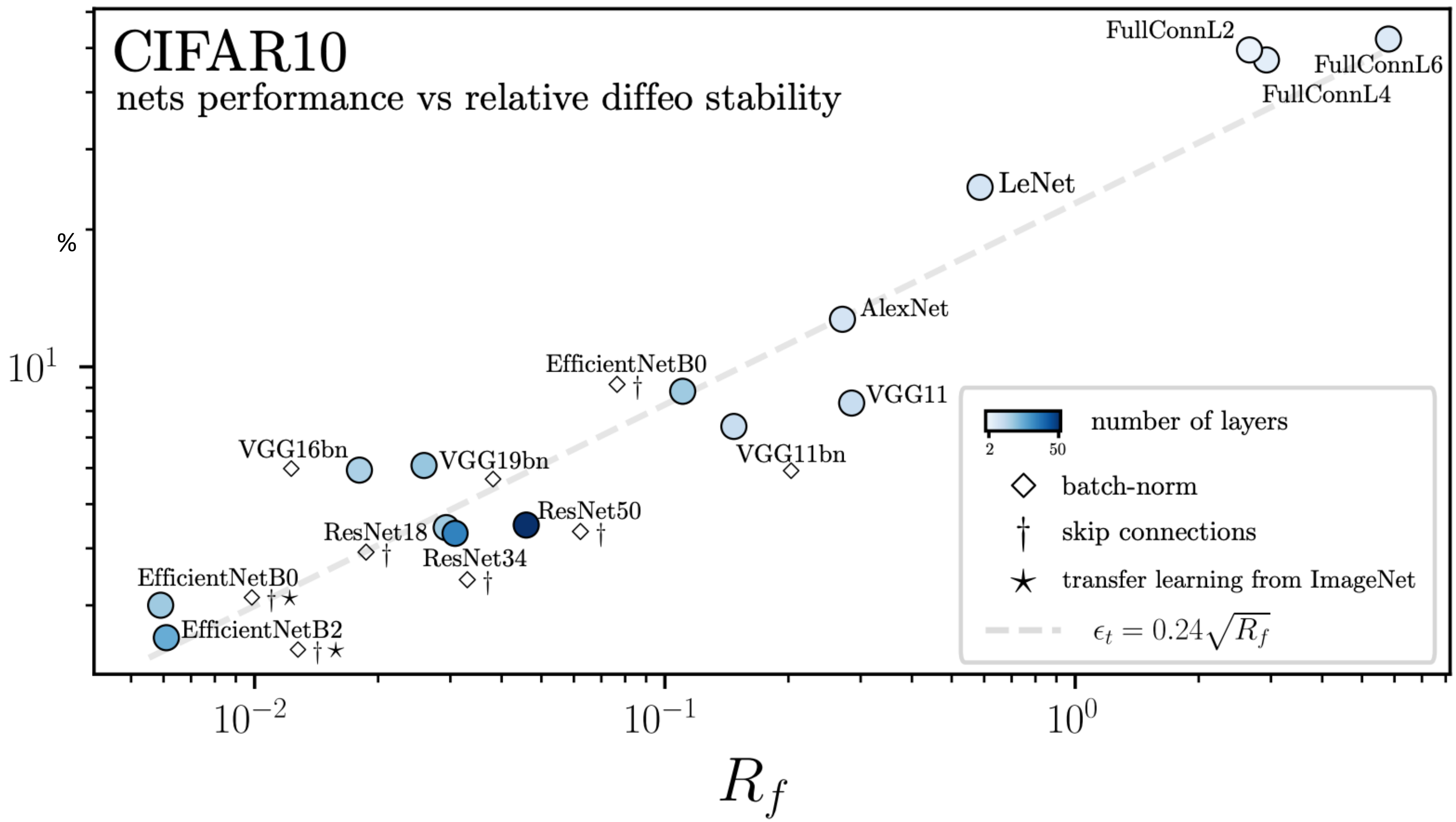
Petrini, Favero, Geiger, MW, NEURIPS 21'

- Test: maximum entropy distribution of smooth transformations

(thermal spring network)

