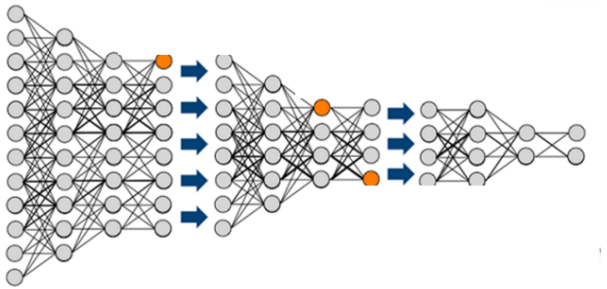
How the compositionality of tasks affect their learnability?

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, A.Scolcchi, U. Tomasini, K. Tyloo*



EPFL SIMONS FOUNDATION FNSNF

Classifying data in large dimension

- Pre-requisite for Artificial Intelligence (and brains):

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

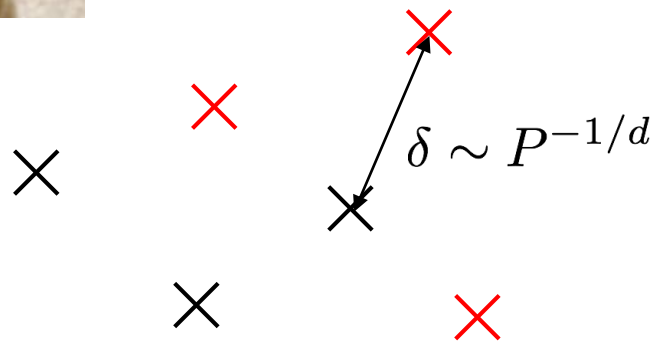
- Example: computer vision. Is it a cat or a dog?

- Learn from examples (supervised learning)

10^6

10^6

- Problem: no algorithms should work, curse of dimensionality



P points in d dimensions

- Data must be highly structured!?

Curse: different level of questions

1) Is there enough information to reconstruct the task?
(may be completely impractical)

2) Are they weird, more practical (much faster), algorithms that work?

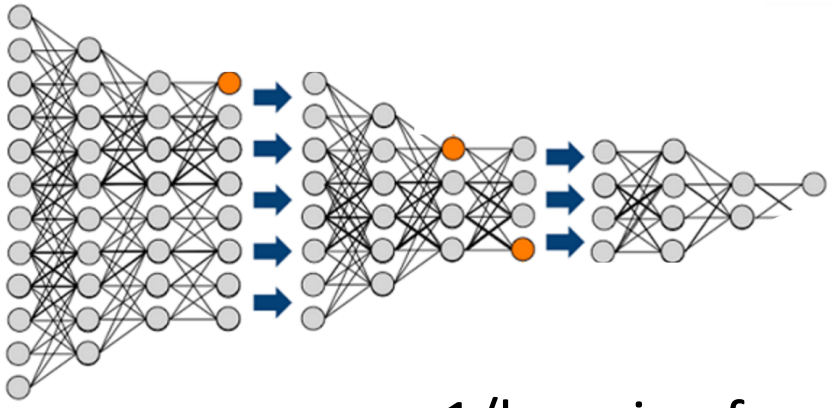
3) Can generic algorithms like neural nets with GD do it, and why?

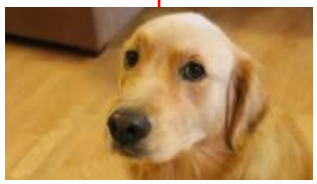
(*) What in the structure of text, images, etc... make them learnable by deep nets?

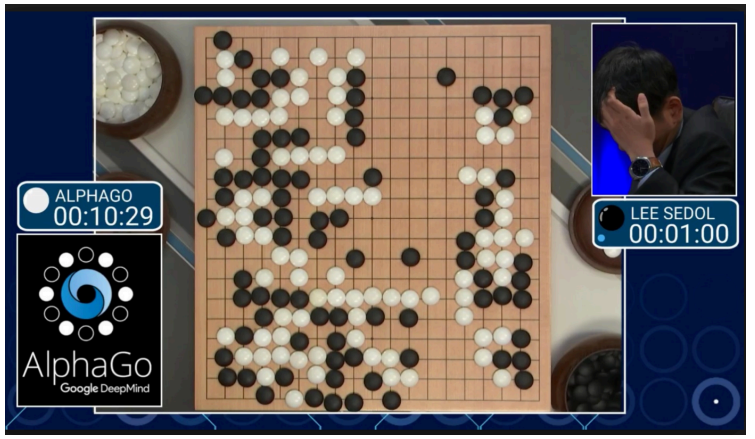
Discussion focuses more on images and CNNs but hopefully more general

(*) relates to practical question: how many data to learn a task?

Deep learning







>m

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

<-m

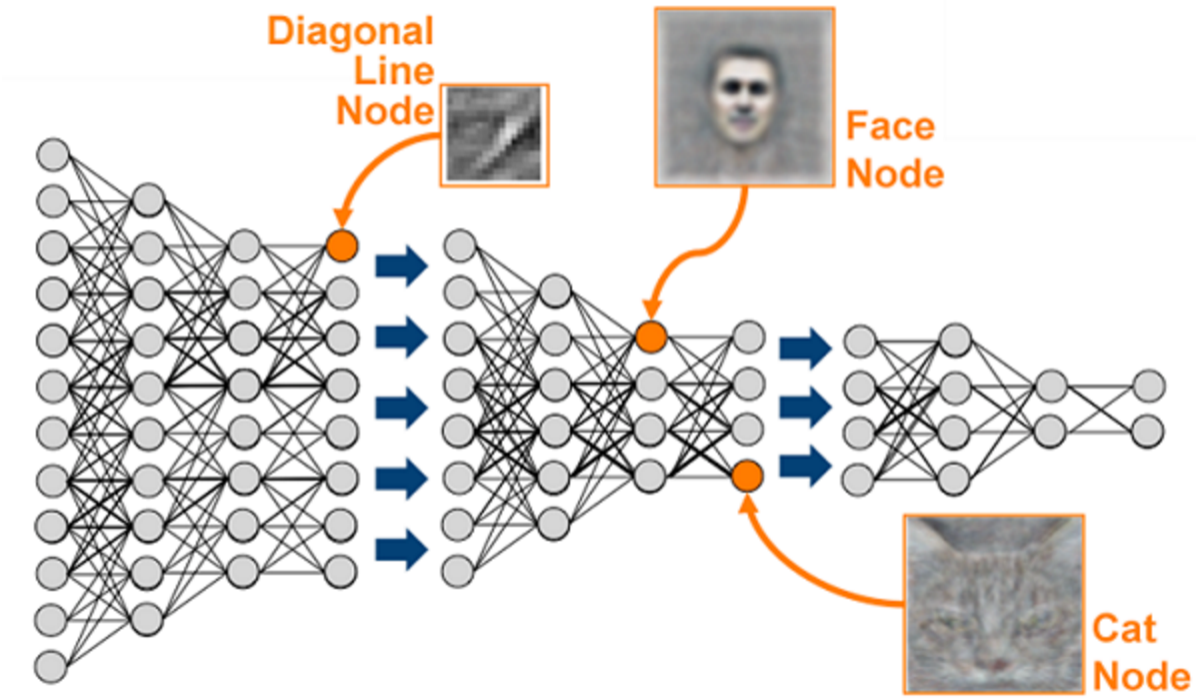
1/learning from example

2/ can predict!

- Powerful! go playing, self-driving car, Chat GPT...
- Principles to understand why it works are lacking

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

Benefits of learning a data representation?



