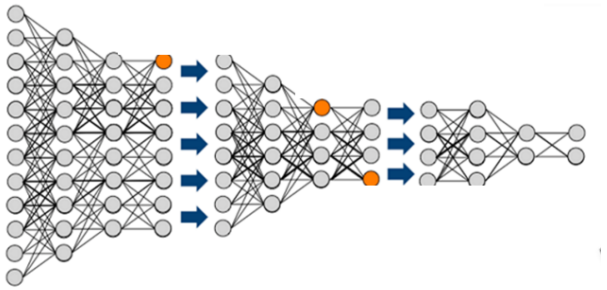
On the learnability of hierarchical data

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*



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

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

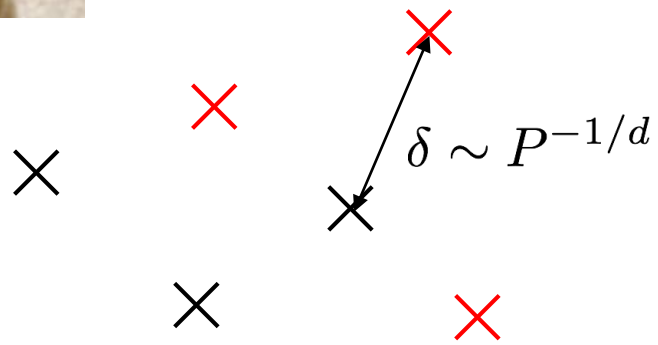
- Example: computer vision. Is it a cat or a dog?

- Learn from examples (supervised learning)

10^6

10^6

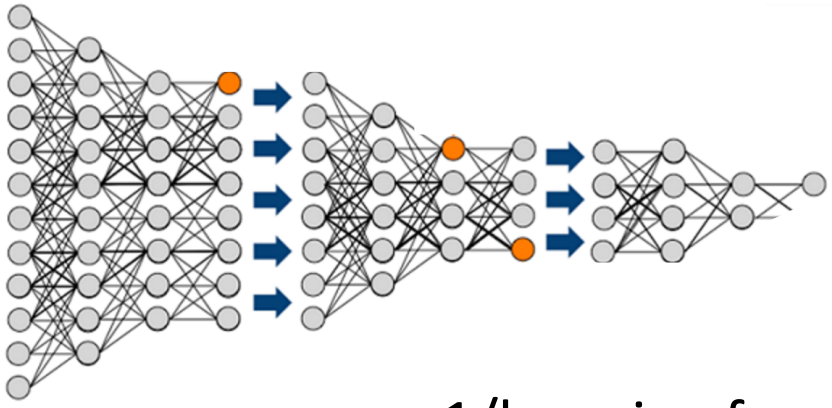
- Problem: no algorithms should work, curse of dimensionality

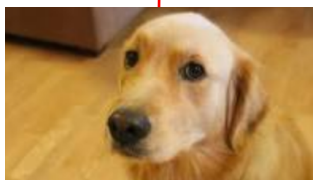


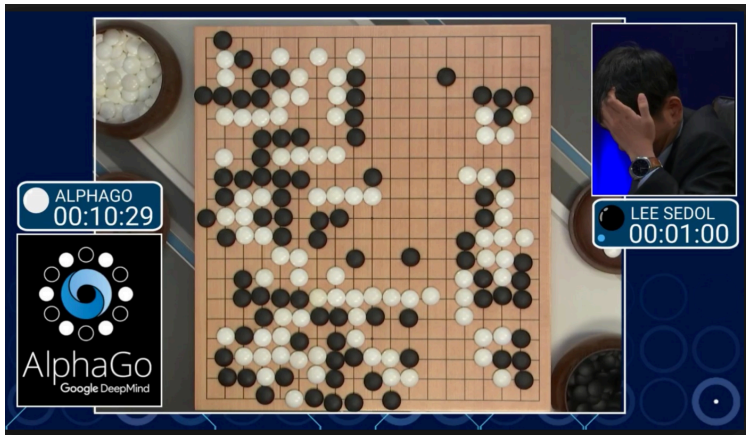
P points in d dimensions

- Data must be highly structured!?

