Non-local manifold learning by regularized auto-encoders

Yoshua Bengio

U. Montreal

Thanks to: Pascal Vincent, Salah Rifai, Yann Dauphin, Grégoire Mesnil, Li Yao, Guillaume Alain + many more

October 22nd, 2013

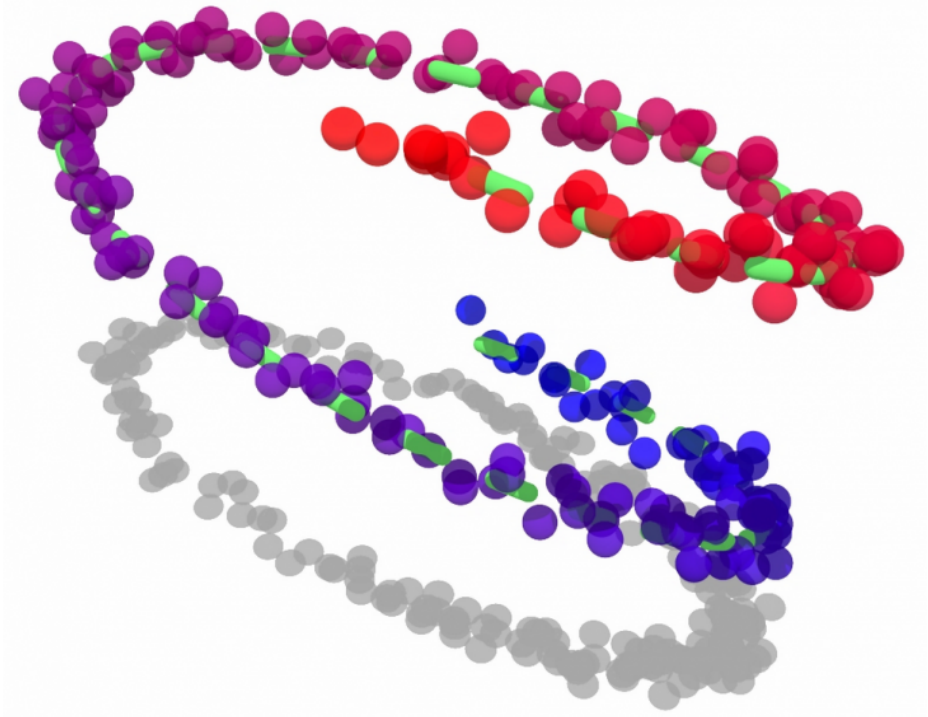
Masterclass lecture, UCL Center for Computational Statistics and Machine Learning

Université de Montréal

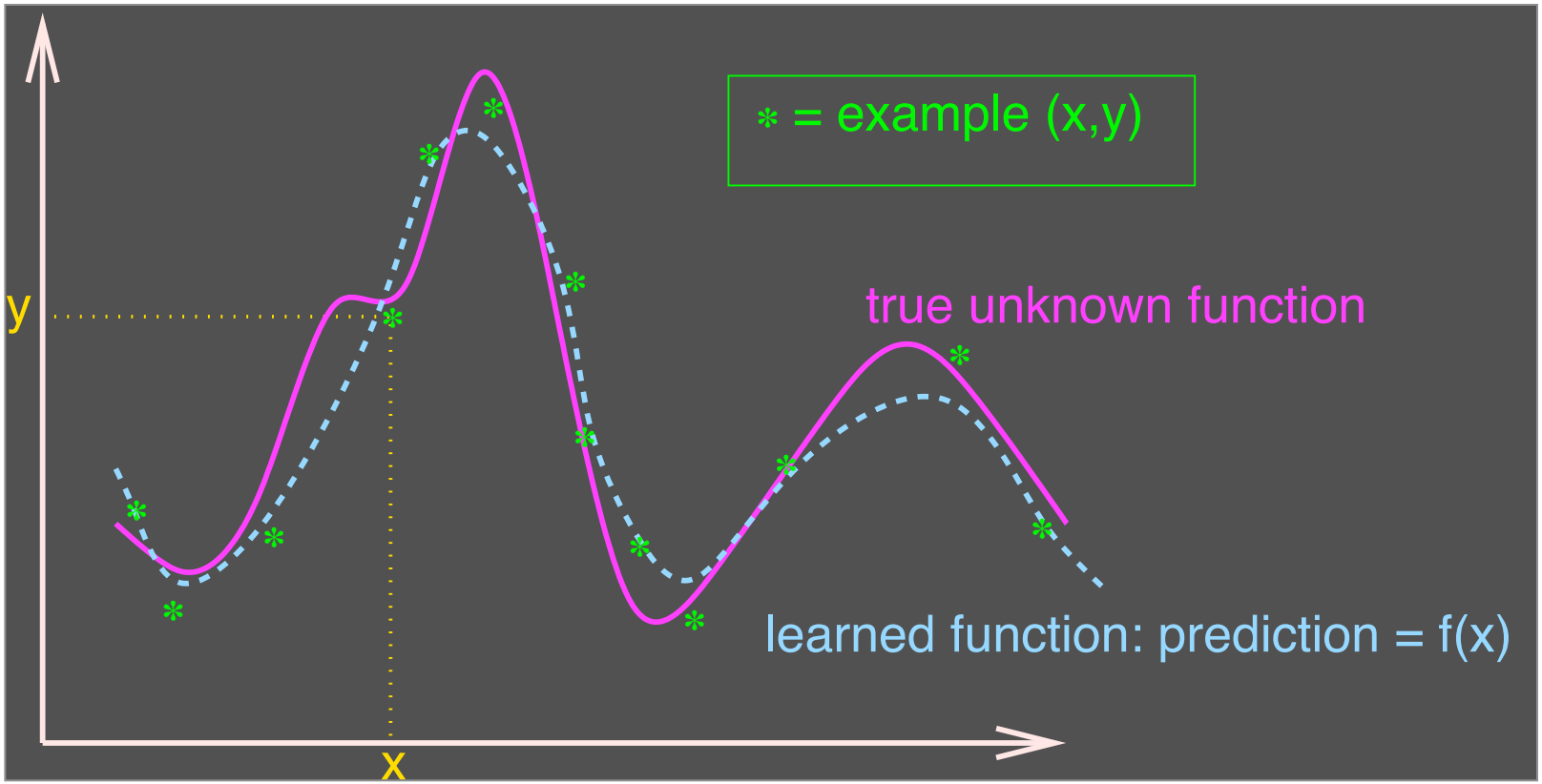
LISA

Geometrical view on machine learning

- Learning as the estimation of a probability function
- Generalization: guessing **where** probability mass concentrates
- Challenge: the curse of dimensionality (exponentially many configurations of the variables to consider)



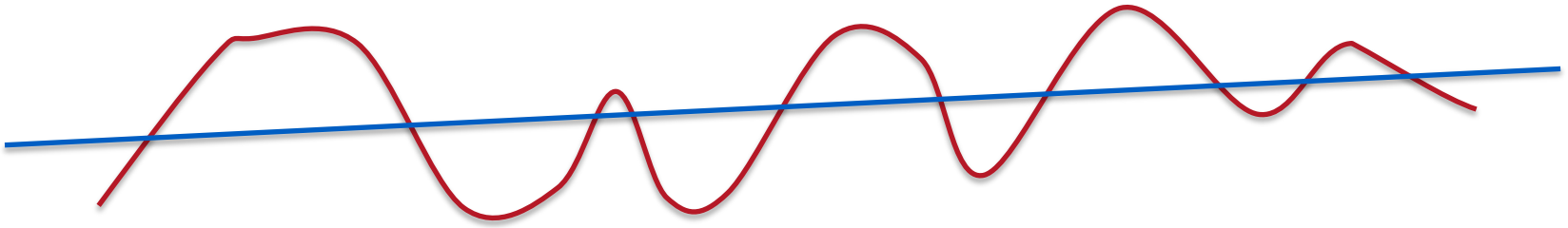
Easy Learning



Not Dimensionality so much as Number of Variations

(Bengio, Dellalleau & Le Roux 2007)

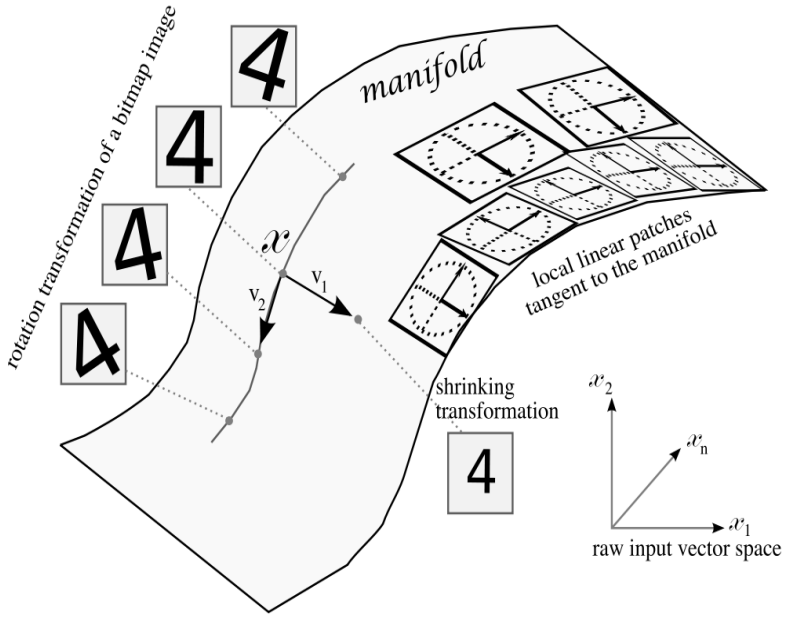
- Theorem: Gaussian kernel machines need at least *k* examples to learn a function that has *2k* zero-crossings along some line



