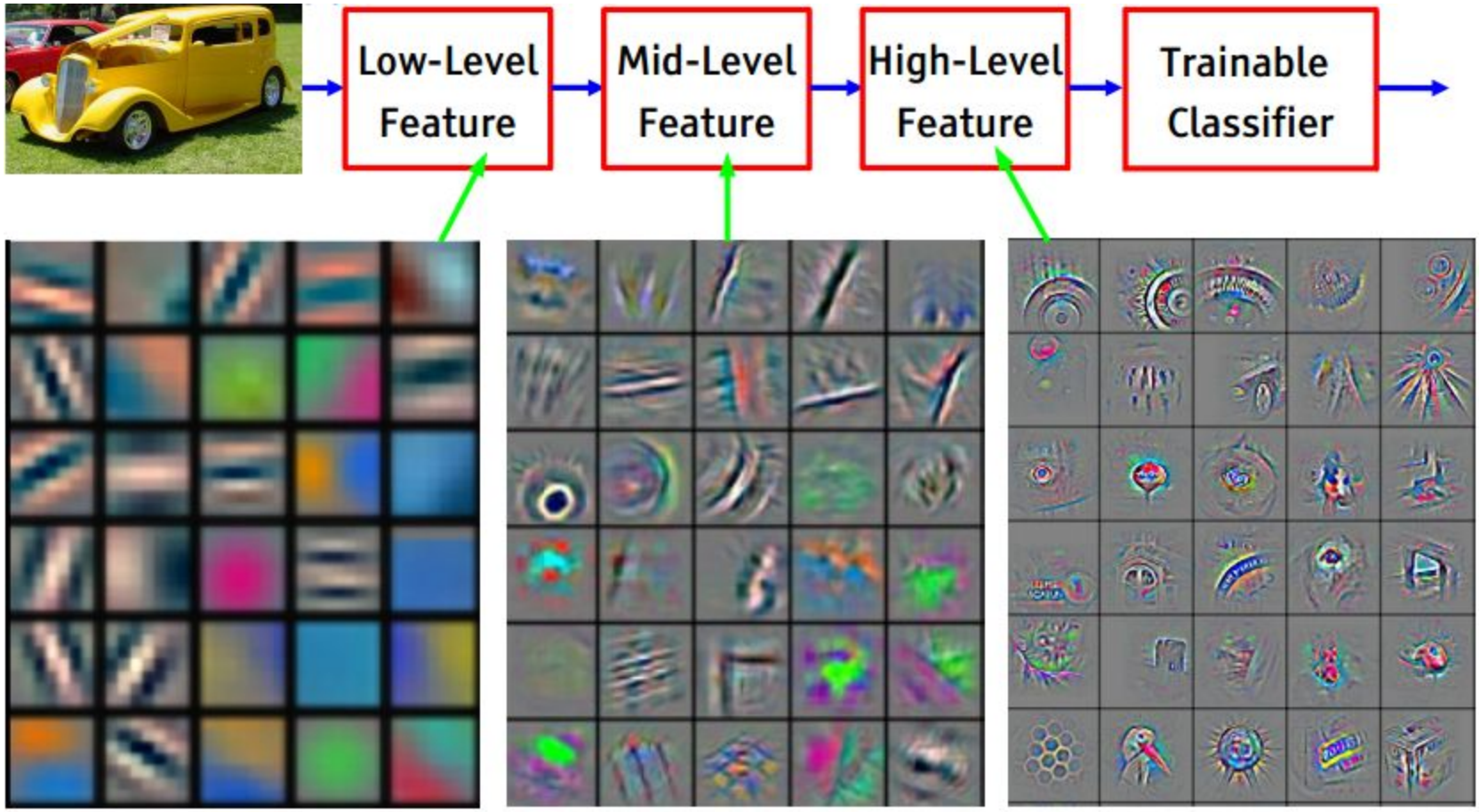
Unidad 5: Aprendizaje de Representaciones

Curso: Redes Neuronales Profundas

Ya vimos las representaciones que se obtienen en las CNN



Zeiler, Matthew D, and Rob Fergus. "Visualizing and understanding convolutional neural networks." *arXiv preprint arXiv:1311.2901* (2013).

¿En qué otros casos surgen o se necesitan representaciones?