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

When is it possible, which advantage? Intuition: lower dimensionality of the problem

-Empirical characterization: invariance (deformations and `semantic' Choice)

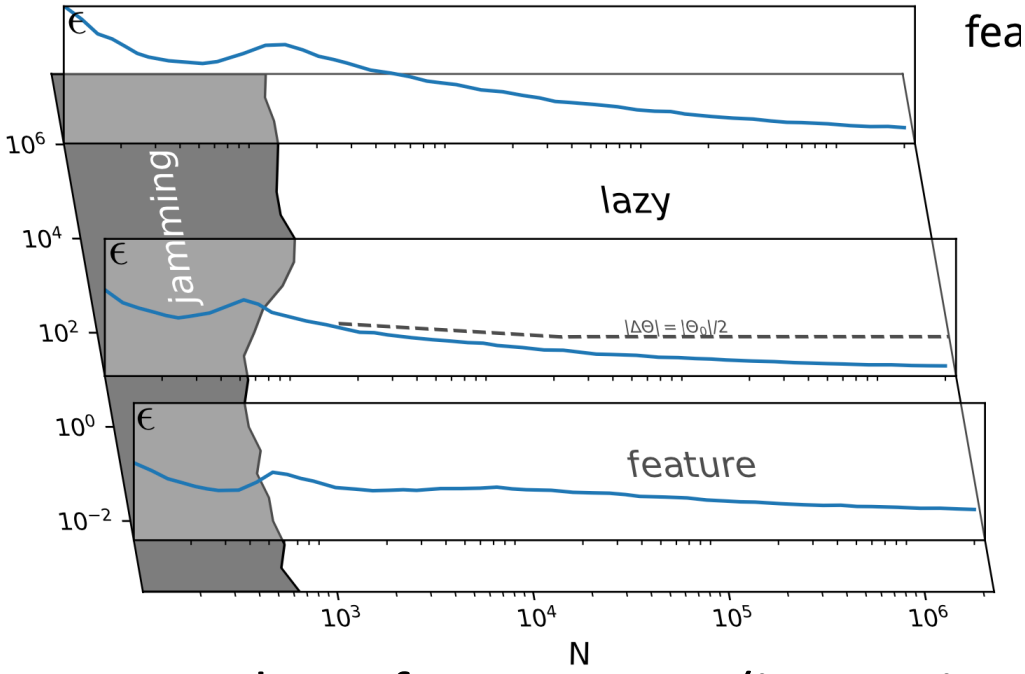
- How many data needed? Simple models

Overparameterization, training regimes: A phase diagram for deep learning

Geiger et al., 21'

- Data sets: e.g. MNIST binary classification
- Fully connected net

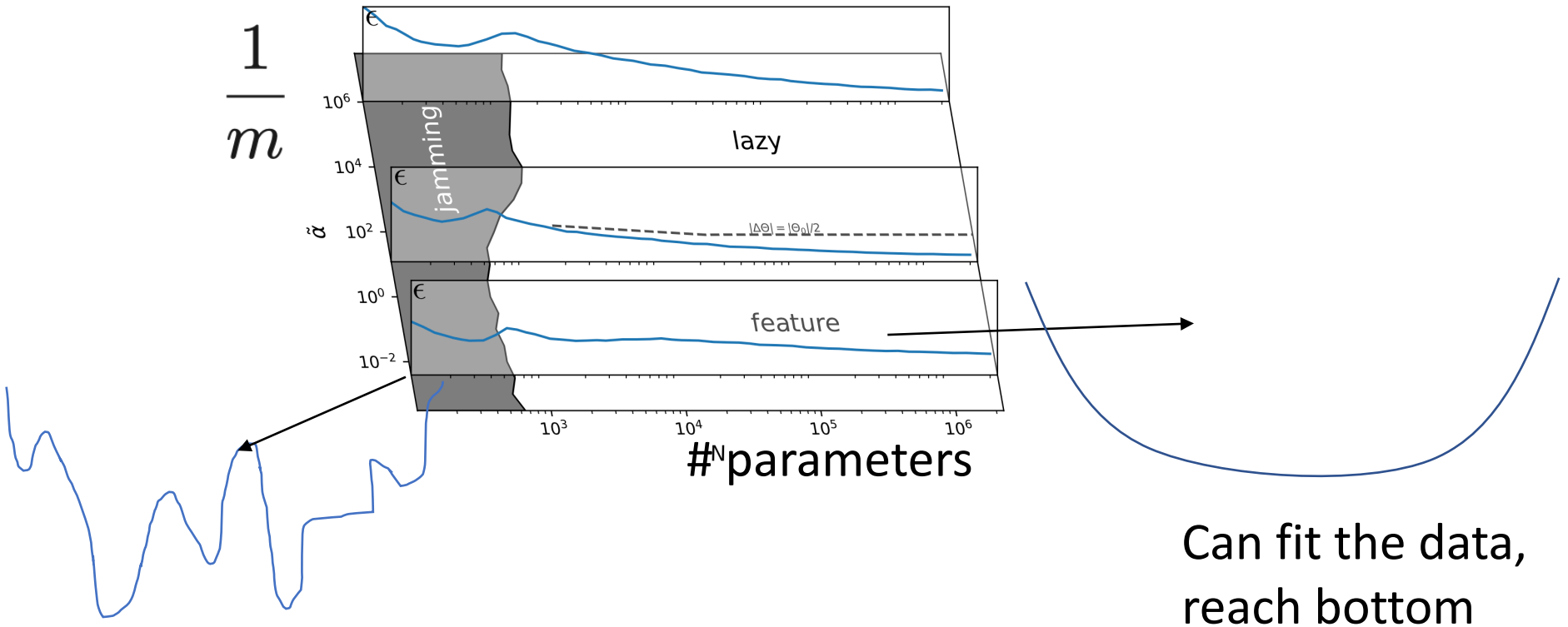
Inverse margin $\frac{1}{m}$



N: Number of parameters (increasing width)

A phase transition separates under-parametrized and over-parametrized regimes

Geiger et al., 2018



Cannot fit the data

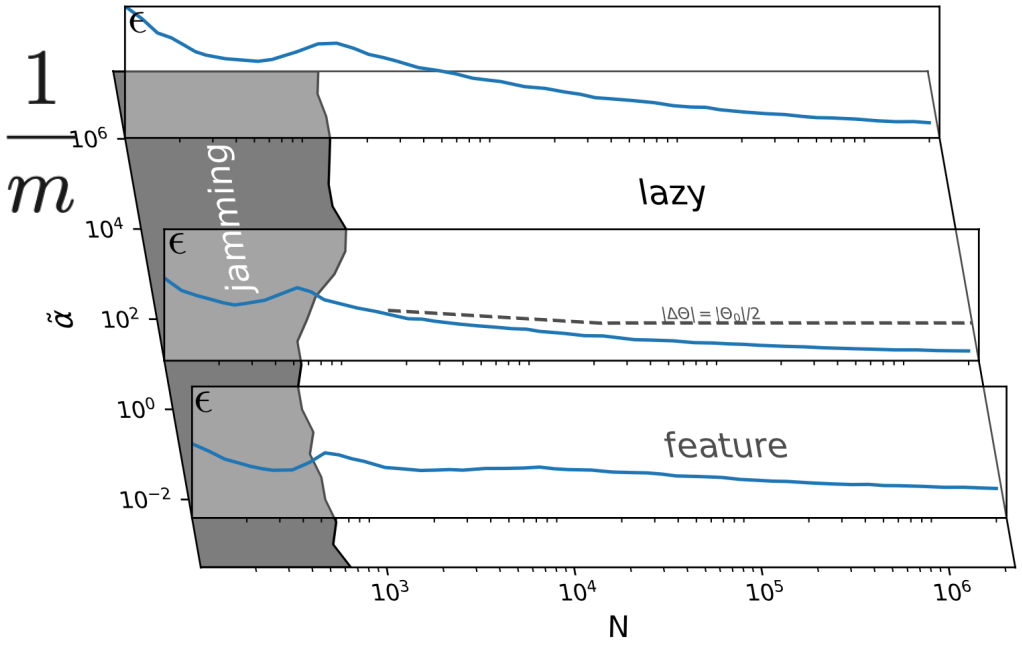
Can fit the data, reach bottom

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

- *Sharp phase transition from rough, glassy to flat landscape*
- *Crank up N, not stuck in bad minima*
- *Overfitting?*