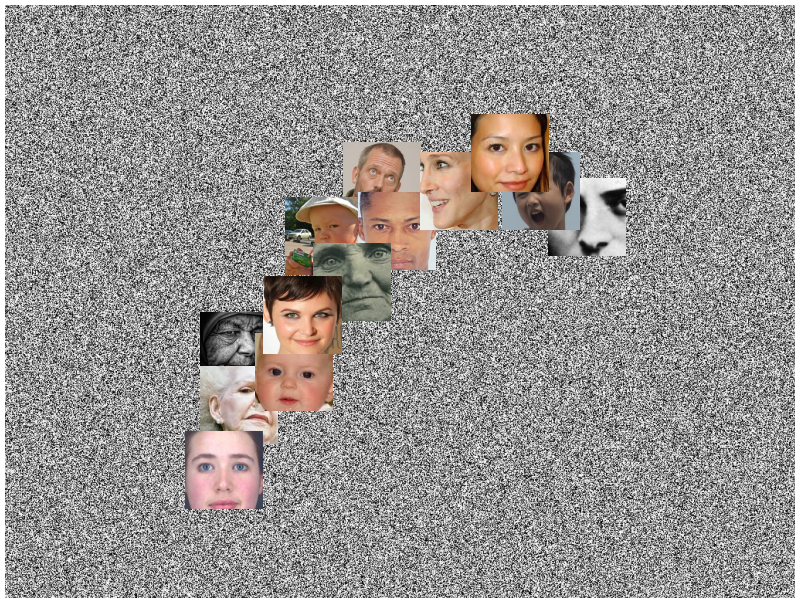
- Theorem: For a Gaussian kernel machine to learn some maximally varying functions over *d* inputs requires $O(2^d)$ examples

However, Real Data Are near Highly Curved Sub-Manifolds

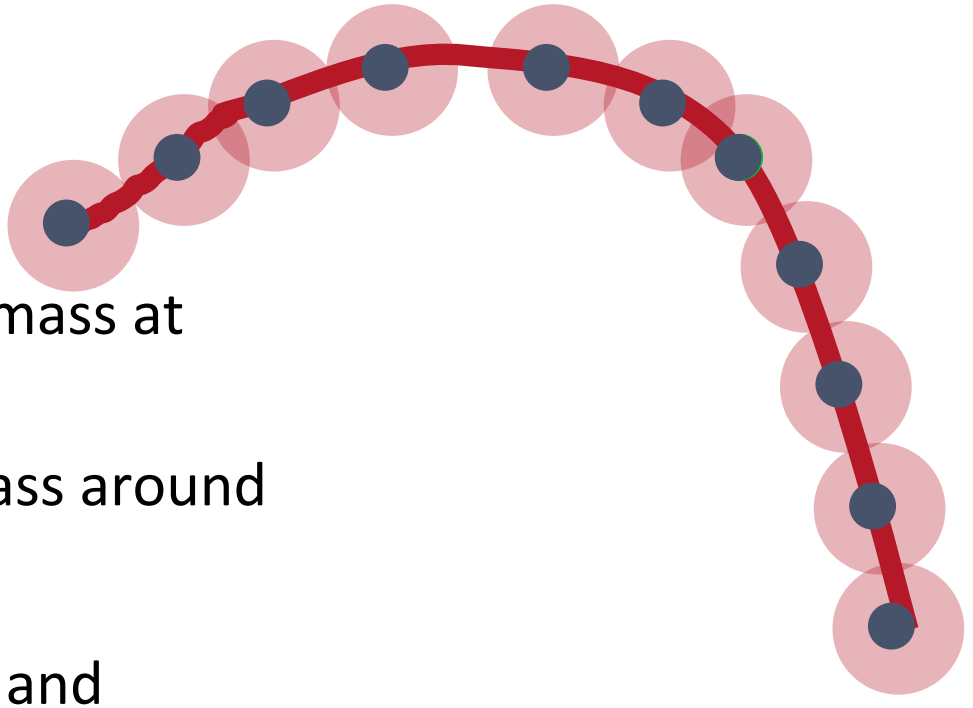
- Additional prior: examples **concentrate** near a lower dimensional "manifold" (region of high density with only few operations allowed which allow small changes while staying on the manifold)



- variable dimension locally?
- Soft # of dimensions?



Putting Probability Mass where Structure is Plausible



- Empirical distribution: mass at training examples
- Smoothness: spread mass around
- Insufficient
- Guess some 'structure' and generalize accordingly

Is there any hope to generalize non-locally?

Yes! Need good priors!

Bypassing the curse

We need to build compositionality into our ML models

Just as human languages exploit compositionality to give representations and meanings to complex ideas

Exploiting compositionality gives an exponential gain in representational power

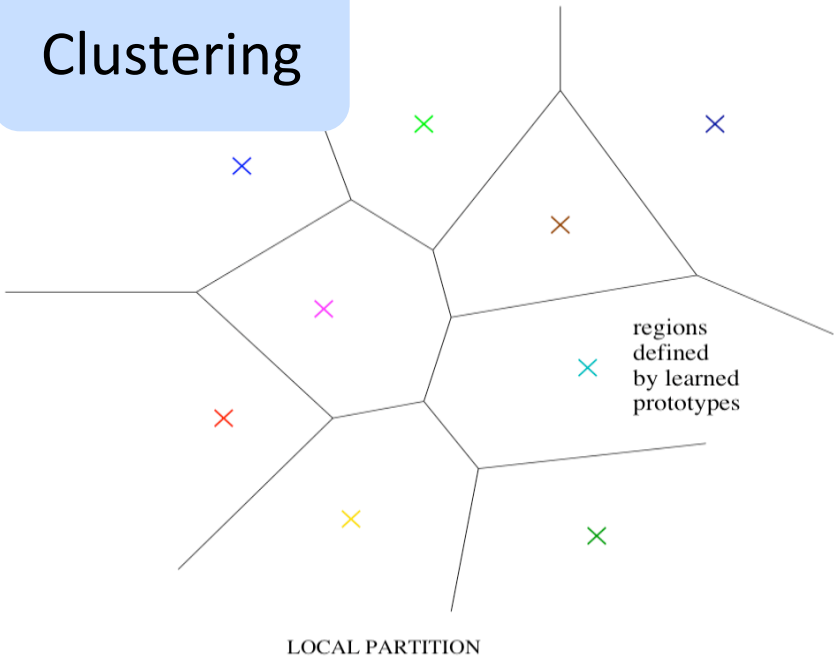
Distributed representations / embeddings: feature learning