Clasificación

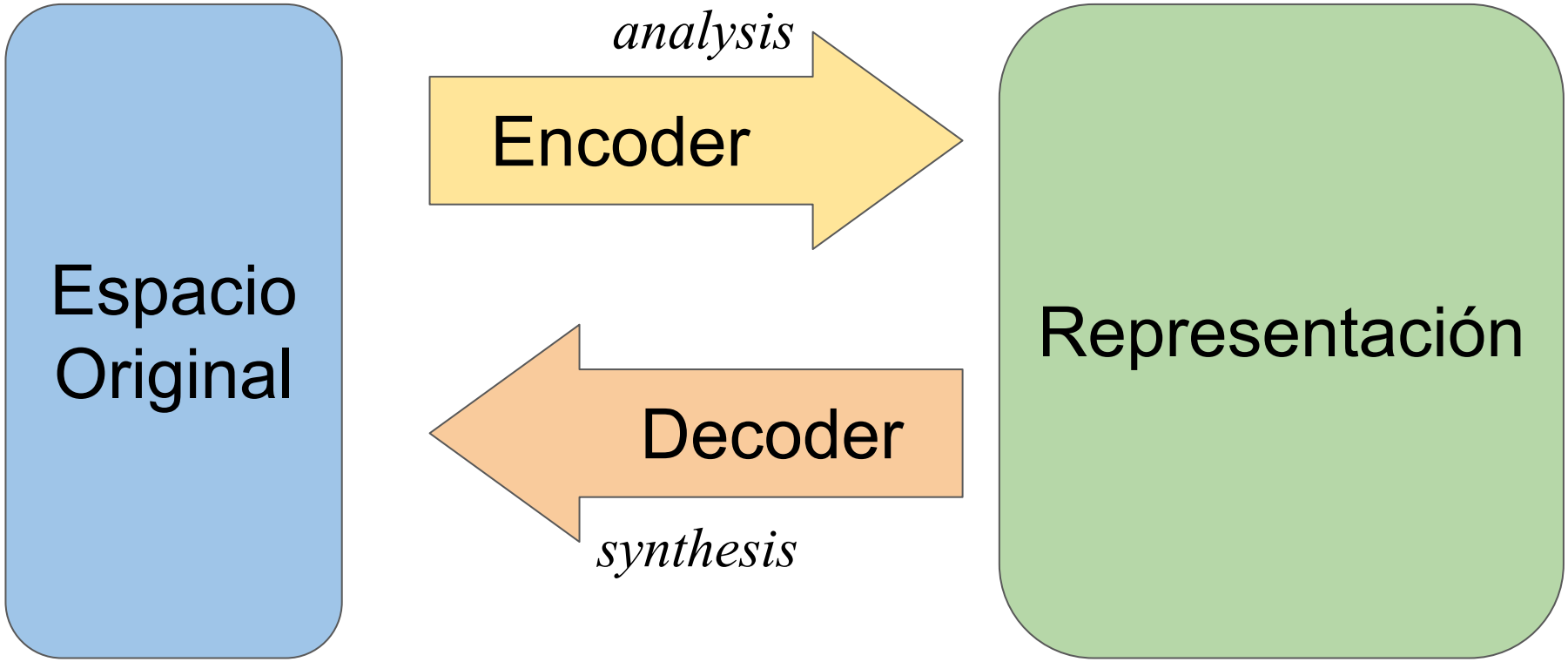
Visualización

Generación

Exploración

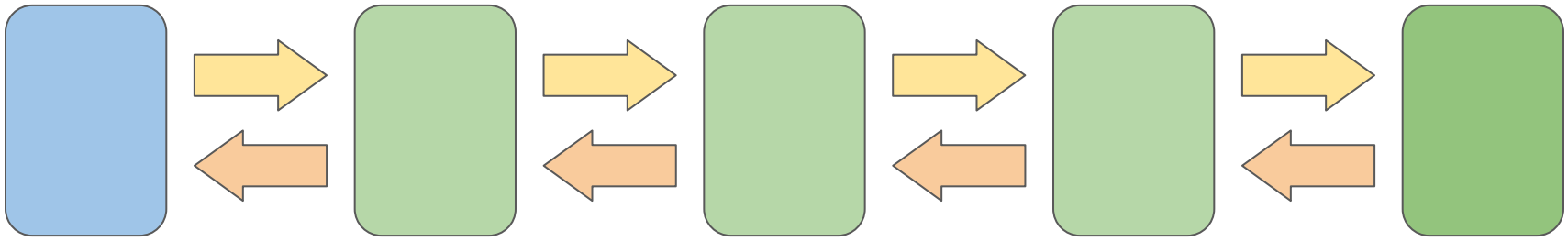
Compresión

Transformación de los datos



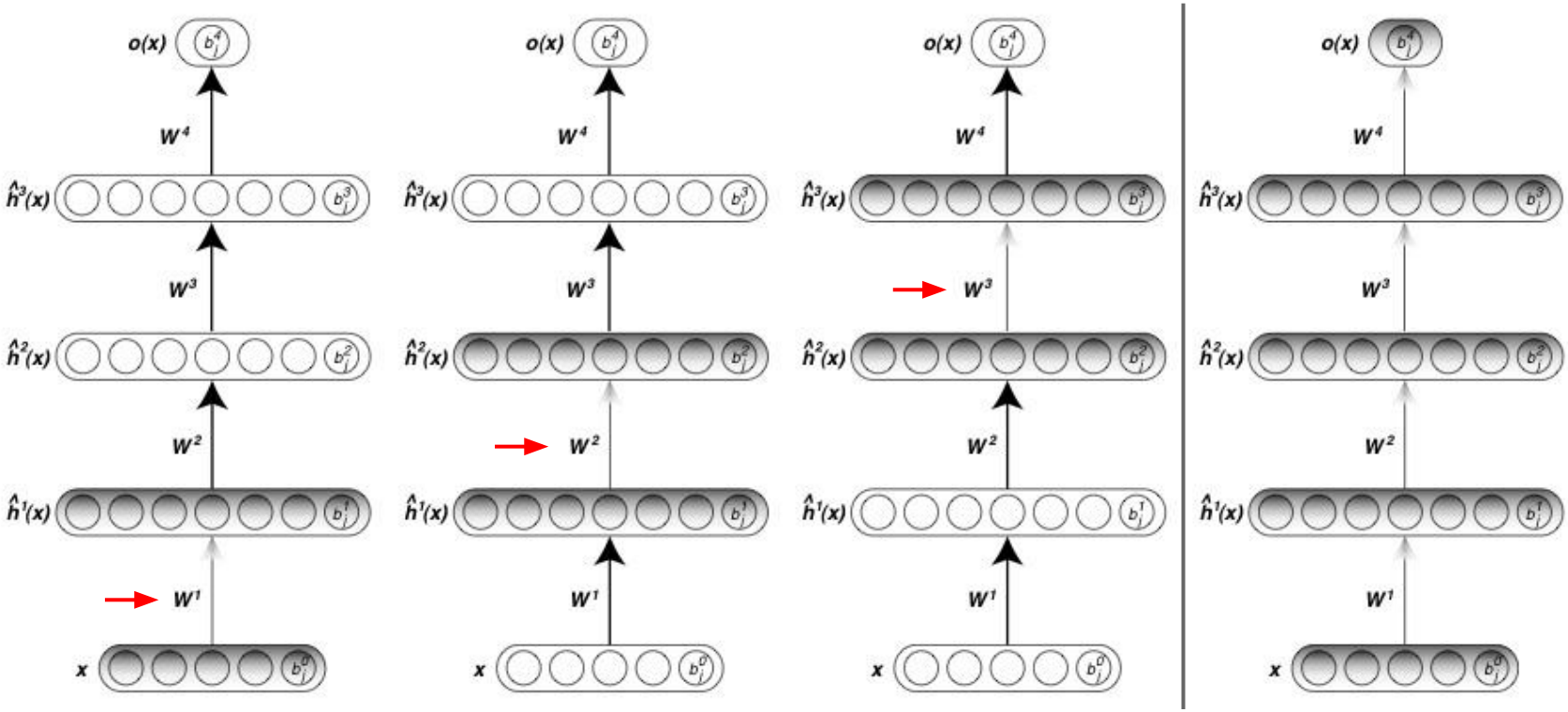
Transformación de los datos

Modelo profundo



concatenación
de encoders (decoders)
no lineales

Greedy layer-wise training of deep networks



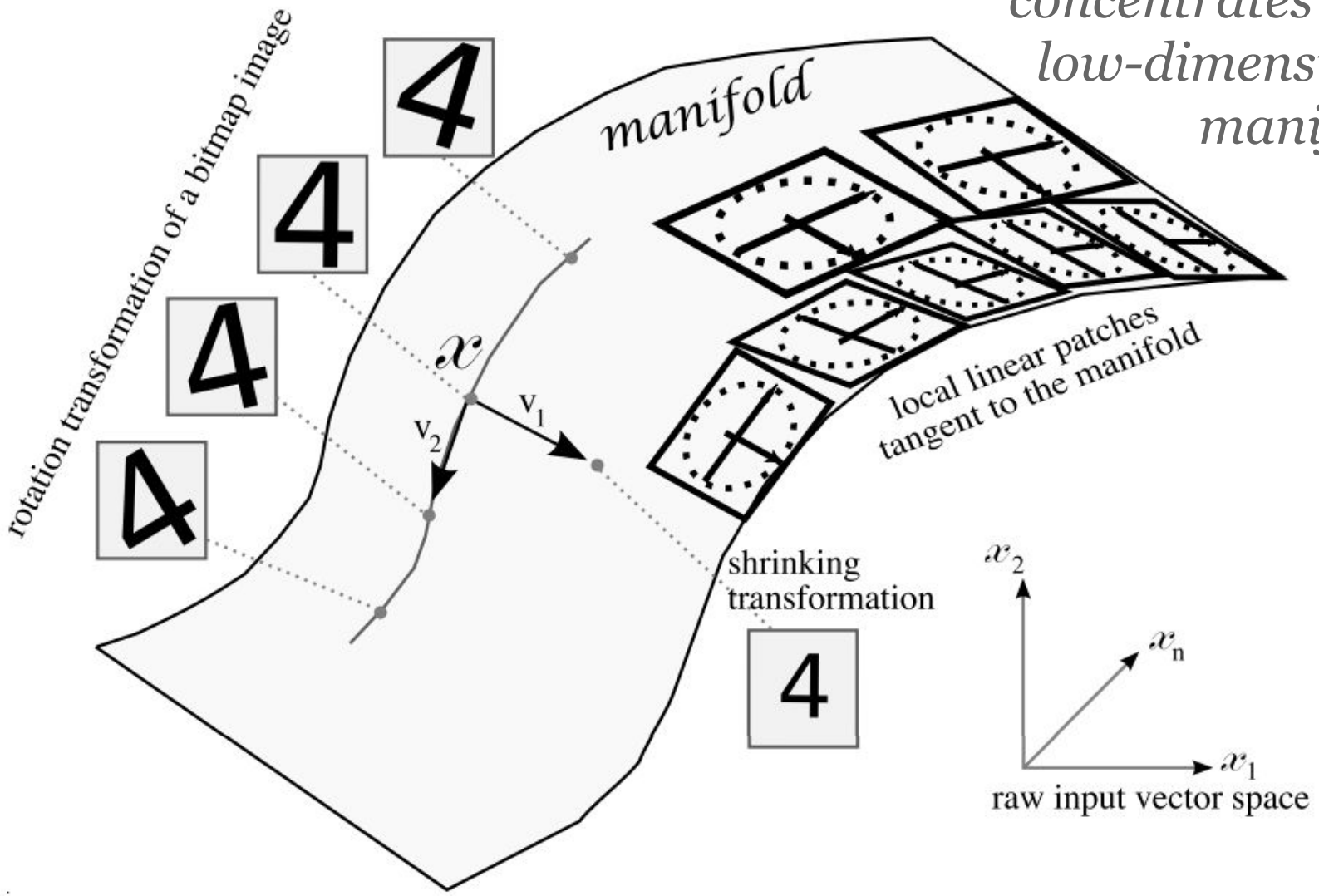