Overfitting? Instead, a 'double descent'

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

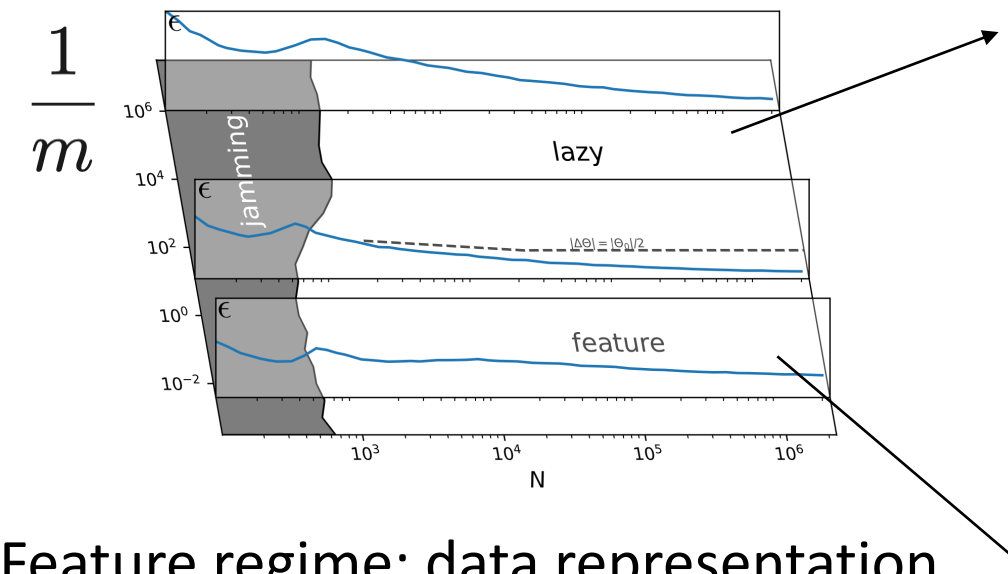


- Test error ε: probability to make a mistake
- No over-fitting as $N \to \infty$!!!
- Two interesting scaling regime: jamming and $N \to \infty$

(scaling arguments in *Geiger et al, 19'*)

Two limiting algorithms as $N \to \infty$

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

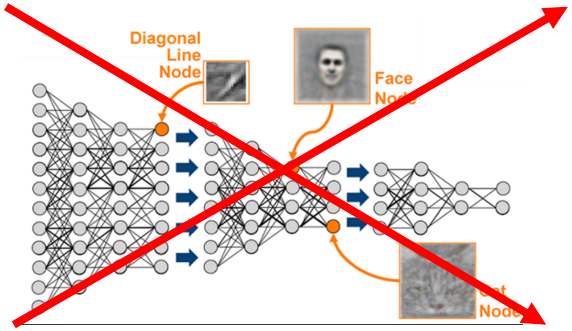


Lazy regime:

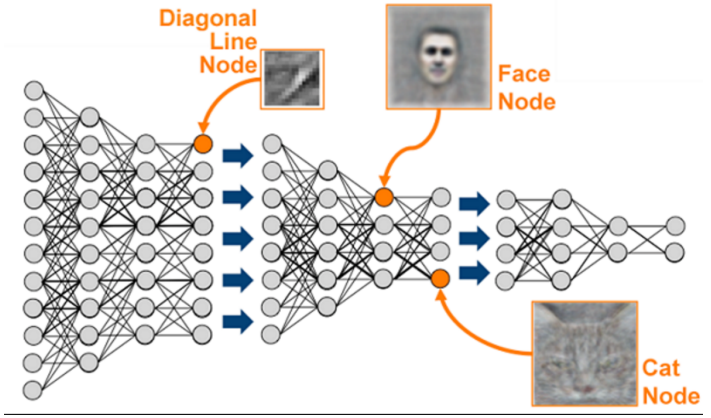
No features learnt!

tiny changes of weights

Sufficient to fit data



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



For modern architectures and images, Feature is better

What are CNNs good at, in both regimes?

Separation lazy vs feature regimes: some limitation of literature

- Mostly concerns fully connected (FC) nets (Barron space vs RKHS):

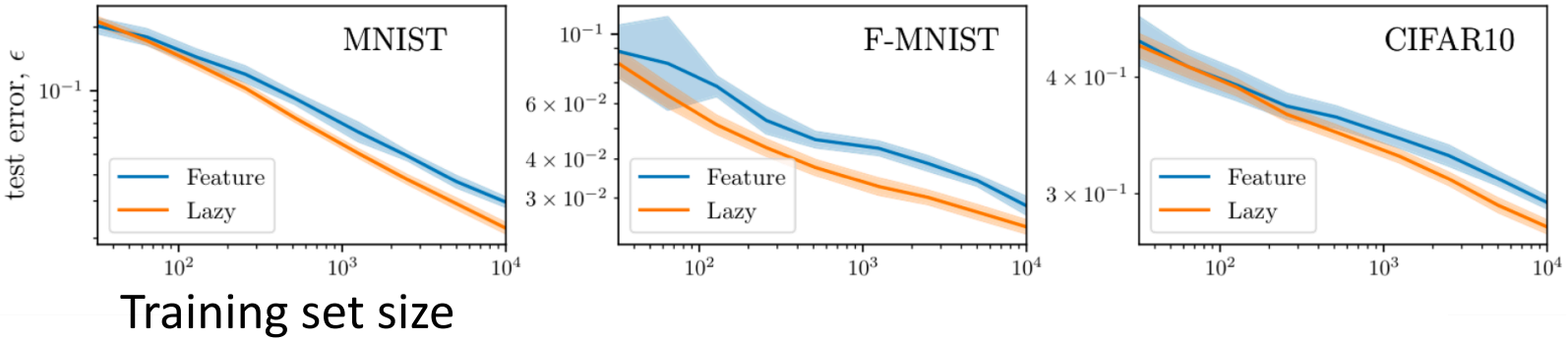
Anisotropic function $f(\mathbf{x}) = f(\mathbf{x_s}) \quad \mathbf{x} \in \mathbb{R}^d \quad \mathbf{x}_s \in \mathbb{R}^s$

$s \ll d$

- These results are needed as a first step, but do not answer (*):

- FC nets do not perform well

- For gradient descent, lazy outperforms feature regime for FC nets!