Deep learning







>m

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

<-m

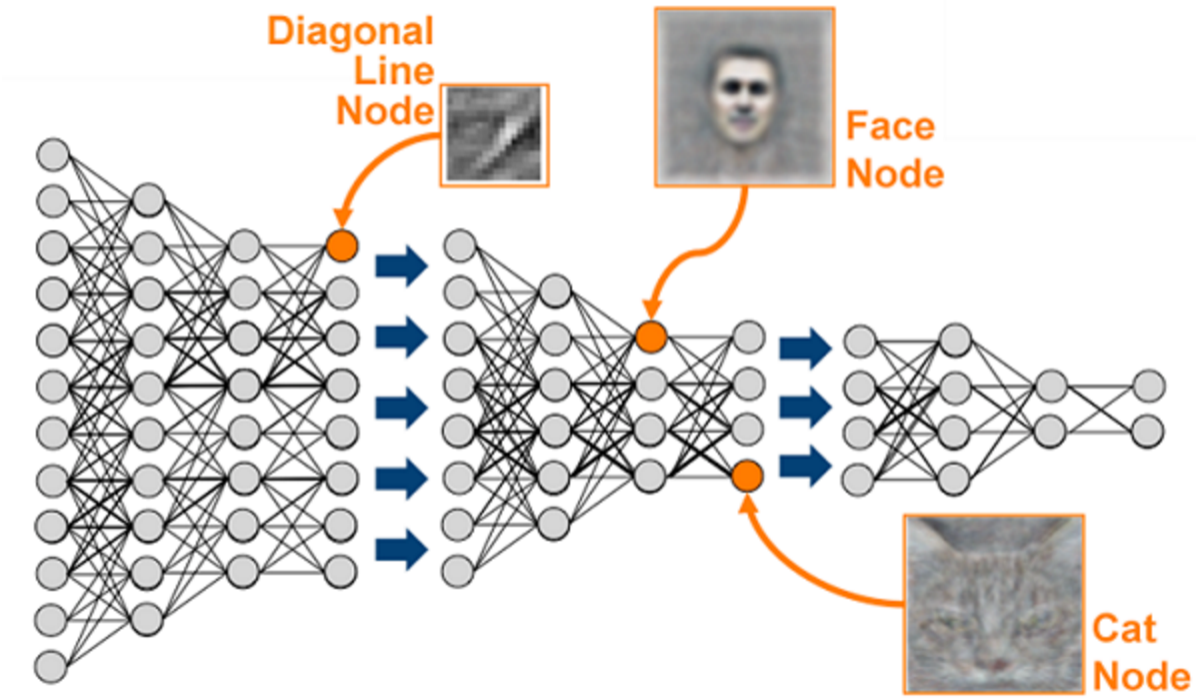
1/learning from example

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

- invariance to `semantic' Choice

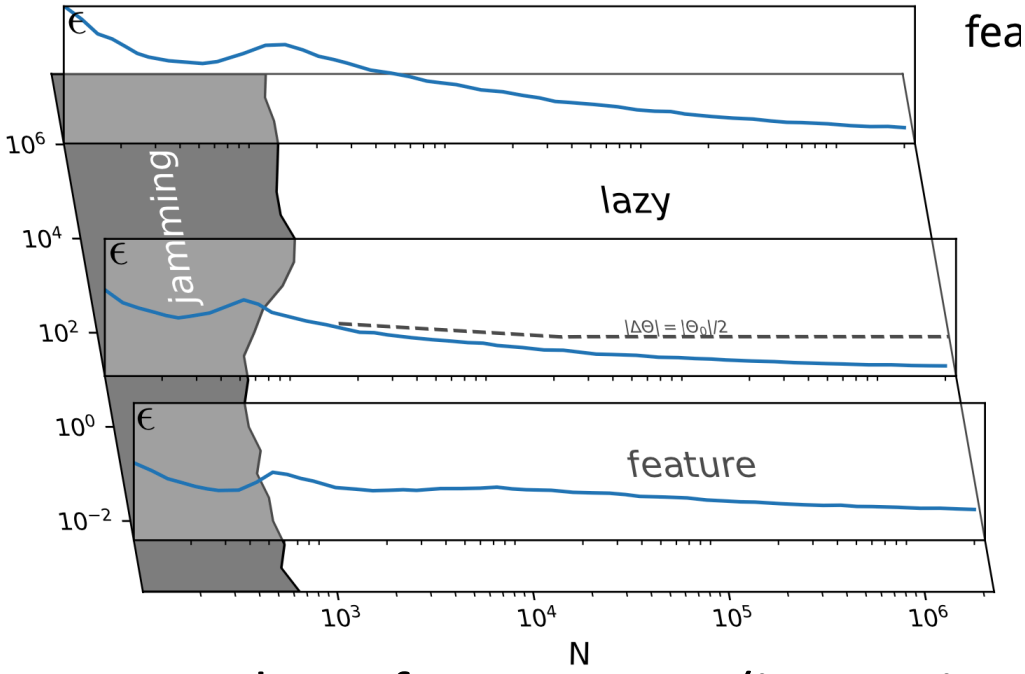
- How many data needed? Simple models

Overparameterization, training regimes: A phase diagram for deep learning

Geiger et al., 21'

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

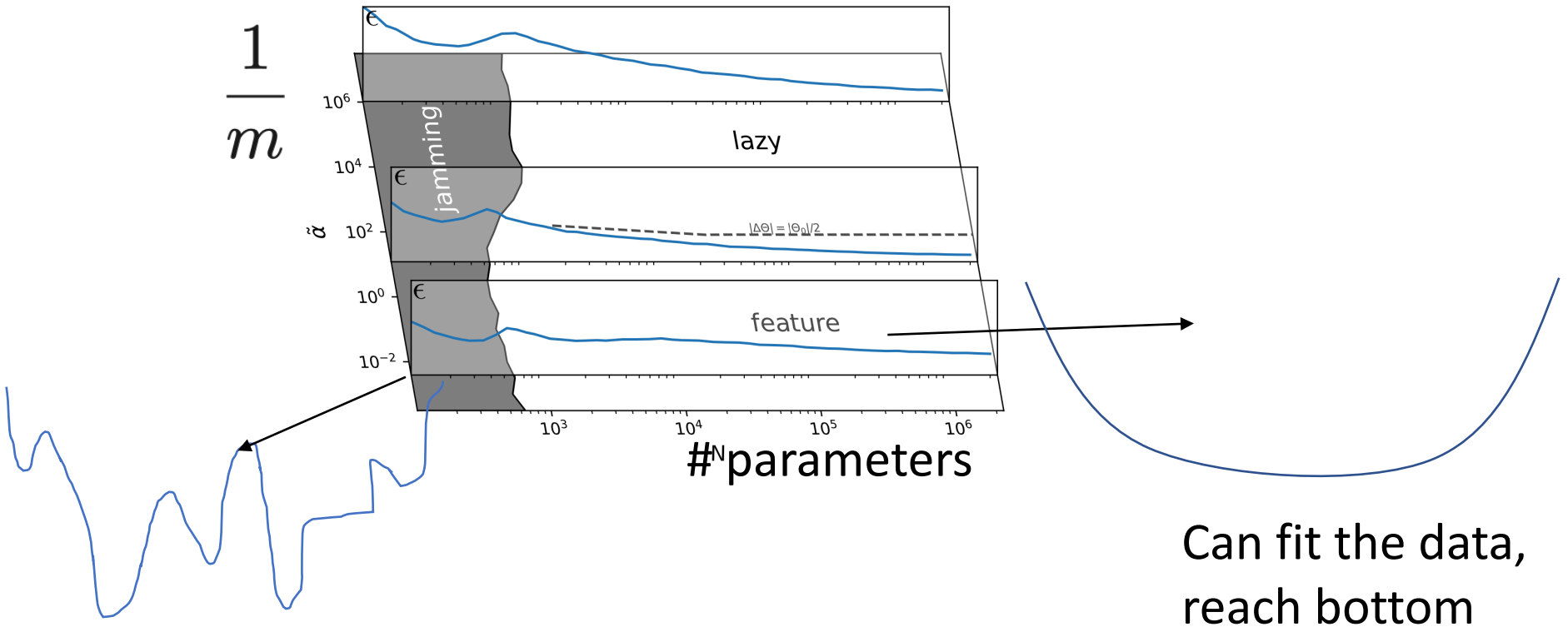
Inverse margin $\frac{1}{m}$



N: Number of parameters (increasing width)

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

Geiger et al., 2018



Cannot fit the data

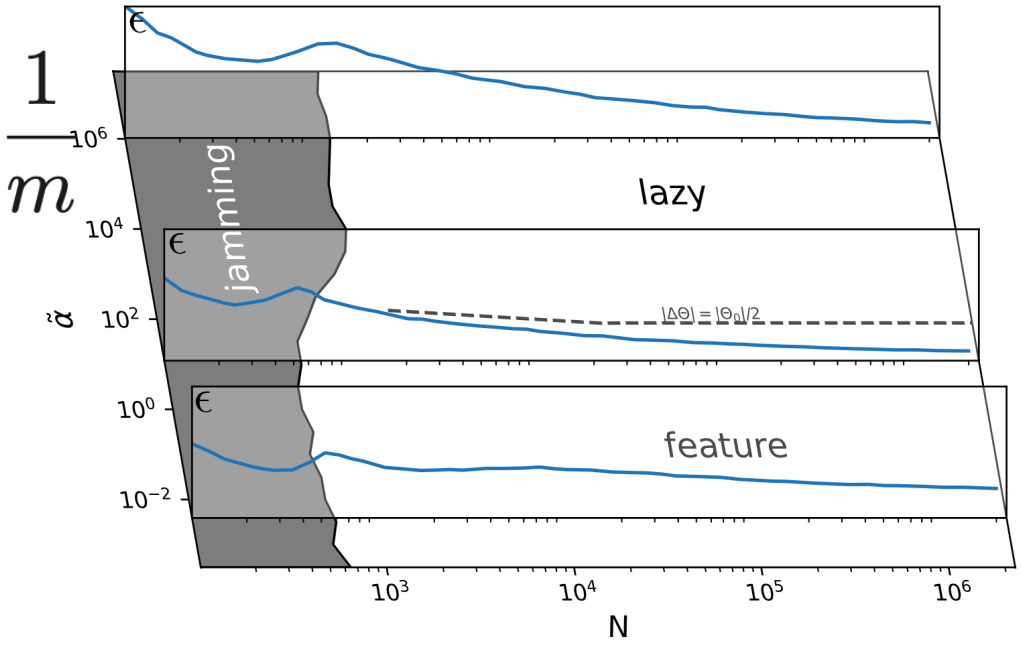
Can fit the data, reach bottom

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'