Sensitivity to diffeo: $\boxed{R_f \sim \langle \| f(\tau x) - f(x) \|^2 \rangle_{x,\tau}}$

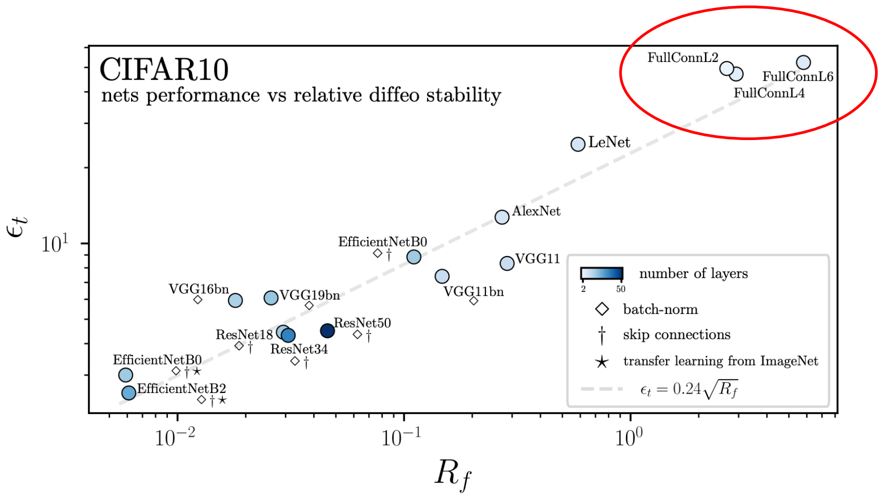
Sensitivity to smooth transformations strongly correlates to performance *Petrini, Favero, Geiger, MW, NEURIPS 21'*



- Supports small sensitivity to diffeo R_f key to performance.
- It is learnt! At initialization, $R_f \approx 1$.
- Continuously develops through depth

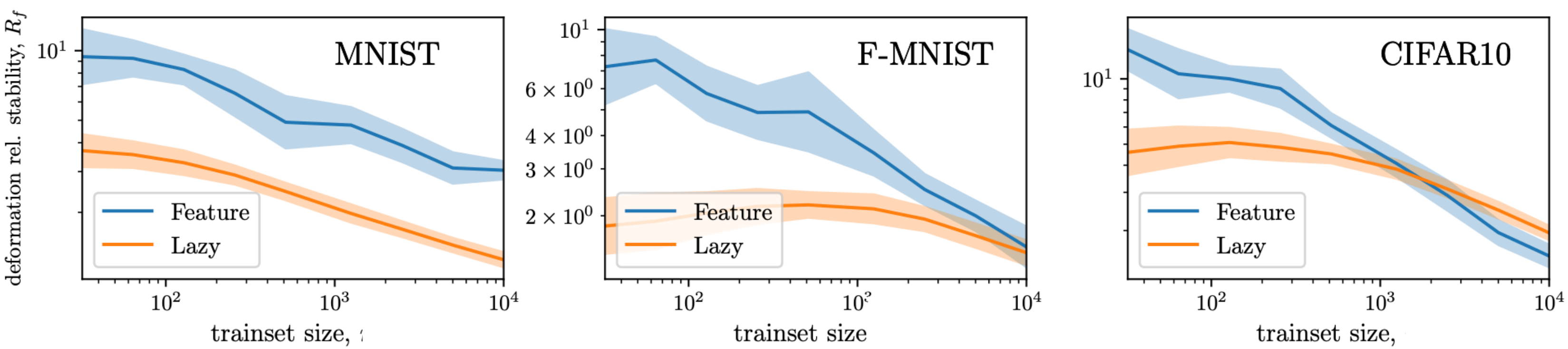
Sensitivity to diffeo in fully connected nets

Petrini, Cagnetta, Vanden Einjnden, MW arxiv 22'



- Fully connected nets become more sensitive to diffeo after training
- Feature learning is detrimental for them.

Suggests learning features is detrimental in fully connected nets
Because it makes them less stable to diffeo



Why is it so?