Deep architecture: multiple levels of feature learning

Prior: compositionality is useful to describe the world around us efficiently

The need for distributed representations

Clustering



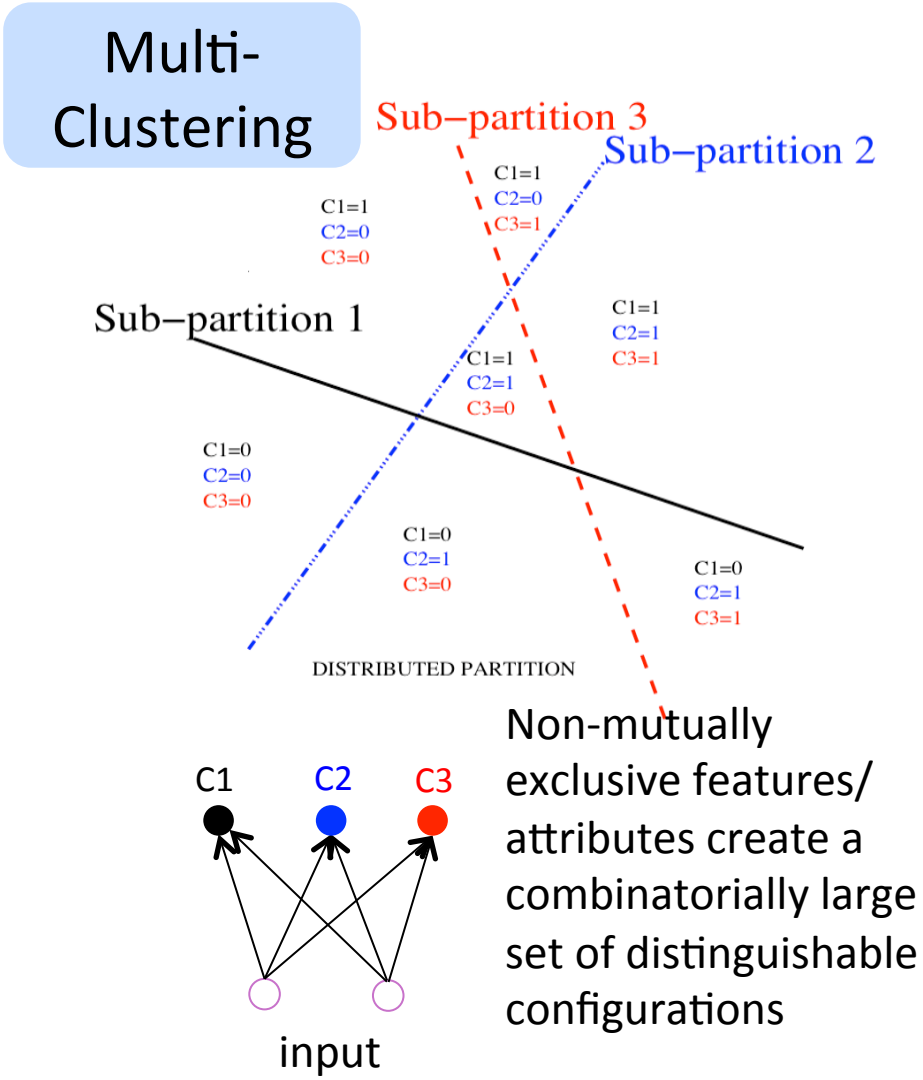
- Clustering, Nearest-Neighbors, RBF SVMs, local non-parametric density estimation & prediction, decision trees, etc.

- Parameters for each distinguishable region
- **# of distinguishable regions is linear in # of parameters**

→ No non-trivial generalization to regions without examples

The need for distributed representations

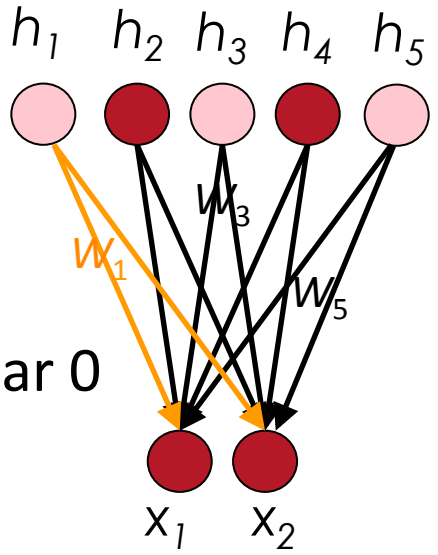
- Factor models, RBMs, Neural Nets, Sparse Coding, Deep Learning, etc.
- Each parameter influences many regions, not just local neighbors
- **# of distinguishable regions grows almost exponentially with # of parameters**
- **GENERALIZE NON-LOCALLY TO NEVER-SEEN REGIONS**

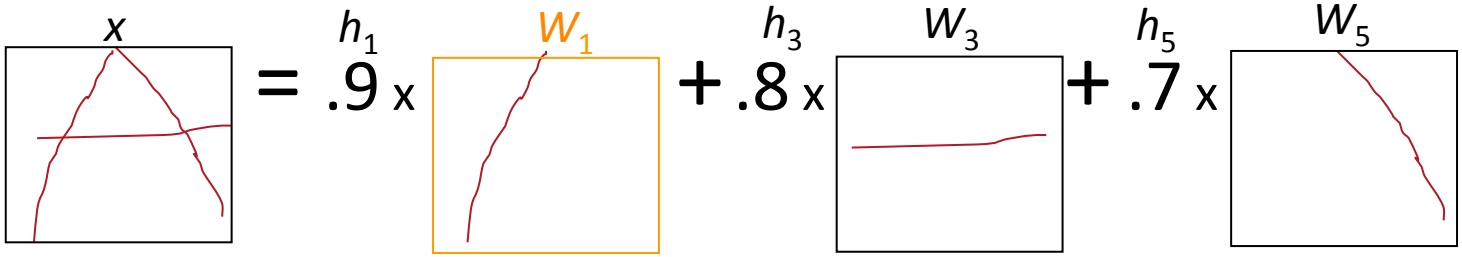


Directed Factor Models: $P(x,h)=P(h)P(x|h)$

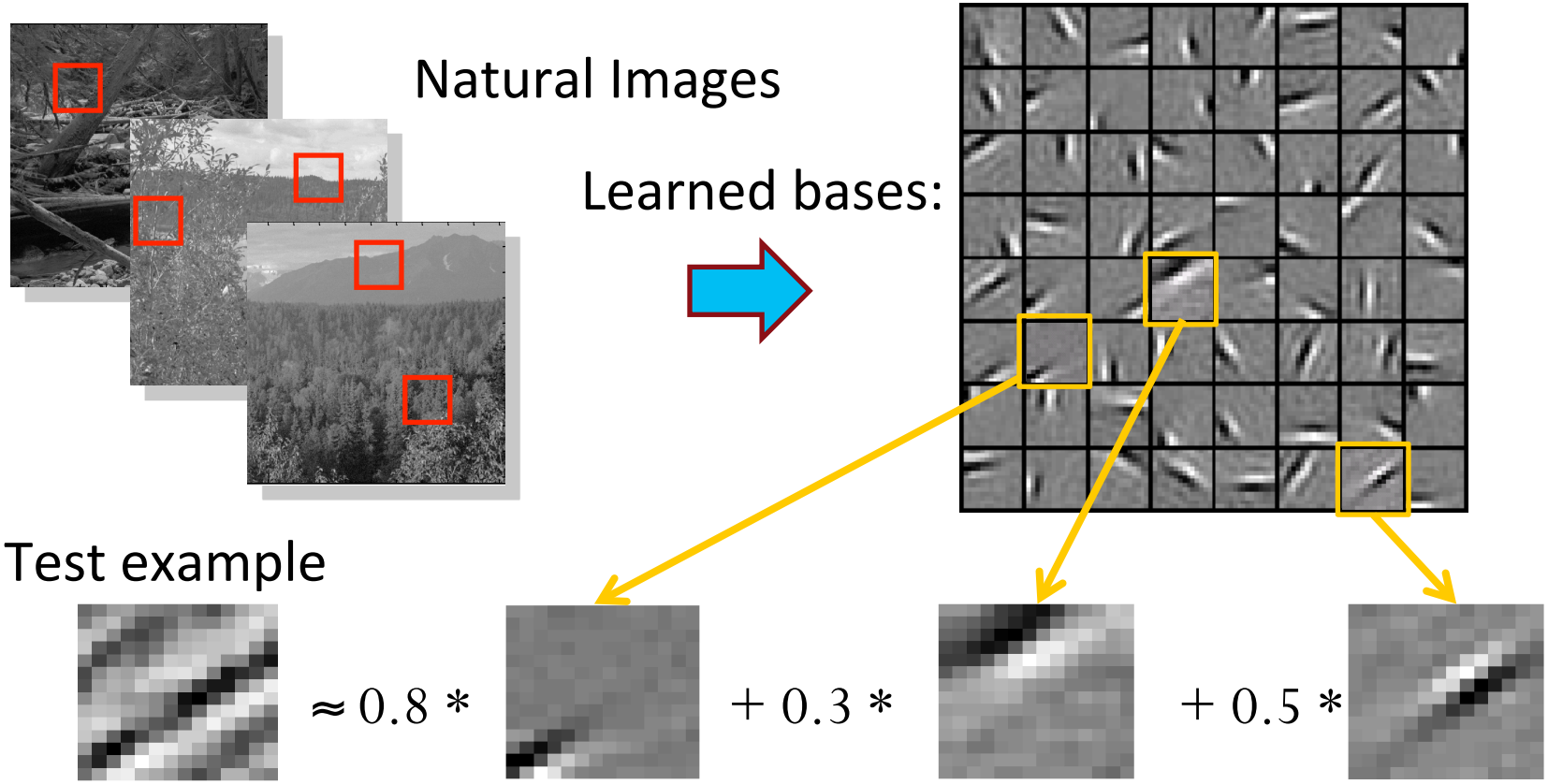
factors prior likelihood

- P(h) factorizes into P(h_1) P(h_2)…
- Different priors:
  - PCA: P(h_i) is Gaussian
  - ICA: P(h_i) is non-parametric
  - **Sparse coding**: P(h_i) is concentrated near 0
- Likelihood is typically Gaussian x | h with mean given by $W^T h$
- Inference procedures (predicting h, given x) differ
- Sparse h: x is explained by the weighted addition of selected filters h_i