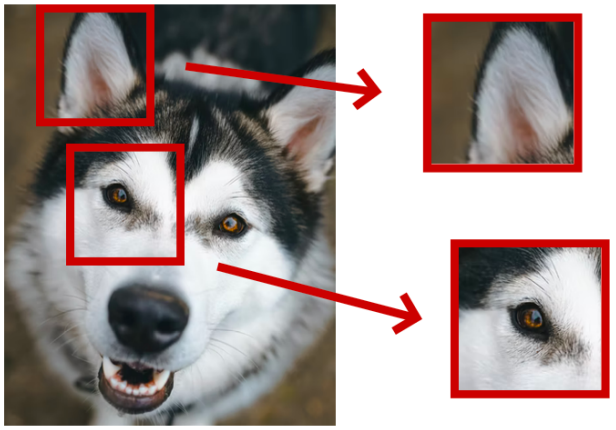


Geiger, Spigler, Jacot, MW, j stat 20', Petrini et al. Neurips 22'
Lee et al., 20'

Anisotropy not adequate structure for (*)

Curse of dimensionality

Which properties of the data make them learnable? Some (among others) ideas for images:

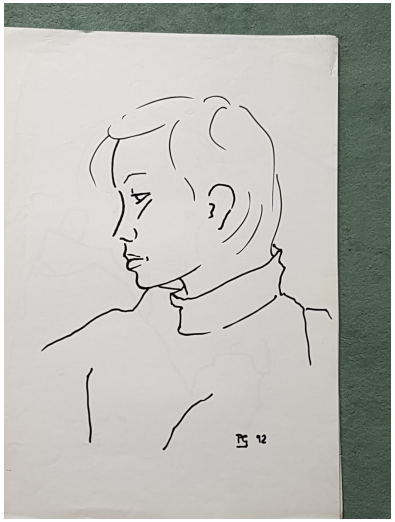


1. Locality: The task depends on the presence of local features

2. The task is combinatorial/hierarchical

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

3. The task is 'sparse'

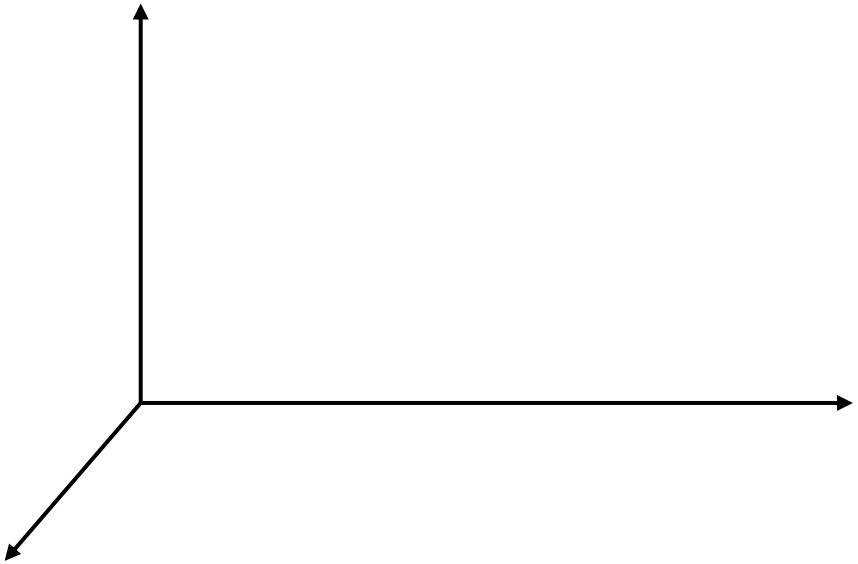


Very limited quantitative understanding...

Presumably also relevant for text

Goal: phase diagram for sample complexity P* when learning feature

Sparsity (s_0) [Umberto's talk next week]



Locality (s)

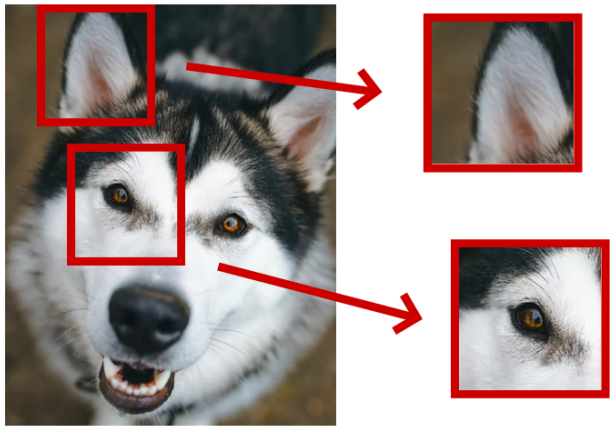
Combinatoriality
(L, v, m)

CNNs in lazy regime adapt to the locality scale

Favero, Cagnetta, MW NEURIPS 21'
Cagnetta, Favero, MW, ICLR 23'

- Regression task is local:

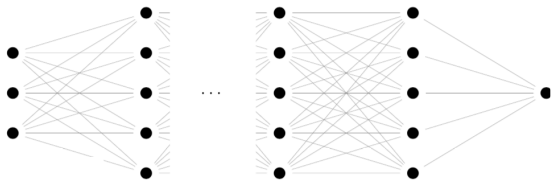
$$f^* = \sum_{i=1}^{d} g_i(\mathbf{x}_i)$$



$\mathbf{x}_i \in \mathbb{R}^s$

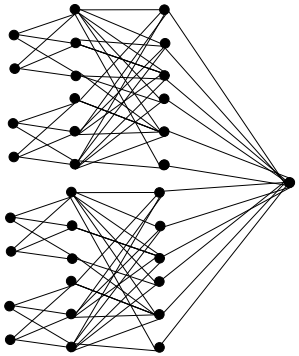
$\mathbf{x} \in \mathbb{R}^d$

- Architecture:



Lazy Fully-connected net

cursed $P^* \sim e^{\mathcal{O}(d)}$



Convolutional net (CNN)

- Local
- hierarchical

Not cursed, nearly optimal

$P^* \sim e^{\mathcal{O}(s)}$

- *Locality can be very helpful*
- *Hierarchical structure allows for adaptivity to scale s*
- *Negative result: still cursed if task involves long distance dependencies*

Hierarchically Compositional data

Example: $f(x_1, x_2, x_3, x_4) = g(g_1(x_1, x_2), g_2(x_3, x_4))$

[and more iterations]

Results:

- Deep networks represent these functions efficiently *Poggio et al., '17*

- These functions can be reconstructed with poly(d) samples *Schmidt-Hieber, '20*

- Generative models of hierarchical data.

E. Mossel 16', E. Malach and Shai Shalev-Shwarz 18',20'

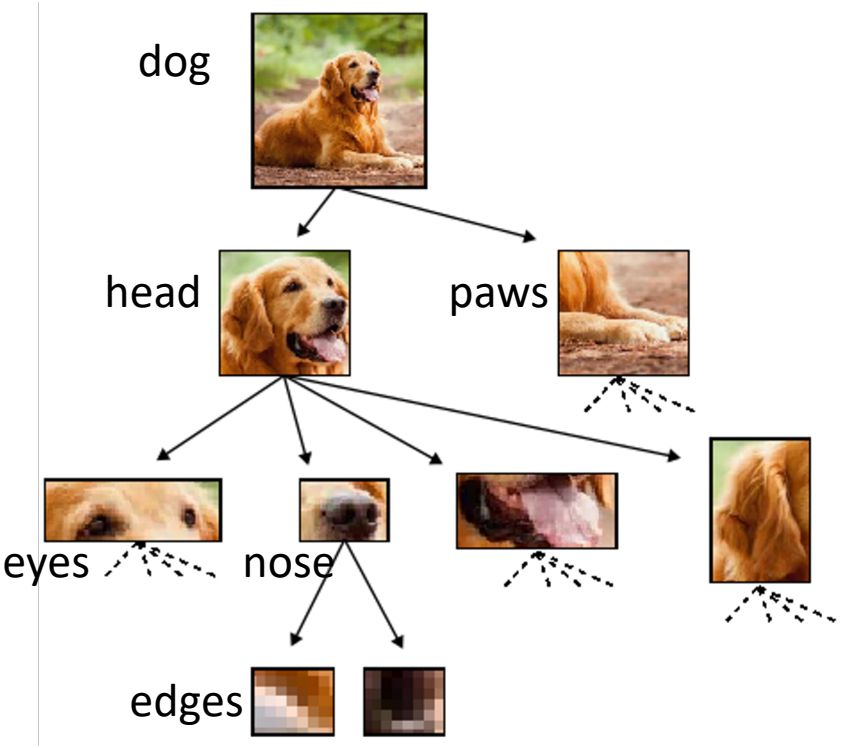
Analysed with Sequential algorithms (include clustering step)