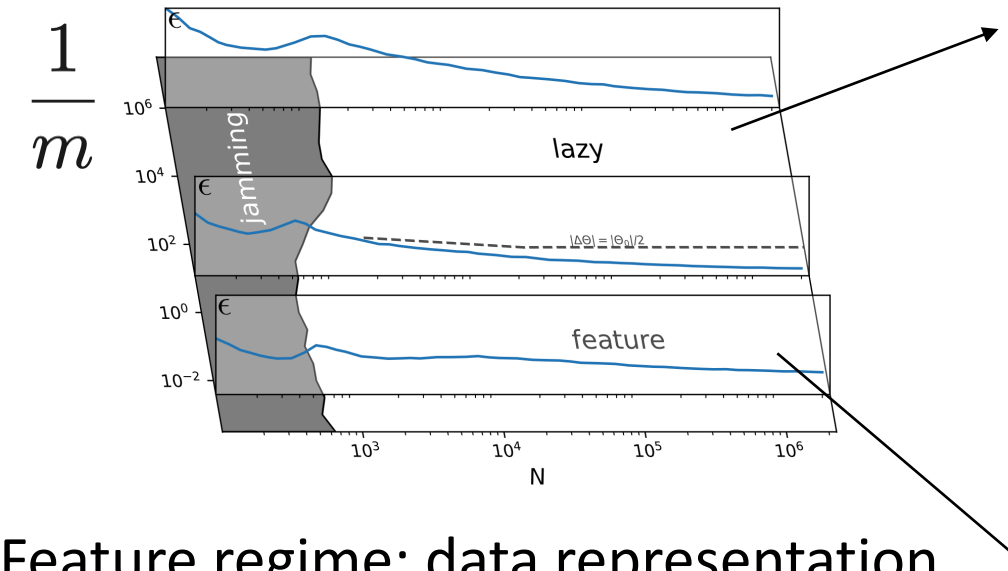


- Test error ε: probability to make a mistake
- No over-fitting as $N \to \infty$!!!
- Two interesting scaling regime: jamming and $N \to \infty$

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

Two limiting algorithms as $N \to \infty$

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

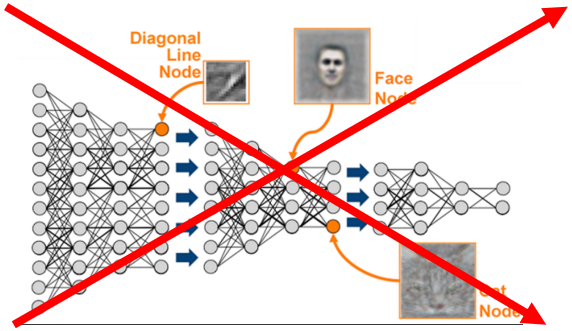


Lazy regime:

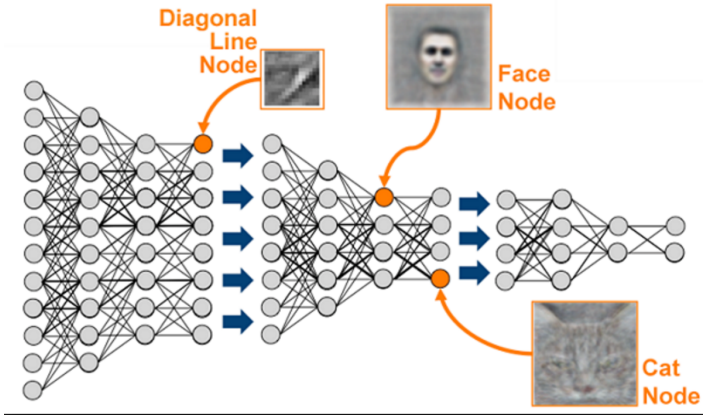
No features learnt!

tiny changes of weights

Sufficient to fit data



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

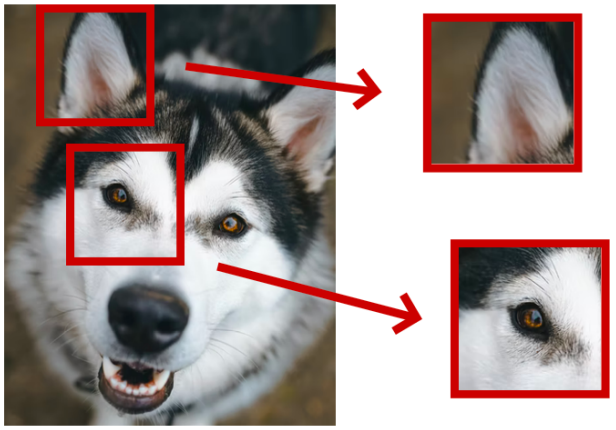


For modern architectures and images, Feature is better

What are CNNs good at, in both regimes?

Curse of dimensionality

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

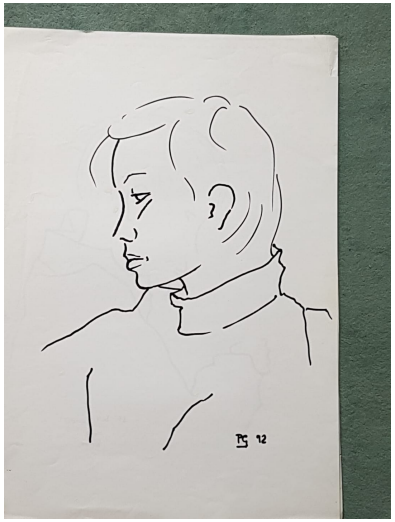


1. Locality: The task depends on the presence of local features

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...

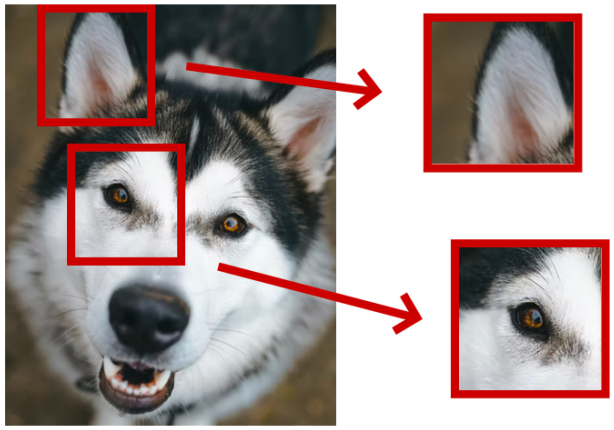
Presumably also relevant for text

CNNs in lazy regime adapt to the locality scale

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

- Regression task is local:

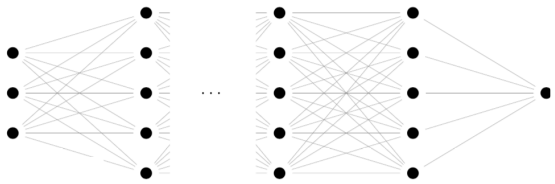
$$f^* = \sum_{i=1}^{d} g_i(\mathbf{x}_i)$$