Learning sparse features is detrimental for smooth tasks

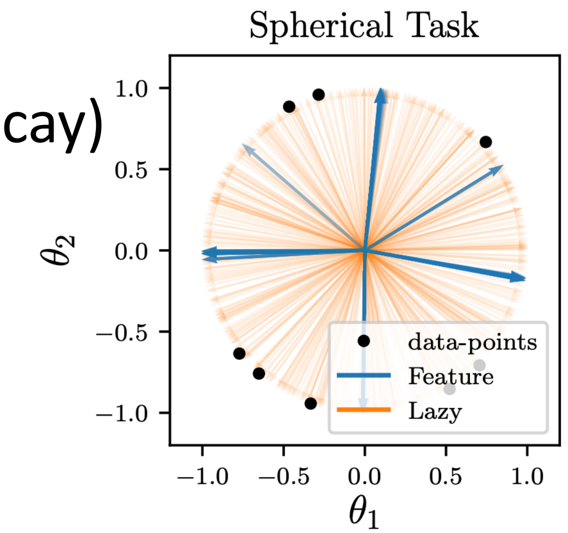
Petrini, Cagnetta, Vanden Einjnden, MW arxiv 22'

- Often learning features lead to sparse

Representations (small initialization or weight decay)

Srebro's lecture

effective neurons ~ size of training set



- It is a disadvantage if the target, true function

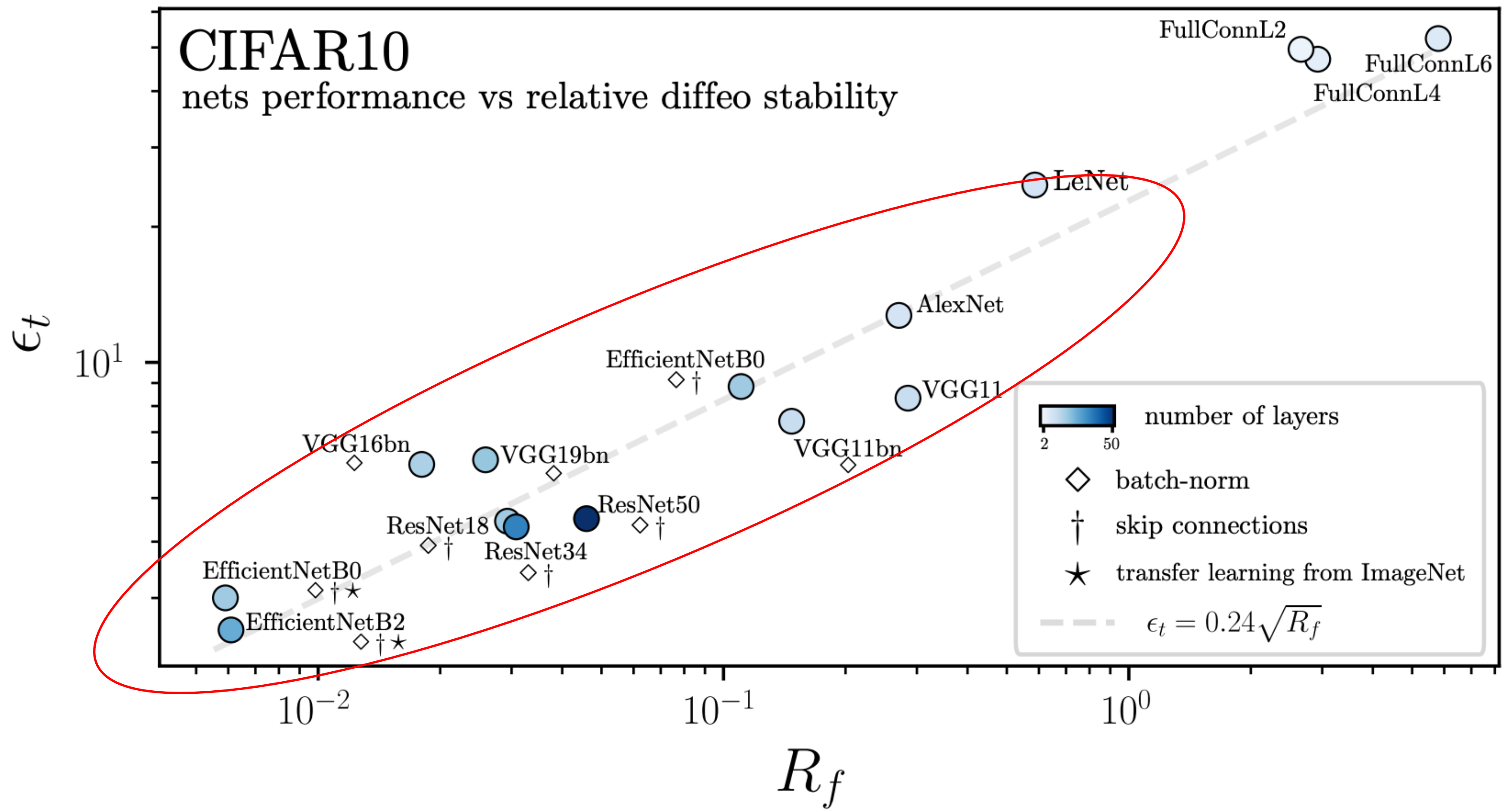
is smooth enough. E.g. constant function, data on sphere:

$$\epsilon(P) \sim P^{-\beta}$$

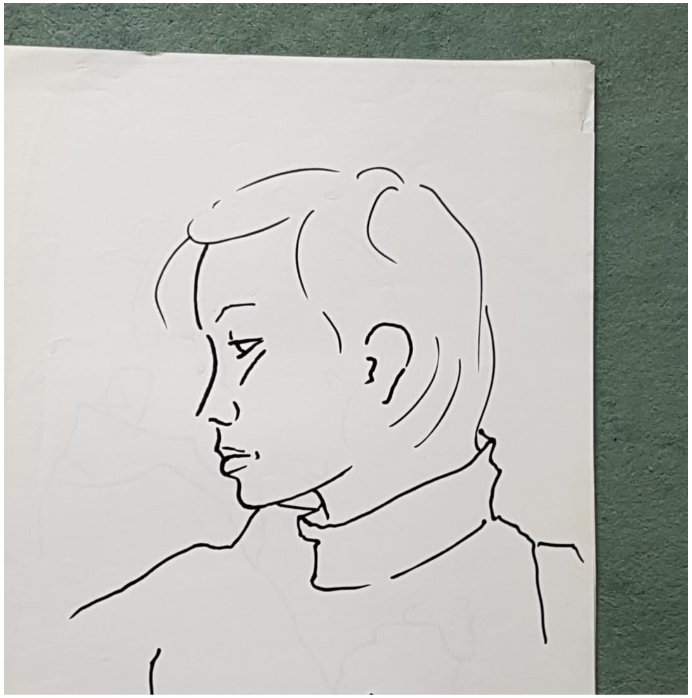
Large d   Lazy: $\beta = 2$ Feature: $\beta = 1$

Learning sparse features leads to rougher, less smooth predictors

How Stability to diffeo is learnt?



How is stability learnt? A Hypothesis



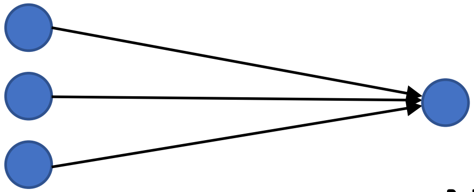
- object depends on local features

- Hierarchical: object made of local features, themselves made of sub-features etc...

  e.g. poggio et al, 20'