Hinton, Geoffrey E, and Ruslan R Salakhutdinov. "Reducing the dimensionality of data with neural networks." *Science* 313.5786 (2006): 504-507.

Hinton, Geoffrey, Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets." *Neural computation* 18.7 (2006): 1527-1554.

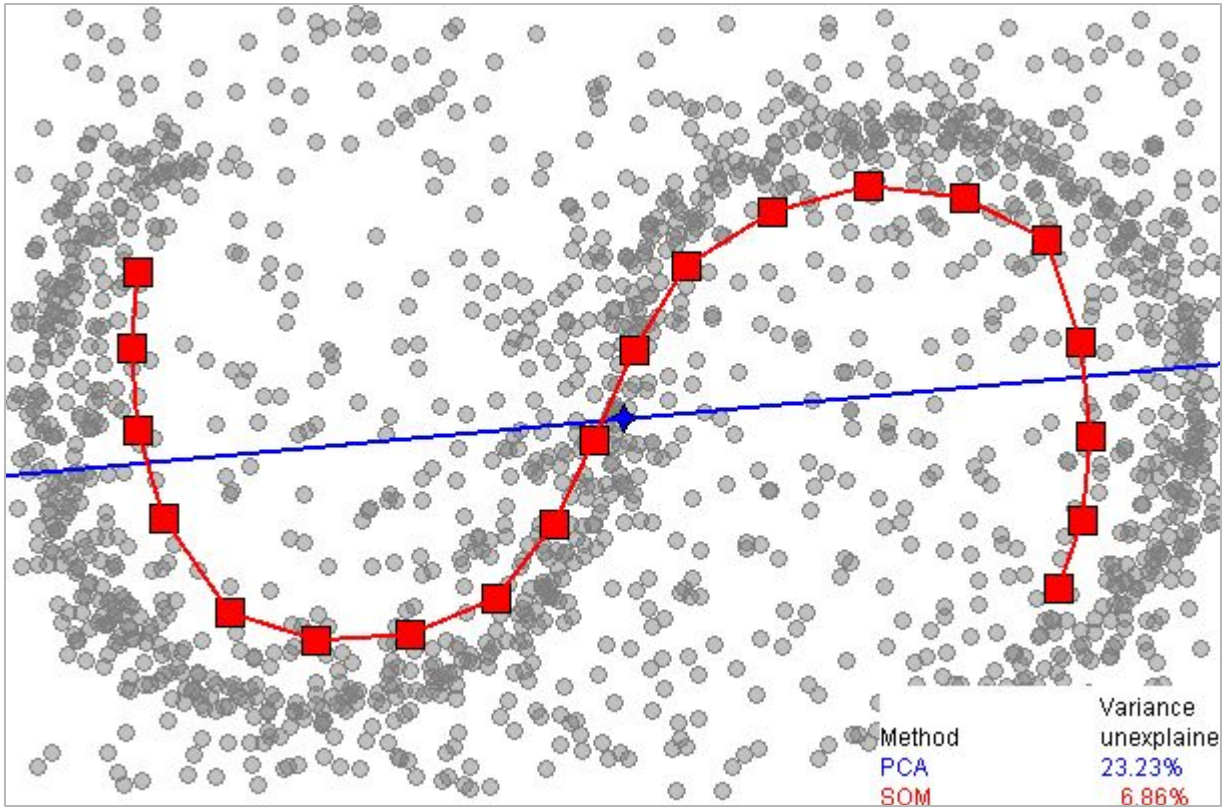
Bengio, Yoshua et al. "Greedy layer-wise training of deep networks." *Advances in neural information processing systems* 19 (2007): 153.