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

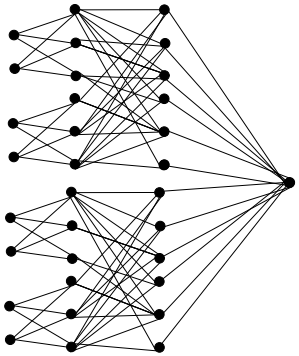
$\mathbf{x} \in \mathbb{R}^d$

- 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*

- 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

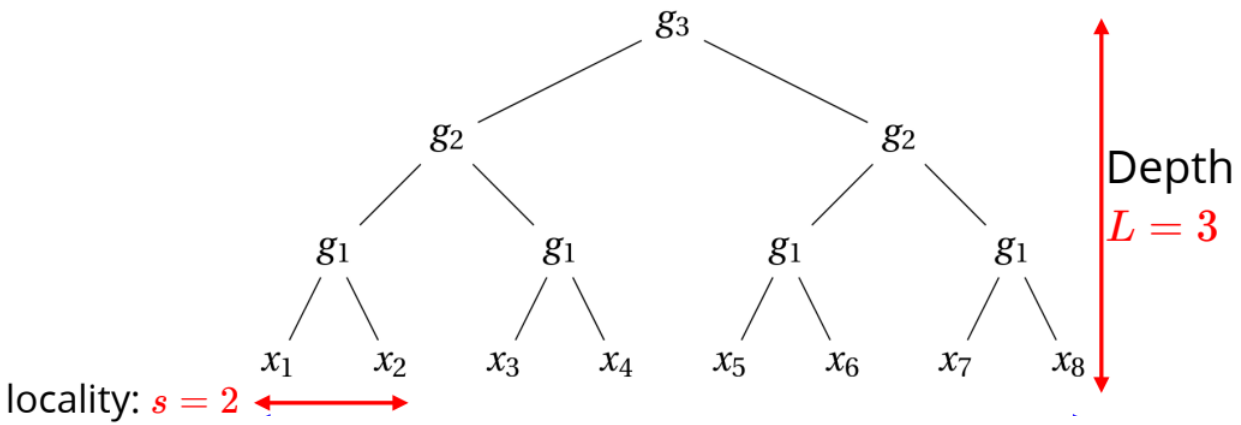
How # data needed depends on combinatorial nature of the task?

Hierarchical Generative Models

- Generates randomly P data and label (x,y), according to some frozen rules

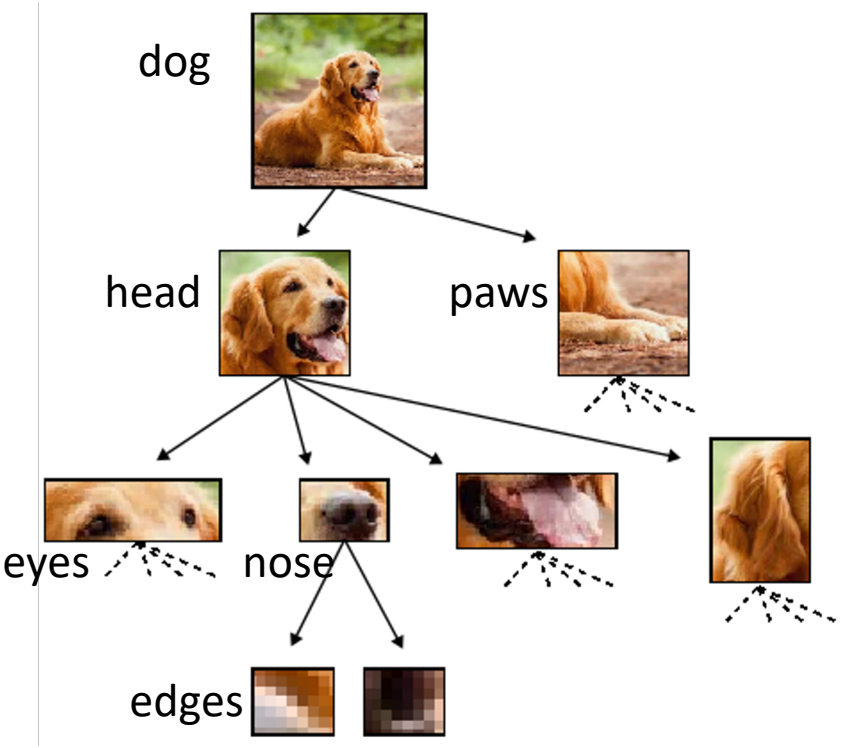
- Task y=f*(x) presents a hierachical structure

$$f^*(\boldsymbol{x}) = g_3(g_2(g_1(x_1, x_2), g_1(x_3, x_4)), g_2(g_1(x_5, x_6), g_1(x_7, x_8))).$$



- study how CNNs learn the task (supervised training)

Hierarchical Generative Models



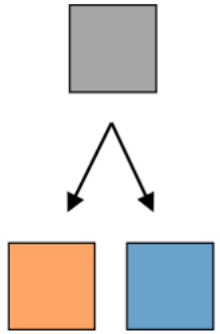
- n_c classes
- Inputs are generated via a hierarchy of L compositions
- One high-level feature corresponds to s sub-features
- Individual sub-features can be shared. finite vocabulary of size v
- A high-level feature can be decomposed into m strings of s subfeatures

Hierarchical Generative Models

class

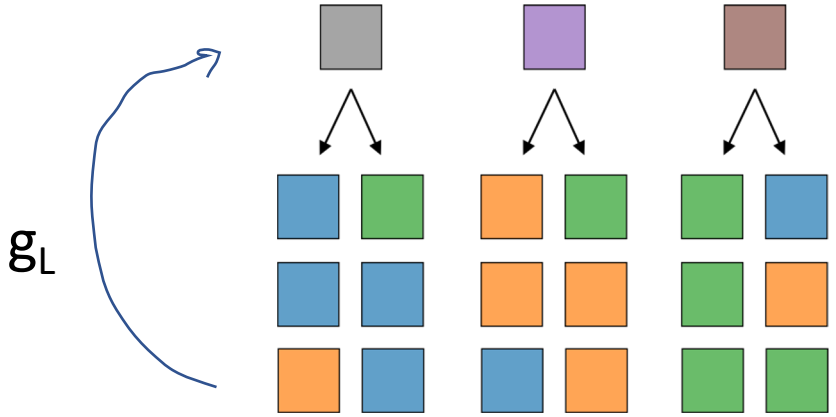
- choose randomly a class among n_c ones (fixes y)

Hierarchical Generative Models



- choose randomly a class among n_c ones

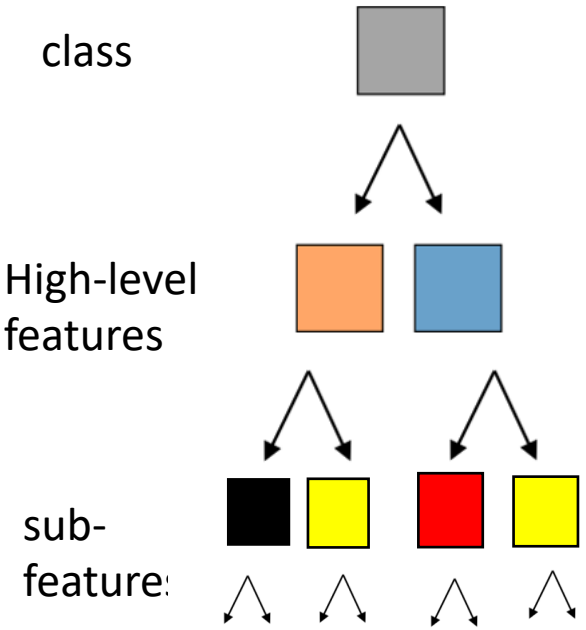
- Consider the m possible ways to represent this class by strings of s features. Choose one randomly



