The Information Bottleneck Theory of [simple] Deep Learning

Statistical Physics and Machine Learning - Back Together
Cargese, 2018

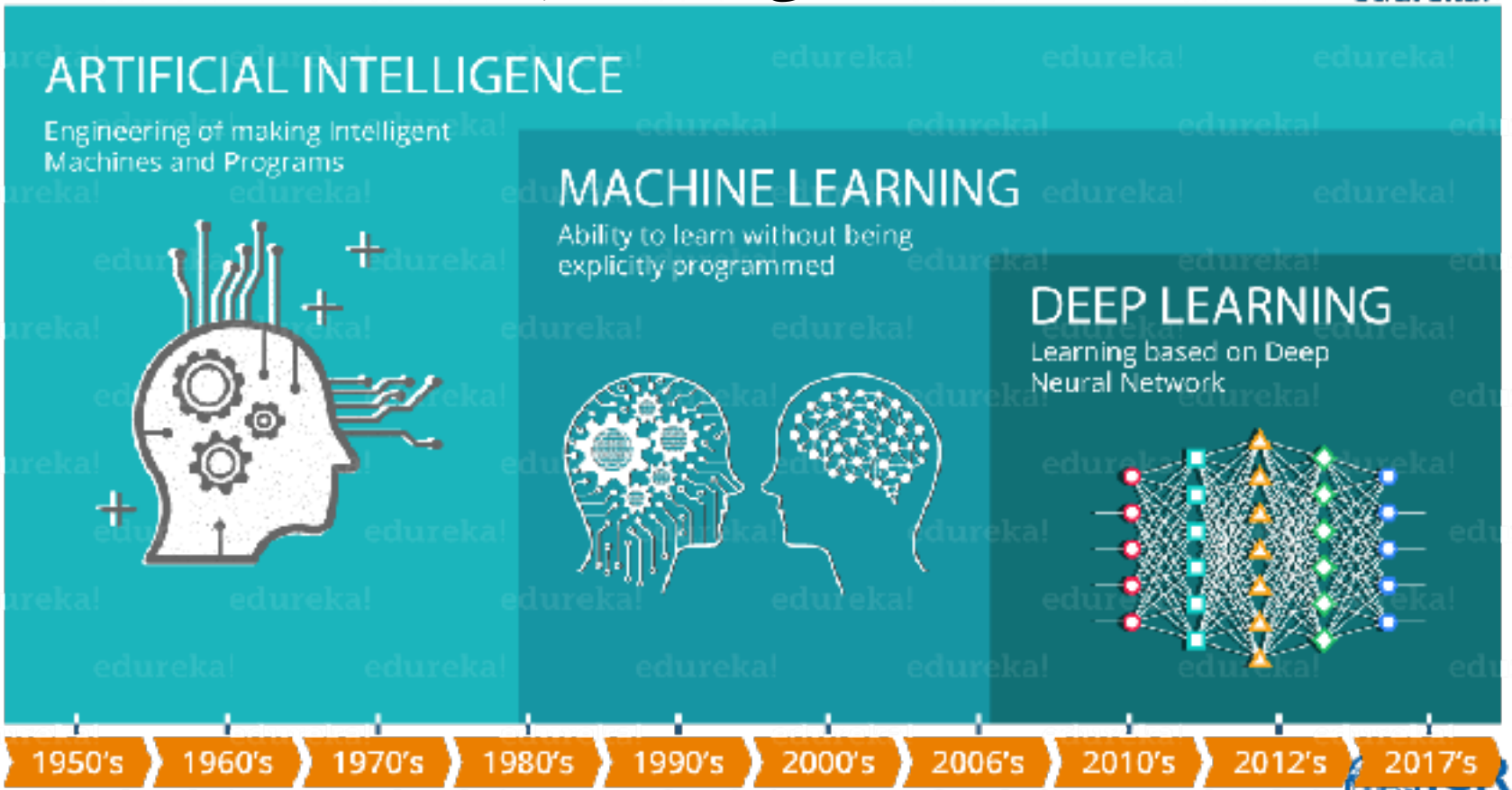
Naftali Tishby

School of Engineering and Computer Science

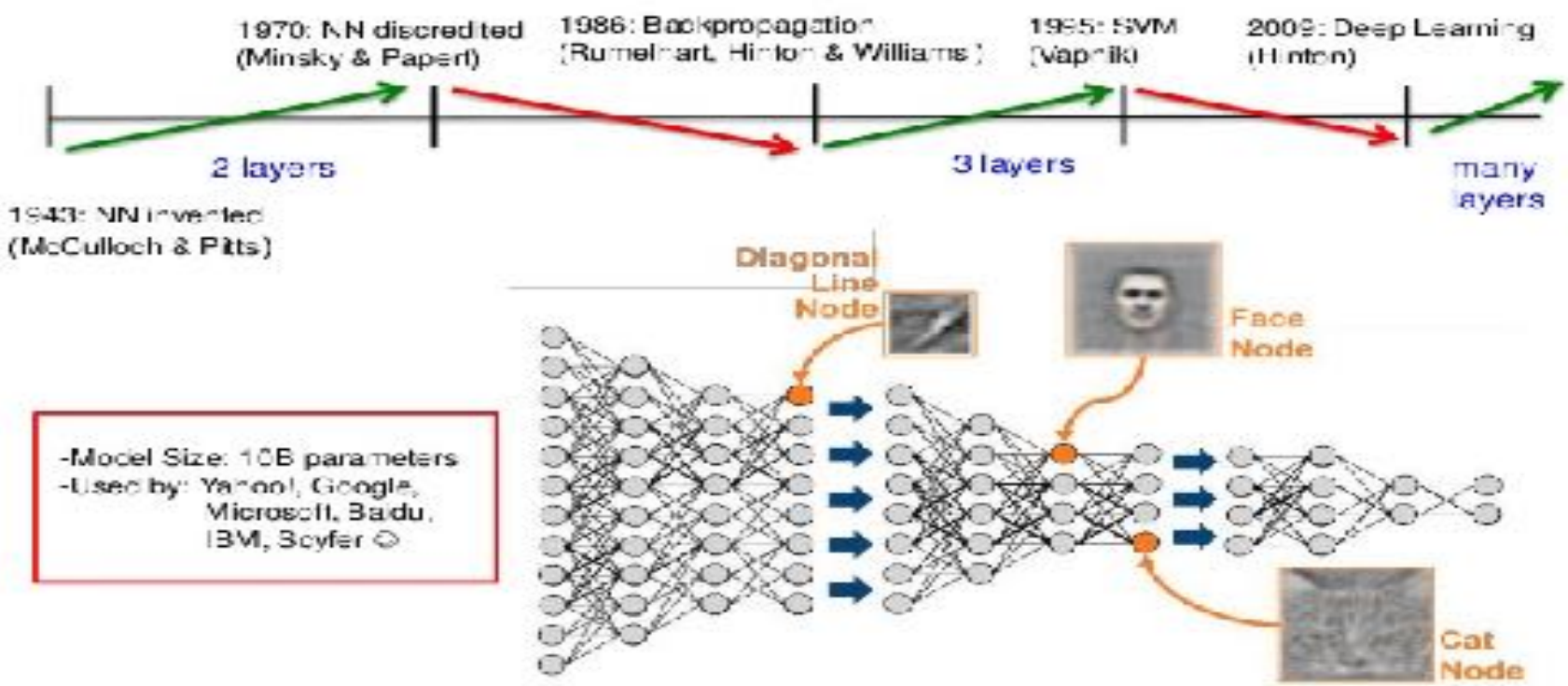
The Edmond & Lily Safra Center for Brain Sciences

Noga Zaslavsky
Ravid Schwartz-Ziv

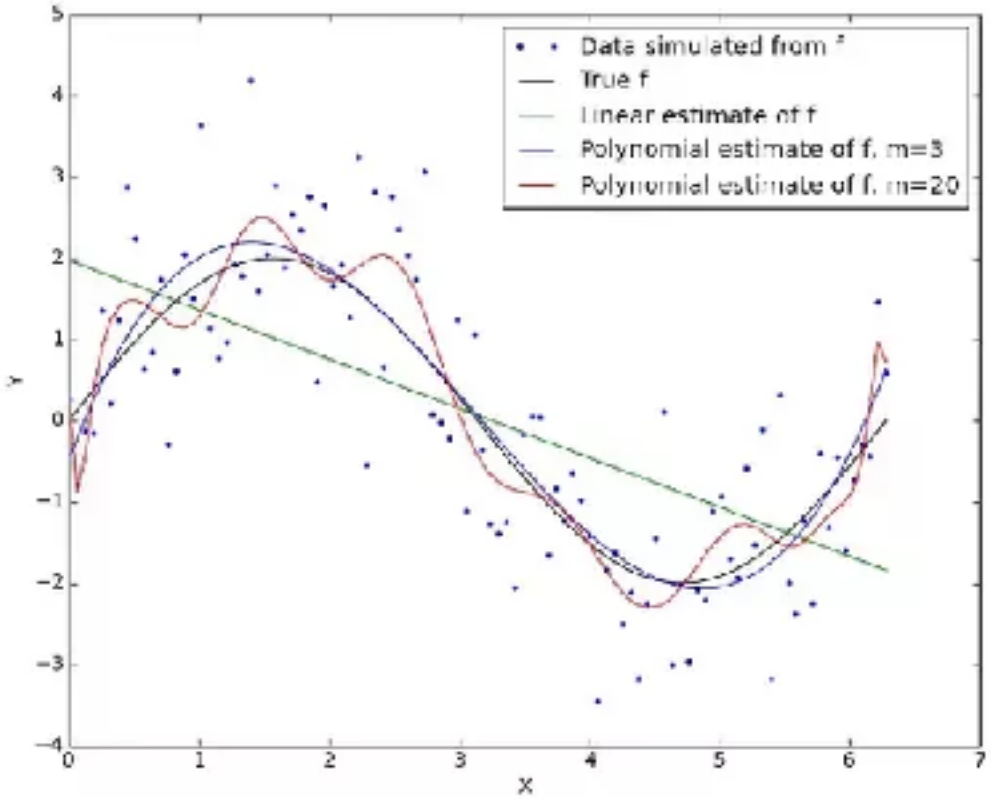
The Deep Learning revolution



Deep Learning: Neural-Nets strike back



Is it more than high-dimensional curve fitting?



- Classical [least-squares] regression
- Main issue: **generalization**
- How to avoid **over-fitting?**
- # of data points ~ # of parameters
- Choose the right hypotheses class!

We begin to obtain some new understanding…

We combine 3 different ingredients:

- **Rethinking Statistical Learning Theory**
  - Worse case PAC bounds → typical case architecture free bounds…
  - From expressivity/Hypothesis class → Input Compression bounds
- **Information Theory** (statistical mechanics…)
  - Large scale learning – *Typical* input patterns
  - → Concentration of the Mutual Information values
  - → Huge parameter space - exponentially many optimal solutions
- **Stochastic dynamics of the training process**
  - Convergence of SGD to locally-Gibbs (Max Entropy) weight distribution
  - → The mechanism of representation compression in Deep Learning
  - → Convergence times – explains the benefit of the hidden layers

The match between DL and the Information Bottleneck

Main results:

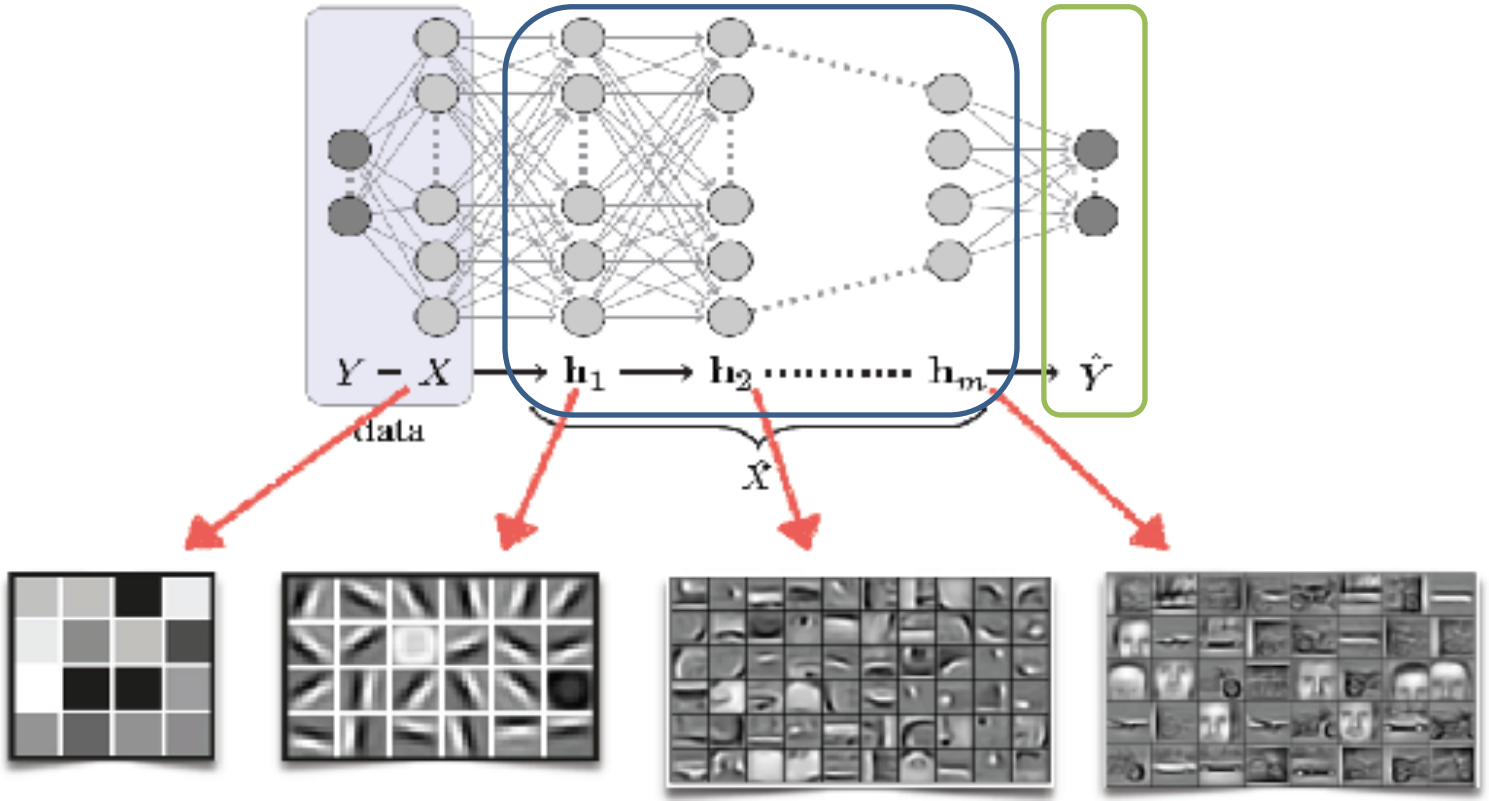
- **Optimality: The layers converge to the [finite-sample] IB bound**
  - DL can achieve optimal (model free, rule dependent) sample complexity-accuracy tradeoff
  - Through the diffusion/noisy phase of the Stochastic Gradient Descent optimization
  - Which compresses the representation by "forgetting" irrelevant details
- **Benefit of the Hidden Layers**
  - The benefit is mostly computational – boosting the compression!
  - The location of the optimal layers is determined by the problem
- **Interpretability**
  - Full layers can have clear – problem specific - meaning, NOT single neurons (in general)!
- **Design principles**
  - DL is good for stochastic, compressible rules.
  - Layers final position is related to critical points of the Information Bottleneck
  - Concrete predictions on the layers organization and weights meaning

Known issues & important reservations

Objections to the theory:

- **Information estimation [requires quantization or noise, not scalable? …]**
  - Not needed for training, only as a tool for understating!
  - Requires finite precision or quantization – CORRECT!
  - Mutual Information values concentrate & become MORE stable the larger the problem!
- **Compression/Information loss not necessary [ResNets, RevNets,i-RevNets,…]**
  - Compression comes from unit saturation, not seen with RelU's (Saxe 2018) – WRONG!
  - Indeed, good generalization can be achieved without apparent layer compression.
  - Similar to the classical physics paradox of reversible microscopic laws & Macroscopic Entropy increase…
  - No "forgetting" of non-informative features (really?)
- **Stochastic Gradients not needed [no convergence to local Gibbs distribution]**
  - Good generalization achieved without stochastic gradients in INFINITE TIMES! How?
  - Convergence to Gibbs (MaxEnt) distribution is only local (in each layer).
  - The benefits of the stochasticity is dynamical (computational), but also in saving training data!
  - There is important INFORMATION in the mini-batch fluctuations!
- **Is the IB bound relevant?**
  - It actually gives concreate predictions and interpretation of the layers & weights.
  - May explain biological neural network organization… our ultimate motivation.

Deep Neural Nets and Information Theory ??