Manifold hypothesis

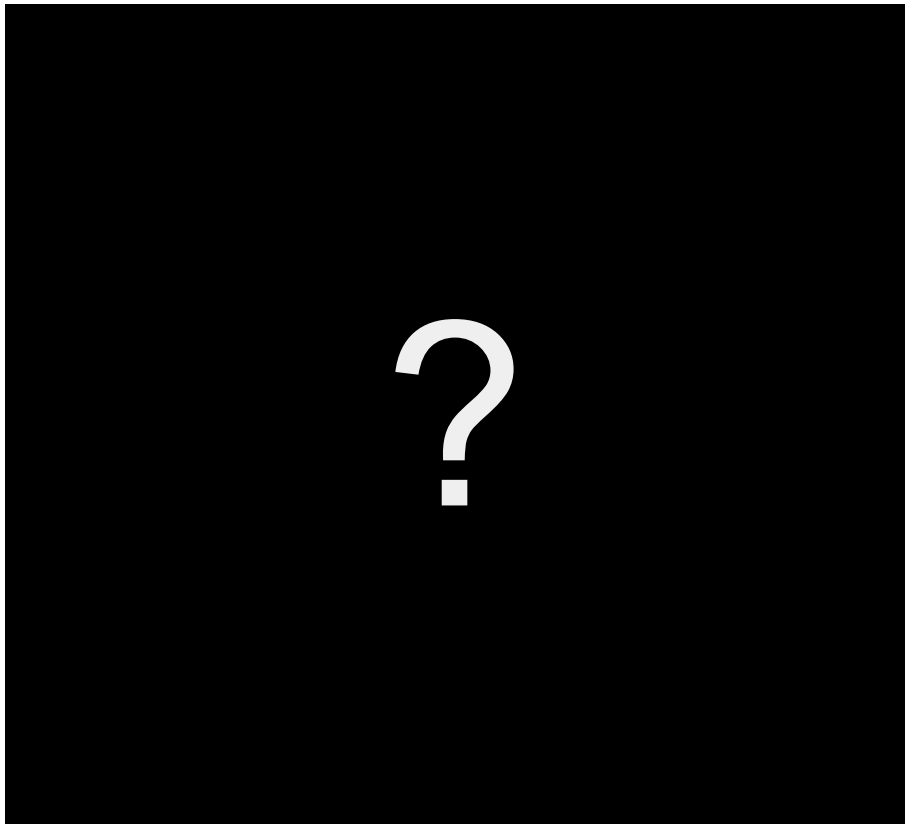
data density concentrates near low-dimensional manifolds