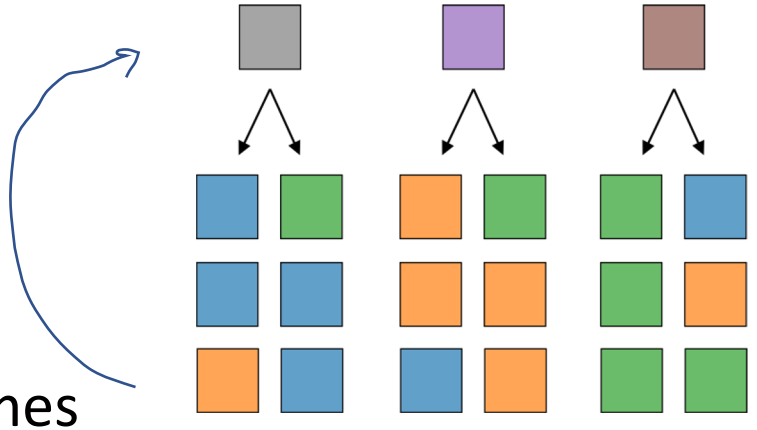
m=3,s=2,v=3

Frozen rules

Hierarchical Generative Models

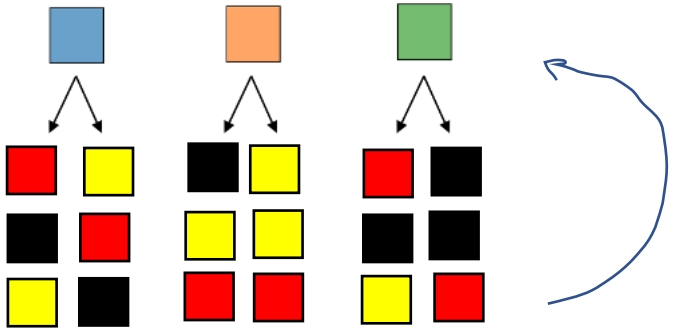


- choose randomly a class among n_c ones
- Consider the m possible ways to represent this class by strings of s features. Choose one randomly



m=3,s=2,v=3

- Repeat the process iteratively L times

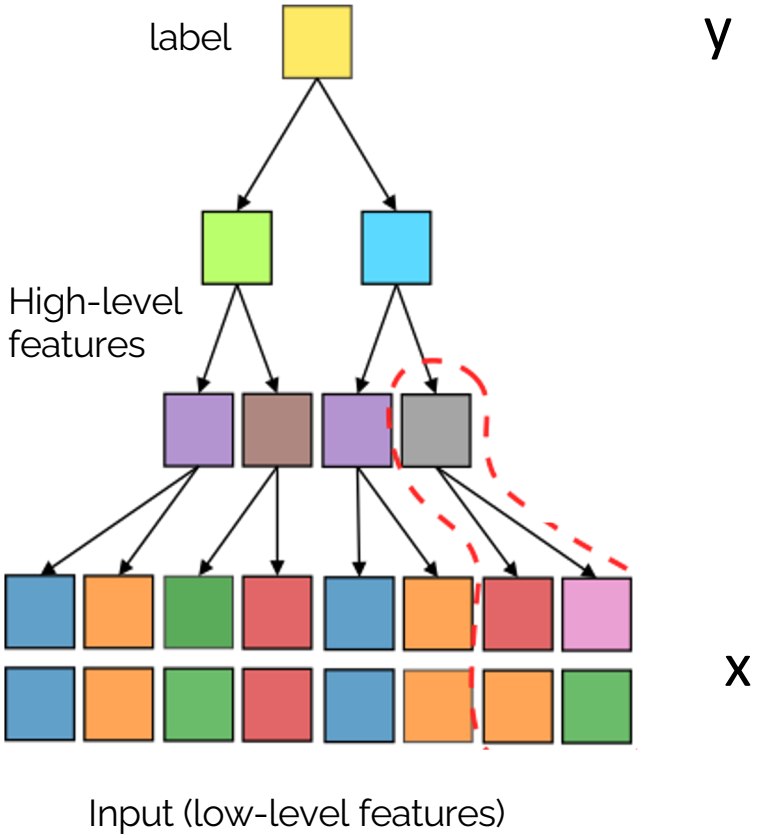


$$m \leq v^{s-1}$$

Number and encoding of inputs

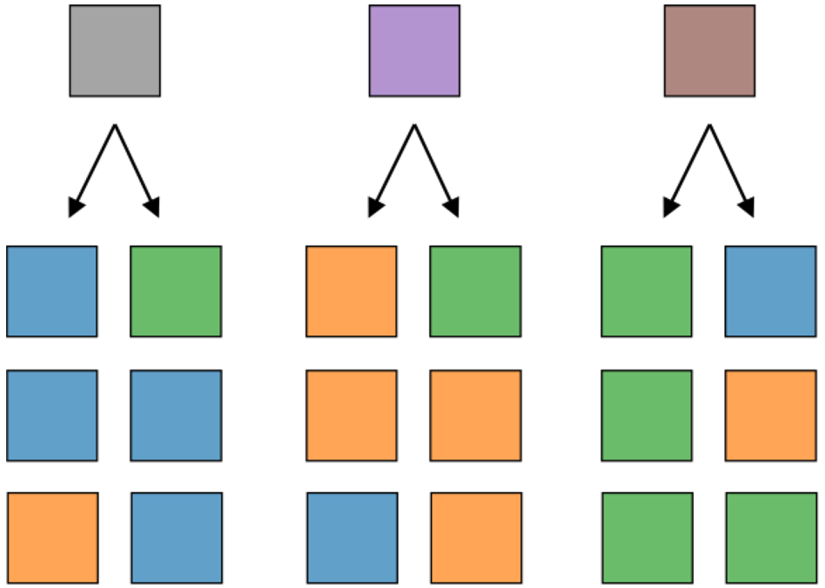
- Dimension of input data $d=s^L$
- Input: one-hot encoding of features
- Number of data generated

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



Random Hierarchy Model (RHM): frozen rules are random

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



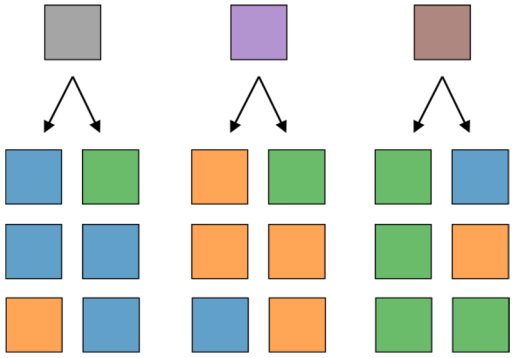
(random)