- High-dimensional because relative distance between features is not fixed

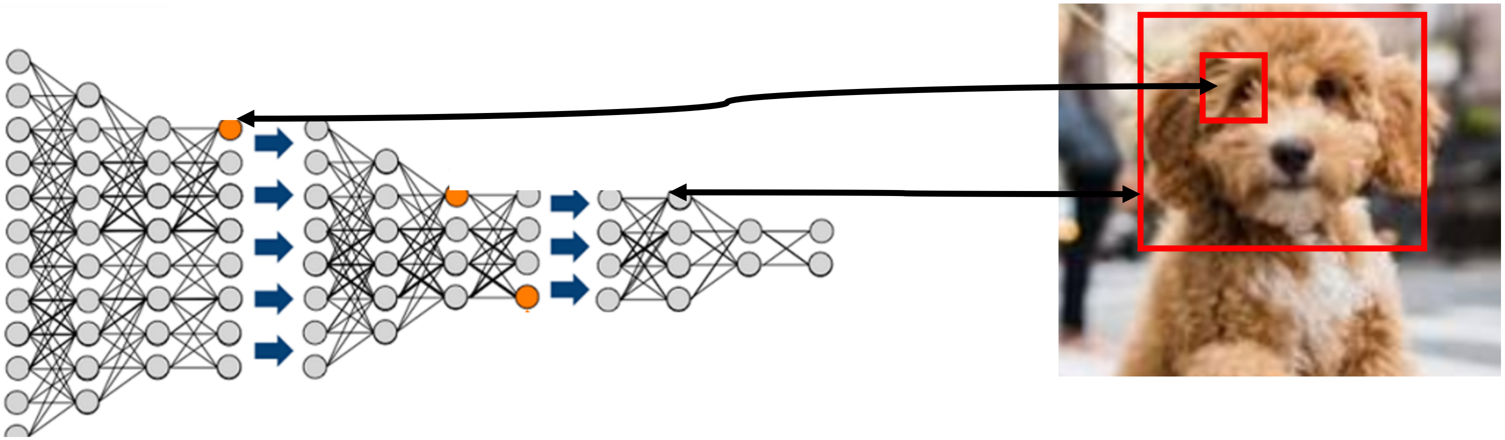
- This information must be lost: 'pooling' on correct scale (some evidence in *Ruderman et al. 18'*)



Neurons coding for 'nose' at specific locations

Neuron coding for 'nose' somewhere around

Adaptative pooling



<u>Hypothesis:</u>   In the feature learning regime, neurons learn how to pool on the correct scale. It:

- increases stability toward smooth deformations
- Effectively lowers the dimension of the problem (helps to beat the curse)

<u>Missing:</u> Simple hierarchical toy models of data to understand this adaptive pooling, and its effect on performance

Conclusion

1. Afraid of bad minima in the loss landscape?

Just crank up the number of parameters

2. Afraid of overfitting?

No worries, deep learning converges to well-defined algorithms as

N diverges, causing second descent

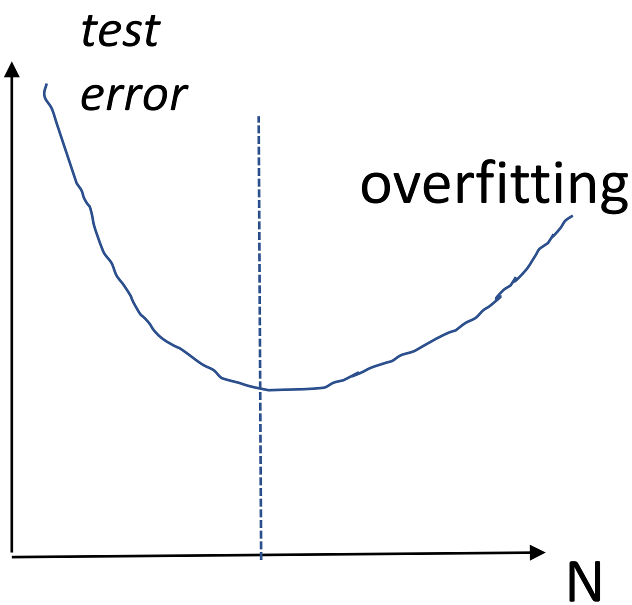
3. Afraid of the curse of dimensionality?