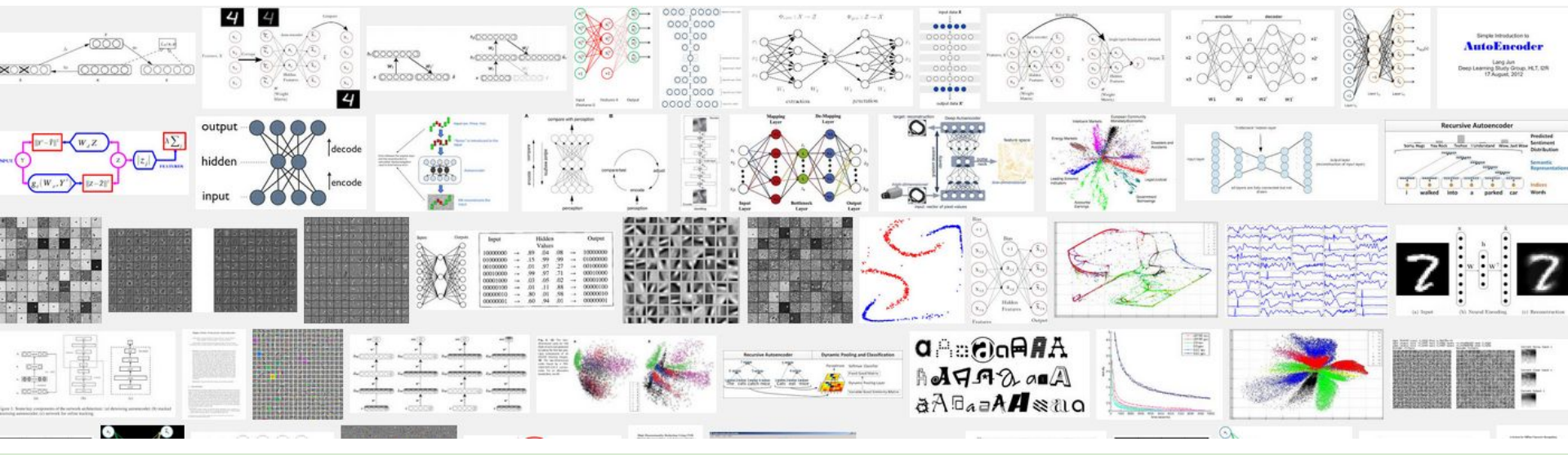
Manifold Learning



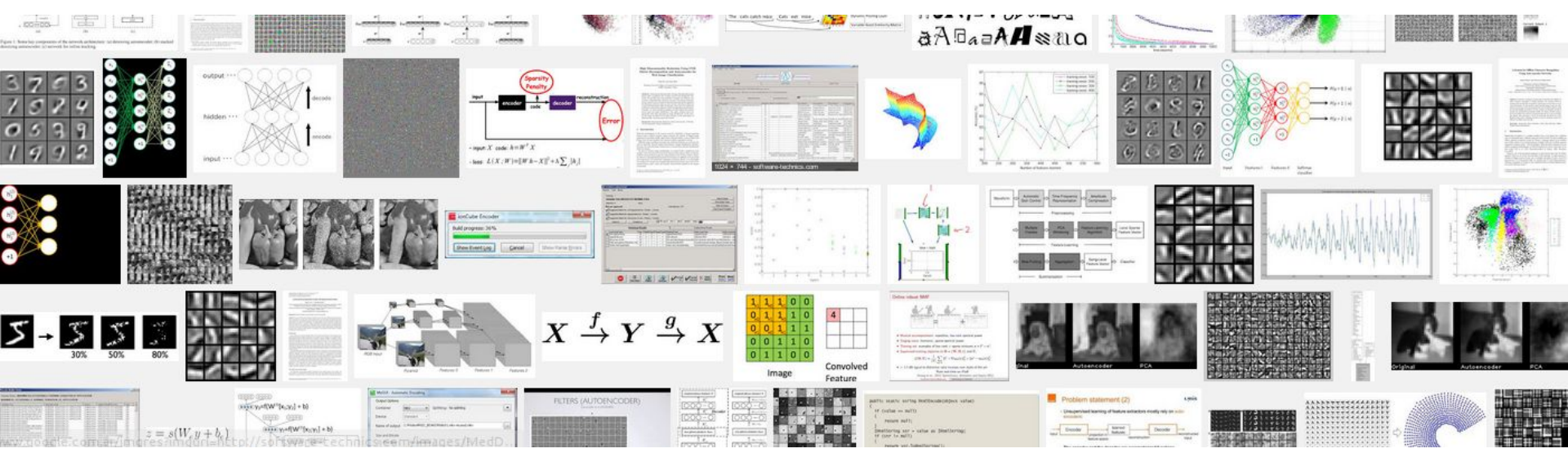
Manifold en altas dimensiones



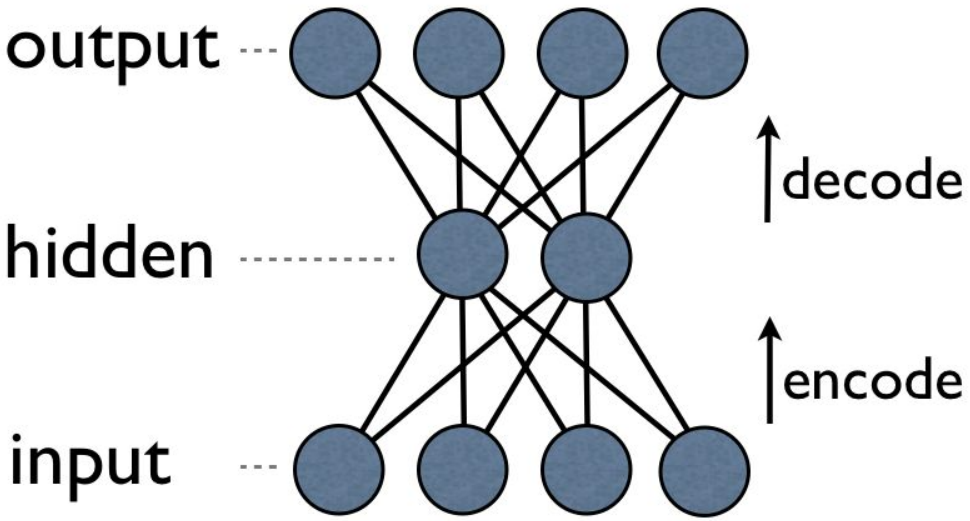
Modelos generativos



Auto-Encoders



Auto-Encoder



$$z = s(W'y + b')$$

$$y = s(Wx + b)$$

tied weights $W' = W^T$ (optional)