-> Correlations between output and local portions of the input are key

Here: study CNN with gradient descent

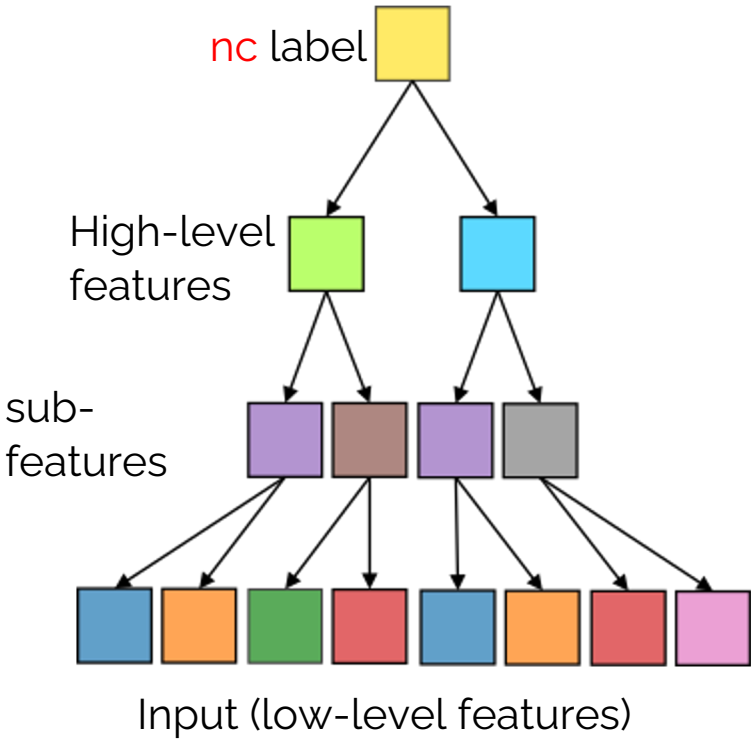
Introduce a data model which is rich, yet correlations computable

Hierarchical Generative Models



- n_c classes
- Inputs are generated via a hierarchy of L compositions
- One high-level feature corresponds to s sub-features
- Individual sub-features can be shared. finite vocabulary of size v
- A high-level feature can be decomposed into m strings of s subfeatures

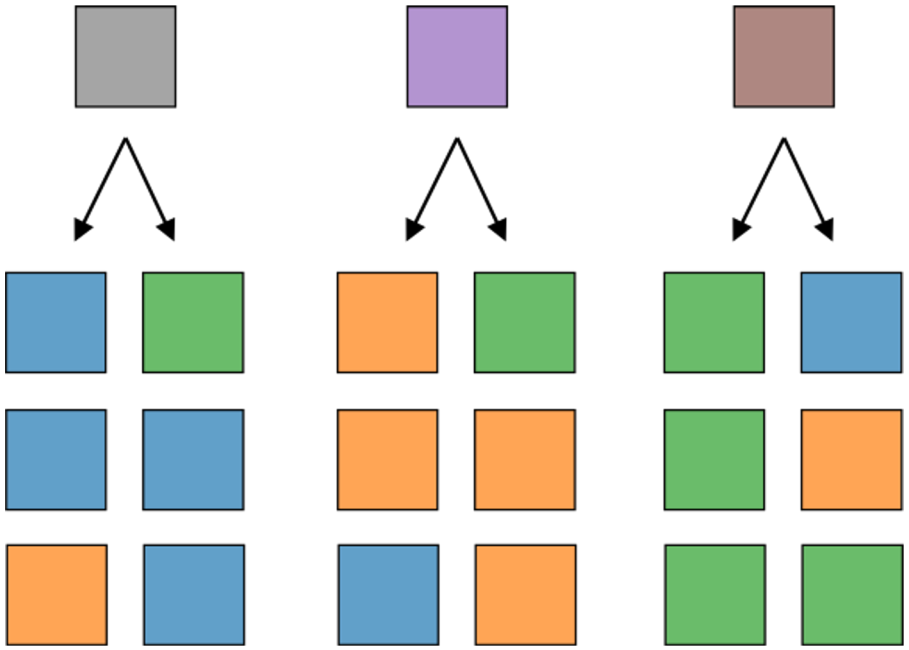
Hierarchical Generative Models



Example with L=3, s=2
(directly generalizable to images
In two dimensions)

- n_c classes
- Inputs are generated via a hierarchy of L compositions
- One high-level feature corresponds to s sub-features
- Individual sub-features can be shared. finite vocabulary of size v
- A high-level feature can be decomposed into m strings of s subfeatures

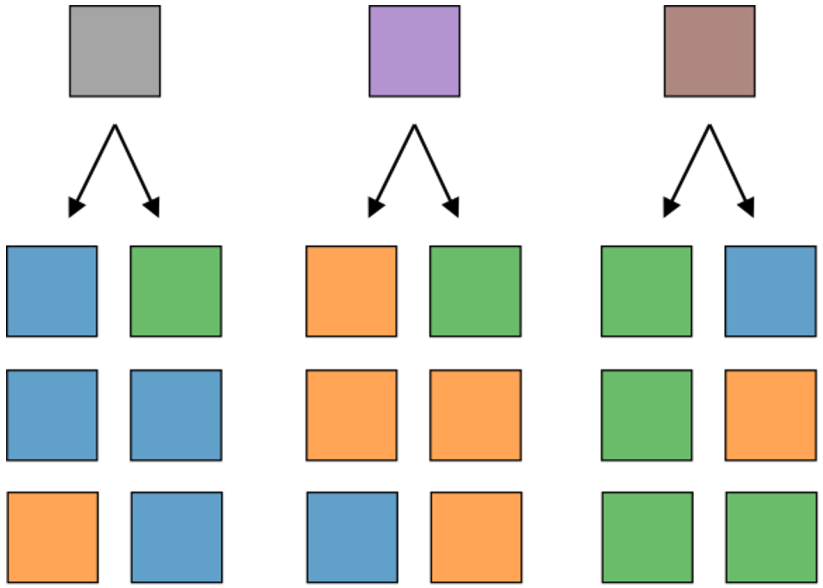
Semantic Synonyms



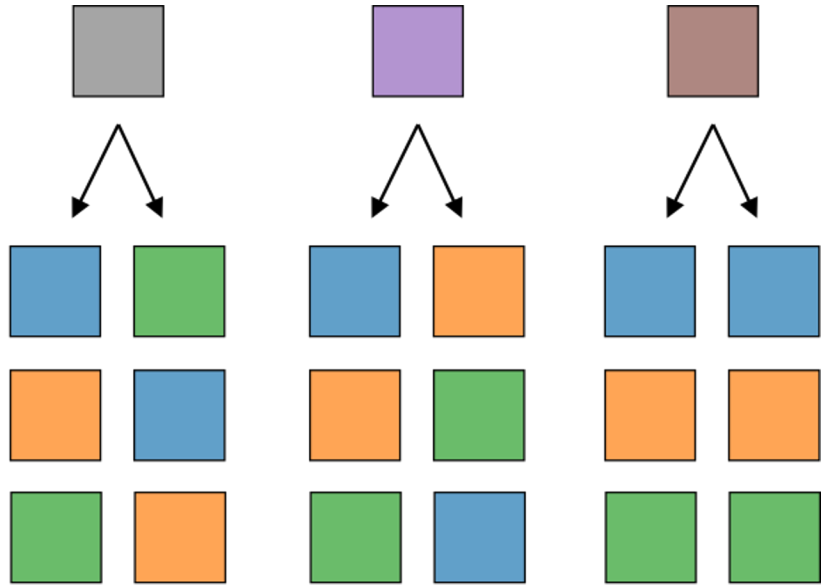
At each level, high-level features can be represented with m **distinct** s**-tuples** of low-level features (**synonyms**)