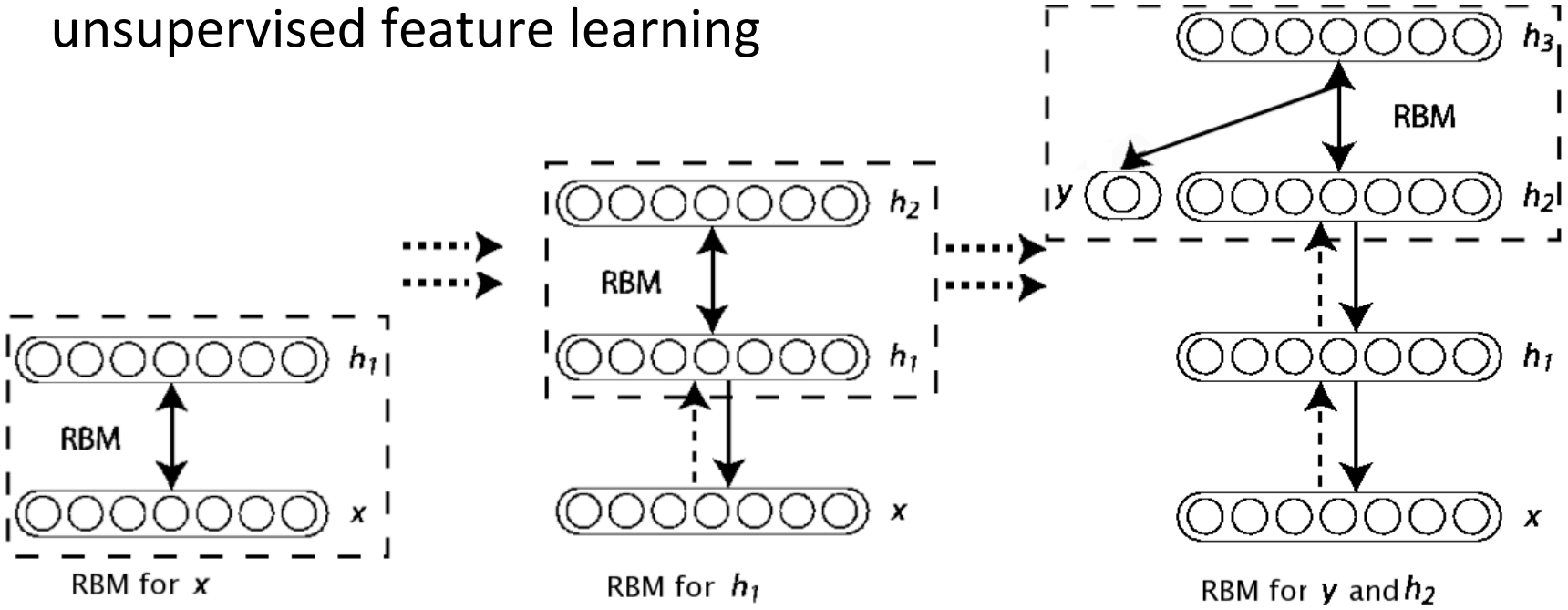
Sparse autoencoder illustration for images



$[h_1, \ldots, h_{64}] = [0, 0, \ldots, 0, \mathbf{0.8}, 0, \ldots, 0, \mathbf{0.3}, 0, \ldots, 0, \mathbf{0.5}, 0]$
(feature representation)

Stacking Single-Layer Learners

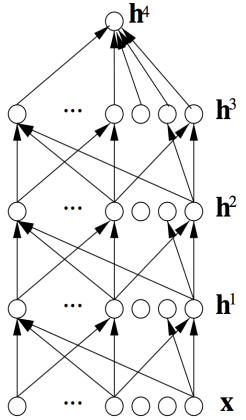
- PCA is great but can't be stacked into deeper more abstract representations (linear x linear = linear)
- One of the big ideas from Hinton et al. 2006: layer-wise unsupervised feature learning



Stacking Restricted Boltzmann Machines (RBM) → Deep Belief Network (DBN)

Auto-Encoders & Variants: Learning a computational graph

Computational Graphs

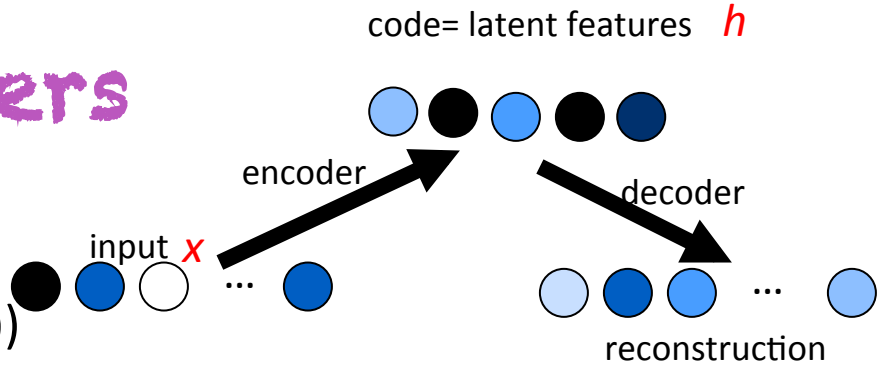


- Operations for particular task
- Neural nets' structure = computational graph for $P(y|\boldsymbol{x})$
- Graphical model's structure ≠ computational graph for inference
- Recurrent nets & graphical models

➔ **family of computational graphs sharing parameters**

- *Could we have a parametrized family of computational graphs defining "the model"?*

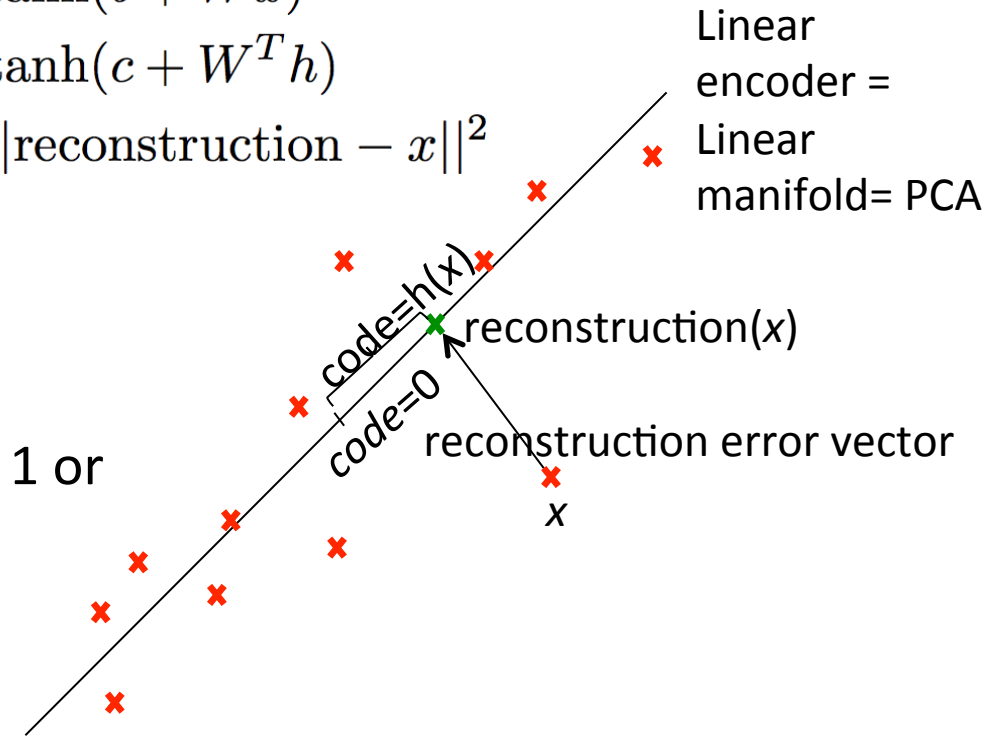
Simple Auto-Encoders



- MLP whose target output = input
- Reconstruction=decoder(encoder(input))

e.g.

$$h = \tanh(b + Wx)$$
$$\text{reconstruction} = \tanh(c + W^T h)$$
$$\text{Loss } L(x, \text{reconstruction}) = ||\text{reconstruction} - x||^2$$



- With bottleneck, code = new coordinate system
- Encoder and decoder can have 1 or more layers
- Training deep auto-encoders notoriously difficult

Contractive Auto-Encoders

(Rifai, Vincent, Muller, Glorot, Bengio ICML 2011; Rifai, Mesnil, Vincent, Bengio, Dauphin, Glorot ECML 2011; Rifai, Dauphin, Vincent, Bengio, Muller NIPS 2011)

$$\text{reconstruction}(x) = g(h(x)) = \text{decoder}(\text{encoder}(x))$$

Training criterion:

$$\mathcal{J}_{CAE}(\theta) = \sum_{x \in D_n} \lambda \sum_{ij} \left(\frac{\partial h_j(x)}{\partial x_i} \right)^2 + L(x, \text{reconstruction}(x))$$