Some Information Theory basics

- The KL-distribution divergence:

for any two distributions p(x) & q(x) over X:

$$D[p(x)\,\|\,q(x)] = \sum_x p(x)\log\frac{p(x)}{q(x)} \geq 0$$

- The Mutual Information:

for any two random variables, X , Y:

$$I(X;Y) = D[p(x,y)\,\|\,p(x)\,p(y)] = D[p(x|y)\,\|\,p(x)] = D[p(y|x)\,\|\,p(y)] = \mathrm{H(X)} - \mathrm{H(X\,|\,Y)}$$

- Data Processing Inequality (DPI) & Invariance:

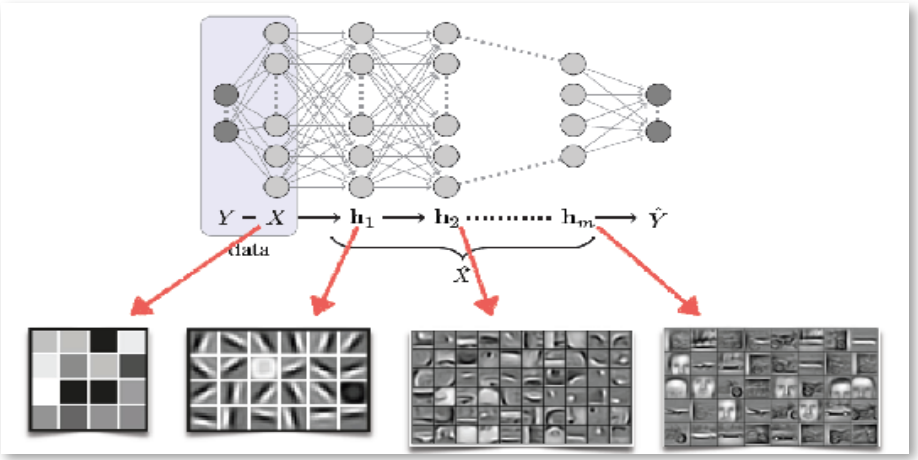
for any Markov chain: $X \rightarrow Y \rightarrow Z$:

$$I(X;Y) \geq I(X;Z)$$

Reparametrization Invariance, for invertible ϕ,ψ :

$$I(X;Y) = I(\phi(X);\psi(Y))$$

What do the DNN Layers represent?

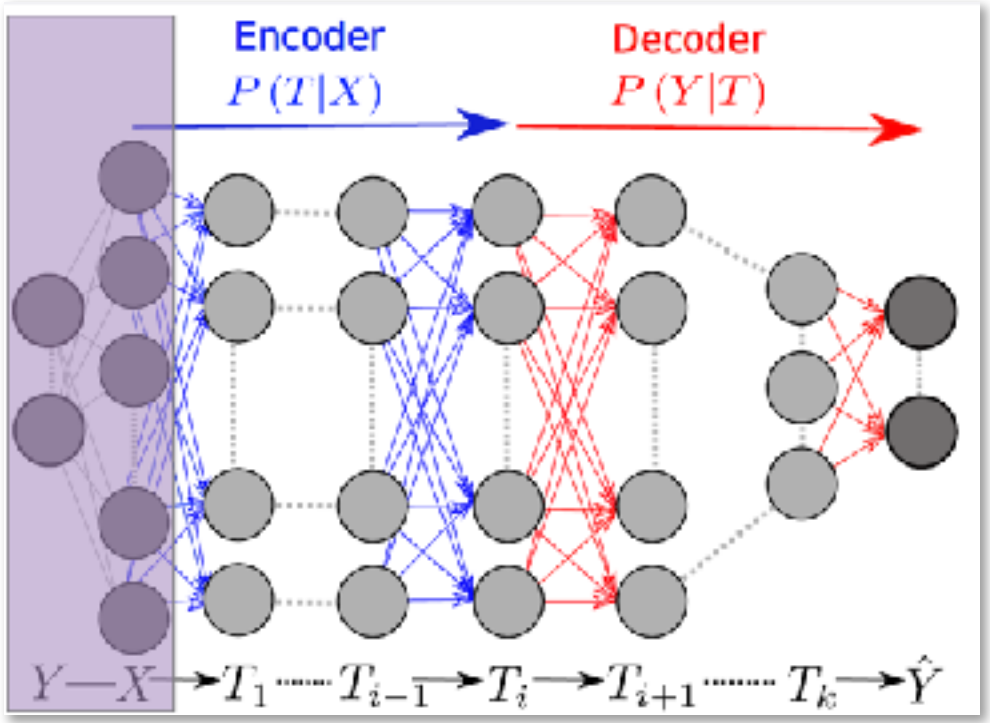


Data Processing Inequalities:

$$H(X) \geq I(X;h_i) \geq I(X;h_{i+1}) \geq I(X;h_{i+2}) \geq \ldots$$
$$I(X;Y) \geq I(h_i;Y) \geq I(h_{i+1};Y) \geq I(h_{i+2};Y) \geq \ldots$$

- A Markov chain of topologically distinct [soft] **partitions** of the input variable X.
- Successive Refinement of Relevant Information
- Individual neurons can be easily "scrambled" within each layer

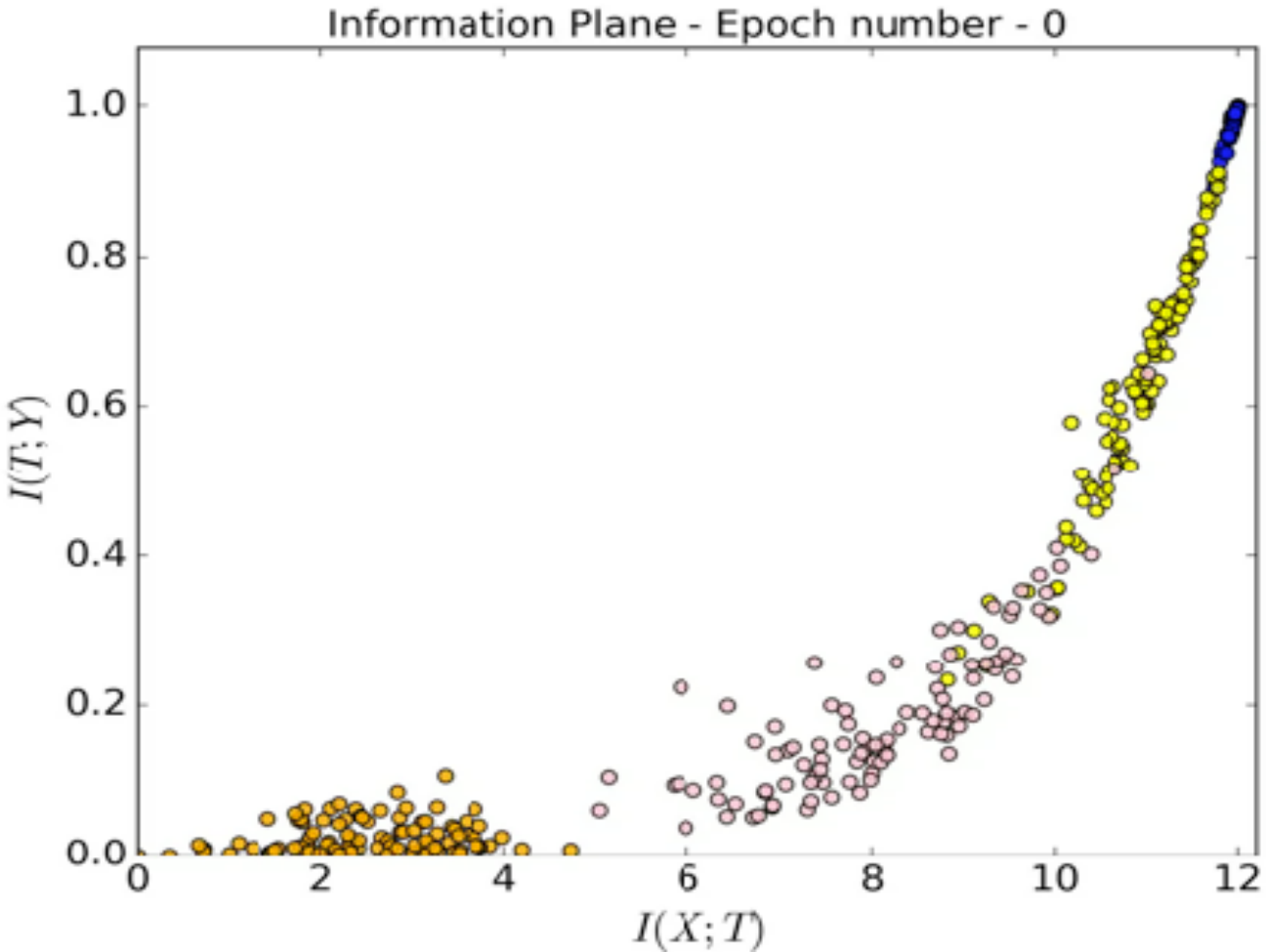
Each layer is characterized by its Encoder & Decoder Information



Theorem (Information Plane):
*For large typical **X**, the sample complexity of a DNN is completely determined by the encoder mutual information, **I(X;T)**, of the last hidden layer; the accuracy (generalization error) is determined by the decoder information, **I(T;Y)**, of the last hidden layer.*

The complexity of the problem shifts from the decoder to the encoder, across the layers…

100 DNN Layers in Info-Plane without averaging

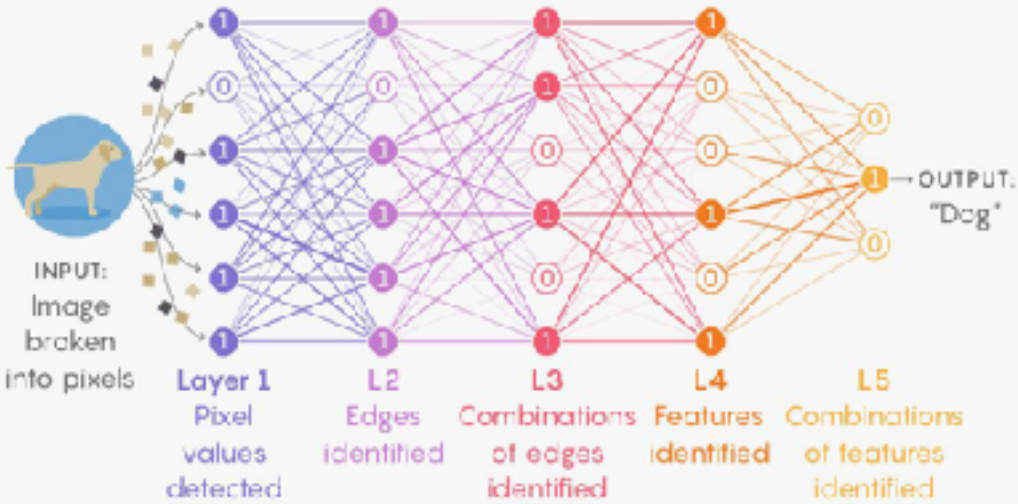


- Is this the general picture?
- Why do the MI values concentrate?
- What do they mean?
- What governs their dynamics?

ICRI-CI

Learning From Experience

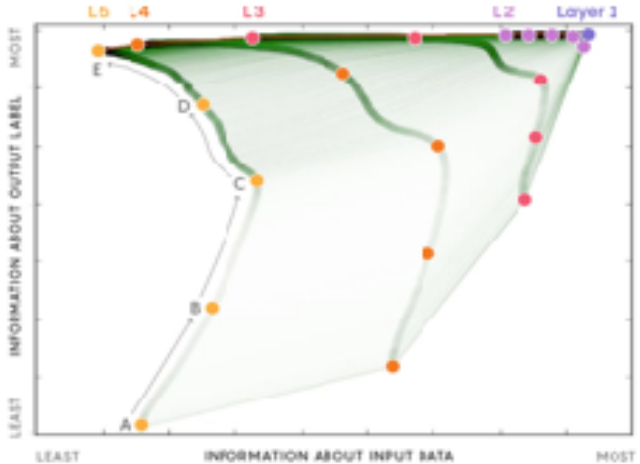
Deep neural networks learn by adjusting the strengths of their connections to better convey input signals through multiple layers to neurons associated with the right general concepts.



When data is fed into a network, each artificial neuron that fires (labeled "1") transmits signals to certain neurons in the next layer, which are likely to fire if multiple signals are received. The process filters out noise and retains only the most relevant features.

Inside Deep Learning

New experiments reveal how deep neural networks evolve as they learn.



A **INITIAL STATE:** Neurons in Layer 1 encode everything about the input data, including all information about its label. Neurons in the highest layers are in a nearly random state bearing little to no relationship to the data or its label.

B **FITTING PHASE:** As deep learning begins, neurons in higher layers gain information about the input and get better at fitting labels to it.

C **PHASE CHANGE:** The layers suddenly shift gears and start to "forget" information about the input.

D **COMPRESSION PHASE:** Higher layers compress their representation of the input data, keeping what is most relevant to the output label. They get better at predicting the label.

E **FINAL STATE:** The last layer achieves an optimal balance of accuracy and compression, retaining only what is needed to predict the label.

The role of stochasticity:
How do we measure Mutual Information?

- The representation invariance of the mutual information raises an interesting question.

- Obviously, the computational complexity of learning is not representation invariant (think about learning from encrypted patterns). **Thus, Information measures can't tell the whole story.**

- Our experiments crucially depends on how we estimate information. We consider 3 types of estimations: **(1) binning the variables. (2) adding noise / stochasticity (3) parametric approximations**. In our experiments we quantized/bin the neuronal output values.

- All assume compressibity/refineability of the variables. They are not robust to arbitrary invertible transformations!

- The assertion that the layers are invertible transformations of the input is NOT robust to small noise and misleading. Binning or assuming stochastic mapping is essential for our information theoretic approach.

- Moreover, the IB is trivial (uninteresting) for completely deterministic rules! I argue that our theory predicts that *without additional structural information on the patterns*, DL can't work for completely deterministic rules, as they can't be distinguished from random (fully mixing) rules!

Rethinking Learning Theory

"Old" Generalization bounds:

$$\varepsilon^2 < \frac{\log|H_\varepsilon| + \log 1/\delta}{2m}$$

ε - generalization error

δ - confidence

m - number of training examples

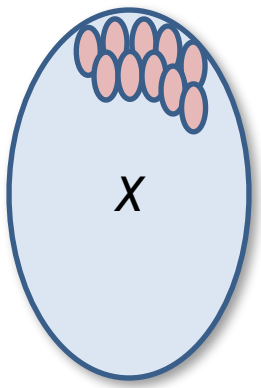
H_ε - ε-cover of the Hypothesis class

typically we assume: $|H_\varepsilon| \sim \left(\frac{1}{\varepsilon}\right)^d$

d - the class (VC,...) dimension

... Don't work for Deep Learning!

Higher expressivity - worse bound!



New: Input Compression bound

$$|H_\varepsilon| \sim 2^{|X|} \to 2^{|T_\varepsilon|}$$

T_ε - ε-partition of the input variable X

with respect to the distortion:

$$d_{IB}(x,t) = D[p(y|x) \| p(y|t)]$$

$$\geq 1/2\ln 2 \, \| p(y|x) - p(y|t) \|_1^2$$

when $p(y|t) = \sum_x p(y|x)p(x|t)$

$$\langle d_{IB} \rangle = I(X;Y) - I(T;Y)$$

small IB distortion , or high $I(T;Y)$,

$\Rightarrow$ small [typical] generalization error

Rethinking Learning Theory…

What are "large typical" patterns?

Typicality emerges when the underling pattern distribution can be asymptotically expressed as a long product of localized conditional probabilities.

E.g. Markov Random Fields, Hidden Markov Models, *pairwise* interaction Hamiltonians in physics, all common Graphical models, etc.

In our case it includes images, speech & text, long molecular sequences, signals generated by localized dynamic systems, etc.

Then, the Shannon-McMillen limit for the entropy exists:

$$\lim_{n\to\infty} -\frac{1}{n}\log p(x_1,\ldots,x_n) = H(X)$$

and *almost all* patterns are *typical* with probability:

$$p(x_1,\ldots,x_n) \simeq 2^{-nH(X)}$$

and also for large enough typical partitions, T:

$$p(x_1,\ldots,x_n \mid T) \simeq 2^{-nH(X|T)}$$

Concentration of Mutual Information

$$I(X;T)=\left\langle \log \frac{p(x|t)}{p(x)} \right\rangle_{X,T} = \left\langle \log \prod_i \frac{p(x_i|Pa(x_i),t)}{p(x_i|Pa(x_i))} \right\rangle_{X,T} = \left\langle \sum_i \log \frac{p(x_i|Pa(x_i),t)}{p(x_i|Pa(x_i))} \right\rangle_{X,T}$$

$$I(T;Y)=\left\langle \log \sum_x p(y|x)p(x|t) - \log p(y) \right\rangle_{Y,T} = \left\langle \log \left[\sum_x p(y|x) \prod_i p(x_i|Pa(x_i),t) \right] - \log p(y) \right\rangle_{Y,T}$$

Proposition:
1. Both I(T;X) and I(T;Y), as defined, concentrate, uniformly, under the partition typicality
assumption.
2. Both can be estimated uniformly well (over the partitions) from a sample of p(X,Y

Rethinking Learning Theory

"Old" Generalization bounds:

$$\varepsilon^2 < \frac{\log |H_\varepsilon| + \log 1/\delta}{2m}$$

ε - generalization error

δ - confidence

m - number of training examples

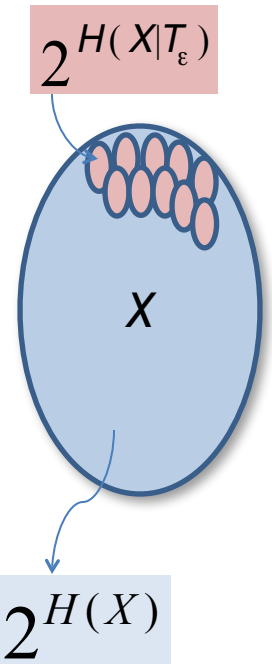
H_ε - ε-cover of the Hypothesis class

typically we assume: $|H_\varepsilon| \sim \left(\frac{1}{\varepsilon}\right)^d$

d - the class (VC,...) dimension

... Don't work for Deep Learning!

Higher expressivity - worse bound!



New: Input Compression bound:

$$|H_\varepsilon| \sim 2^{|X|} \to 2^{|T_\varepsilon|}$$

T_ε - ε-partition of the input variable X

Information Theory: $|T_\varepsilon| \sim 2^{I(T_\varepsilon;X)}$

$$\varepsilon^2 < \frac{2^{I(T_\varepsilon;X)} + \log 1/\delta}{2m}$$

... K bits of compression of X are like a factor of 2^K training examples!