- locality &

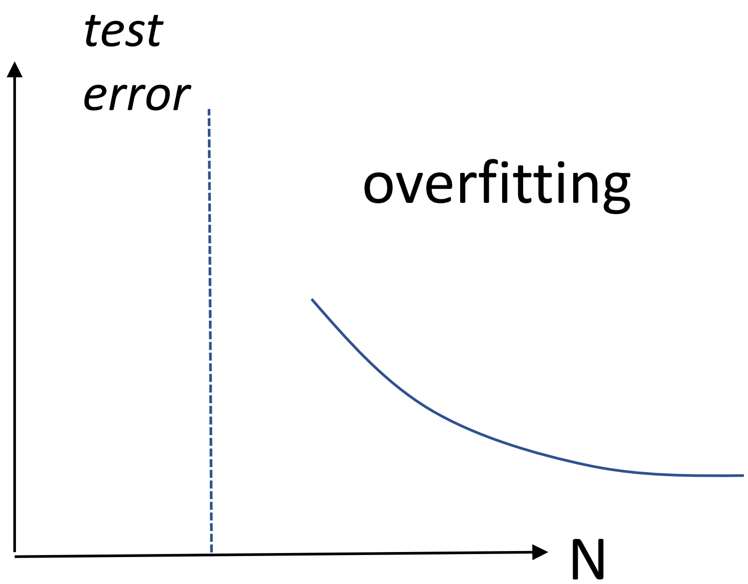
- Stability to smooth transformations appears key to performance

- Suggests that curse can be beaten when an object consists of local parts, made of local subparts etc... whose relative positions can fluctuate. Need models

Why no overfitting?



Naive guess

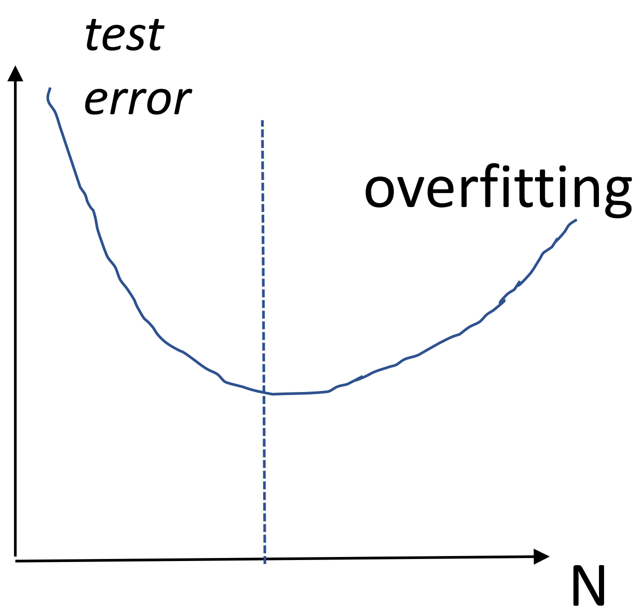


Observation

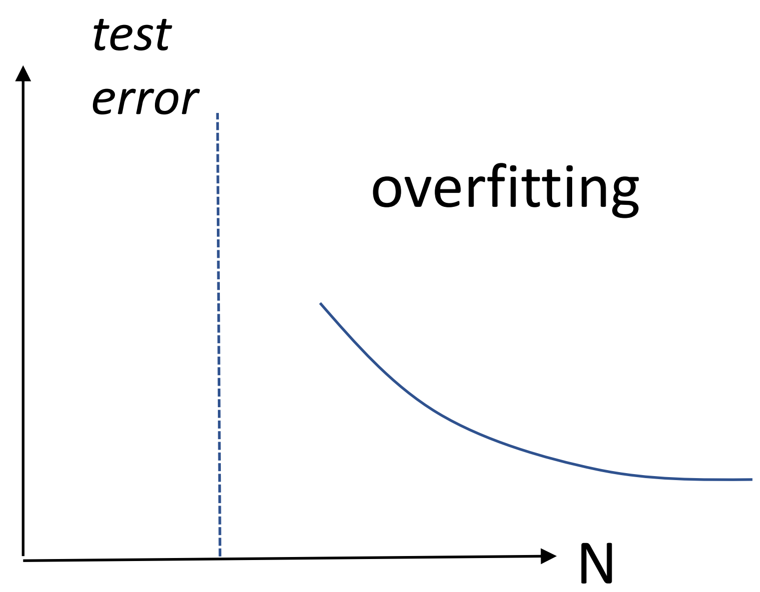
Srebro

Why is increasing N in a regime where data are perfectly fitted beneficial?