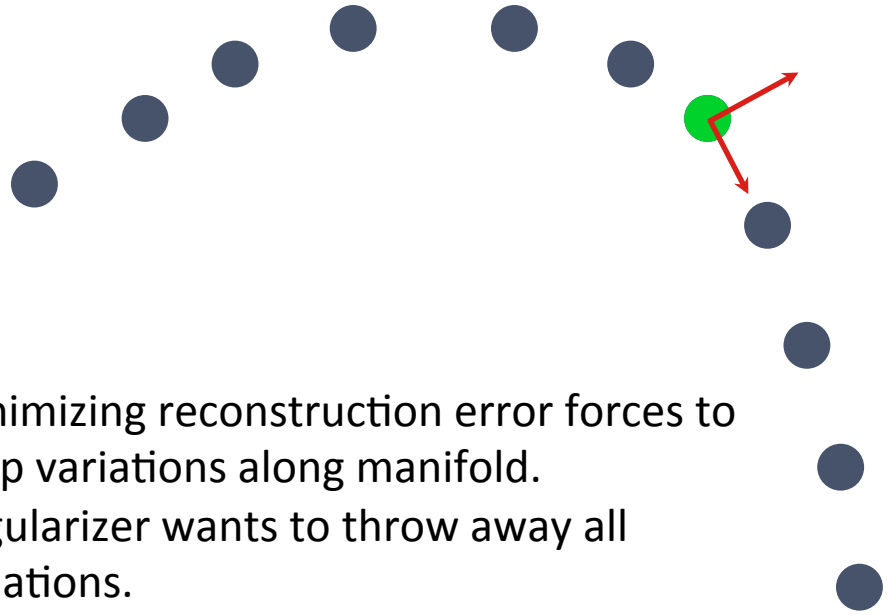
wants contraction in all directions

cannot afford contraction in manifold directions

If $h_j = \text{sigmoid}(b_j + W_j x)$

$(dh_j(x)/dx_i)^2 = h_j^2(1-h_j)^2 W_{ji}^2$

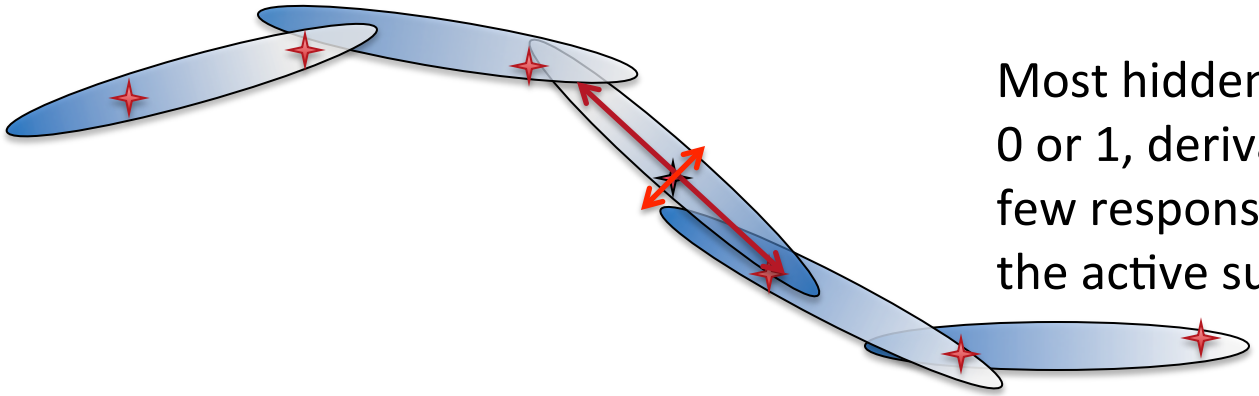
Auto-Encoders Learn Salient Variations, like a non-linear PCA



- Minimizing reconstruction error forces to keep variations along manifold.
- Regularizer wants to throw away all variations.
- With both: keep ONLY sensitivity to variations ON the manifold.

Contractive Auto-Encoders

(Rifai, Vincent, Muller, Glorot, Bengio ICML 2011; Rifai, Mesnil, Vincent, Bengio, Dauphin, Glorot ECML 2011; Rifai, Dauphin, Vincent, Bengio, Muller NIPS 2011)

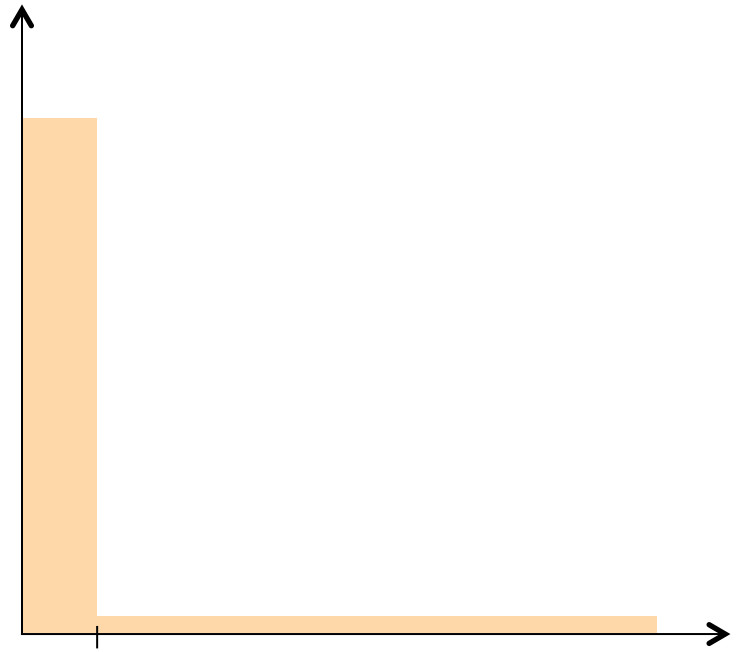


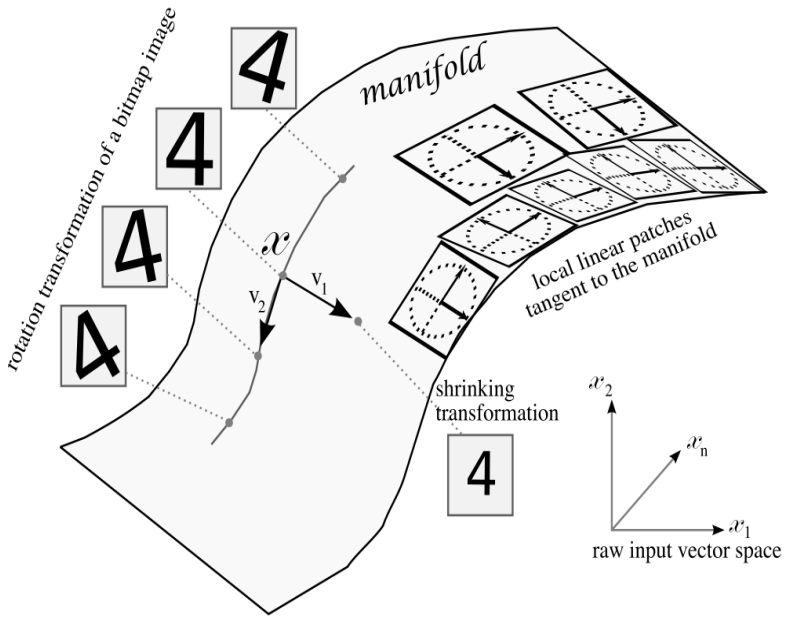
Most hidden units saturate (near 0 or 1, derivative near 0):
few responsive units represent the active subspace (local chart)