The Information Bottleneck (IB) Method

(Tishby, Pereira, Bialek, 1999)

(1) Approximate Minimal Sufficient Statistics:

Markov chain: $Y \rightarrow X \rightarrow S(X) \rightarrow \hat{X}$

$$\hat{X} = \arg\min_{\Sigma(X):I(S(X);Y)=I(X;Y)} I(S(X);X)$$

Relaxation - given $p(X,Y)$:

$$\hat{X} = \arg\min_{p(\hat{x}|x)} I(\hat{X};X) - \beta I(\hat{X};Y)\ ,\ \beta > 0$$

(Shamir, Sabato,T., TCS 2010)

The Information Bottleneck optimality bound

(Tishby, Pereira, Bialek, 1999)

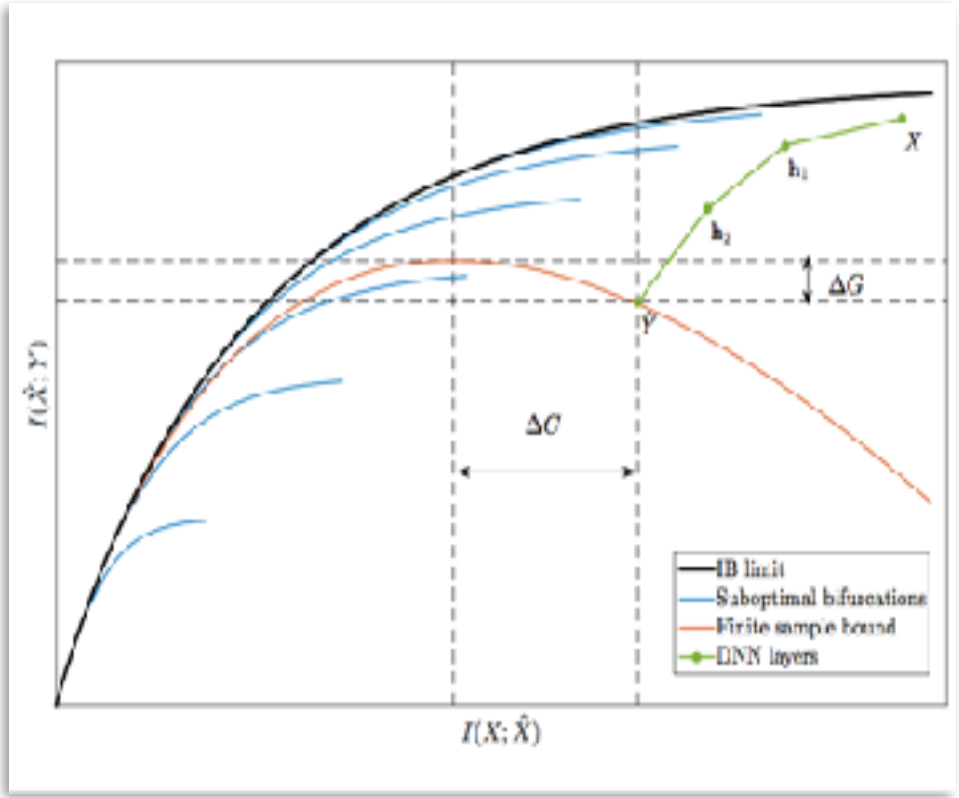
The IB bound optimality equations:

$$\min_{p(\hat{x}|x),Y-X-\hat{X}} I(\hat{X};X) - \beta I(\hat{X};Y)\ ,\ \beta > 0$$

$$\begin{cases} p(x|\hat{x}) = \dfrac{p(x)}{Z(x,\beta)} \exp(-\beta D[p(y|x) \| p(y|\hat{x})]) \\ Z(x,\beta) = \sum_x p(\hat{x}) \exp(-\beta D[p(y|x) \| p(y|\hat{x})]) \\ p(\hat{x}) = \sum_x p(\hat{x}|x)\, p(x) \\ p(y|\hat{x}) = \sum_x p(y|x)\, p(x|\hat{x}) \end{cases}$$

Solved by Arimoto-Blahut like iterations,

but with possibly sub-optimal solutions, bifurcations (!),



Example: Gaussian Information Bottleneck

Gal Chechik + Amir Globerson, Naftali Tishby, Yair Weiss

NIPS 2003, JMLR 2004

ICRI-CI

GIB: Statement of the problem

- Let X and Y be jointly multivariate Gaussian
- Search for another variable T such that

$$p^*(t \mid x) = \underset{p(t|x), Y \to X \to T}{\arg\min} \; L = I(X;T) - \beta I(T;Y)$$

- First, we can prove that the optimal (T,X) are jointly Gaussian, in this case

- Then: T can always be represented as

$T = A X + \xi$, with. $\xi \sim N(0, \Sigma_\xi)$, $A = \Sigma_{tx} \Sigma_x^{-1}$

- Minimize L over the projection A and noise ξ

ICRI-CI

Which covariance matrix?

- Several linear projection algorithms can all be formalized through the eigenvalue/vector problems:

– PCA: Analysis	$eig(\Sigma xx)$	Principal Component
– PLS:	$eig(\Sigma xy\Sigma yx)$	Partial Least Squares
– CCA:	$eig(\Sigma xy\Sigma y^{-1}\Sigma yx\ \Sigma x^{-1})$	Canonical Correlation Analysis
– MLR:	$eig(\Sigma xy\Sigma yx\ \Sigma x^{-1})$	Multi Linear Regression
– MDS: Scaling…	$eig(\Sigma yx\Sigma xy\ )$	Linear Multi Dimensional

What about the dependence on β ?

- For β=*0*, the degenerated null solution (*T=0)* is optimal.

- For infinite β ? We probably want to take all eigenvectors. But how should we weight them?

- What should happen for finite β ?

Deriving the GIB solution

Using the differential entropy of a Gaussian and the conditional covariance expression:

$$h(X) = \tfrac{1}{2}\log\left((2\pi e)^d \mid \Sigma_x \mid\right)$$

$$\Sigma_{t|y} = \Sigma_t - \Sigma_{ty}\left(\Sigma_y\right)^{-1}\Sigma_{yt} = A\Sigma_{x|y}A^T + \Sigma_\xi$$

we write the target function

$$\begin{aligned} L(A,\Sigma_\xi) &= h(T) - h(T \mid X) - \beta\, h(T) + \beta\, h(T \mid Y) \\ &= (1-\beta)\log\left|A\Sigma_x A^T + \Sigma_\xi\right| - \log\left|\Sigma_\xi\right| \\ &\quad + \beta\log\left|A\Sigma_{x|y}A^T + \Sigma_\xi\right| \end{aligned}$$

Lemma: For every (A, Σ_ξ) there is a pair (A', I) such that: $L(A, \Sigma_\xi) = L(A', I)$

Corollary: We can replace Σ_ξ with I

Differentiating *L* w.r.t *A* (matrix derivative!) using:

$$\frac{dL}{dA}\log|ACA^T| = (ACA^T)^{-1}2AC$$

which holds for symmetric ***C***. We have:

$$\frac{dL}{dA} = (1-\beta)\left(A\Sigma_x A^T + I\right)^{-1}2A\Sigma_x + \beta(A\Sigma_{x|y}A^T + I)2A\Sigma_{x|y}$$

comparing to zero, we obtain the stationarity condition:

$$\frac{\beta-1}{\beta}\left[\left(A\Sigma_{x|y}A^T + I\right)\left(A\Sigma_x A^T + I\right)^{-1}\right]A = A\Sigma_{x|y}\Sigma_x{}^{-1}$$

The scalar T case (A is a row vector)

- The condition on A is:

$$\left(\frac{\beta - 1}{\beta}\right)\left(\frac{A\Sigma_{x|y}A^T + I}{A\Sigma_x A^T + I}\right)A = A\left(\Sigma_{x|y}\Sigma_x{}^{-1}\right)$$

- This has two types of solutions:
  - A degenerates to zero
  - A is an eigenvector of $\Sigma_{x|y}\,\Sigma_x^{-1}$

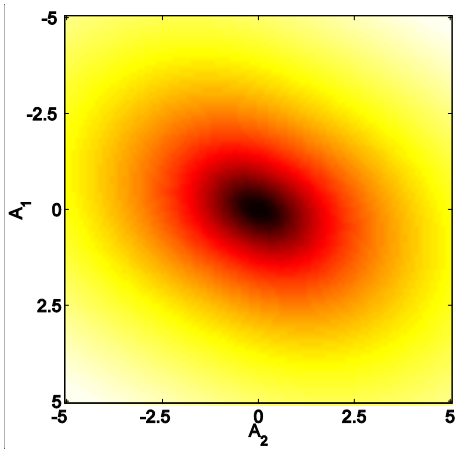
The eigenvector solution

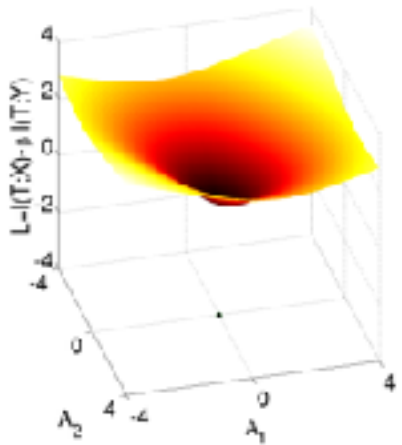
- The eigenvalue satisfies $\lambda = \left(\frac{\beta - 1}{\beta}\right)\left(\frac{A\Sigma_{x|y}A^T + I}{A\Sigma_x A^T + I}\right)$
- Denoting $\frac{A\Sigma_x A^T}{\|A\|^2} = r$ we have: $\frac{\beta(1-\lambda)-1}{r\lambda} = \| A \|^2 > 0$
- Hence an eigenvector solution exists iff $\beta \geq \frac{1}{1-\lambda}$
- Otherwise we get the A=0 solution.
- Which eigenvalue is optimal ?
- The minimal one λ_1

The nature of the solution

$$A = \begin{cases} \frac{\beta\,(1-\lambda_1)-1}{\lambda_1} v_1 & \beta > (1-\lambda_1)^{-1} \\ 0 & \text{otherwise} \end{cases}$$

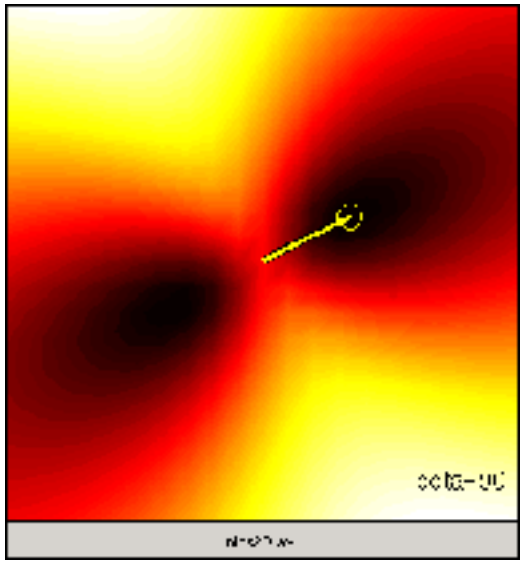
- As an illustration, look at the surface of *L* as a function of A for A = (1x2) vector. For example

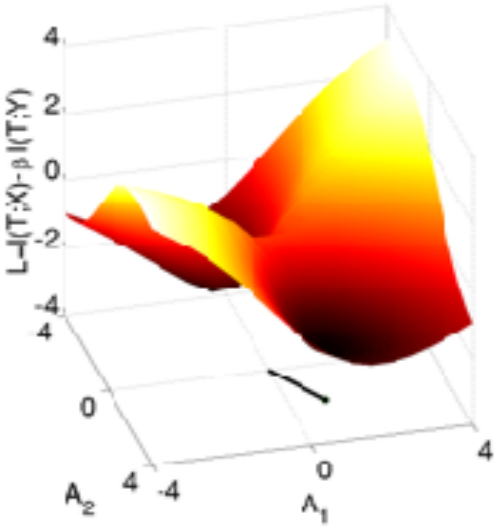




The nature of the solution

- As β increases the eigenvector solution emerges:





The higher dimensional case

- back to: $$\frac{\beta - 1}{\beta}\left[\left(A\Sigma_{x|y}A^T + I\right)\left(A\Sigma_x A^T + I\right)^{-1}\right]A = A\Sigma_{x|y}\Sigma_x{}^{-1}$$

- The rows of A are still in the span of several eigenvectors. We can show that an optimal solution is achieved with the smallest eigenvectors of $\Sigma y|x\Sigma x^{-1}$.
- As β increases A goes through a series of phase- transitions, each adding another eigenvector:

$$A = \begin{cases} [0^T; \boxed{?}; 0^T] & 0 < \beta < \beta_i^c \\ [\alpha_1 \mathbf{v}_1^T; 0^T; \boxed{?}; 0^T] & \beta_1^c < \beta < \beta_2^c \\ [\alpha_1 \mathbf{v}_1^T; \alpha_2 \mathbf{v}_2^T; 0^T; \boxed{?}; 0^T] & \beta_2^c < \beta < \beta_3^c \\ \boxed{?} & \boxed{?} \end{cases} \quad \alpha_i = \frac{\beta(1-\lambda_i)-1}{\lambda_i}$$

$$\text{with } r_i = \mathbf{v}_i^T \Sigma_x \mathbf{v}_i^T \qquad \beta_i^c = (1-\lambda_i)^{-1}$$

ICRI-CI

The GIB Information Bottleneck bound

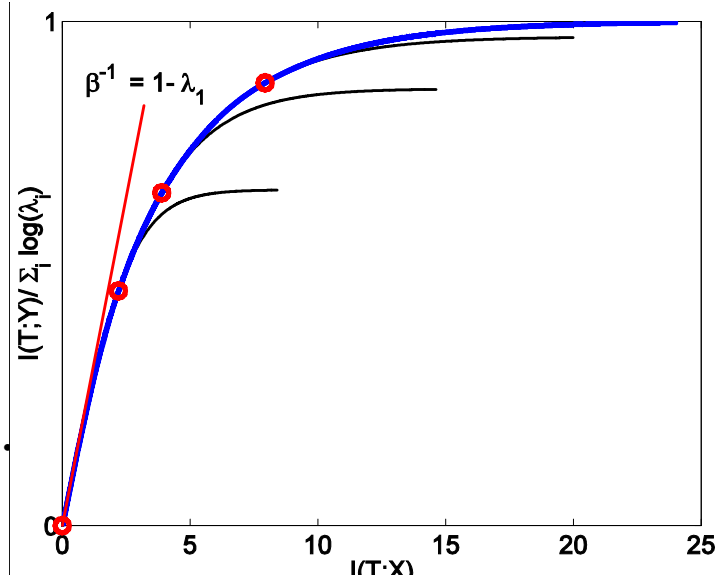
- Can be calculated analytically, as a function of the eigenvalue spectrum:

$$I(T;Y) = I(T;X) - \frac{n_I}{2}\log\left(\prod_{i=1}^{n_I}(1-\lambda_i)^{-n_I} + \exp\left(\frac{2I(T;X)}{n_I}\right)\prod_{i=1}^{n_I}(\lambda_i)^{-n_I}\right)$$

where n_I is chosen such that:

$$\sum_{i=1}^{n_I-1}\log\frac{\lambda_{n_I}}{\lambda_i}\frac{1-\lambda_i}{1-\lambda_{n_I}} \le I(T;X) \le \sum_{i=1}^{n_I}\log\frac{\lambda_{n_I+1}}{\lambda_i}\frac{1-\lambda_i}{1-\lambda_{n_I+1}}$$

- Example: the GIB curve with 4 eigenvalues λ=0.9,0.7,0.5,0.1. Curves for fixed no of eigenvalues are also shown.