Why no overfitting?



Naive guess

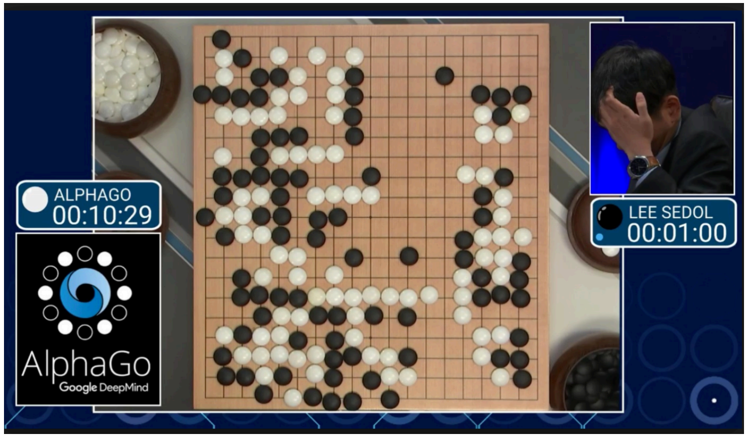


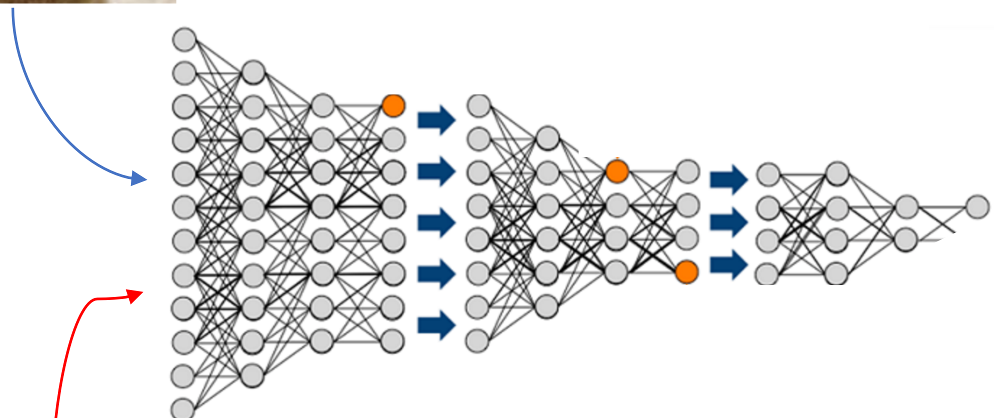
Observation

Srebro

Why is increasing N in a regime where data are perfectly fitted beneficial?

Deep learning

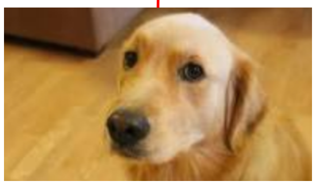




>1

$f_{\mathbf{w}}(\mathbf{x}_i)$

<-1



1/learning from example

2/ can predict!

- Revolution in Artificial Intelligence (go playing, self-driving car...)
- Principles to understand why It works are lacking

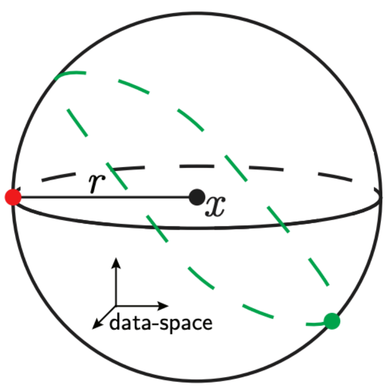
E.g: How many data are needed to learn a given task???