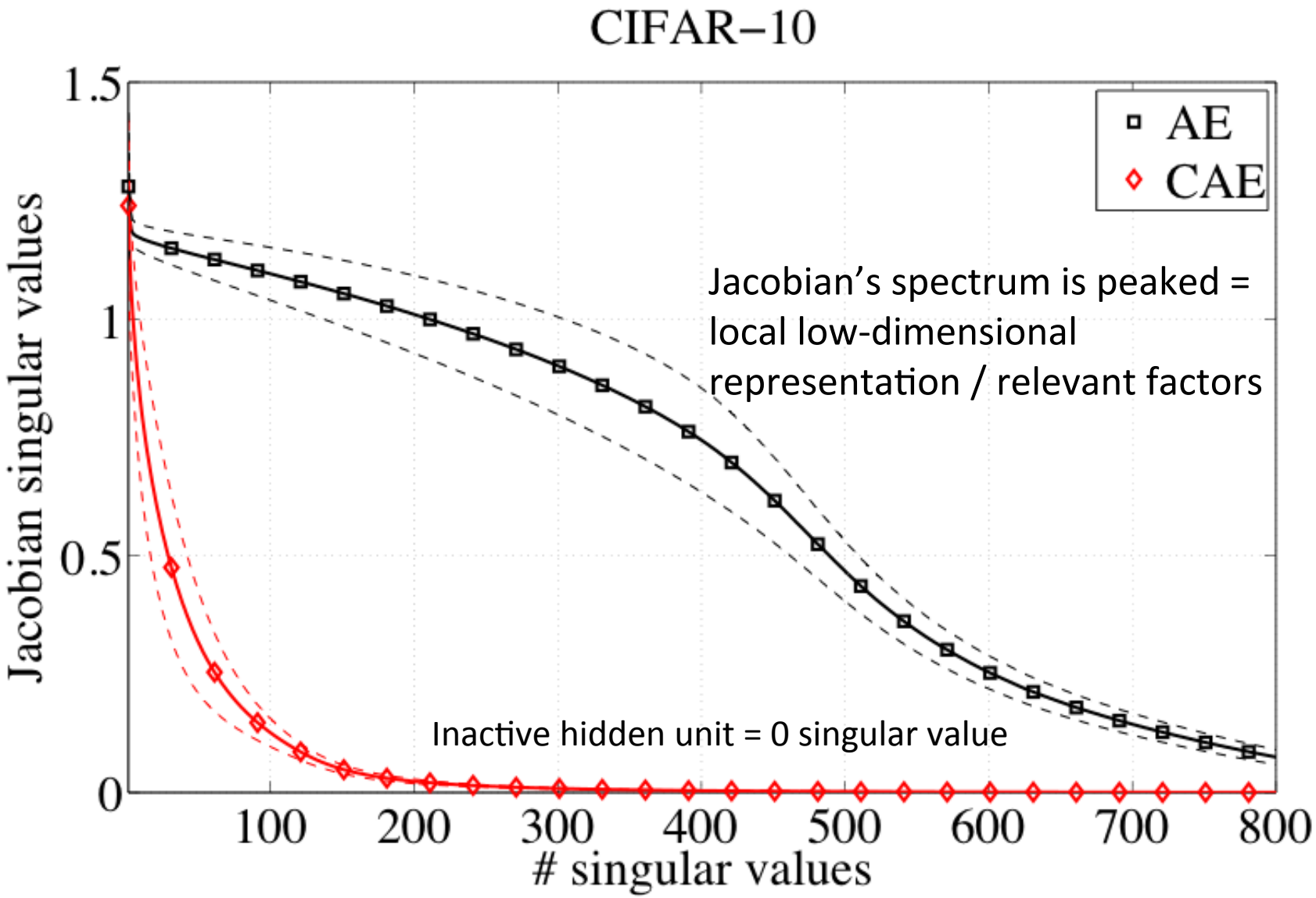
Each region/chart = subset of active hidden units
Neighboring region: one of the units becomes active/inactive
Unlike PCA:
SHARED SET OF FILTERS ACROSS REGIONS, EACH USING A SUBSET,
Multi-clustering instead of clustering

Coordinate System & Eigenspectrum

- Ideal spectrum of dh/dx for manifolds

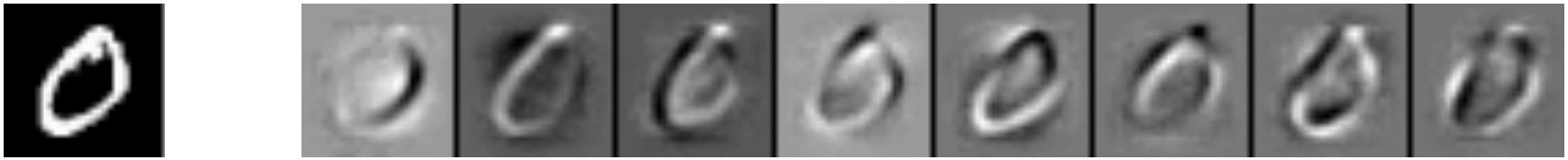


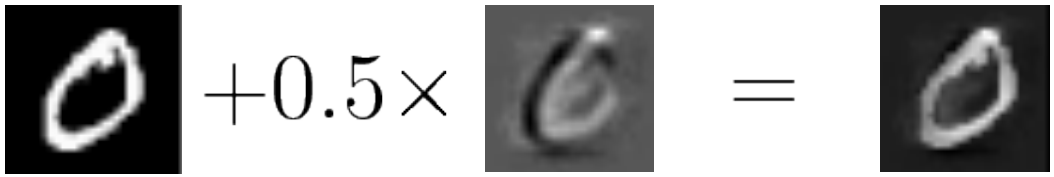



CIFAR−10
AE
CAE
Jacobian's spectrum is peaked = local low-dimensional representation / relevant factors
Inactive hidden unit = 0 singular value
Jacobian singular values
singular values
1.5
1
0.5
0
100
200
300
400
500
600
700
800

Input Point

Tangents

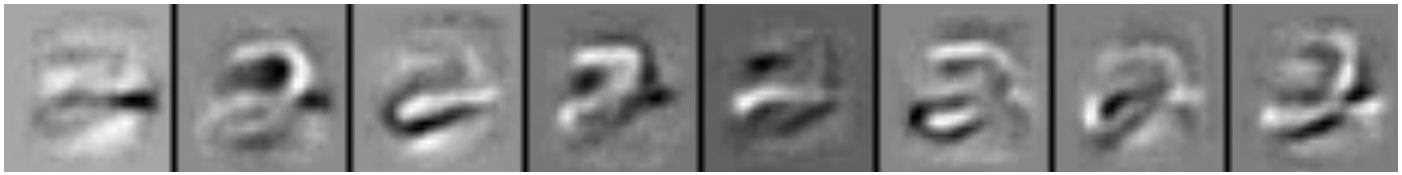




MNIST

Input Point

Tangents



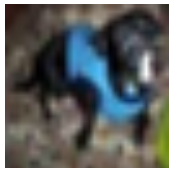
MNIST Tangents

Distributed vs Local (CIFAR-10 unsupervised)

Input Point

Tangents

Local PCA (no sharing across regions)



Contractive Auto-Encoder

Learned Tangent Prop: the Manifold Tangent Classifier

(Rifai et al NIPS 2011)

3 hypotheses:

1. Semi-supervised hypothesis (P(x) related to P(y|x))

2. Unsupervised manifold hypothesis (data concentrates near low-dim. manifolds)

3. Manifold hypothesis for classification (low density between class manifolds)

Learned Tangent Prop: the Manifold Tangent Classifier

Algorithm:

1. Estimate local principal directions of variation U(x) by CAE (principal singular vectors of dh(x)/dx)

2. Penalize f(x)=P(y|x) predictor by || df/dx U(x) ||

Makes f(x) insensitive to variations on manifold at x, tangent plane characterized by U(x).

Manifold Tangent Classifier Results

- Leading singular vectors on MNIST, CIFAR-10, RCV1:

Trading & Markets								
	+gilt	-slow	+matur	-percent	+bln	-anti	+interest	-sen
	+yen	-term	+auction	-sent	+coupon	-predict	+calcul	-californ
	+usda	-debt	+treasur	-pressure	+discount	-belgian	+overnight	-introduc

- **Knowledge-free MNIST: 0.81% error**

K-NN	NN	SVM	DBN	CAE	DBM	CNN	MTC
3.09%	1.60%	1.40%	1.17%	1.04%	0.95%	0.95%	**0.81%**

- Semi-sup.

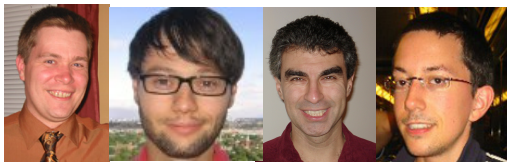
	NN	SVM	CNN	TSVM	DBN-rNCA	EmbedNN	CAE	MTC
100	25.81	23.44	22.98	16.81	-	16.86	13.47	**12.03**
600	11.44	8.85	7.68	6.16	8.7	5.97	6.3	**5.13**
1000	10.7	7.77	6.45	5.38	-	5.73	4.77	**3.64**
3000	6.04	4.21	3.35	3.45	3.3	3.59	3.22	**2.57**

- Forest (500k examples)

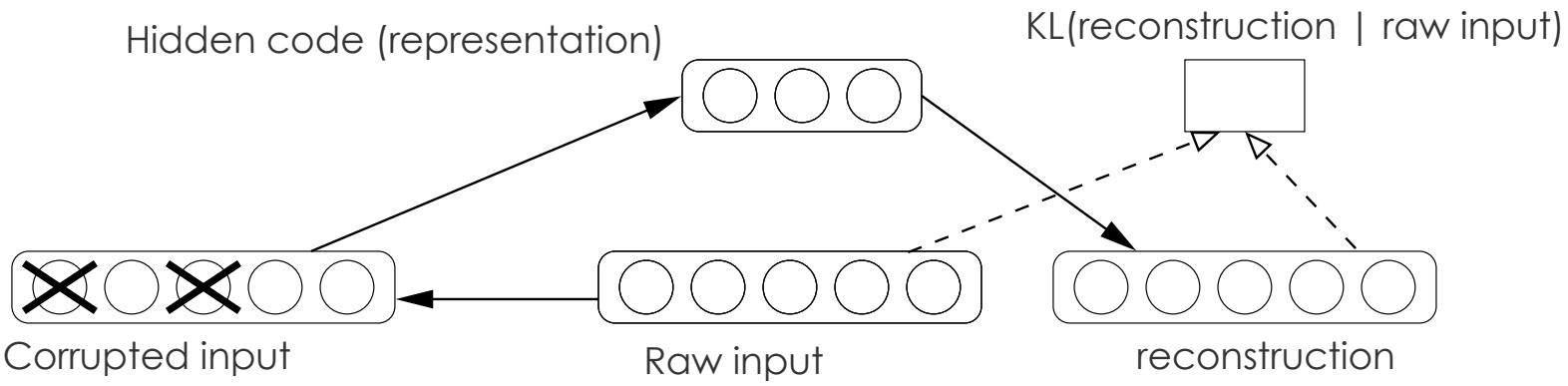
SVM	Distributed SVM	MTC
4.11%	3.46%	**3.13%**

Denoising Auto-Encoder

(Vincent et al 2008)



- Corrupt the input during training only
- Train to reconstruct the uncorrupted input



- Encoder & decoder: any parametrization
- As good or better than RBMs for unsupervised pre-training

Denoising Auto-Encoder

- Learns a vector field pointing towards higher probability direction (Alain & Bengio 2013)

$$r(x)-x \propto d\log p(x)/dx$$

- Some DAEs correspond to a kind of Gaussian RBM with *regularized* Score Matching (Vincent 2011, Swersky et al 2011)

  [equivalent when noise→0]

- Compared to RBM:

No partition function issue,
+ can measure training criterion

Denoising auto-encoders are also contractive!