Rethinking Learning Theory…

… but we need to guarantee the label homogeneity of the -partition with finite samples. Without additional structural information on the inputs (stability, robustness, topology), we must use the stochasticity of the rule and the IB distortion measure:

The ε – partition, T_ε, is with the empirical distortion

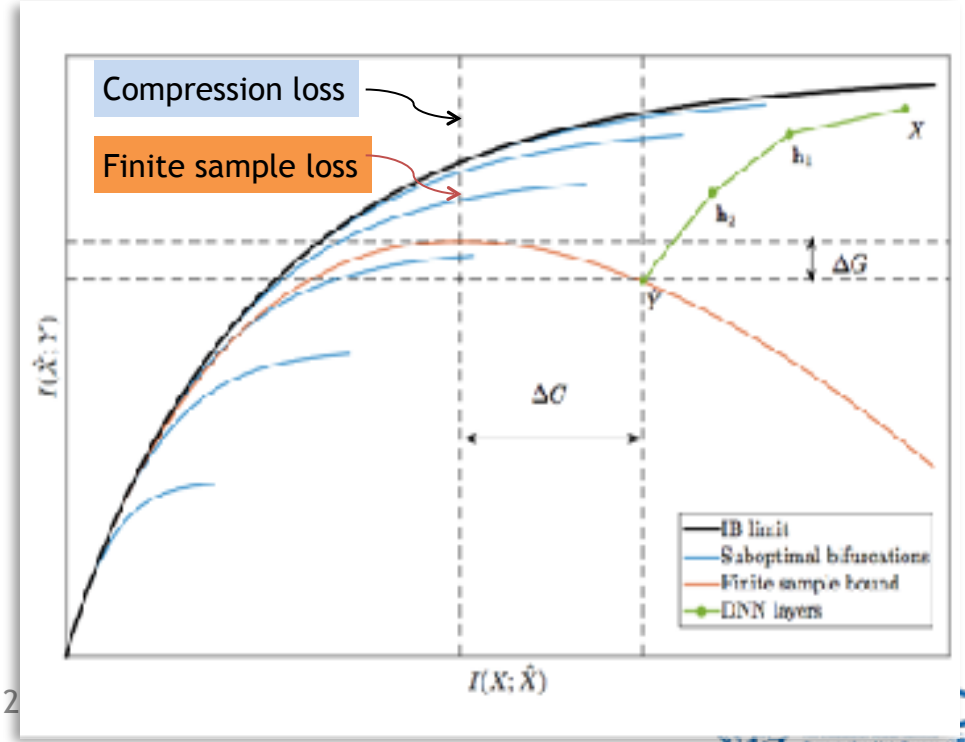
$$d_{IB}(x,t) = D[\,p_{emp}(y|x) \,\|\, p(y|t)]$$

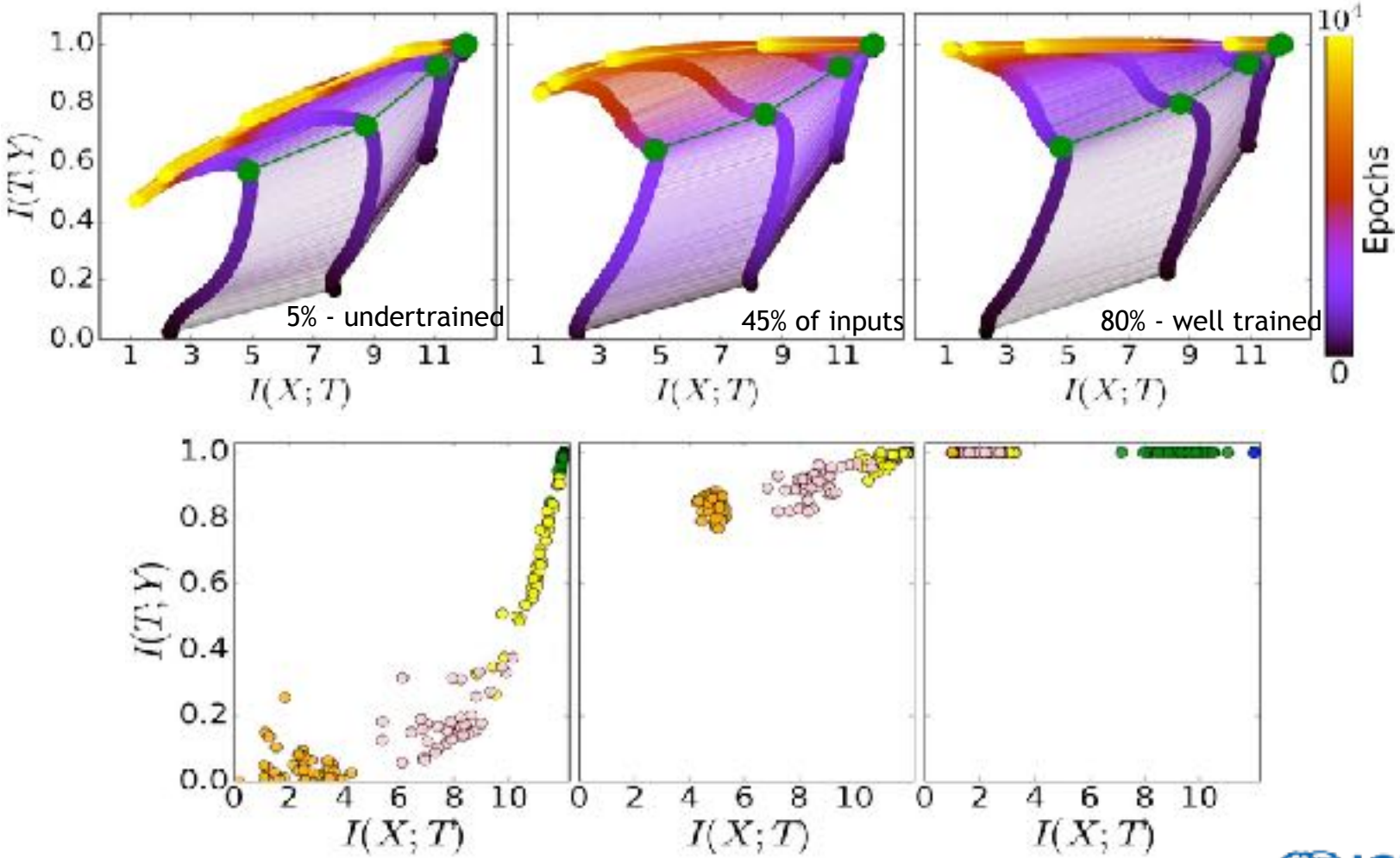
$$\text{as } \left\langle d_{IB} \right\rangle_{emp} = I(X;Y) - \hat{I}_{emp}(T;Y)$$

with a finite sample there is another information loss:

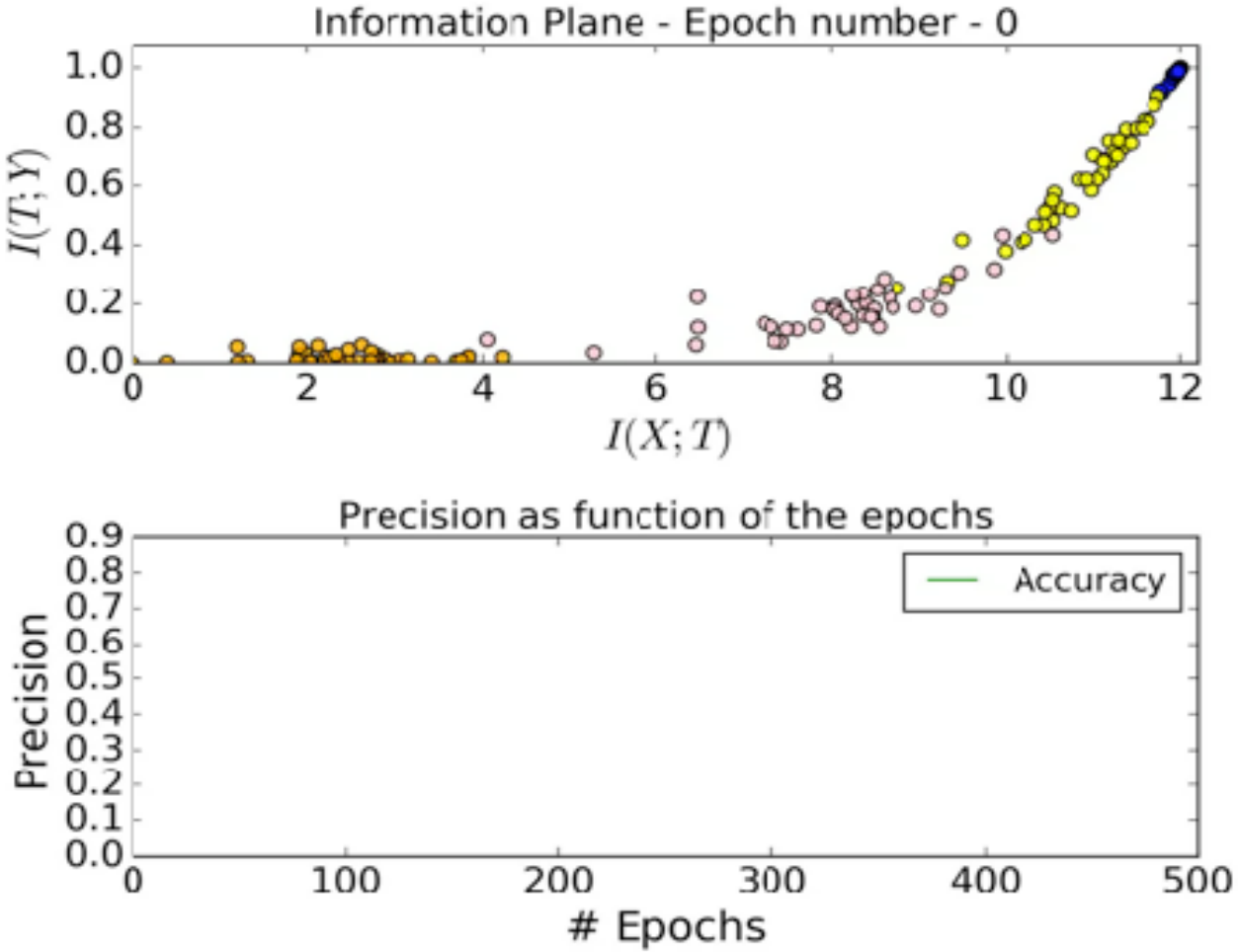
$$I(T;Y) \le \hat{I}_{emp}(T;Y) + O\sqrt{\left(\frac{2^{I(T;X)}|Y|}{m}\right)},$$

both should remain small for good generlaization!

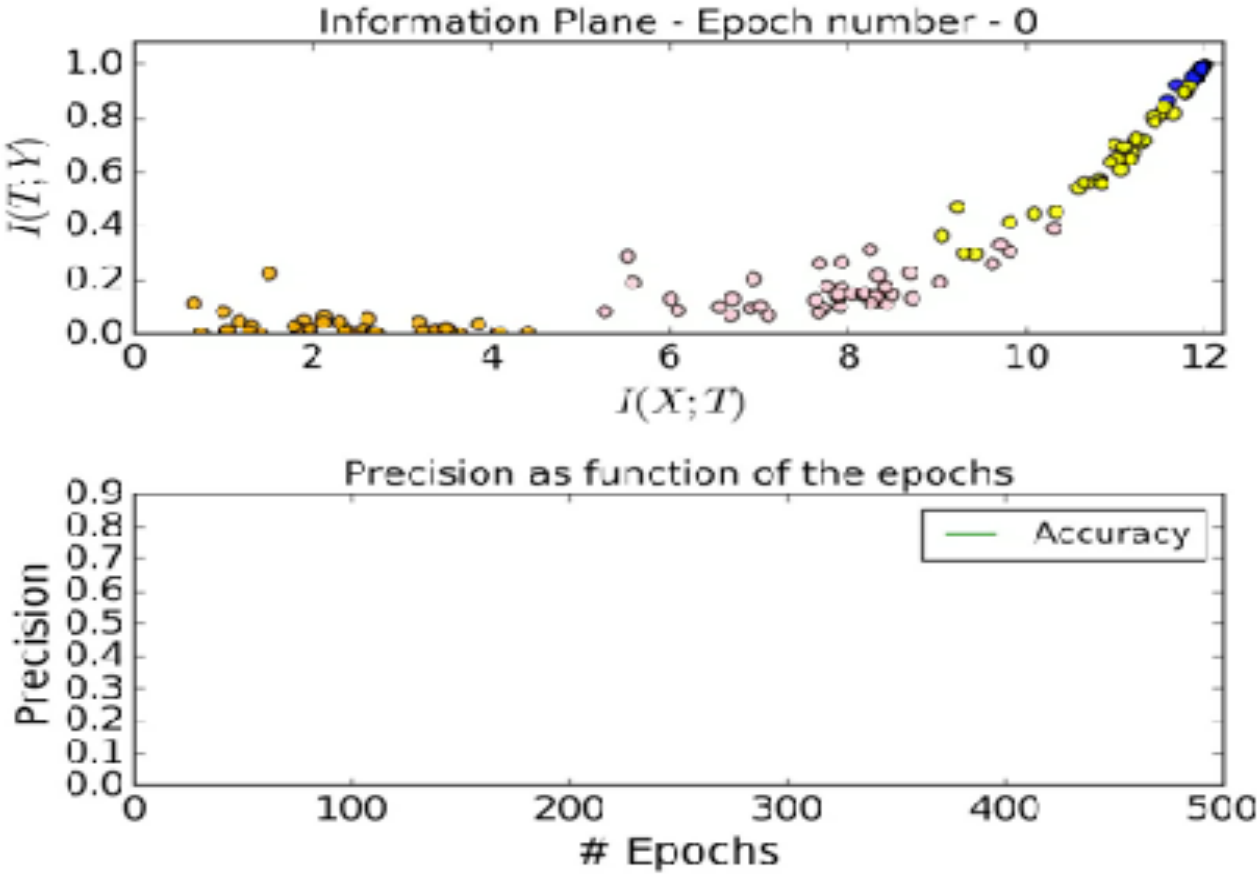


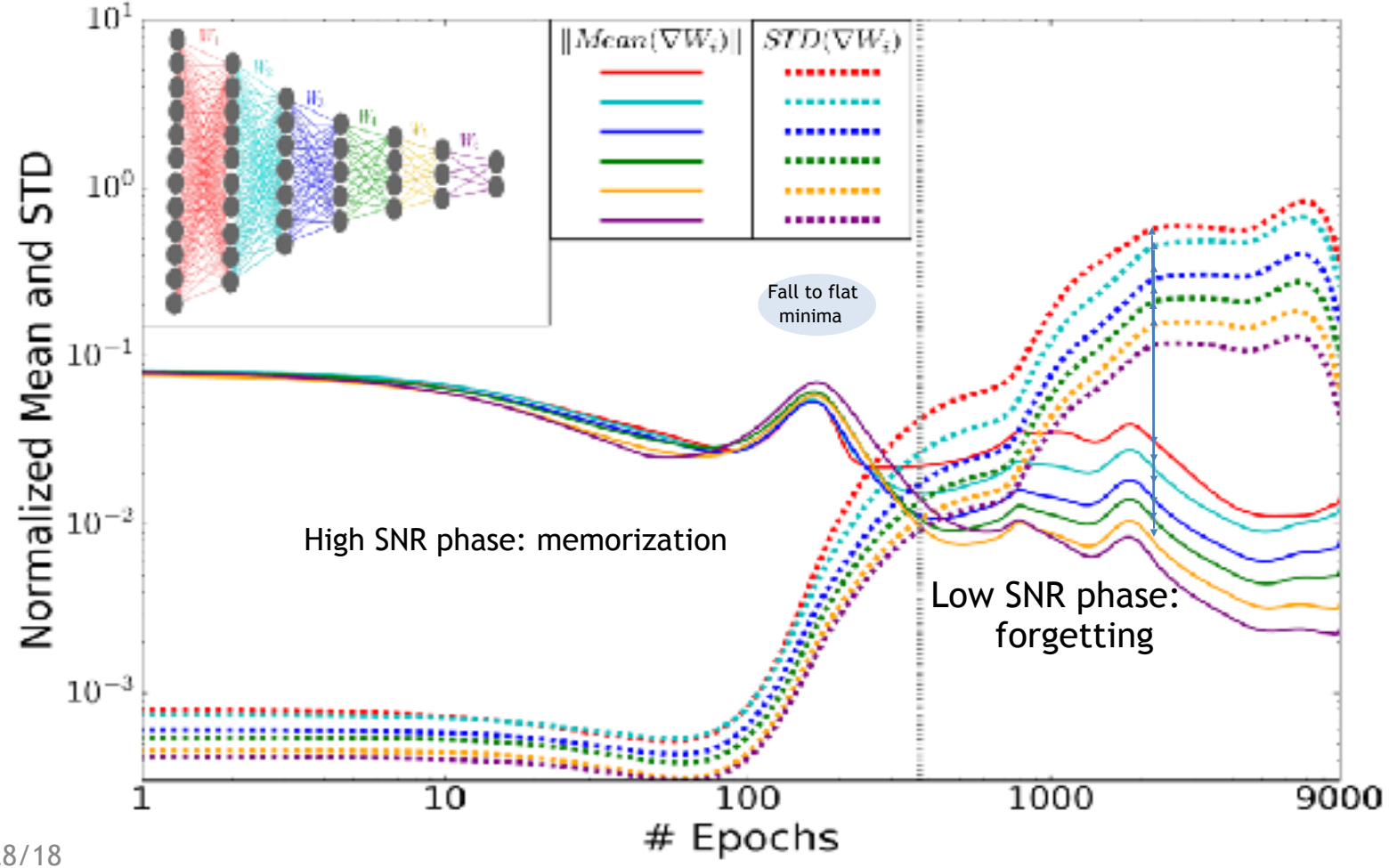

5% - undertrained
45% of inputs
80% - well trained
Epochs
10^4
0
$I(T;Y)$
$I(X;T)$

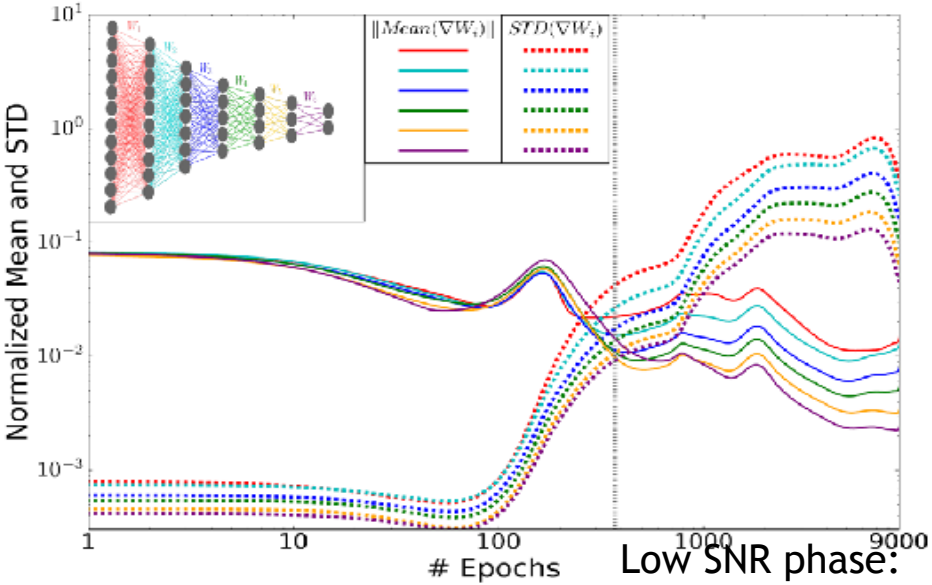
Layers paths with training/generalization error