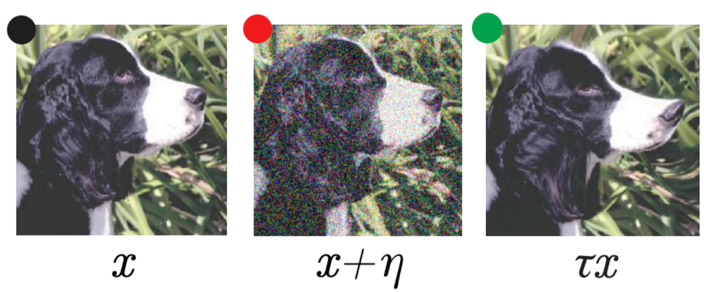
Stability to deformations must be defined in relative terms

- Sensibility of output to smooth deformations not best observable

- Best networks trained on many data become sensitive to noise

- Relative stability to diffeomorphisms R_f





$$R_f = \frac{\langle \| f(\tau x) - f(x) \|^2 \rangle_{x,\tau}}{\langle \| f(x+\eta) - f(x) \|^2 \rangle_{x,\eta}}.$$

1/ local tasks in lazy regime

Favero, Cagnetta, MW NEURIPS 21'

- Regression: approximating some true function f*

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

- Inputs are d-dimensional random sequences

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

$\boldsymbol{x}_i$ *t*-dimensional **patch**

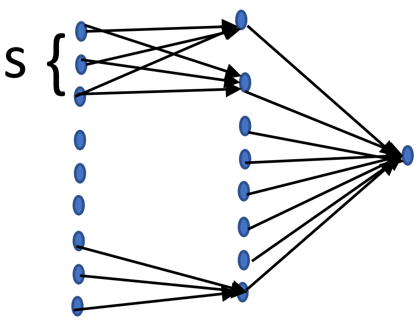
- Task is t-local:

$$f^* = \sum_{i=1}^{d} g_i(\mathbf{x}_i)$$

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

- Student is s-local:

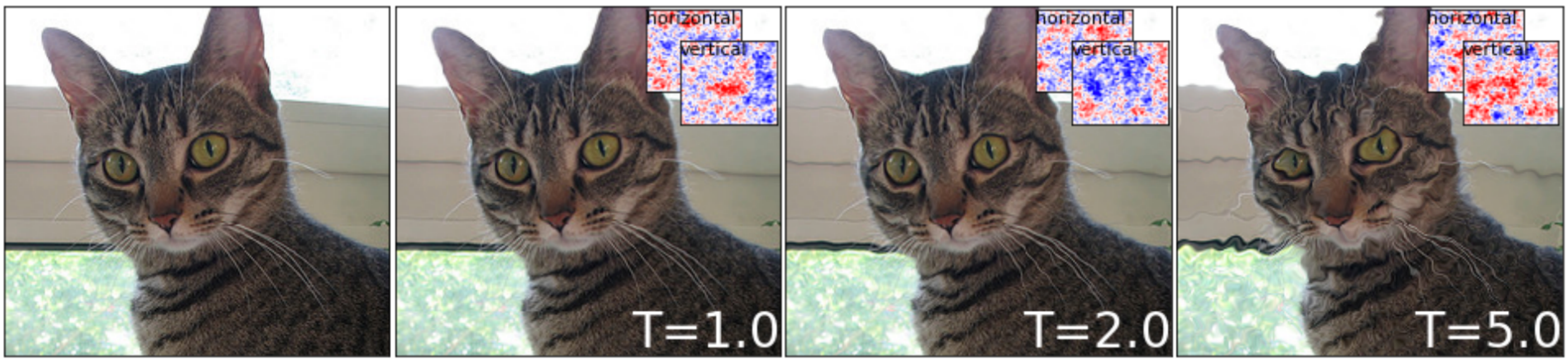
$$K(\mathbf{x}, \mathbf{x}') = \frac{1}{d} \sum_{i=1}^{d} C(\mathbf{x}_i, \mathbf{x}'_i)$$



Patches of size s, Smoothness of Kernel C is α_s

Stability toward smooth deformations?

- Proposition: Deep nets work because they the task is invariant toward smooth deformations, and they learn an invariant representation *Mallat, Bruna.*
- It effectively reduce the dimension of the problem, allowing to beat the curse



- Is it true? Not really supported by existing observations. *Dieleman et al. 16', Azulay et al. 18', Zhang 19'*

- Mechanism for net to become invariant ? Effect on performance?