- Taylor-expand Gaussian corruption noise in reconstruction error:

$$
\begin{aligned}
E\left[\ell(x, r(x+\epsilon))\right] &\approx E\left[\left(x - \left(r(x) + \frac{\partial r(x)}{\partial x}\epsilon\right)\right)^T \left(x - \left(r(x) + \frac{\partial r(x)}{\partial x}\epsilon\right)\right)\right] \\
&= E\left[\|x - r(x)\|^2\right] + \sigma^2 E\left[\left\|\frac{\partial r(x)}{\partial x}\right\|_F^2\right]
\end{aligned}
$$

- Yields a contractive penalty in the reconstruction function (instead of encoder) proportional to amount of corruption noise

What is the probabilistic interpretation of denoising auto-encoders?

Can we sample from the learned distribution?

First Theoretical Results on Probabilistic Interpretation of Auto-Encoders (Vincent 2011, Alain & Bengio 2013)

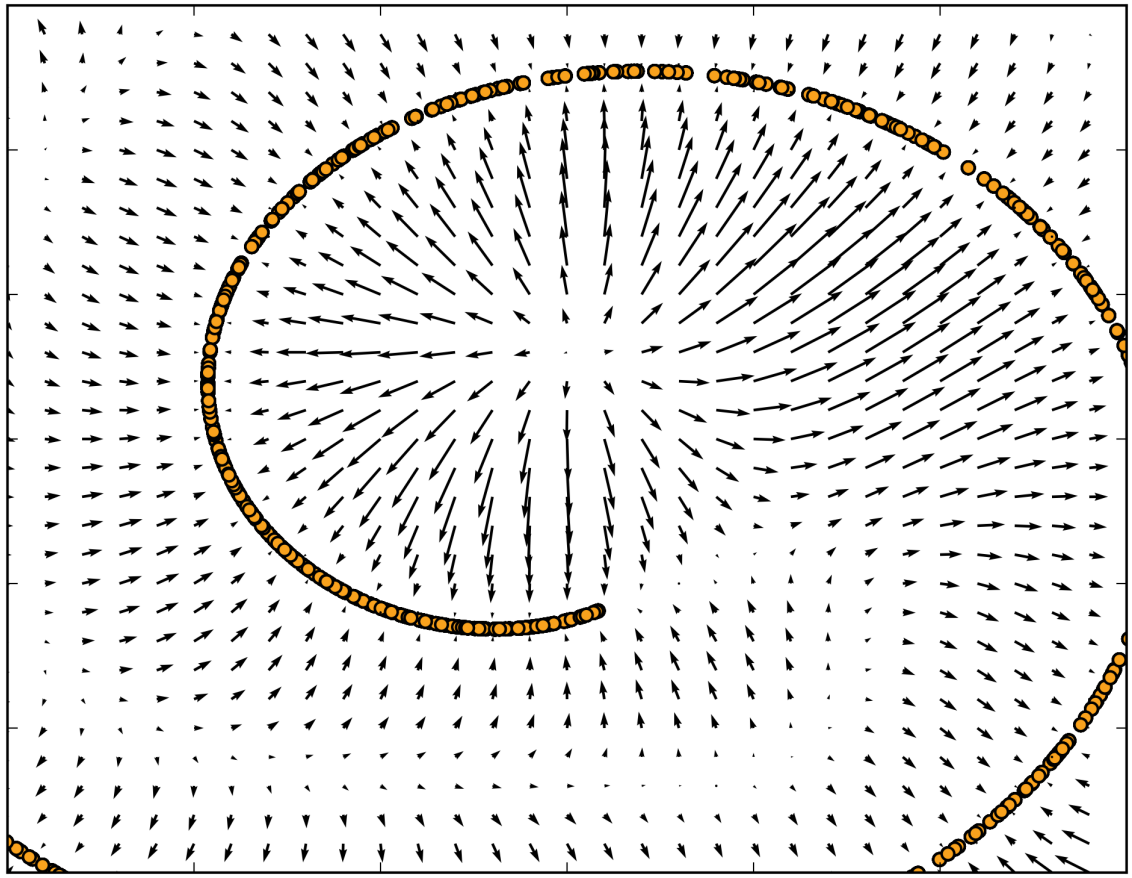
- Continuous X
- Gaussian corruption
- Noise $\sigma \rightarrow 0$
- Squared reconstruction error $||r(X+\text{noise})-X||^2$

$(r(X)-X)/\sigma^2$ estimates the score $d \log p(X) / dX$

- Langevin + Metropolis-Hastings can be used to approximately sample from such a model, but mixing was poor

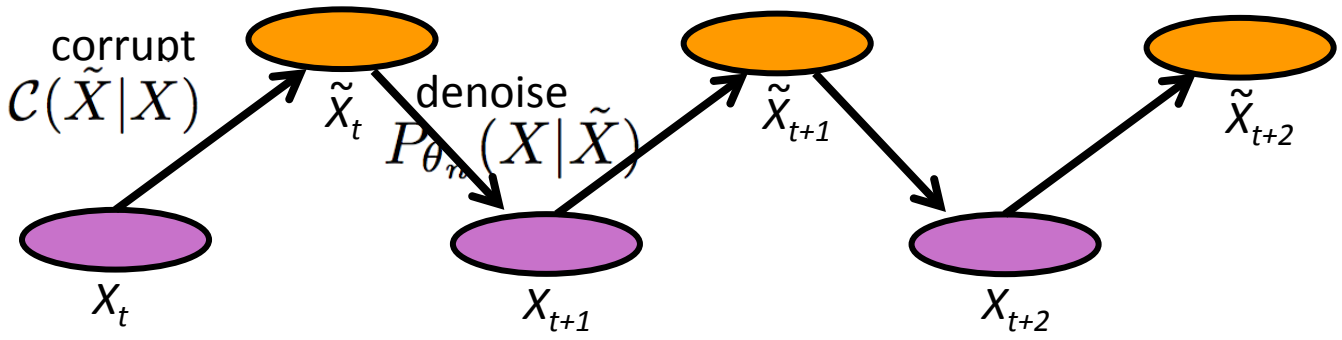
Learning a Vector Field that Estimates a Gradient Field

- Continuous inputs
- Gaussian corruption
- Squared error
- Reconstruction(x)-x estimates dlogp(x)/dx
- Zero reconstruction error could be either local min or local max of density



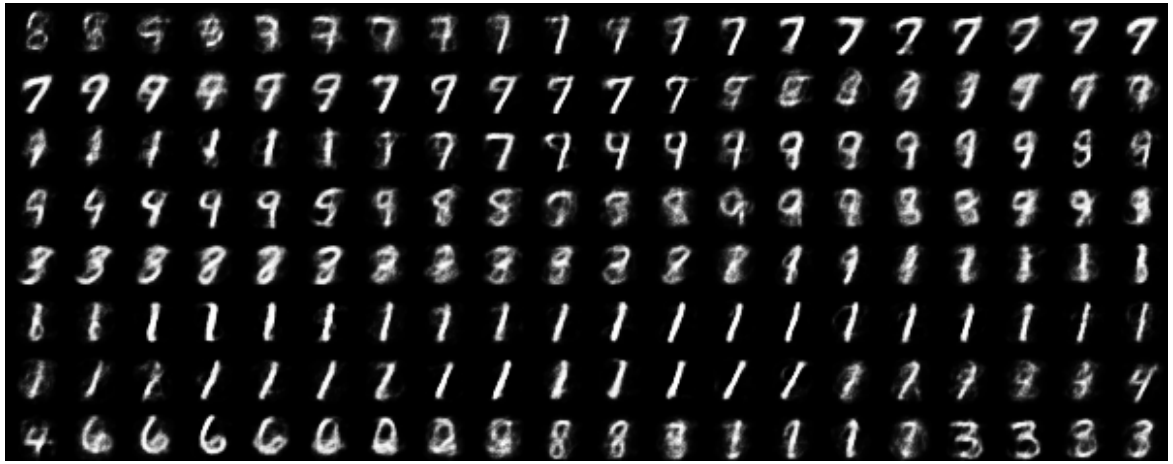
New Result: Denoising Auto-Encoder Markov Chain (NIPS'2013)