Random Hierarchy Model (RHM): random rules

Petrini, Cagnetta, Tomasini, Favero, MW arxiv 23'



(random)



(homogeneous)

- At each level of the hierarchy: Each v feature is decomposed on m strings of length s, uniformly chosen out of the v^s possible ones.

$$m \leq v^{s-1}$$

- Generates correlations between an isolated sub-feature and the High-level feature

Simple properties of the Random Hierarchy Model

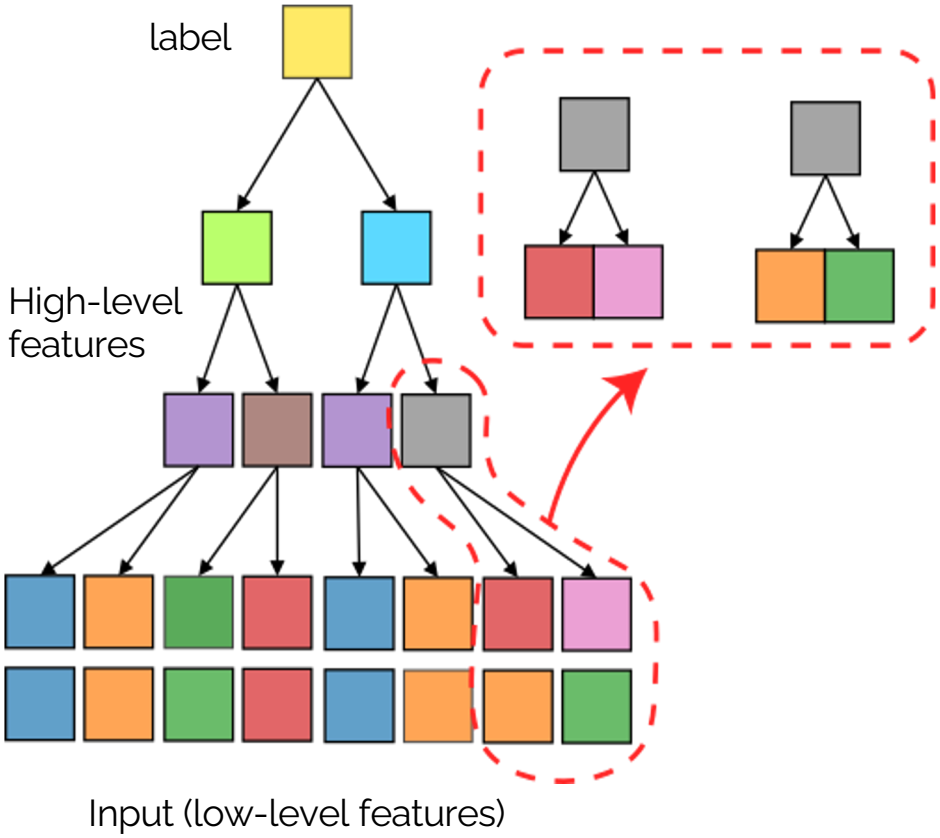
- Dimension of data $d=s^L$
- Number of data generated

$$P_{\max} = n_c m^{\frac{s^L - 1}{s-1}}$$

- All data generated in the maximal case $m=v^{s-1}$

- Invariant for exchange of **synonyms**

- random rules induces **computable correlations** between input features and label



Characteristic sample sizes of the RHM

Estimate of Information Theorical bound

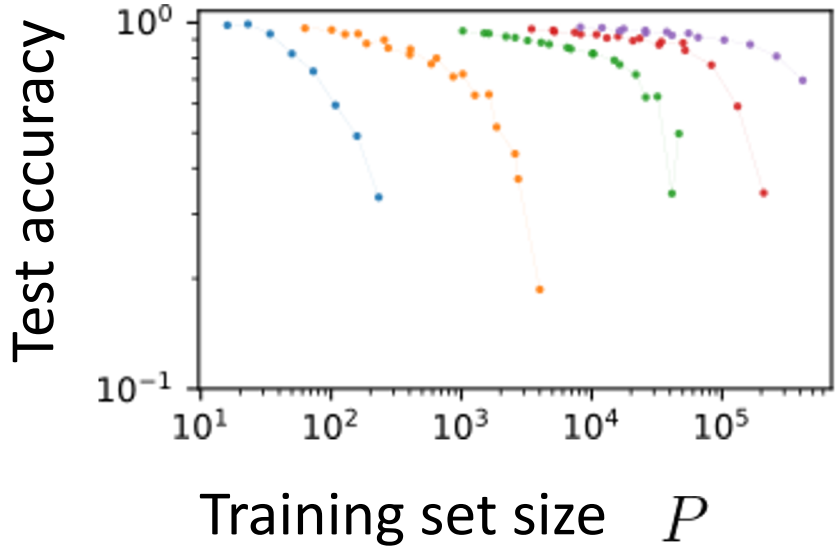
$P_{min} \leq Lv^s$ **logarithmic** in d

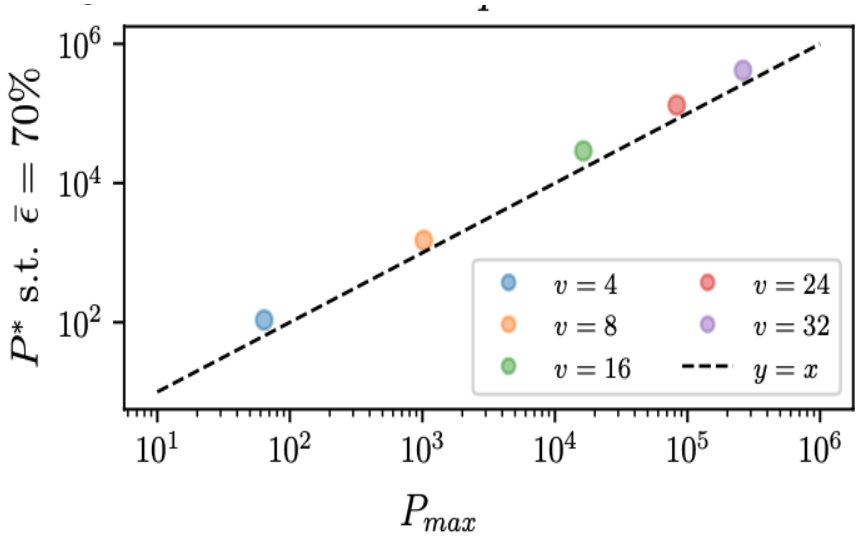
Sample Complexity?? Polynomial or exponential in d?