Reconstruction Error

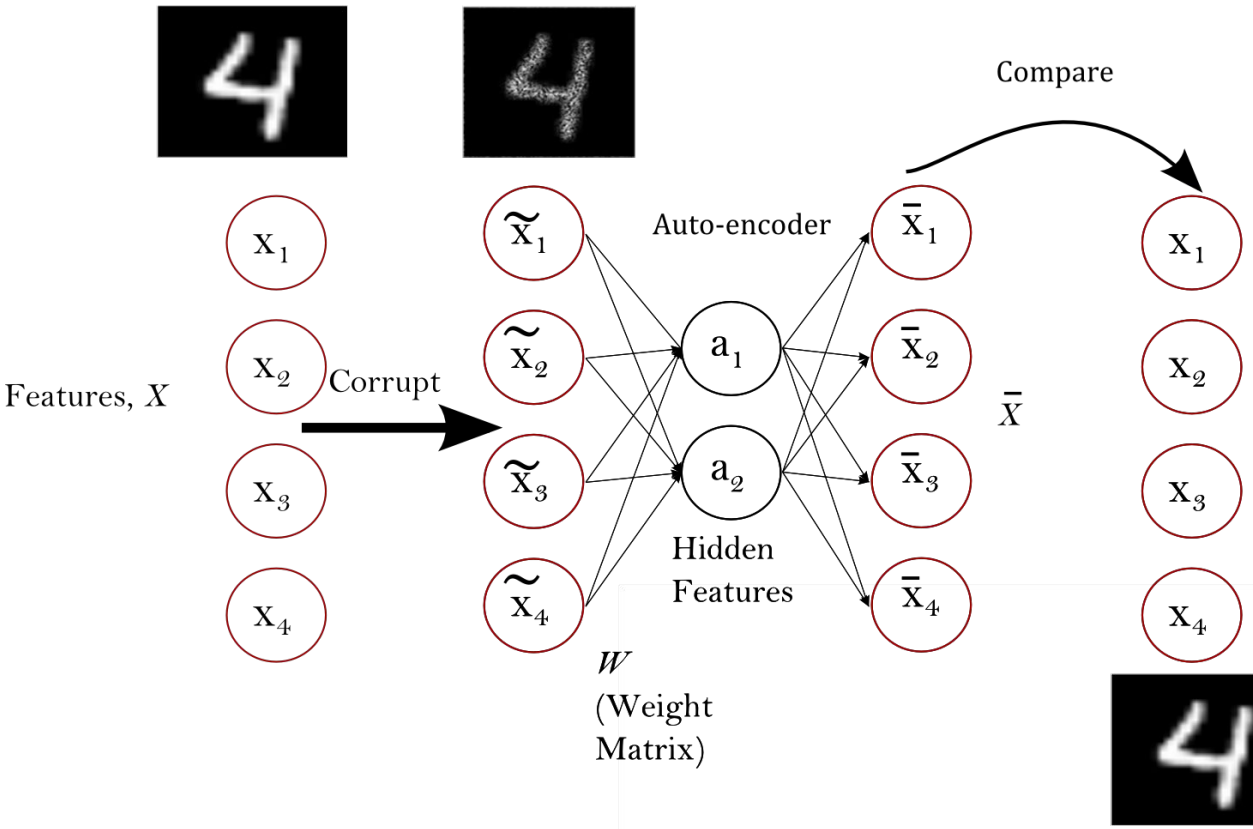
squared error

$$L(x, z) = \|x - z\|^2$$

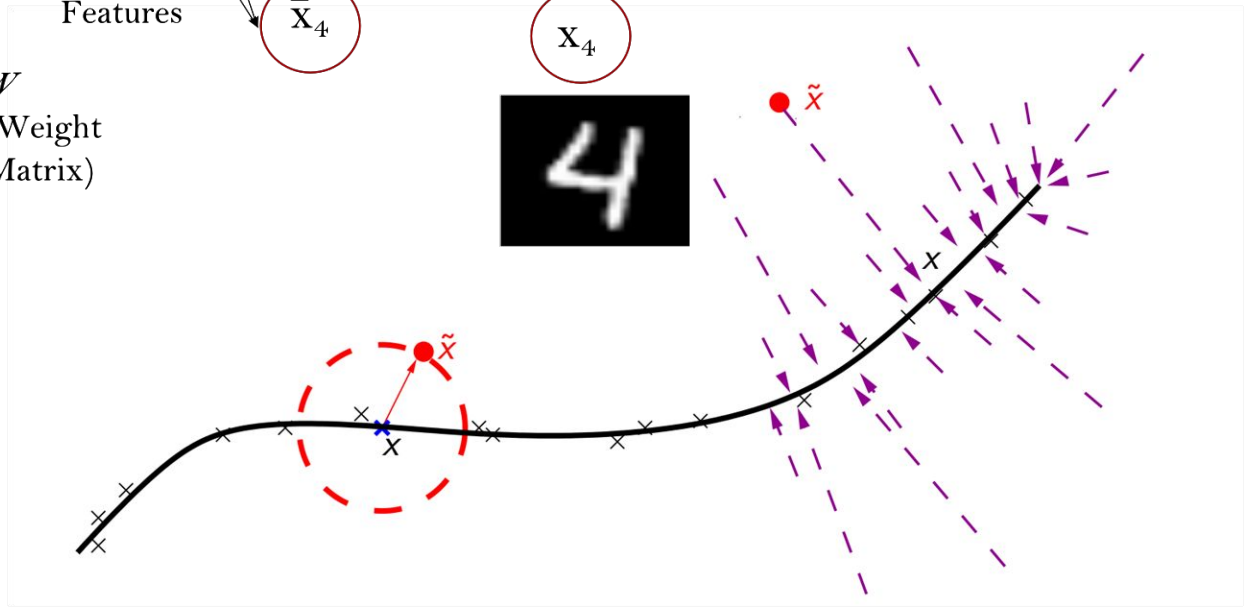
cross-entropy

$$L_H(x, z) = -\sum_{k=1}^{d} [x_k \log z_k + (1 - x_k) \log(1 - z_k)]$$

Denoising Auto-Encoders



We train the auto-encoder to reconstruct the input from a corrupted version of it.



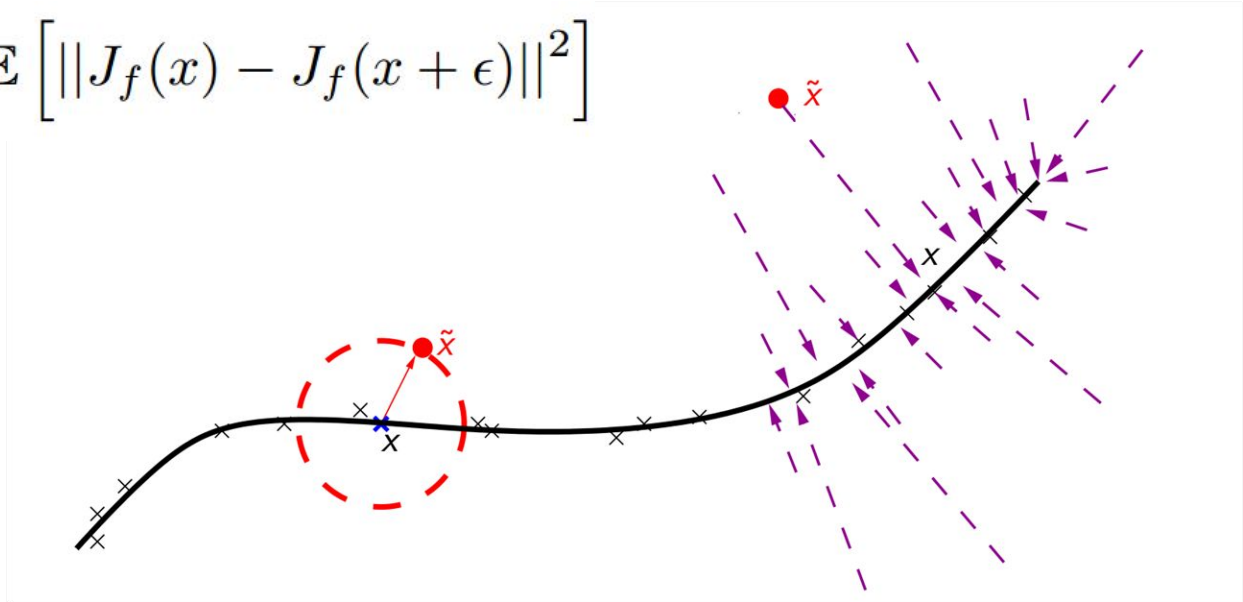
Contractive Auto-Encoders

First-order contractive Auto-Encoder

$$\mathcal{J}_{\mathrm{CAE}}(\theta) = \sum_{x \in D_n} L(x, g(f(x))) + \lambda \|J_f(x)\|^2$$