Layers paths with generalization error - committee

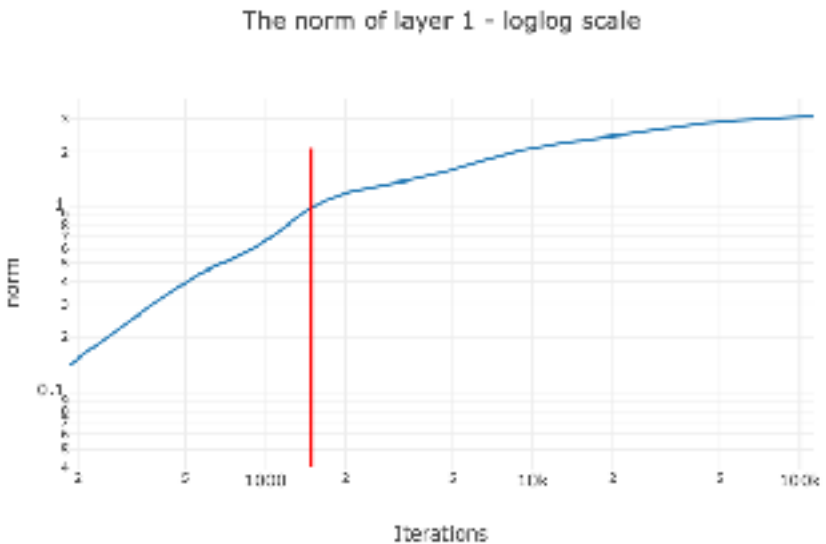



Normalized Mean and STD
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
$\|Mean(\nabla W_i)\|$
$STD(\nabla W_i)$
Fall to flat minima
High SNR phase: memorization
Low SNR phase: forgetting
1
10
100
1000
9000
Epochs



High SNR phase: memorization

Low SNR phase: forgetting

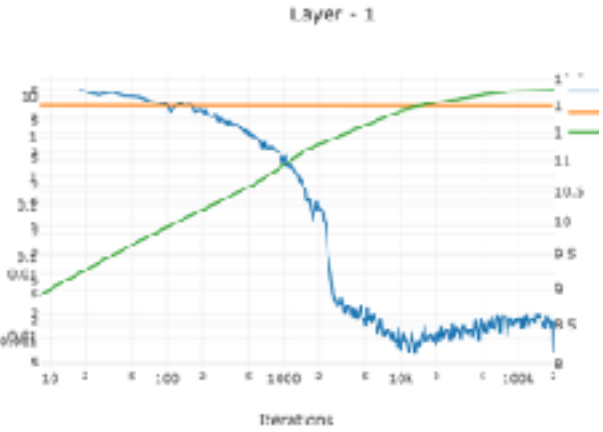


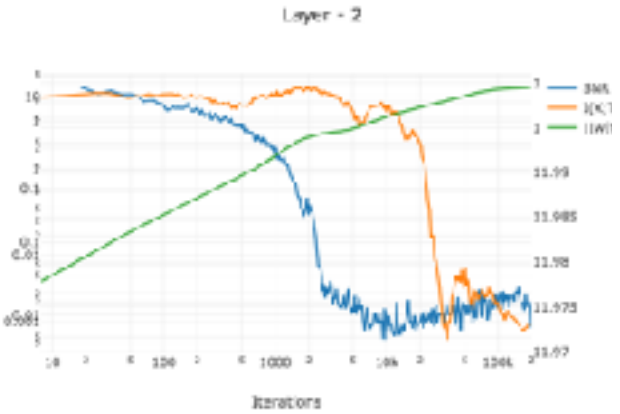
High SNR phase: Linear drift

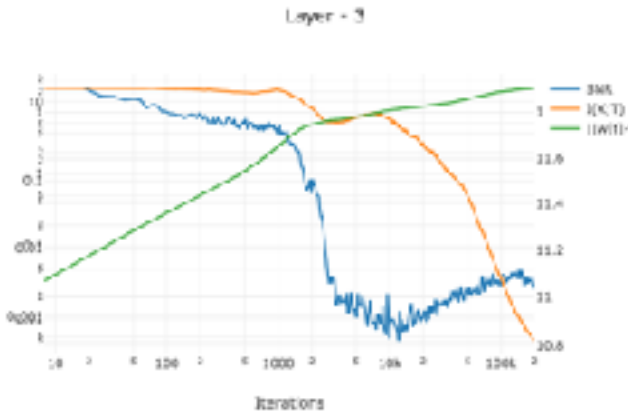
Low SNR phase: Diffusion

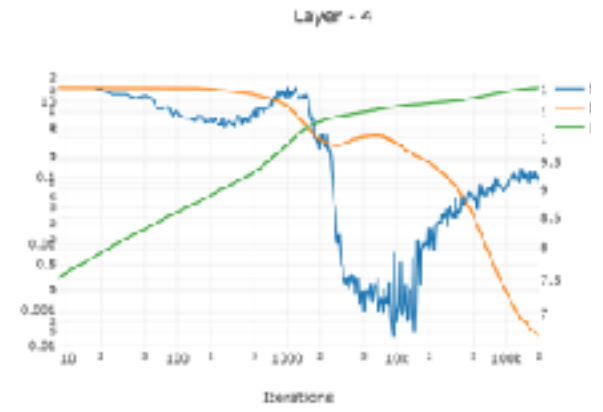
In the noisy phase the weights diffuse and grow like $\mathcal{O}(\sqrt{t})$

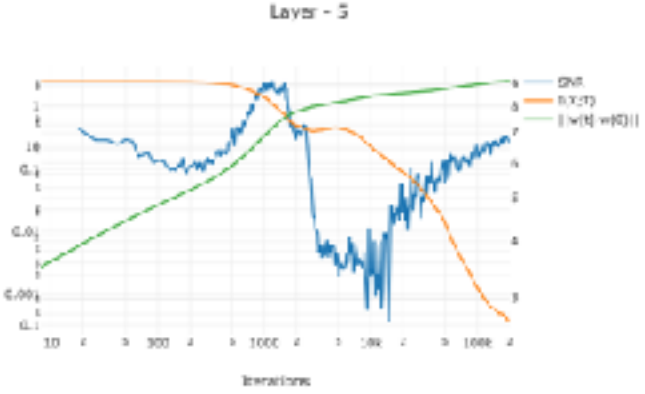
Gradients SNR, Diffusion & Compression – all layers

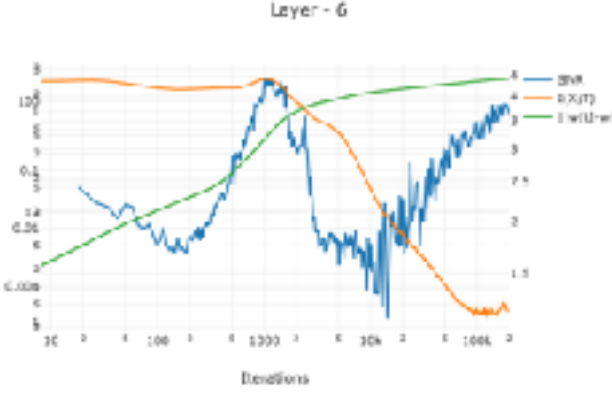




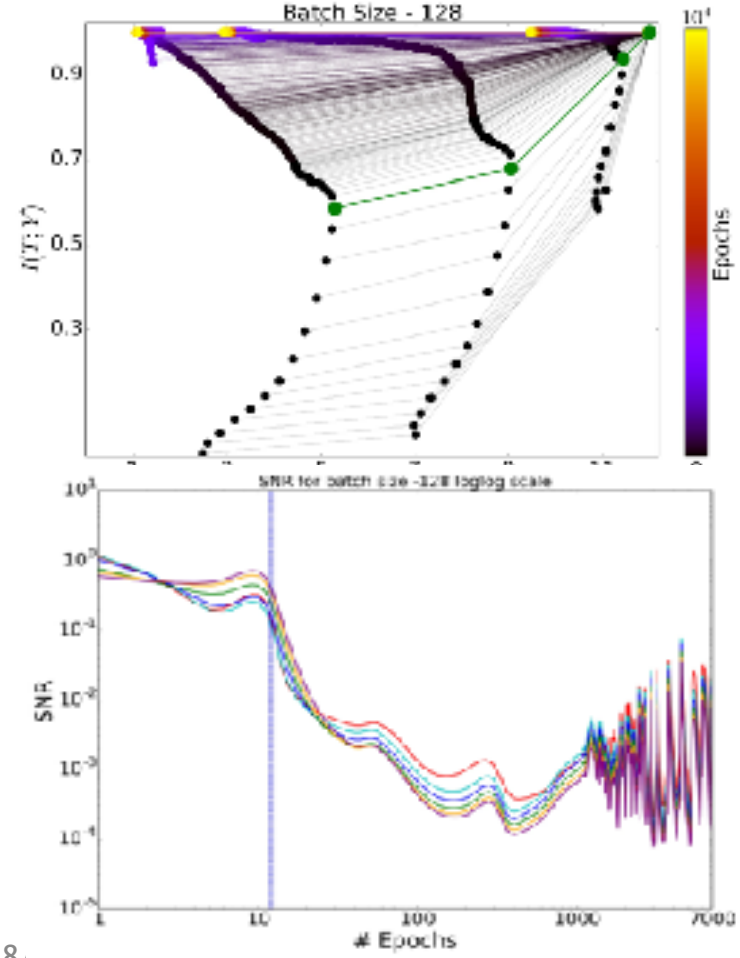


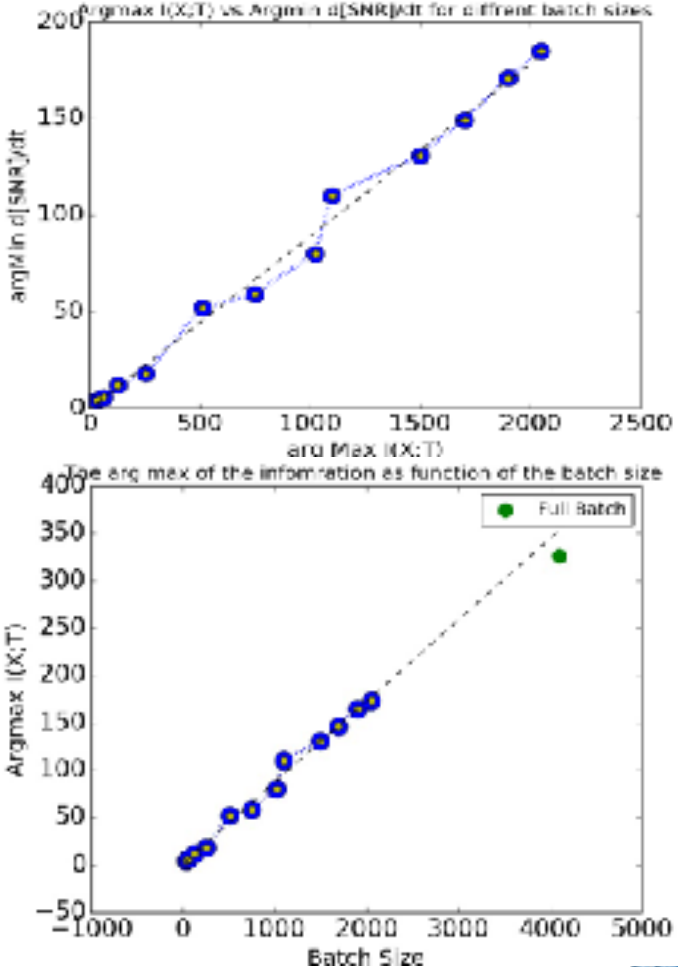






The role of the batch size





Break

Relevant and Irrelevant local dimensions

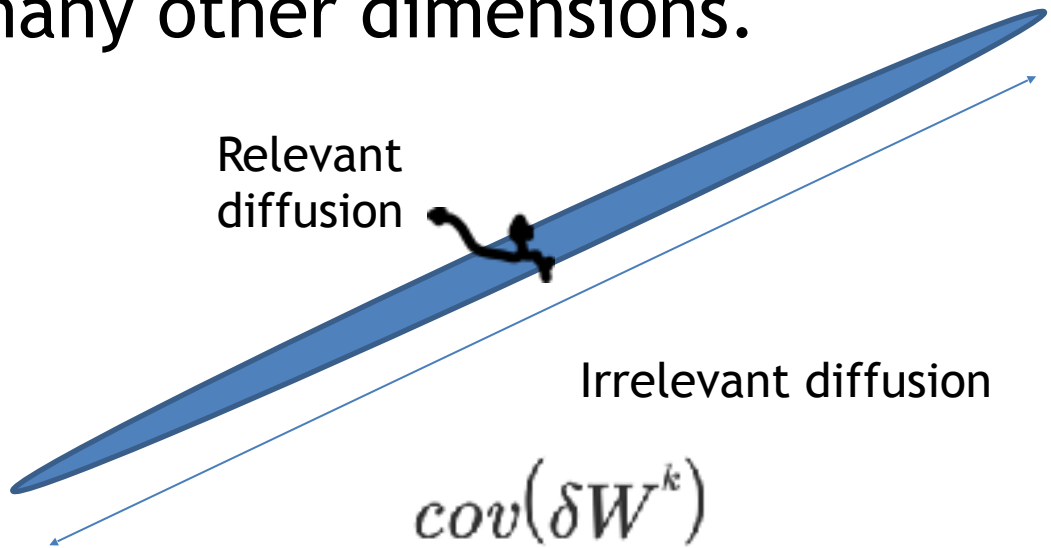
- The covariance matrix of the gradients is very narrow in the relevant local dimensions and very wide in the many other dimensions.

$$W^k \rightarrow W^k_{cca} + \delta W^k$$

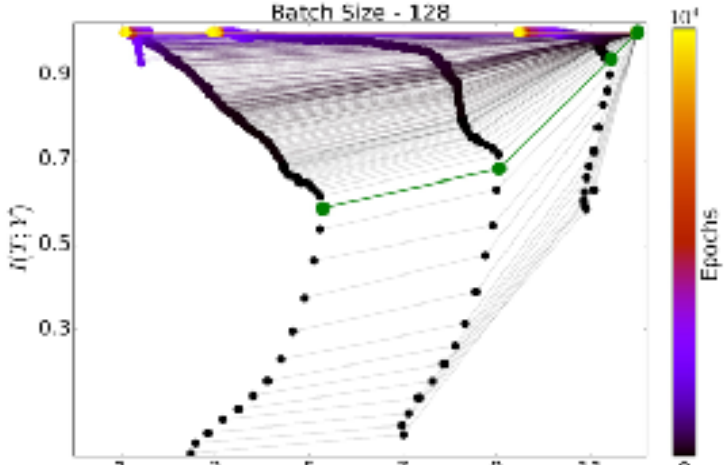
$$T^{k+1} = \sigma\left(W^k_{cca} T^k + \xi^k\right)$$

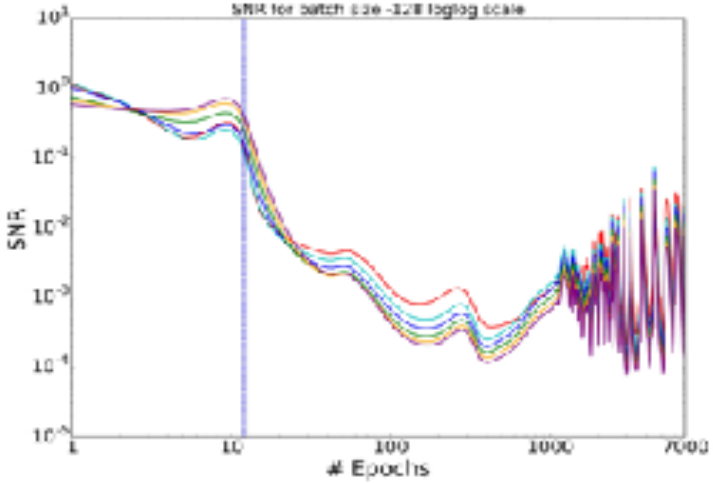
$$\xi^k = \delta W^k T^k \sim N\left(0, cov(\delta W^k)\right)$$

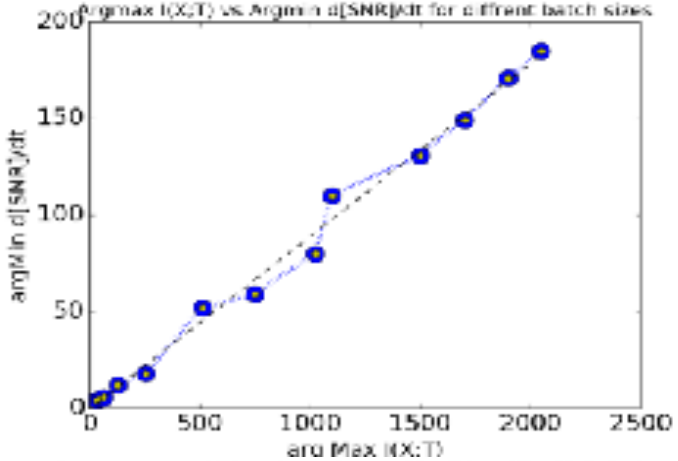
$$I(T_k; T_{k+1}) \leq \frac{1}{2} \log\left(1 + \frac{\|W^{\blacktriangle}_{cca}\|}{\|\delta W^{\blacktriangle}\|}\right)$$