$P_{\max} = n_c m^{\frac{s^L - 1}{s-1}}$ **exponential** d

Shallow Fully-connected nets are cursed, as well as lazy deep CNNs

- maximal case $m=v^{s-1}$



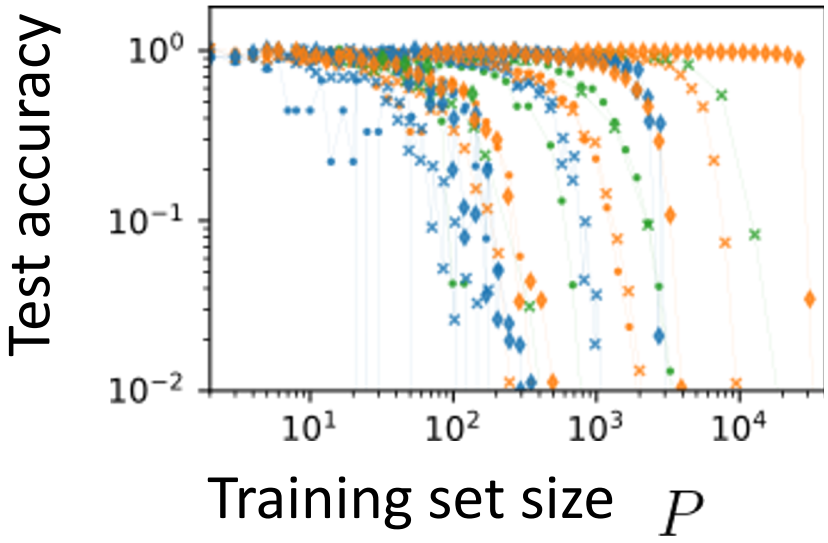


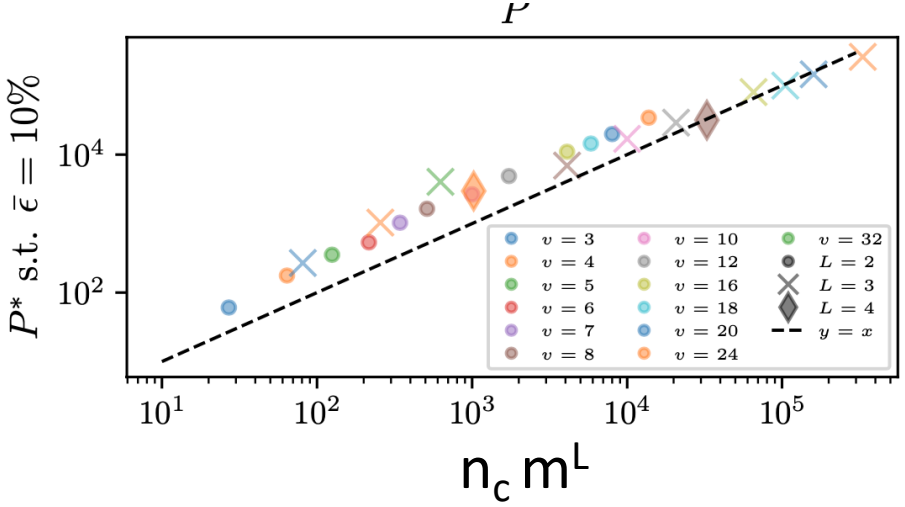
- $P^* \sim P_{max}$ sample complexity is exponential in $d=s^L$

- the same is observed for lazy deep CNNs

Sample complexity of deep CNNs

- Deep CNNs with matching architectures (L hidden layers, filter Size s)





$$P^* \sim n_c m^L$$

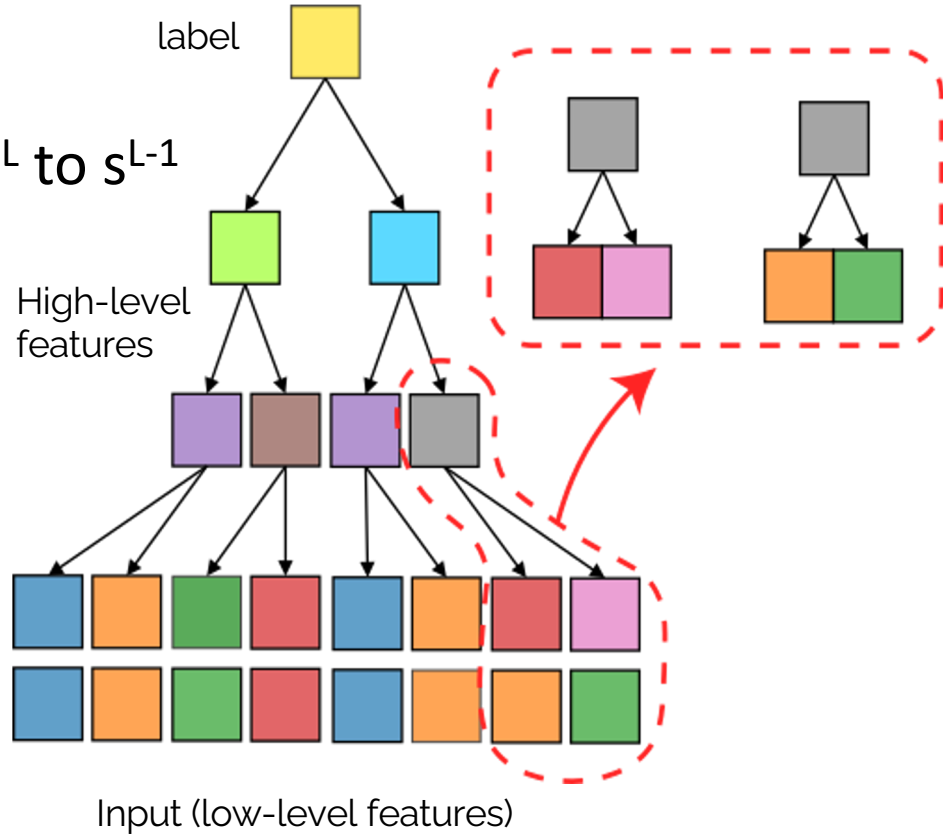
Polynomial in $d=s^L$

- Preliminary: same dependence on m found in deep FC nets, With prefactor exponential in L (beats the curse too)

How to learn the RHM?

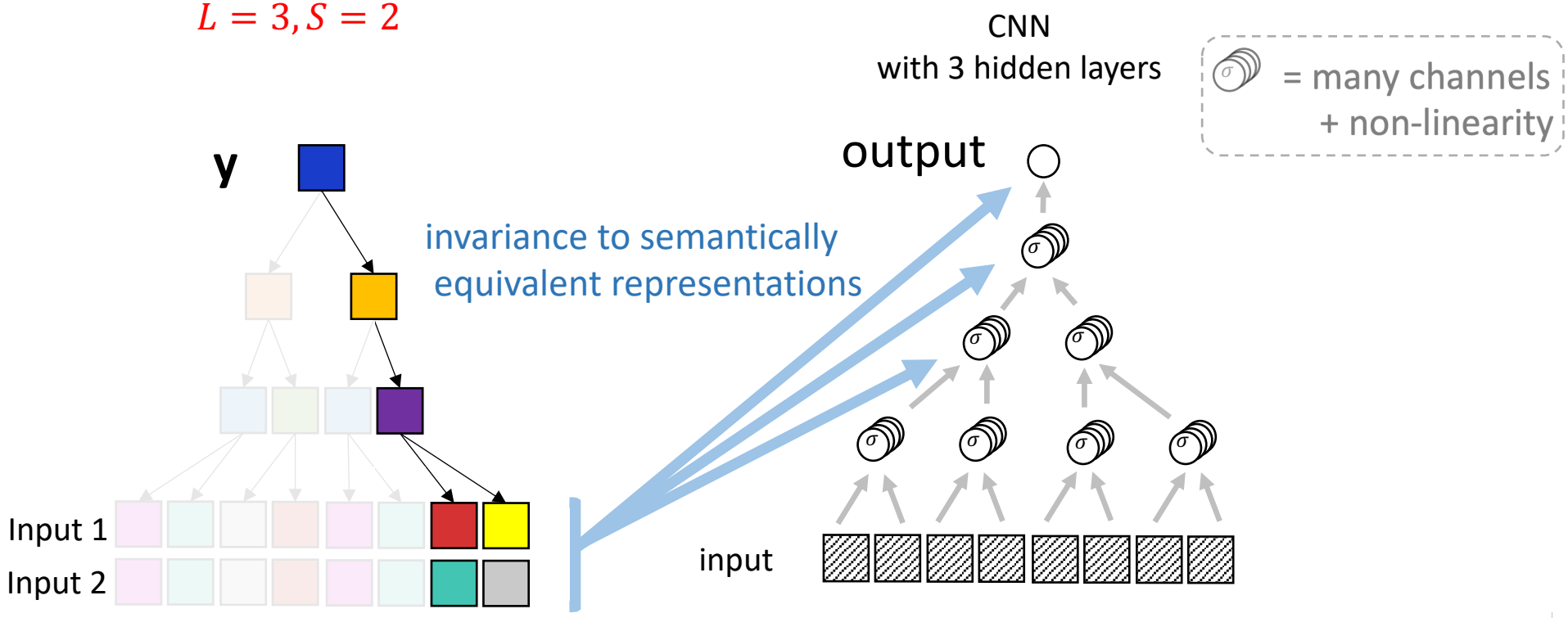
- Intuitive approach: learn the groups of synonyms of low-level patches
- Use the fact that they have the same correlation with the output
- Reduction of dimension from s^L to s^{L-1}
- Iterate L times

Is it the case for deep CNNs?