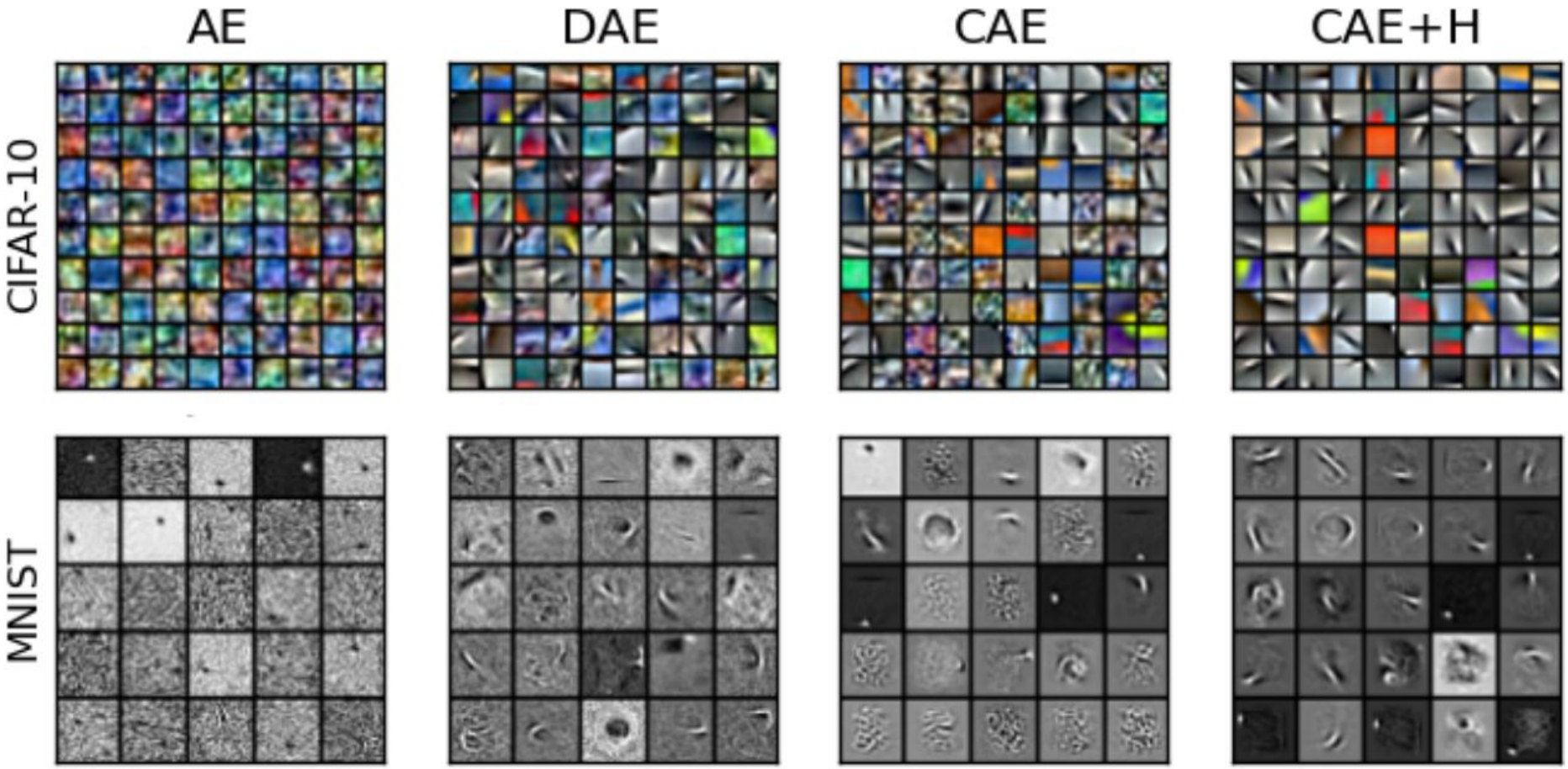
Higher order regularization:

$$\|H_f(x)\|^2 = \lim_{\sigma \to 0} \frac{1}{\sigma^2} \mathbb{E}\left[\|J_f(x) - J_f(x+\epsilon)\|^2\right]$$

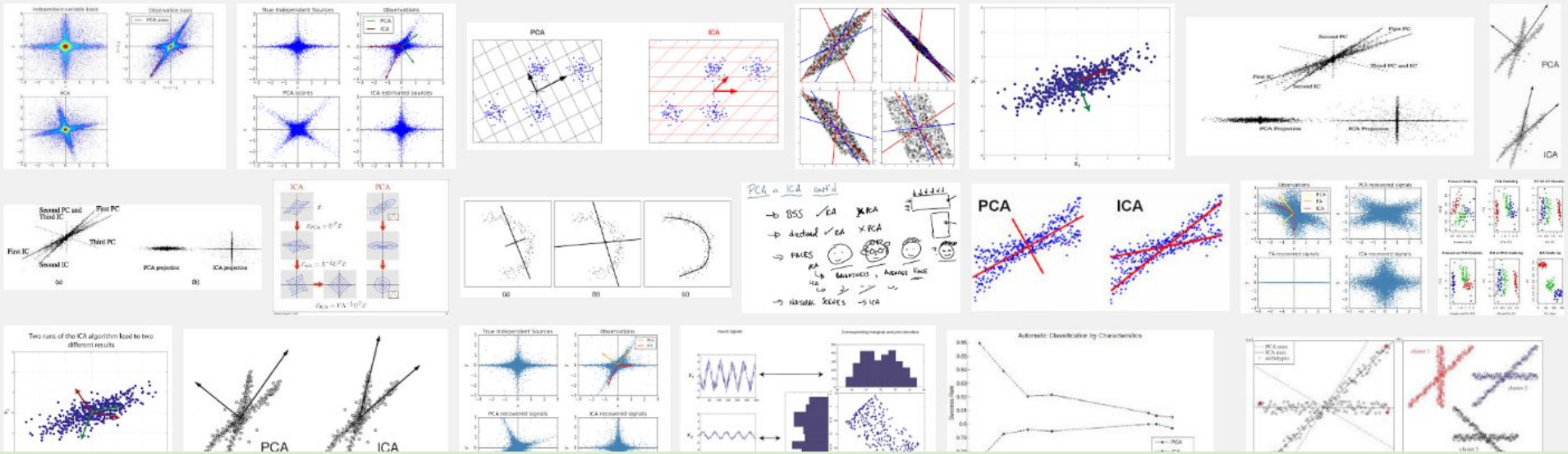


Rifai, Salah, et al. "Higher order contractive auto-encoder." *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2011. 645-660.