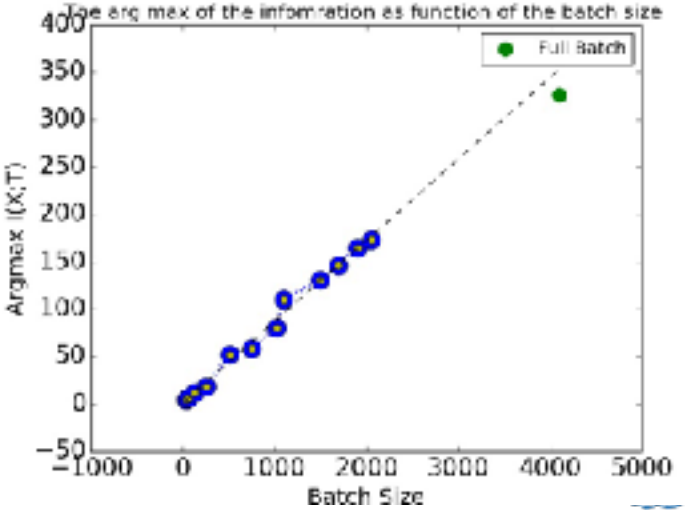


The role of the batch size

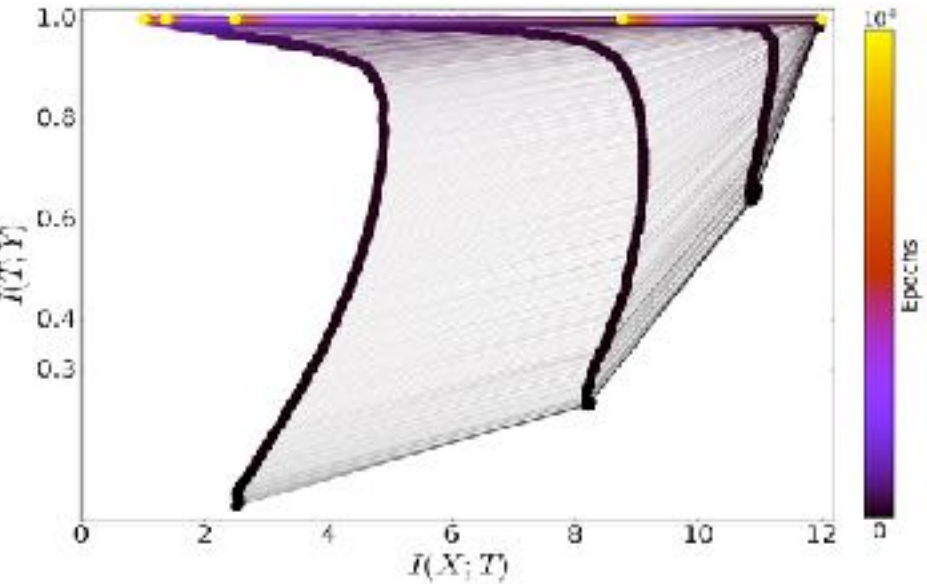


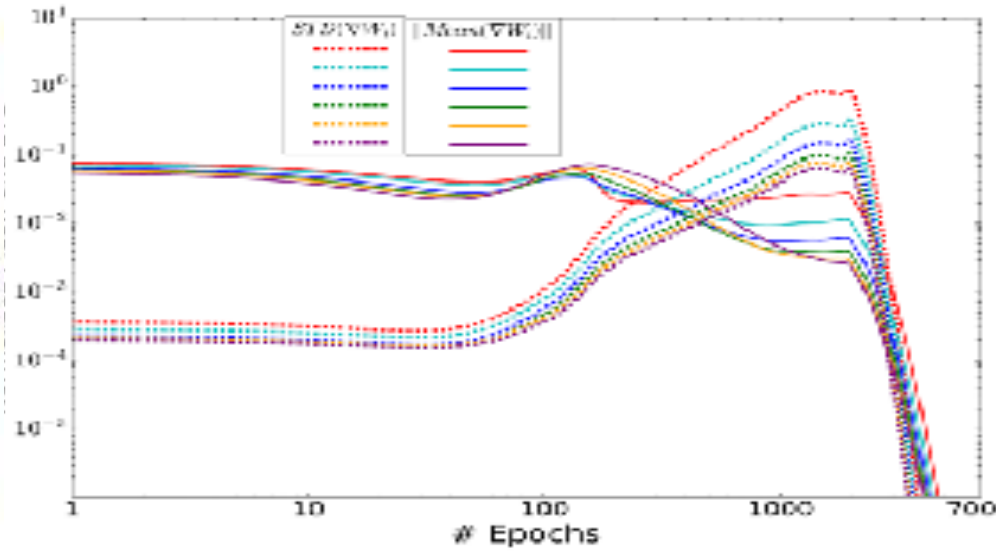






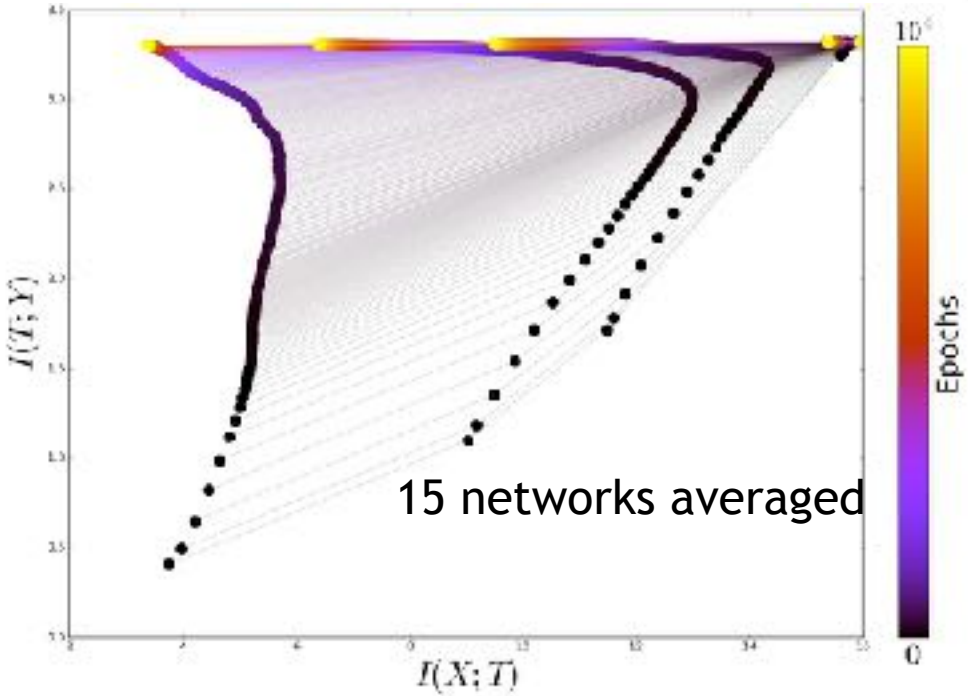
Is it the general picture? Yes!

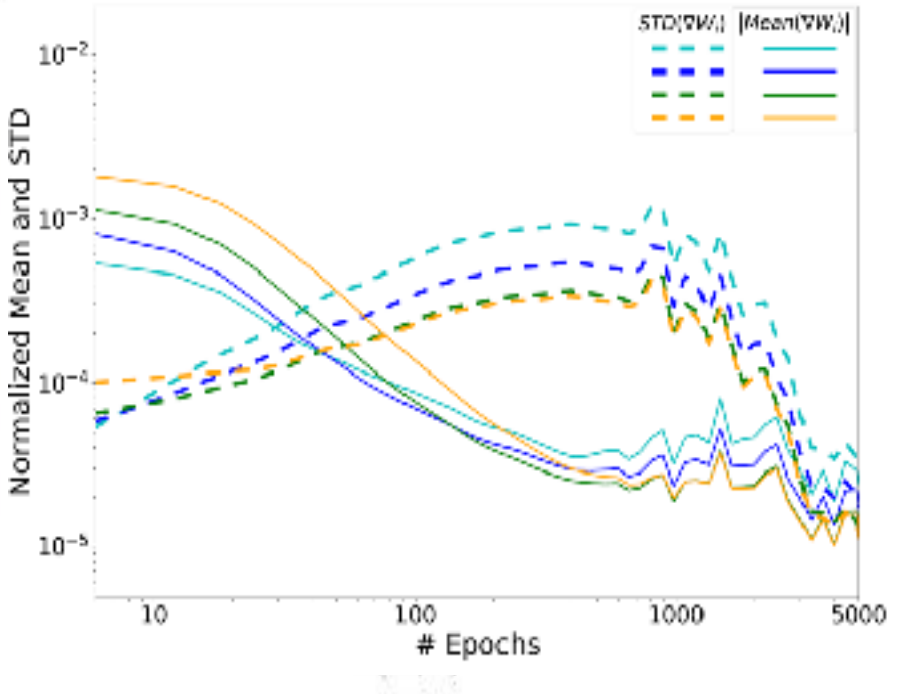




6 layer committee machine

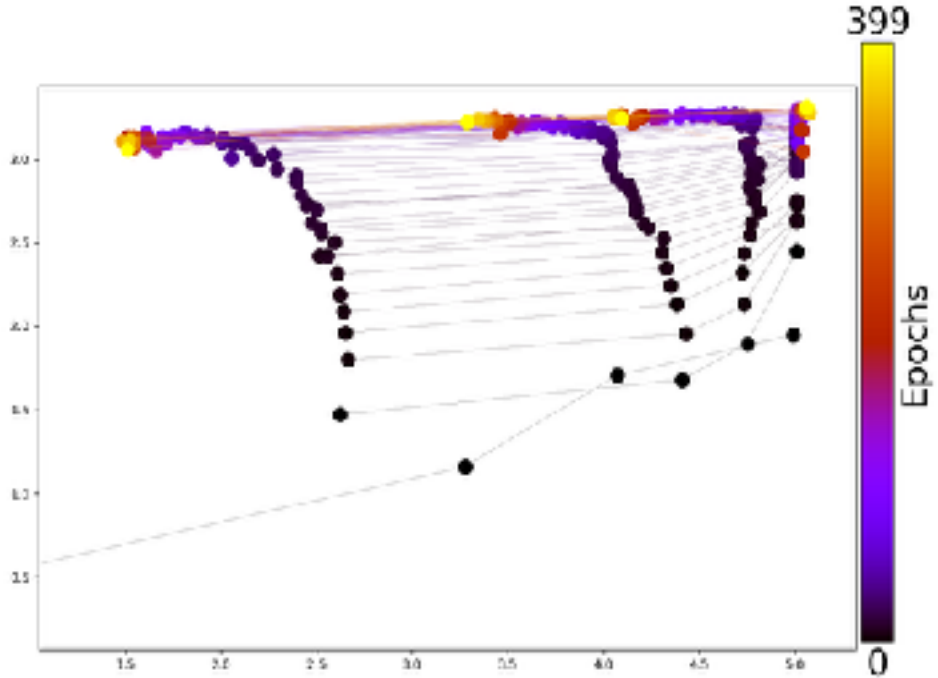
… and for “Real-world” problems? Yes!

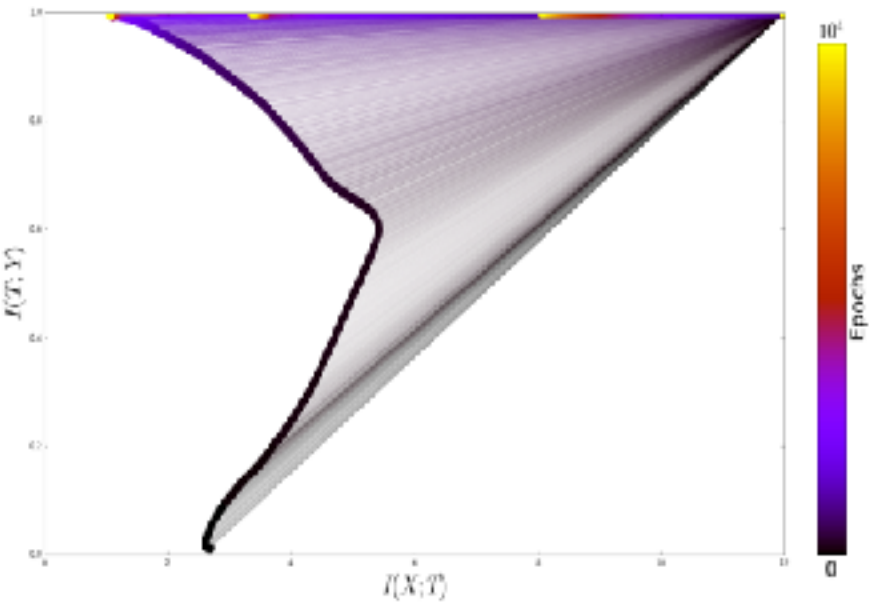




MNIST handwriting digit recognition with RelU’s a CNN architecture

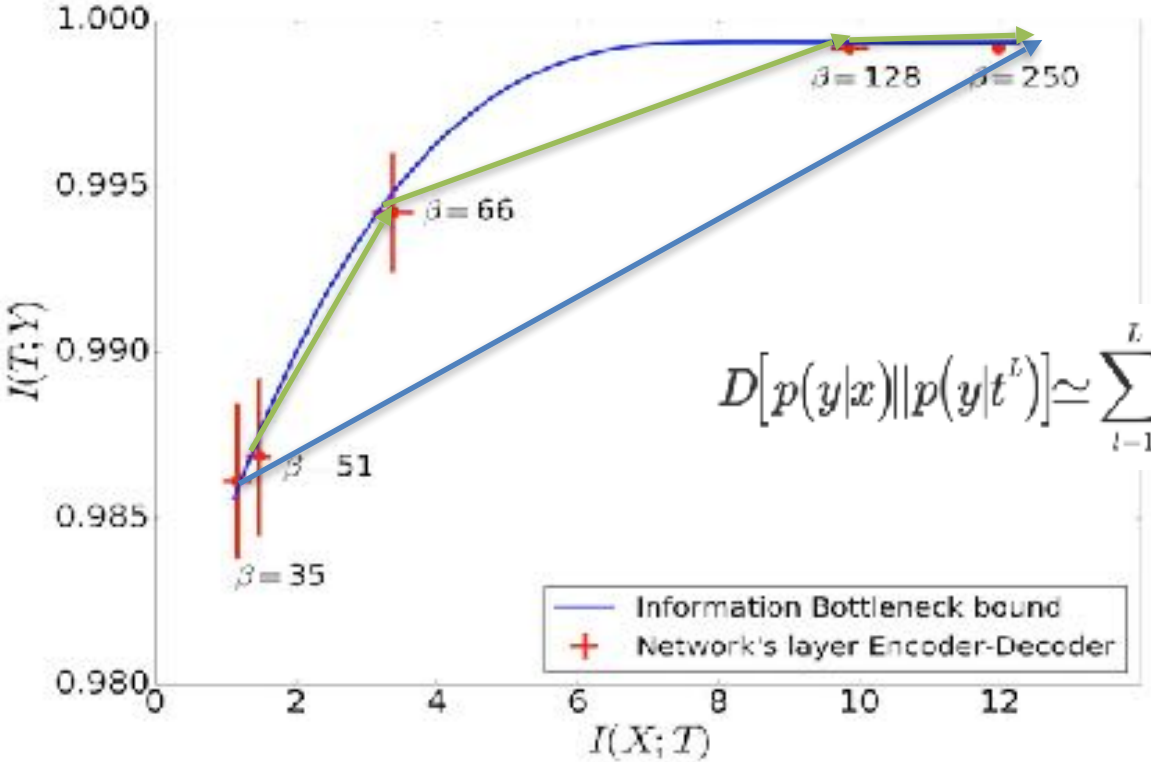
Is it the general picture? Yes!





CIFAR 10 object recognition task

Non decreasing layer widths - notice last hidden



$$D\left[p(y|x)\|p(y|t^L)\right] \simeq \sum_{l=1}^{L} D\left[p(y|t^{l-1})\|p(y|t^l)\right]$$

- Layers of optimal DNN converge to [a successively refineable approximation of] the optimal finite-sample IB limit information-curve
- Layers must be in “different topological phases” of the IB solutions
- The DNN encoder & decoder for each layer satisfy the IB self-consistent equations

Compression through Stochastic Relaxation

Noisy relaxation (SGD) on training error (with m examples):

$$\frac{\partial W_k}{\partial t} = -\nabla E(W_k \mid X^{(m)}) + \beta_k^{-1}\xi(t)\,, \quad \text{layer } k, \quad \xi \sim N(0,1),\ \beta_k - \textit{decoder } k\textit{-layer noise}$$

$\Rightarrow$ Maximum Entropy: $P_{Gibbs}(W_k \mid X^{(m)}) \propto \exp\left(-\beta_k E(W_k \quad X^{(m)})\right)$

with additive [cross-entropy] traininng error and i.i.d. samples, using Bayes rule with the quenched W:

$$P_{Gibbs}(X \mid W_k) = P_{Gibbs}(X \mid T_k) \propto \exp\left(-\tilde{\beta}_k D_{KL}[p(Y \mid X) \| p(Y \mid T)]\right)$$

This is precisely the IB optimal encoder with $\tilde{\beta}_k$ – *the encoder k-layer noise*

Since $I(X;T_k) = H(X) - H(X \mid T_k)$, Max Entropy of the weights $\rightarrow$ Min $I(X;T_k)$

thus SGD converges, layer by layer, to a maximally compressed representation,

which is a SUCCESSIVELY REFINEABLE APPROXIMATION of the optimal IB bound!