- $\mathcal{P}(X)$: true data-generating distribution
- $\mathcal{C}(\tilde{X}|X)$: corruption process
- $P_{\theta_n}(X|\tilde{X})$: denoising auto-encoder trained with *n* examples $X, \tilde{X}$ from $\mathcal{C}(\tilde{X}|X)\mathcal{P}(X)$, probabilistically "inverts" corruption
- T_n : Markov chain over *X* alternating $\tilde{X} \sim \mathcal{C}(\tilde{X}|X)$, $X \sim P_{\theta_n}(X|\tilde{X})$



Samples from a Denoising Auto-Encoder Markov Chain

- Trained on MNIST
- 1 hidden layer, factorized Bernouilli output distribution
- Consecutive samples (no skip)



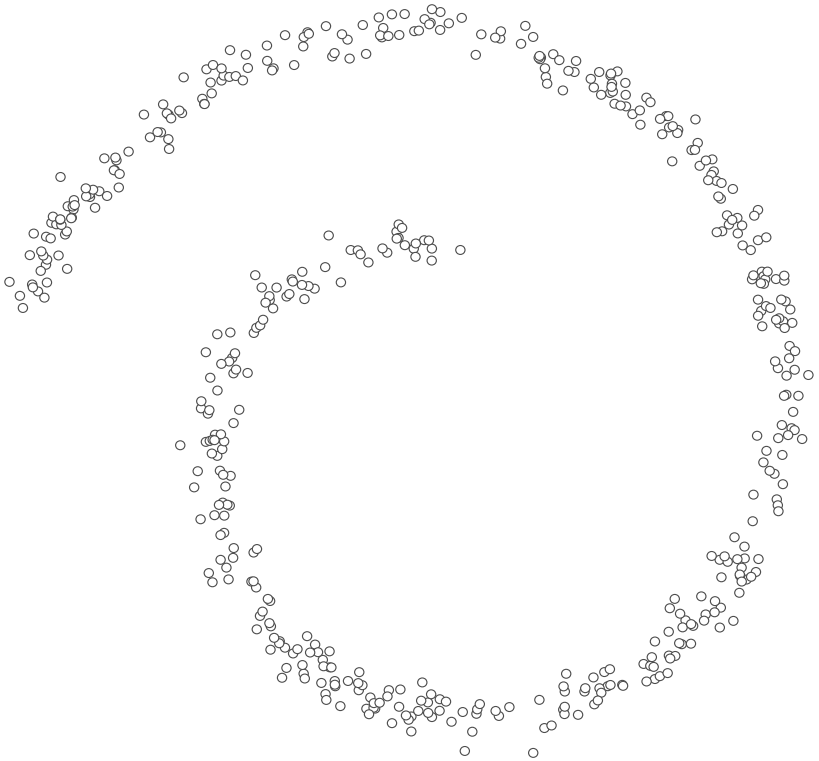
Theorem

- Denoising AE are consistent estimators of the data-generating distribution through their Markov chain, so long as they consistently estimate the conditional denoising distribution and the Markov chain converges.

$$\text{Making } P_{\theta_n}(X|\tilde{X}) \text{ match } \mathcal{P}(X|\tilde{X}) \text{ makes } \pi_n(X) \text{ match } \mathcal{P}(X)$$

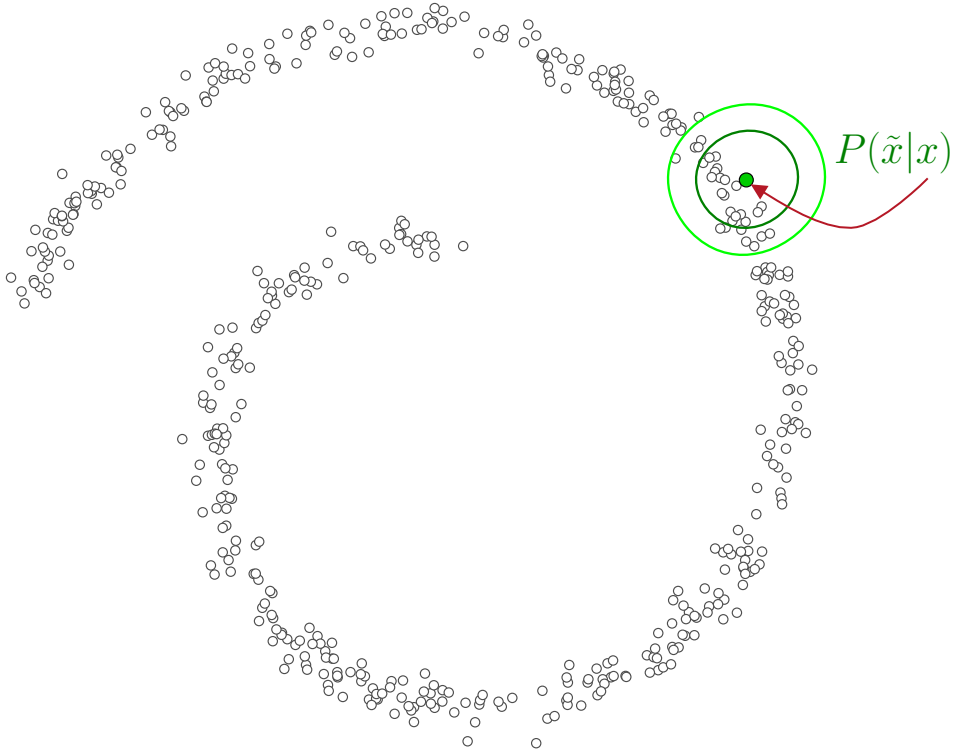
denoising distr. truth stationary distr. truth

Learning with a simpler normalization constant, a nearly unimodal conditional distribution instead of a complicated multimodal one



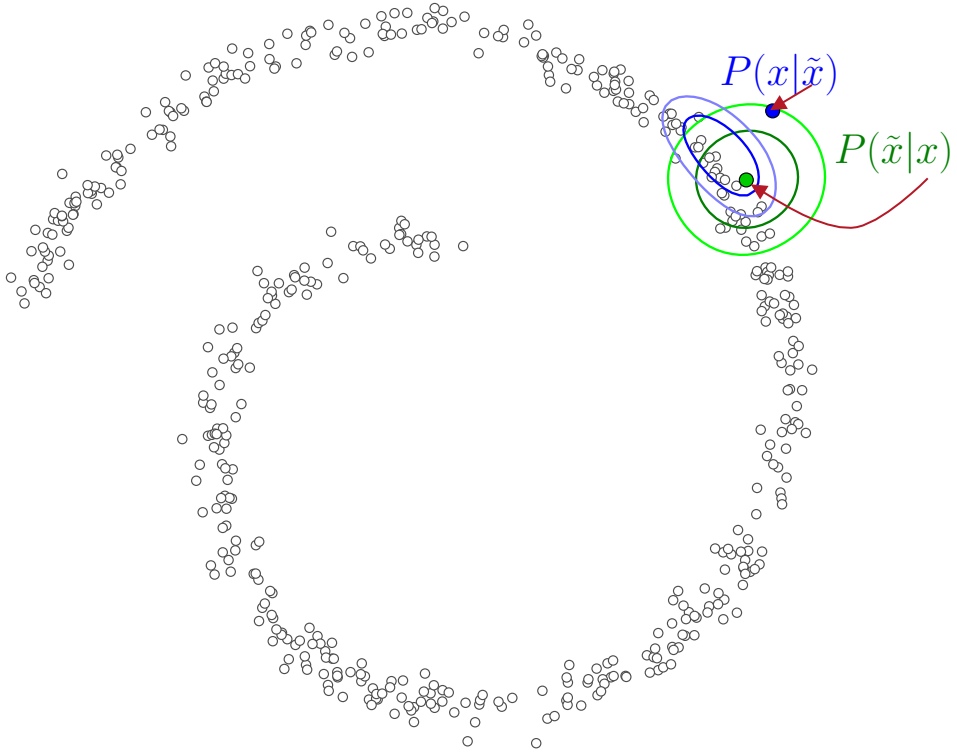
Thanks:
Jason Yosinski

Learning with a simpler normalization constant, a nearly unimodal conditional distribution instead of a complicated multimodal one



Thanks:
Jason Yosinski

Learning with a simpler normalization constant, a nearly unimodal conditional distribution instead of a complicated multimodal one



Thanks:
Jason Yosinski

Conclusions

- Unsupervised learning = guessing where to put probability mass
- AI tasks → manifold structure
- Regularized auto-encoders capture manifold structure
- Regularized auto-encoders can now be viewed as generative models
- The mystery of their probabilistic interpretation has now been mostly solved (at least for the denoising case).

LISA team: Merci! Questions?

LISA team: Merci! Questions?