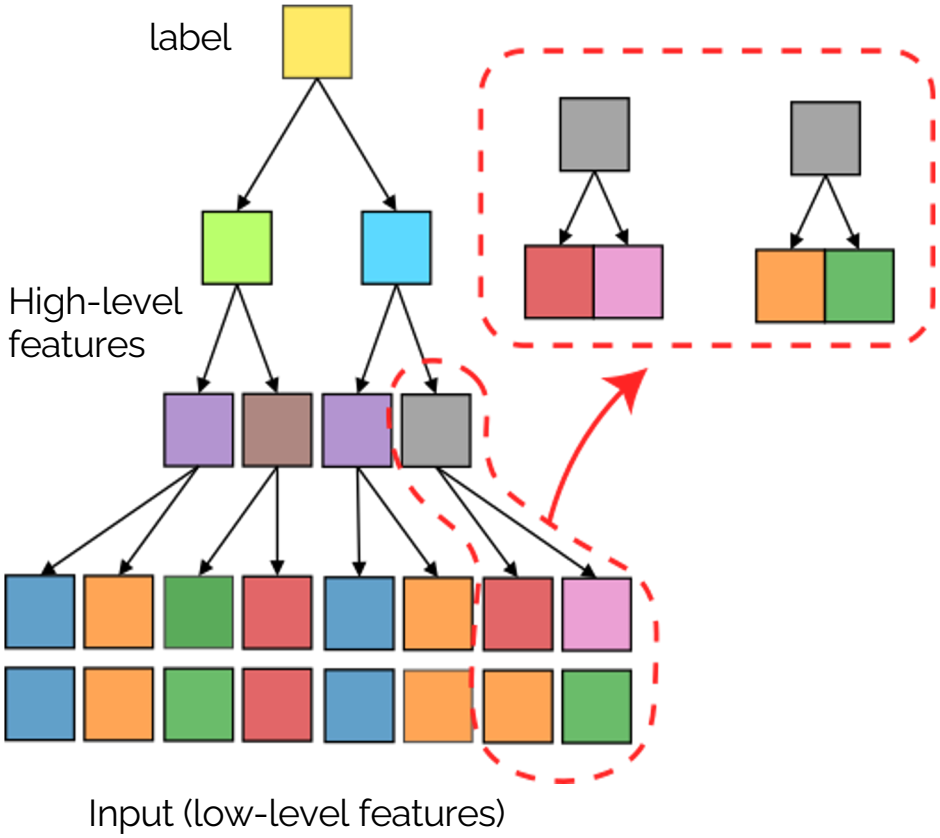
- At each level of the hierarchy: Each v feature is decomposed on m strings of length s, uniformly chosen out of the v^s possible ones.

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

Key properties of the Random Hierarchy Model

- Semantic synonyms

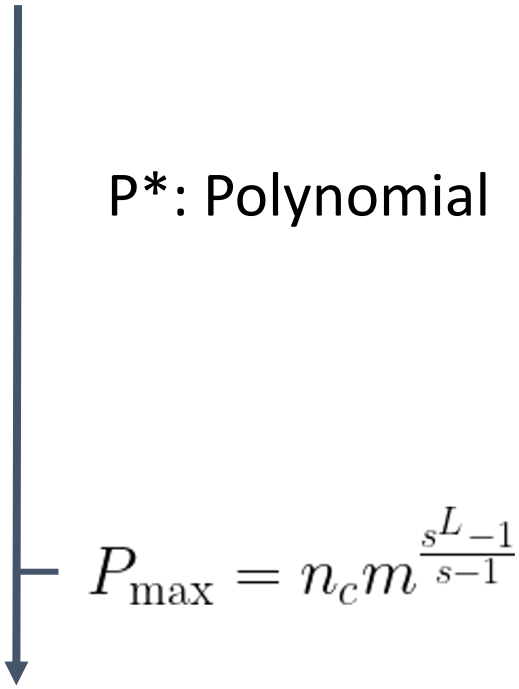




- Label Invariant for exchange of **synonyms**

- random rules induces **computable correlations** between each pixel, or groups of pixels, and label

data needed to learn such combinatorial tasks? (sample complexity P*)

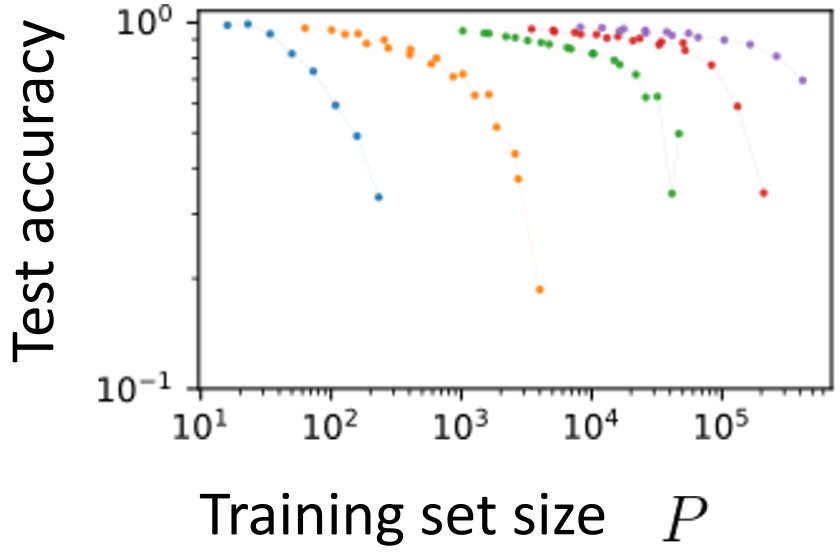


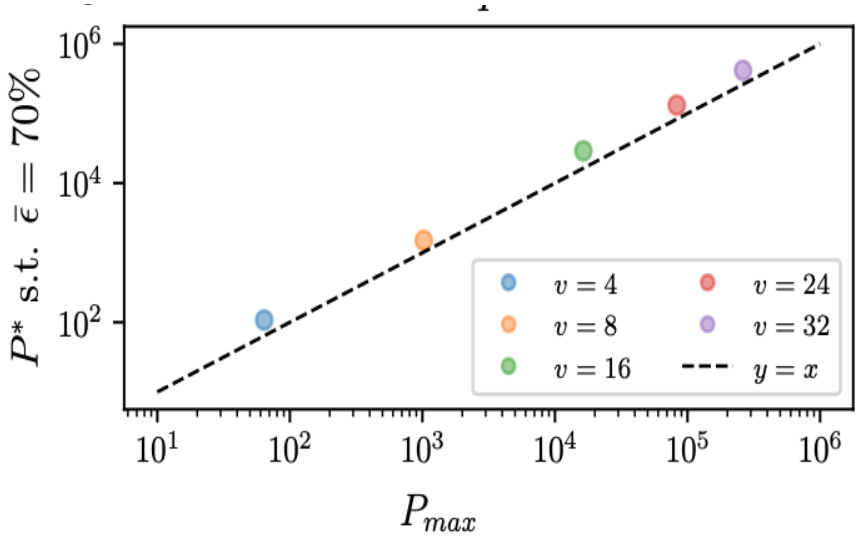
P*: Polynomial or exponential in d?