Local weights Gibbs and optimal IB representations

Noisy relaxation (SGD) on training error (with m examples):

$$\frac{\partial W_k}{\partial t} = -\nabla E(W_k \mid X^{(m)}) + \beta_k^{-1}\xi(t)\ , \quad \text{layer } k, \qquad \xi \sim N(0,1),\ \beta_k - \textit{decoder } k\textit{-layer noise}$$

$\Rightarrow$ Maximum Entropy: $P_{Gibbs}(W_k \mid X^{(m)}) \propto \exp\left(-\beta_k E(W_k \quad X^{(m)})\right)$

with additive [cross-entropy] traininng error and i.i.d. samples, using Bayes rule with the quenched W:

$$P_{Gibbs}(X \mid W_k) = P_{Gibbs}(X \mid T_k) \propto \exp\left(-\tilde{\beta}_k D_{KL}[p(Y \mid X) \| p(Y \mid T)]\right)$$

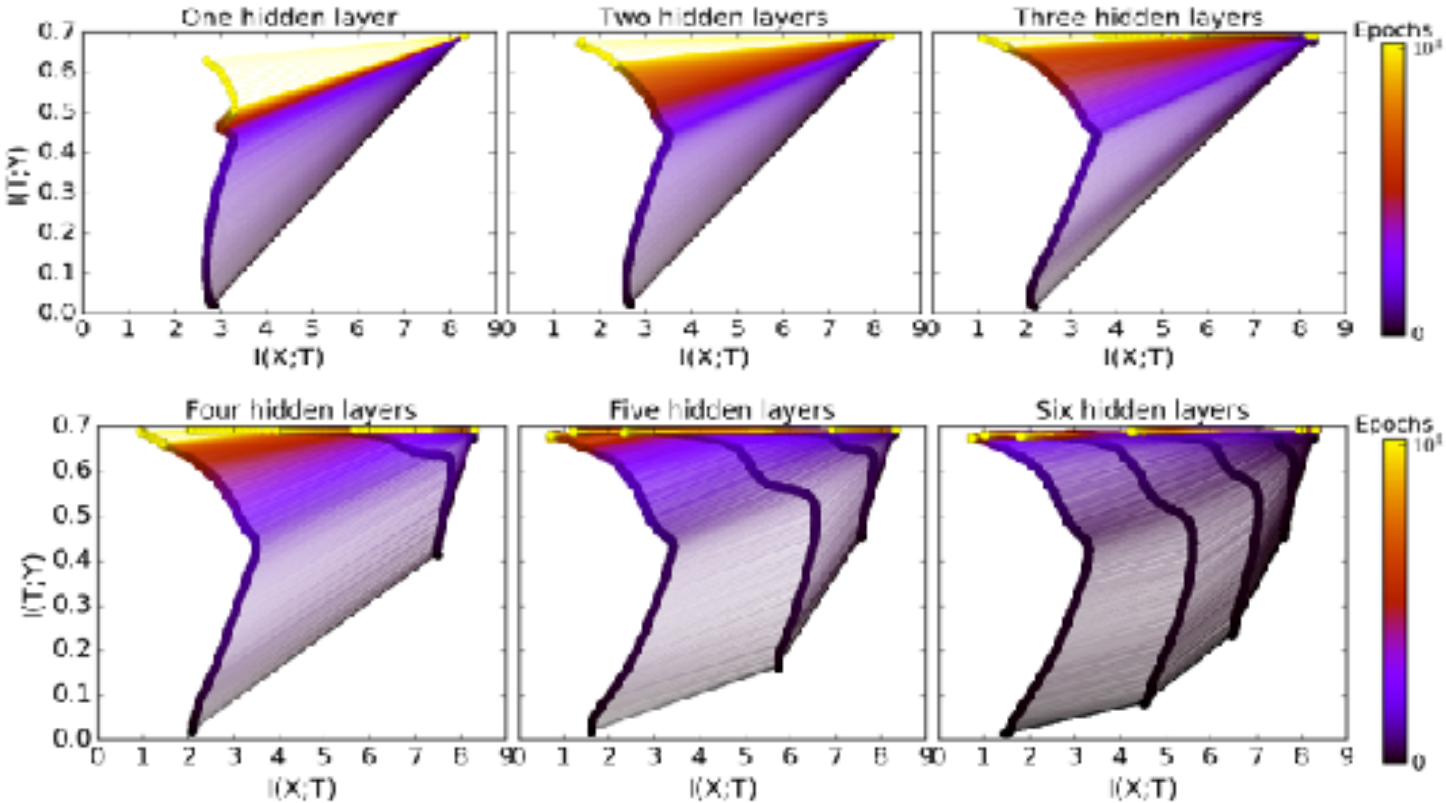
This is precisely the IB optimal encoder with $\tilde{\beta}_k$ – *the encoder k-layer noise*

Since $I(X;T_k) = H(X) - H(X \mid T_k)$, Max Entropy of the weights $\rightarrow$ Min $I(X;T_k)$

thus SGD converges, layer by layer, to a maximally compressed representation,

which is a SUCCESSIVELY REFINEABLE APPROXIMATION of the optimal IB bound!

The benefit of the hidden layers



More layers take much FEWER training epochs for good generalization.

The optimization time depend super-linearly (exponentially?) on the compressed information, delta Ix, for each layer.

Relaxation times and the benefit of the hidden layers

Noisy relaxation (SGD): $\frac{\partial W_k}{\partial t} = -\nabla E(W_k) + \beta_k^{-1}\xi(t)$, layer k, $\xi \sim N(0,1)$

$\Rightarrow$ Maximum Entropy (via Focker-Planck): $P_{Gibbs}(W_k) \propto \exp\left(-\beta_k E(W_k)\right)$

Relaxation time for non-strongly convex error: $\Delta t_k \sim \exp(\Delta S_k)$

Denote the layer compression be: $\Delta S_k = I(X;T_k) - I(X;T_{k-1})$

Since $\exp\left(\sum_k \Delta S_k\right) \gg \sum_k \exp\left(\Delta S_k\right) > \max_k \exp\left(\Delta S_k\right) \Rightarrow$

Exponential boost in the relaxation time with K layers!

Equilibration of Information Flow through the layers

The Information Capacity between two layers is bounded by the Gaussian capacity:

$$C_G(W_k) = \tfrac{1}{2}\log\left(1+\frac{P_k}{N_k}\right) = \tfrac{1}{2}\log\left(1+SNR_k\right)$$

The stochastic relaxation decreases the SNR of the irrelevant channels

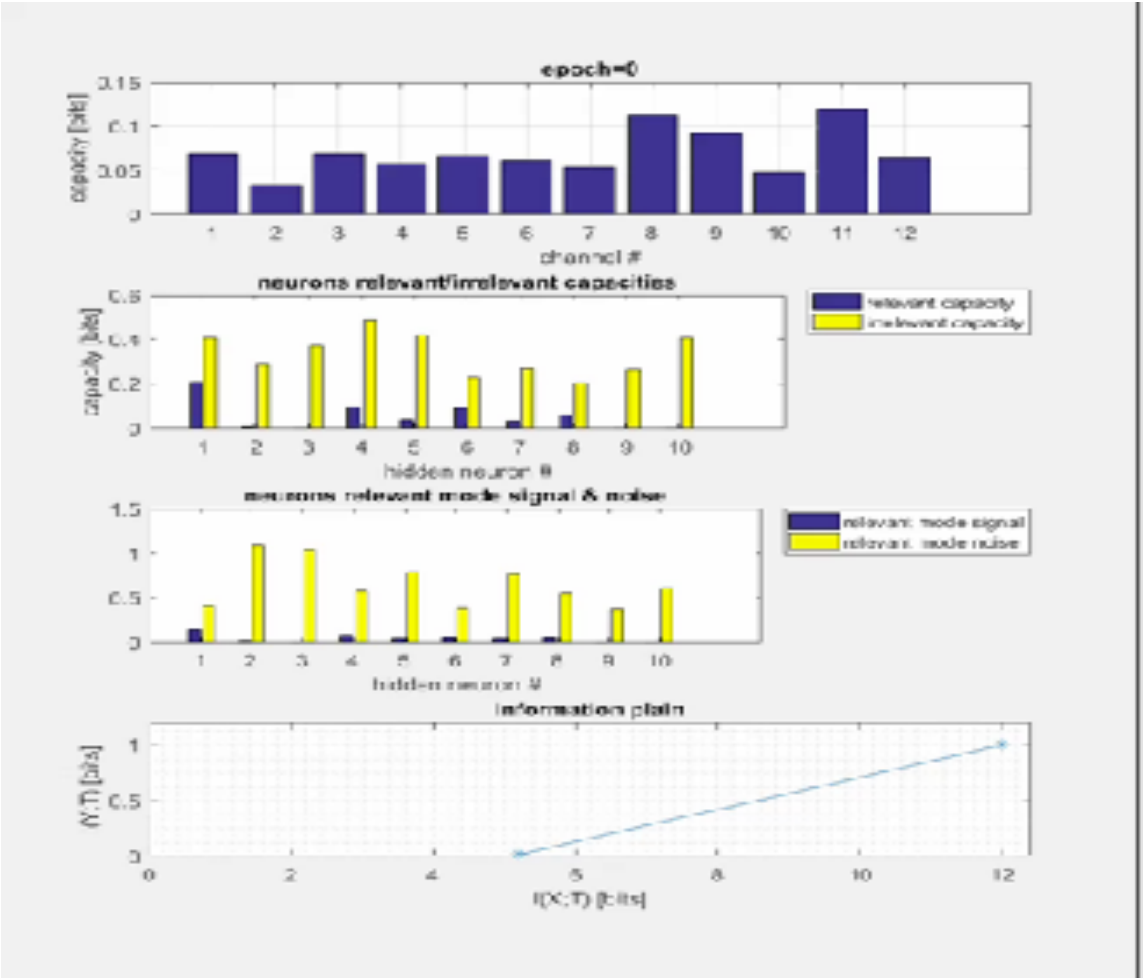
$\Rightarrow$ In optimized DNN only the relevant information $I(X;Y)$ flows through the network

$\Rightarrow$ $\mathrm{SNR}_k \approx const.$ we see this in the simulations.

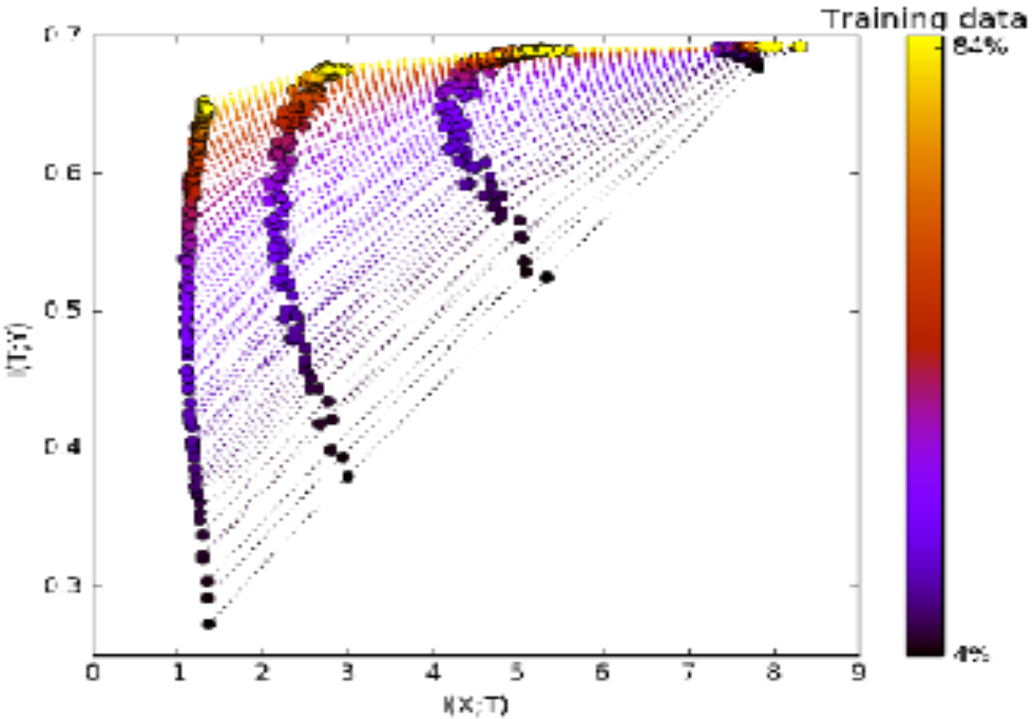
This can determine the final layers locations in the Information Plane...

Unless the stochastic relaxation stops through critical slowing down near phase transitions !