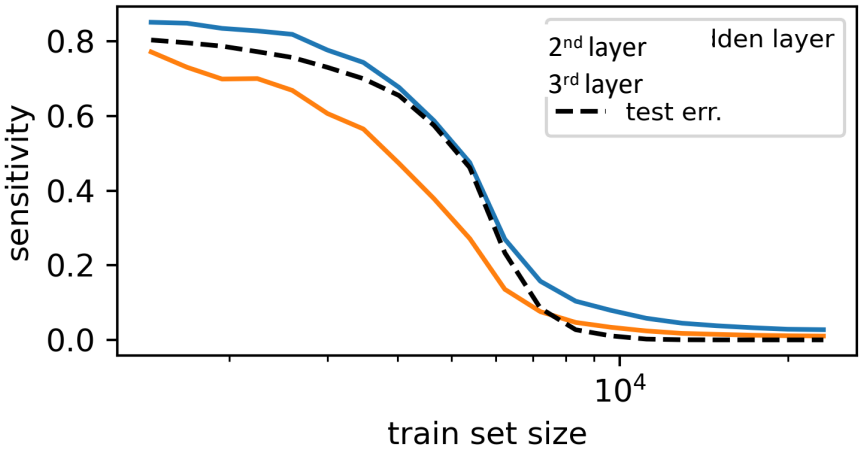
Invariant representation to synonyms emerges at P*

$L = 3, S = 2$

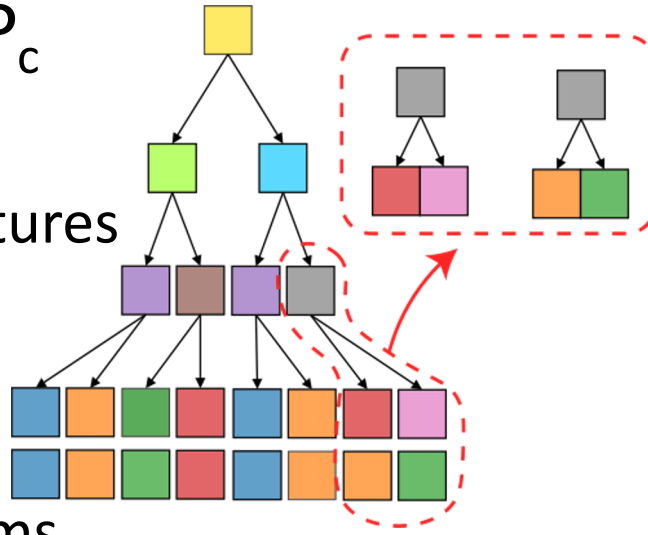


- Strong correlation between invariance and performance!

- Sensitivity to synonyms decreases at $P_c \sim P^*$



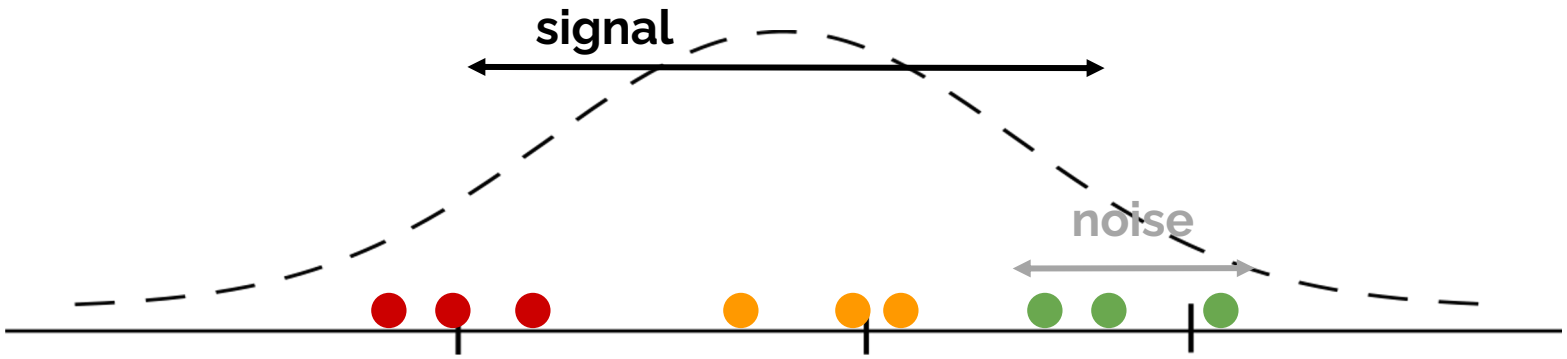
Simple argument for P_c



- Compute correlations between the sub-features at a given location and a class

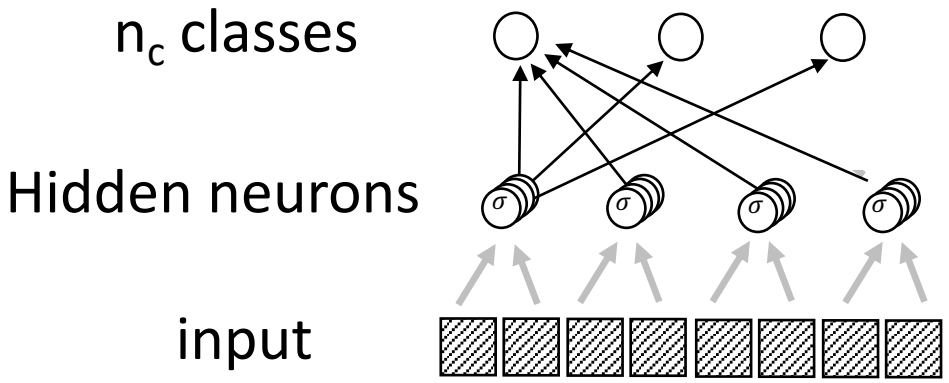
- Distribution of correlations has some Variance. It is the `signal': identical for synonyms

- For a finite training set, the estimation of correlation is noisy



- Signal = noice when $P = n_c m^L$ (precisely = P_c and P*)

Simple argument for P_c



- For such a one hidden layer and proper initialization/set-up, one step of GD leads to a representation with low sensitivity to synonyms if signal is larger than noise. In this set-up

$$P_c = n_c m^L$$

- Note (experts): P*=P_c not obvious. Sequential methods using clustering do slightly better than CNNs:

$$P^*_{sequential} = \sqrt{n_c} m^L$$

Conclusions

1. In the lazy regime, CNN beats the curse for local functions, not for hierarchical ones

2. Introduce the Random Hierarchical Model (RHM):

   Hierarchical task (supervised learning), Hierarchical data (SSL)

3. CNNs in feature regime can beat the curse when data hierarchical

4. Gives estimates of sample complexity assuming the hierarchical structure of data $P^* \sim n_c m^L$. Gives reasonable (crude) estimate, e.g. CIFAR L=3, m=5-20, 10 classes $P^* \in [10^3, 10^5]$

5. Predict that good performance is reached when stability to synonyms is achieved. Testable?