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

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

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



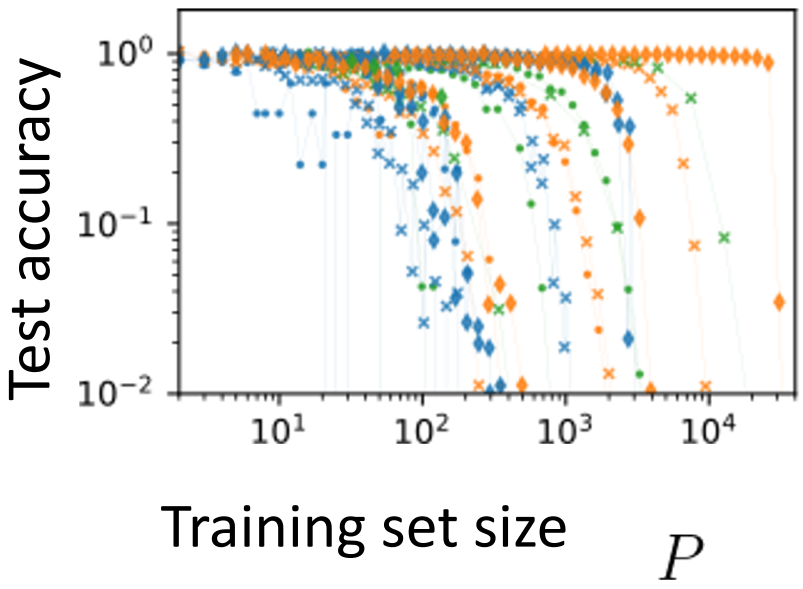


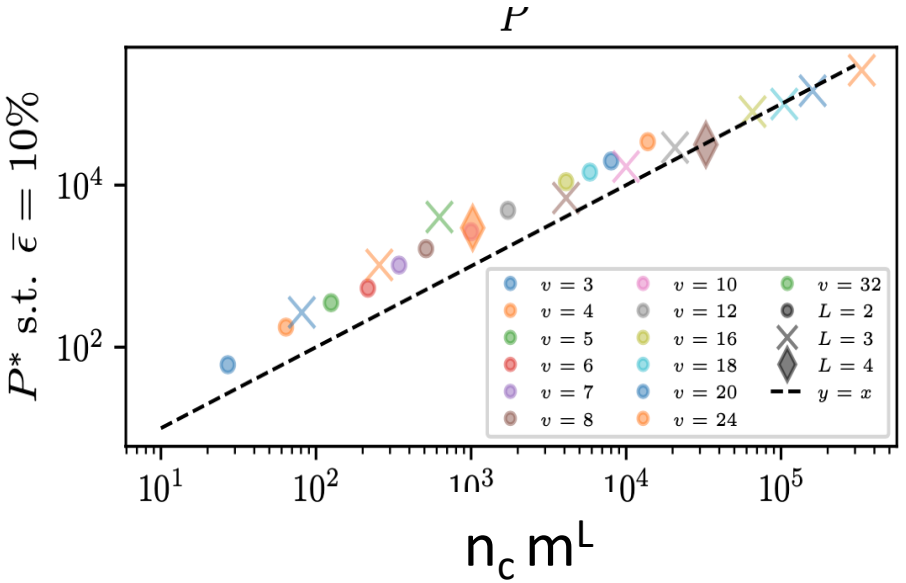
- $P^* \sim P_{max}$ sample complexity is exponential in $d=s^L$

- the same is observed for lazy deep CNNs

Sample complexity of deep CNNs

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





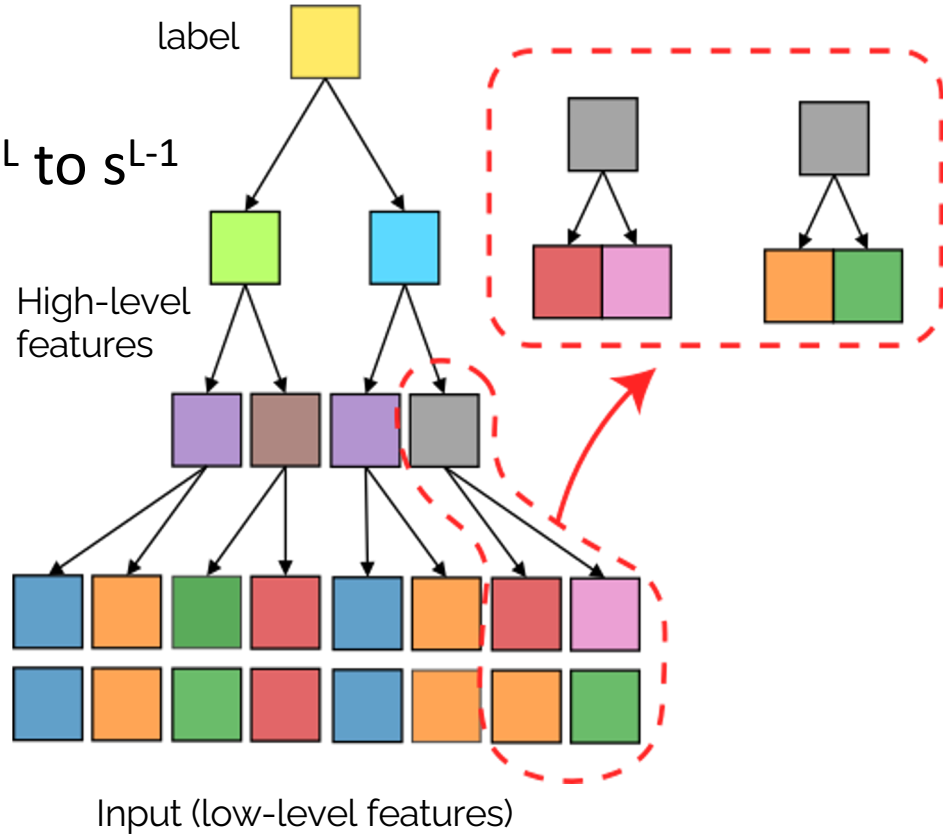
$$P^* \sim n_c m^L$$

Polynomial in $d=s^L$!! Beats the curse

How to learn the RHM?

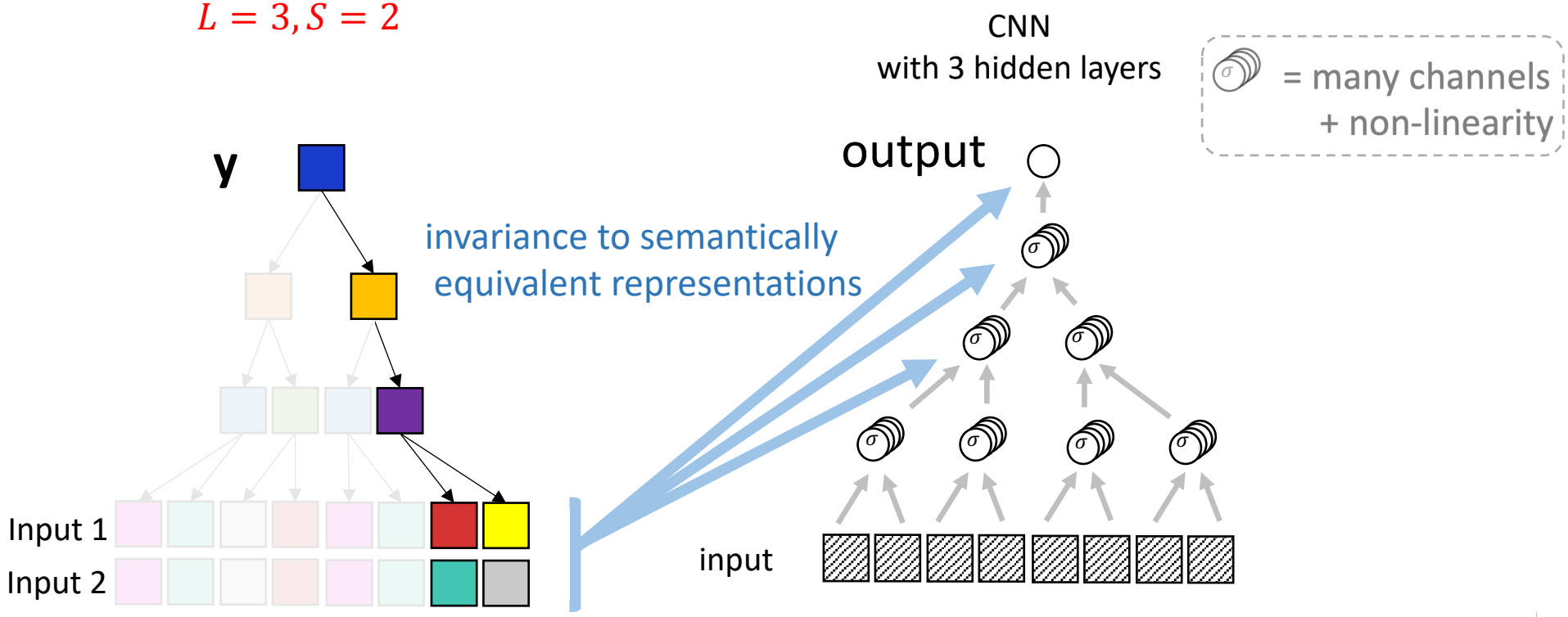
- Intuitive approach: learn the groups of synonyms of low-level patches
- Use the fact that they have the same correlation with the output
- Reduction of dimension from s^L to s^{L-1}
- Iterate L times