Layers relevant capacity equilibration

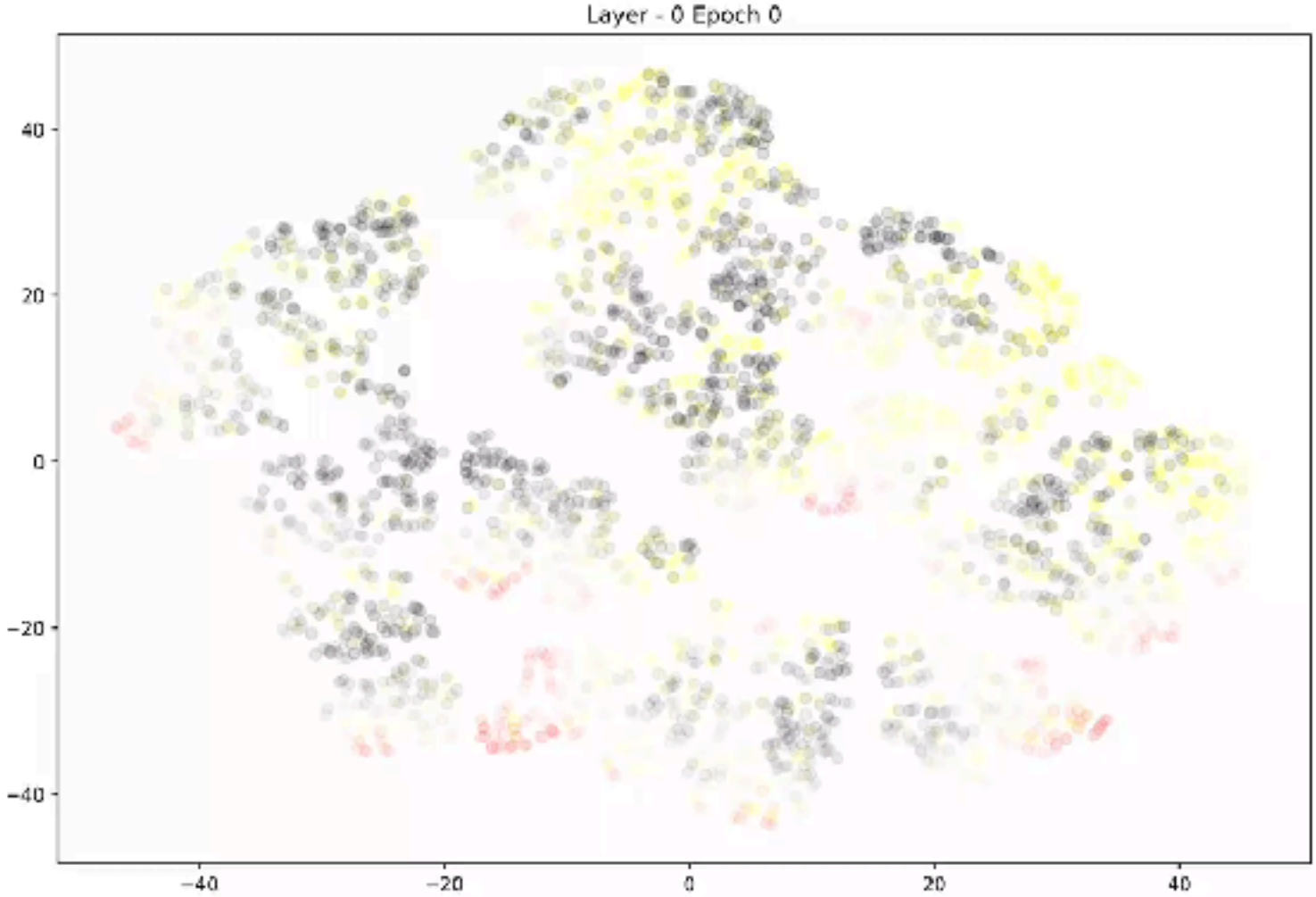


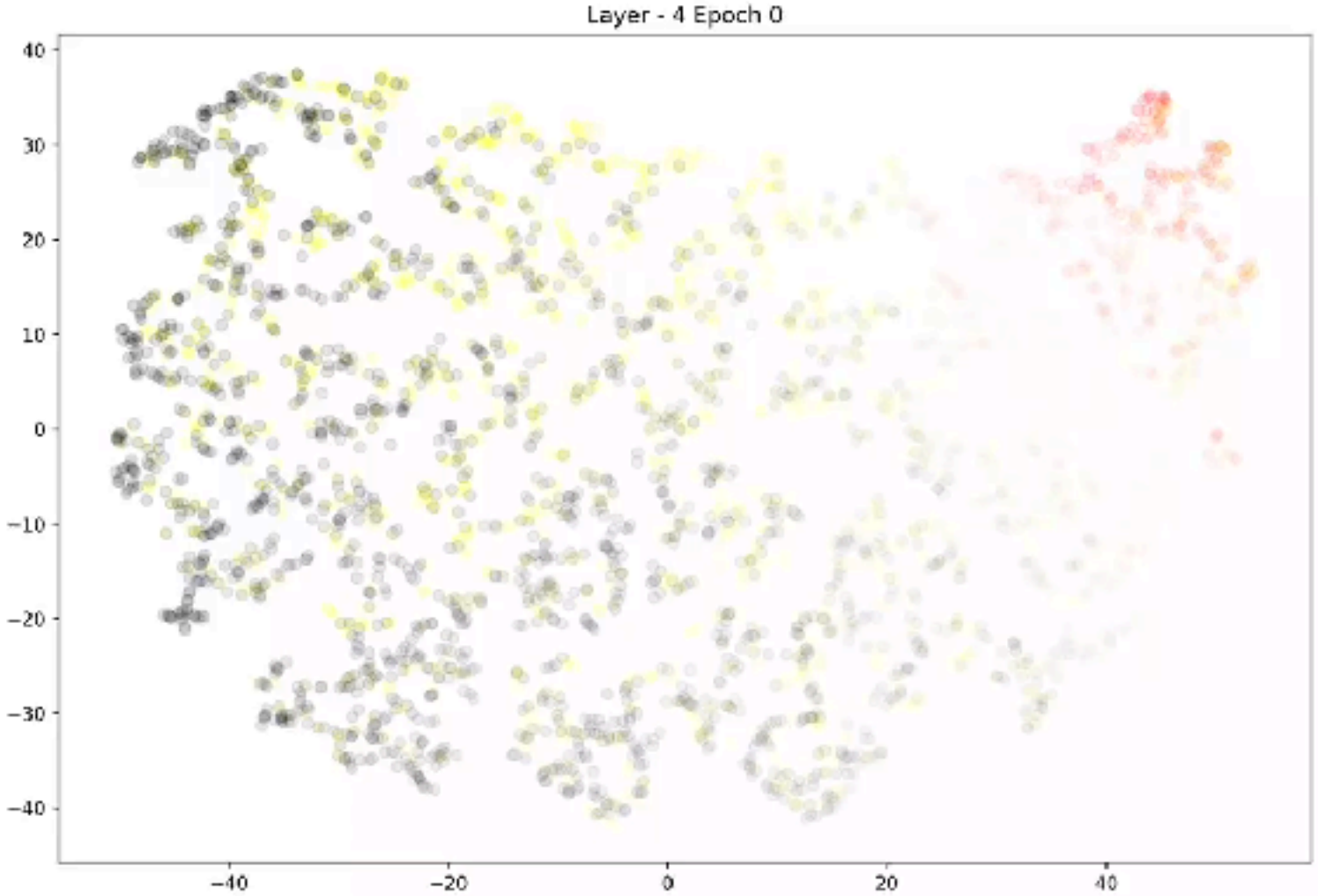
Simulations by Tomer Barak



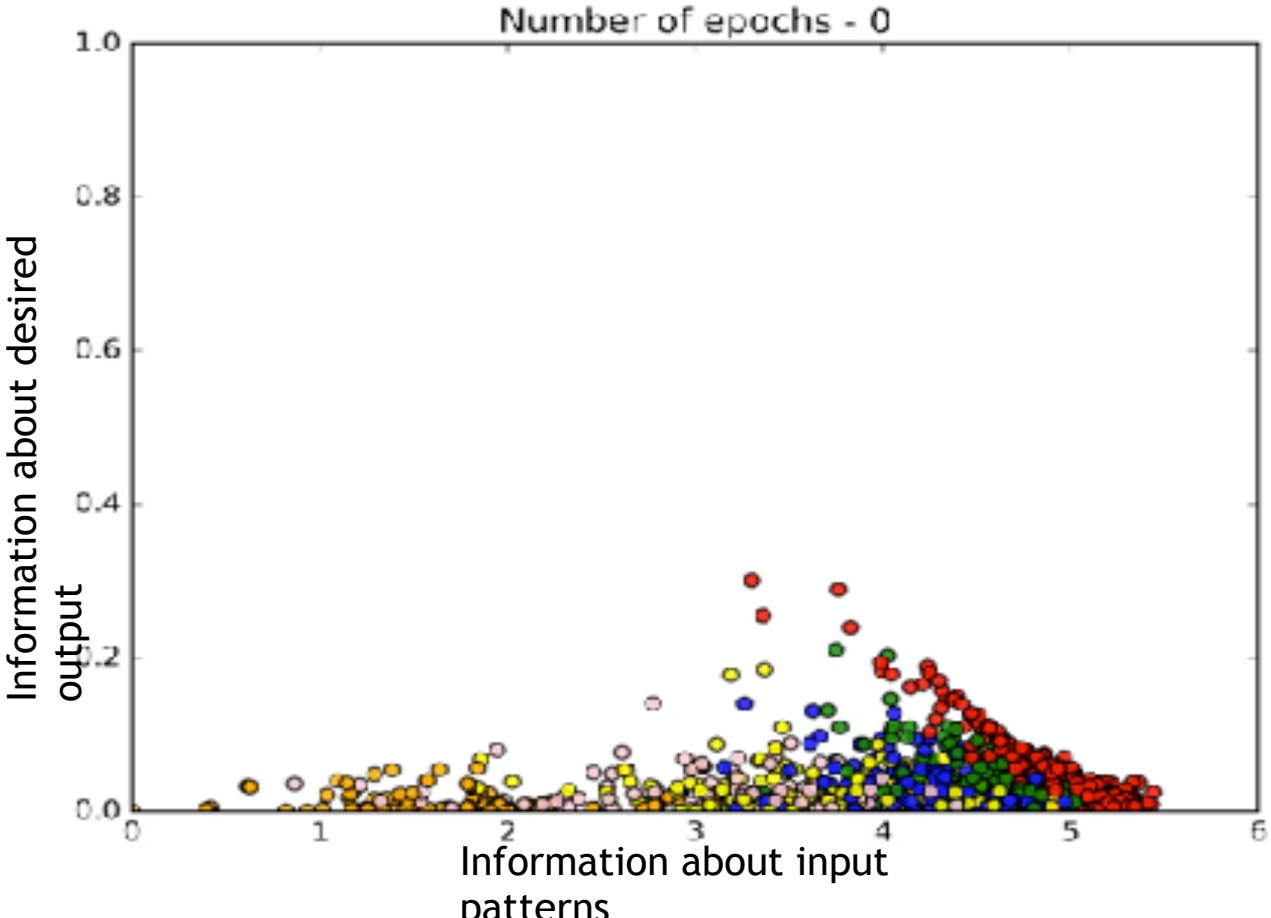
- Fitting larger training data require more information in the hidden layers.

- It is the **mutual-information of the last hidden layer**, which determines generalization (unlike standard hypothesis class bounds)

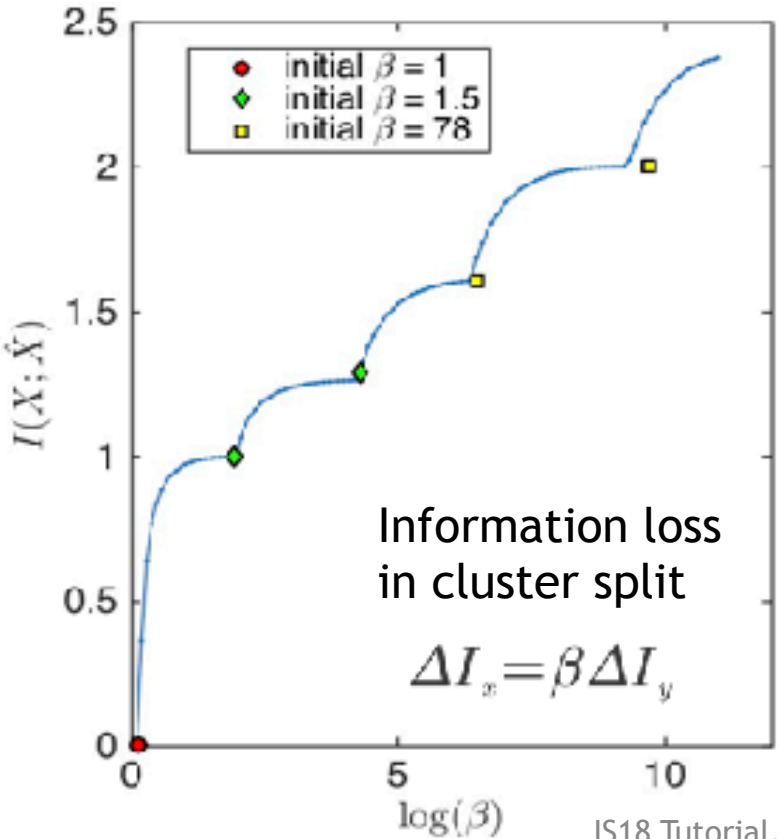

Layer - 0 Epoch 0
40
20
0
−20
−40
−40
−20
0
20
40

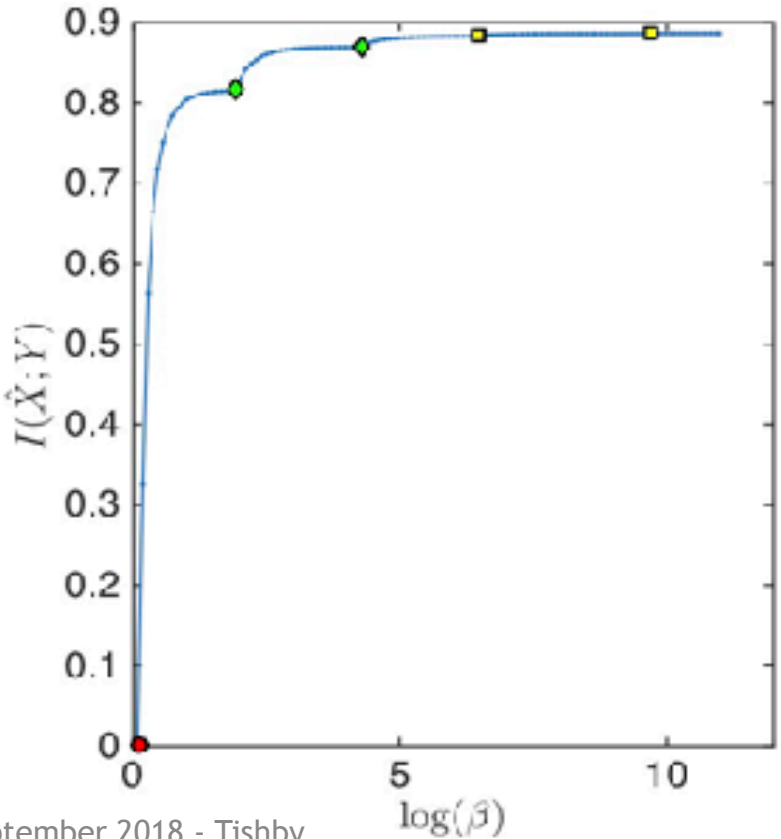

Layer - 4 Epoch 0
40
30
20
10
0
−10
−20
−30
−40
−40
−20
0
20
40

What do single neurons represent?



Second order phase transitions on the IB curve





The IB bifurcation (phase-transitions) points

The IB bifurcation points can be found as follows:

$$p_\beta(x|\hat{x}) = \frac{p(x)}{Z(x,\beta)} \exp(-\beta D[p(y|x) \| p_\beta(y|\hat{x})])$$

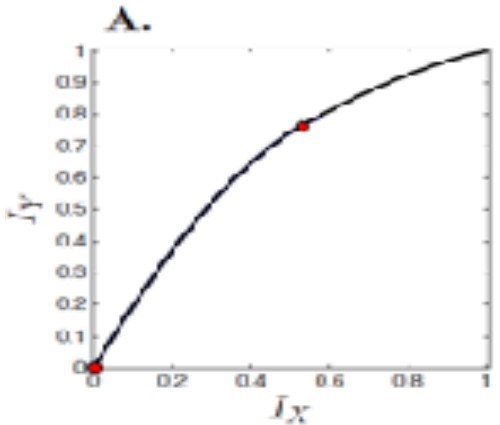
or $$\ln p_\beta(x|\hat{x}) = \ln \frac{p(x)}{Z(x,\beta)} - \beta D[p(y|x) \| p_\beta(y|\hat{x})]$$

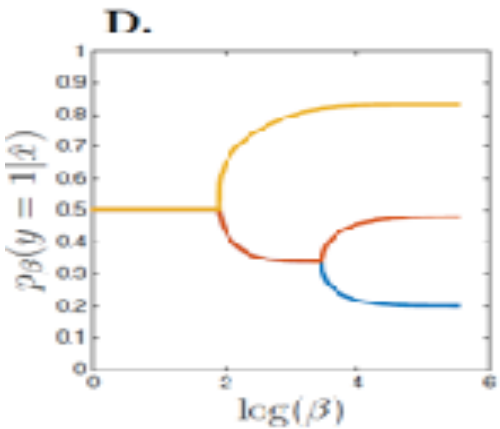
then:

$$\frac{\partial \ln p_\beta(x|\hat{x})}{\partial \hat{x}} = \beta \sum_y p(y|x) \frac{\partial \ln p_\beta(y|\hat{x})}{\partial \hat{x}}$$

similarily: $p_\beta(y|\hat{x}) = \sum_x p(y|x)\, p_\beta(x|\hat{x})$

$$\frac{\partial \ln p_\beta(y|\hat{x})}{\partial \hat{x}} = \frac{1}{p_\beta(y|\hat{x})} \sum_x p(y|x)\, p_\beta(x|\hat{x}) \frac{\partial \ln p_\beta(x|\hat{x})}{\partial \hat{x}}$$





The IB bifurcation (phase-transitions) points

Defining the matrices:

$$C_{xx'}(\hat{x},\beta) = \sum_y \frac{p(y|x)}{p_\beta(y|\hat{x})} p_\beta(x'|\hat{x}) p(y|x)$$

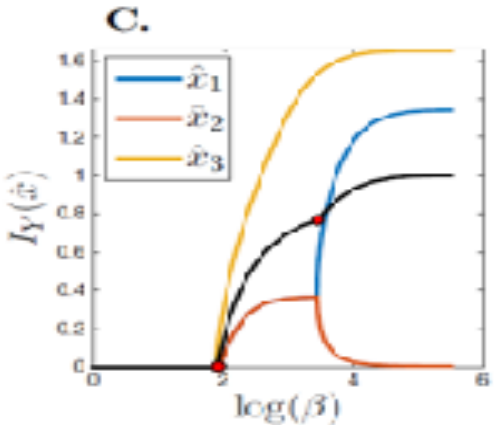
$$C_{yy'}(\hat{x},\beta) = \sum_x \frac{p(y|x)}{p_\beta(y|\hat{x})} p_\beta(x|\hat{x}) p(y'|x)$$

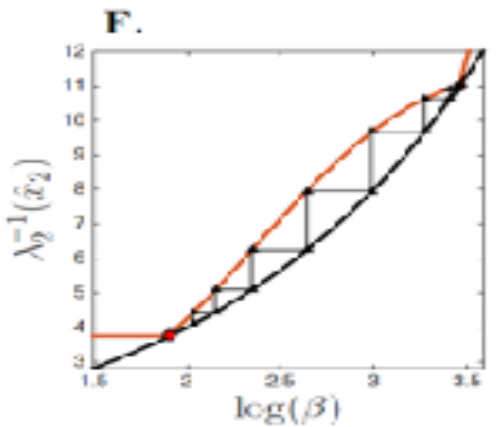
these equations can be combined into two (non-linear) eigenvalue problems:

$$\left[I - \beta C_{xx'}(\hat{x},\beta)\right] \frac{\partial \ln p_\beta(x'|\hat{x})}{\partial \hat{x}} = 0$$

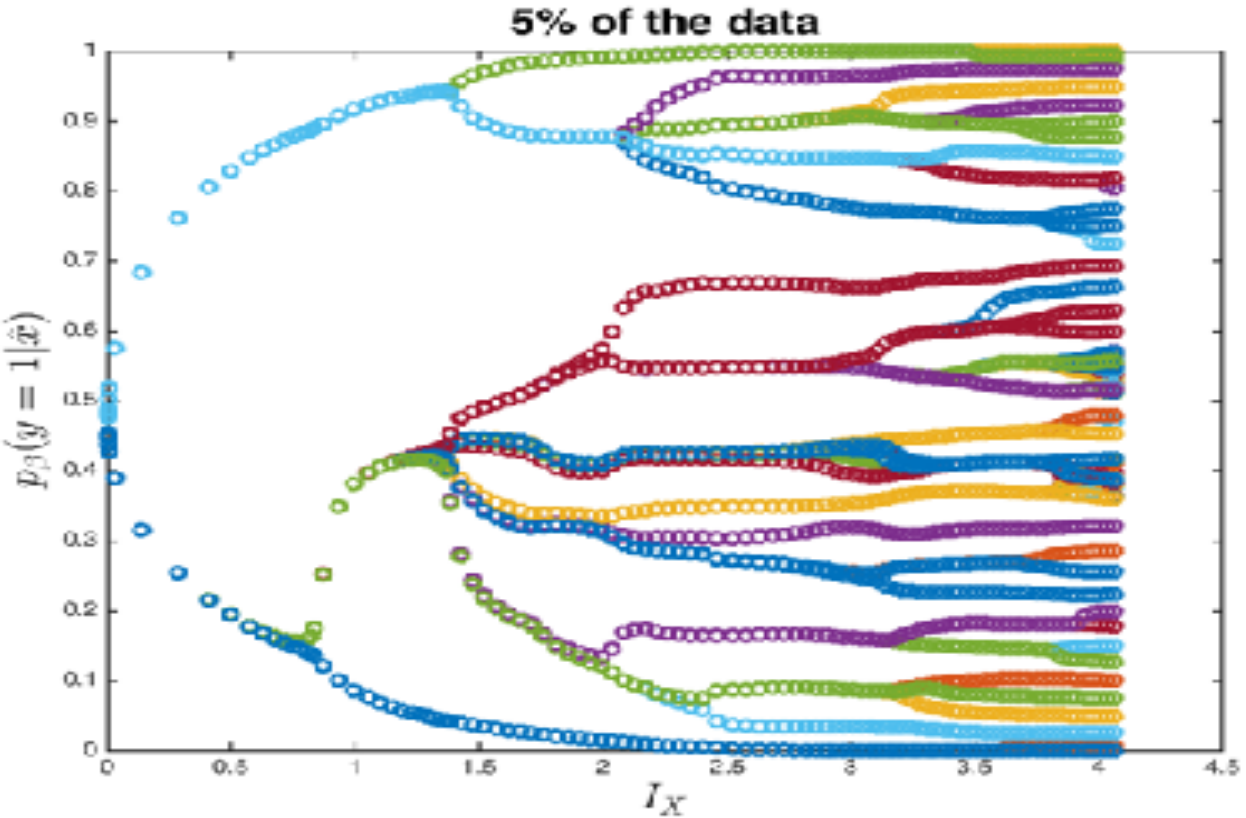
$$\left[I - \beta C_{yy'}(\hat{x},\beta)\right] \frac{\partial \ln p_\beta(y'|\hat{x})}{\partial \hat{x}} = 0$$

These eigenvalue problems have non-trivial solutions (eigenvectors) only at the critical bifurcation points (second order phase transitions).





Bifurcation diagrams in symmetric rule: layers diffusion slows down at phase transitions



Summary

- **The Information Plane provides a unique visualization of DL**
  - Most of the learning time goes to compression
  - Layers are learnt bottom up – and "help" each other
  - The layers converge to special (critical?) points on the IB bound
- **The advantage of the layers is mostly computational**
  - Relaxation times are super-linear (exponential?) in the Entropy gap
  - Hidden layers provide intermediate steps and boost convergence time
  - Hidden layers help in avoiding critical slowing down
- **Further directions**
  - Exactly solvable DNN models (through symmetry & group theory)
  - New/better learning algorithms & design principles
  - Predictions on the organization of biological layered networks …

Thank you!