Is it the case for deep CNNs?



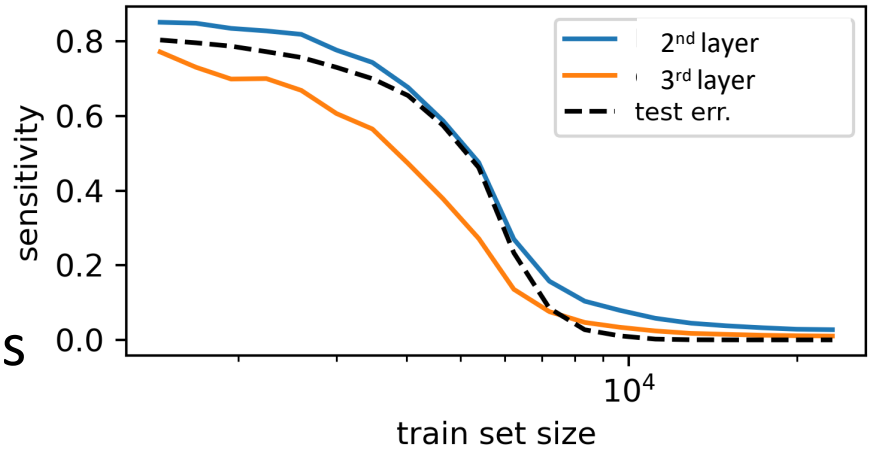
Invariant representation to synonyms emerges at P*

$L = 3, S = 2$

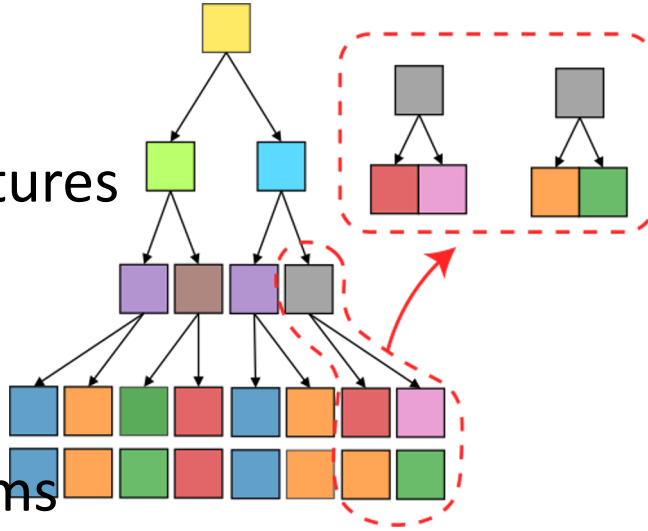


- Strong correlation between invariance and performance!

- Sensitivity to synonyms decreases at P*



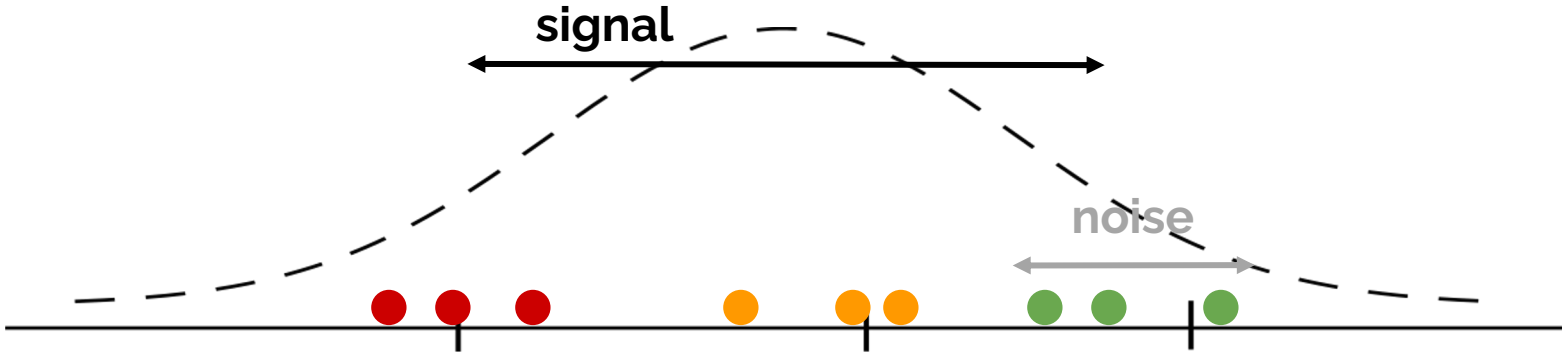
Simple argument for P*



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

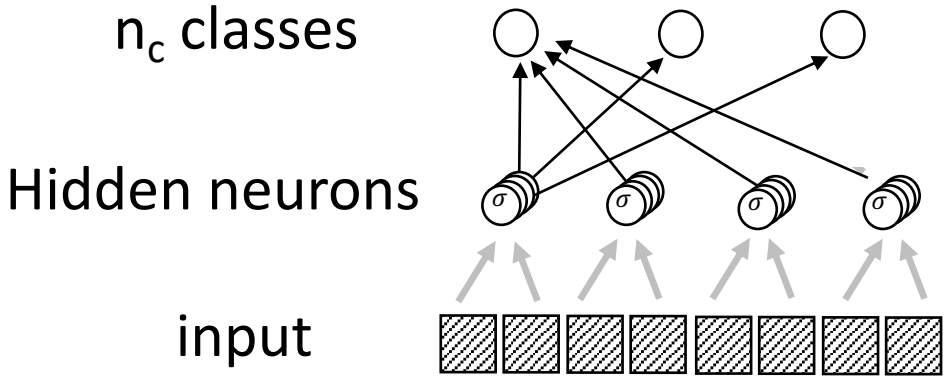
- Distribution of correlations has some Variance. It is the `signal': identical for synonyms

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



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

Simple argument for P^*



- GD sensitive to these correlations
- one step of GD -> representation with low sensitivity to synonyms if signal > noise. Invariance thus appears for

$$P^* = n_c m^L$$

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 distribution (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. Test in real data?