Contractive Auto-Encoders

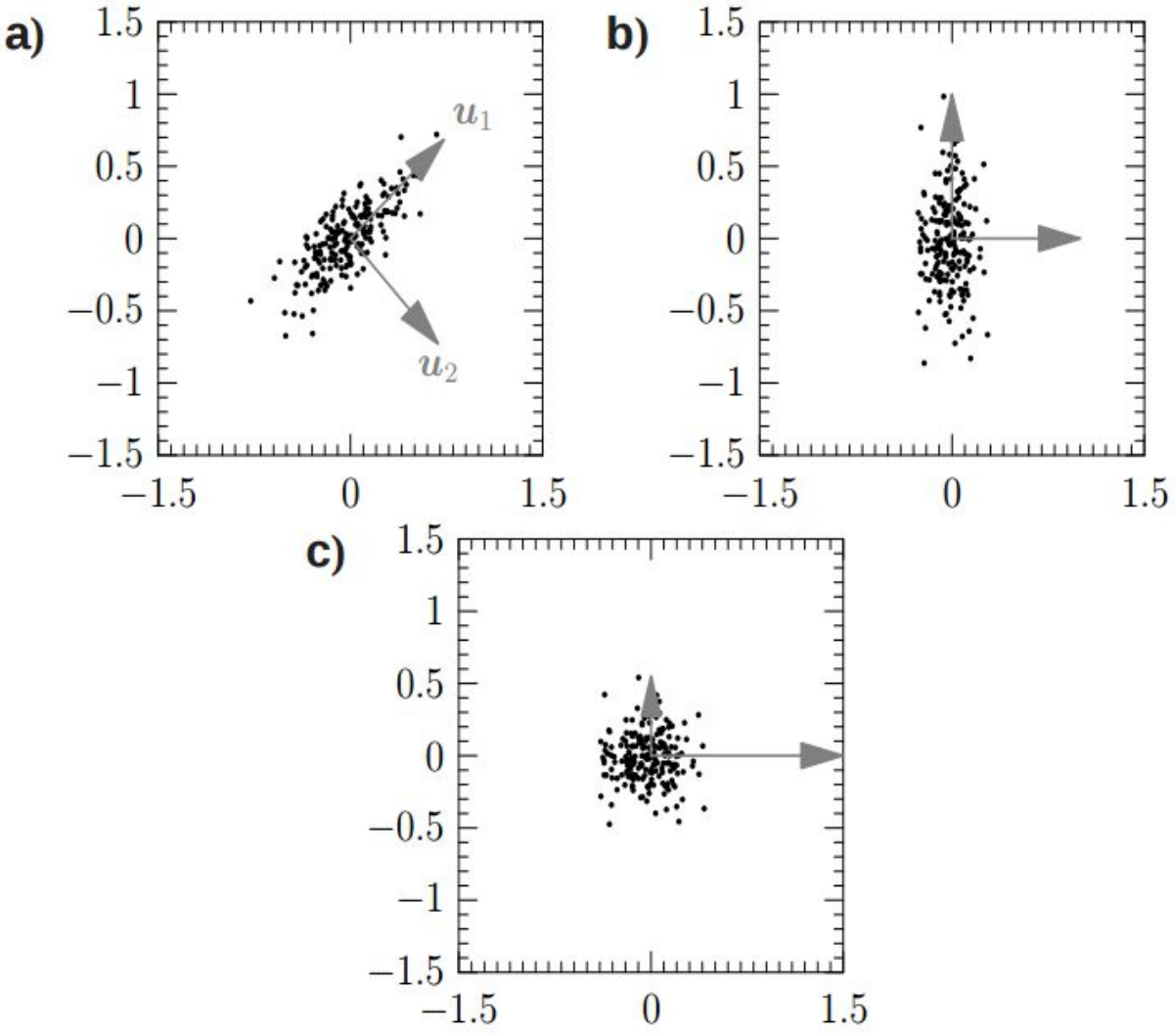


Rifai, Salah, et al. "Higher order contractive auto-encoder." *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2011. 645-660.

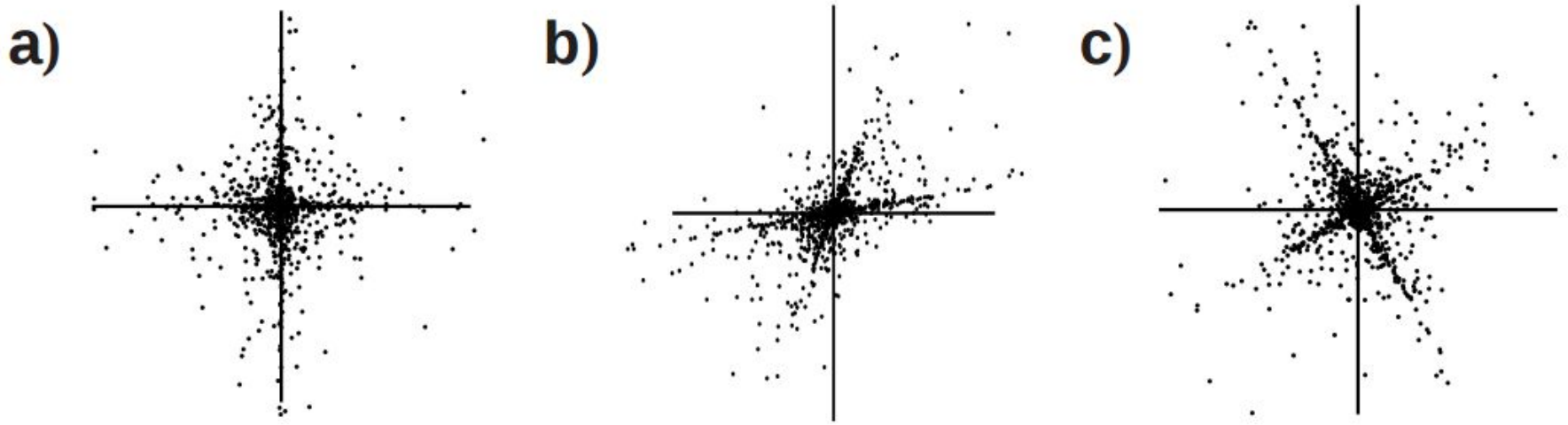

PCA
ICA
PCA
ICA
Original
PCA Recon
ICA Recon
PCA 2nd Dimention
PCA 1st Dimention
PCA
ICA
MSD
Number of reconstructing components
Dimensionality reduction
PCA, SVD, MDS, ICA,
and friends
Jure Leskovec
Machine Learning recitation
April 27 2006
Non-Gaussian data

PCA & ICA

Principal Component Analysis (PCA)



Independent Component Analysis (ICA)



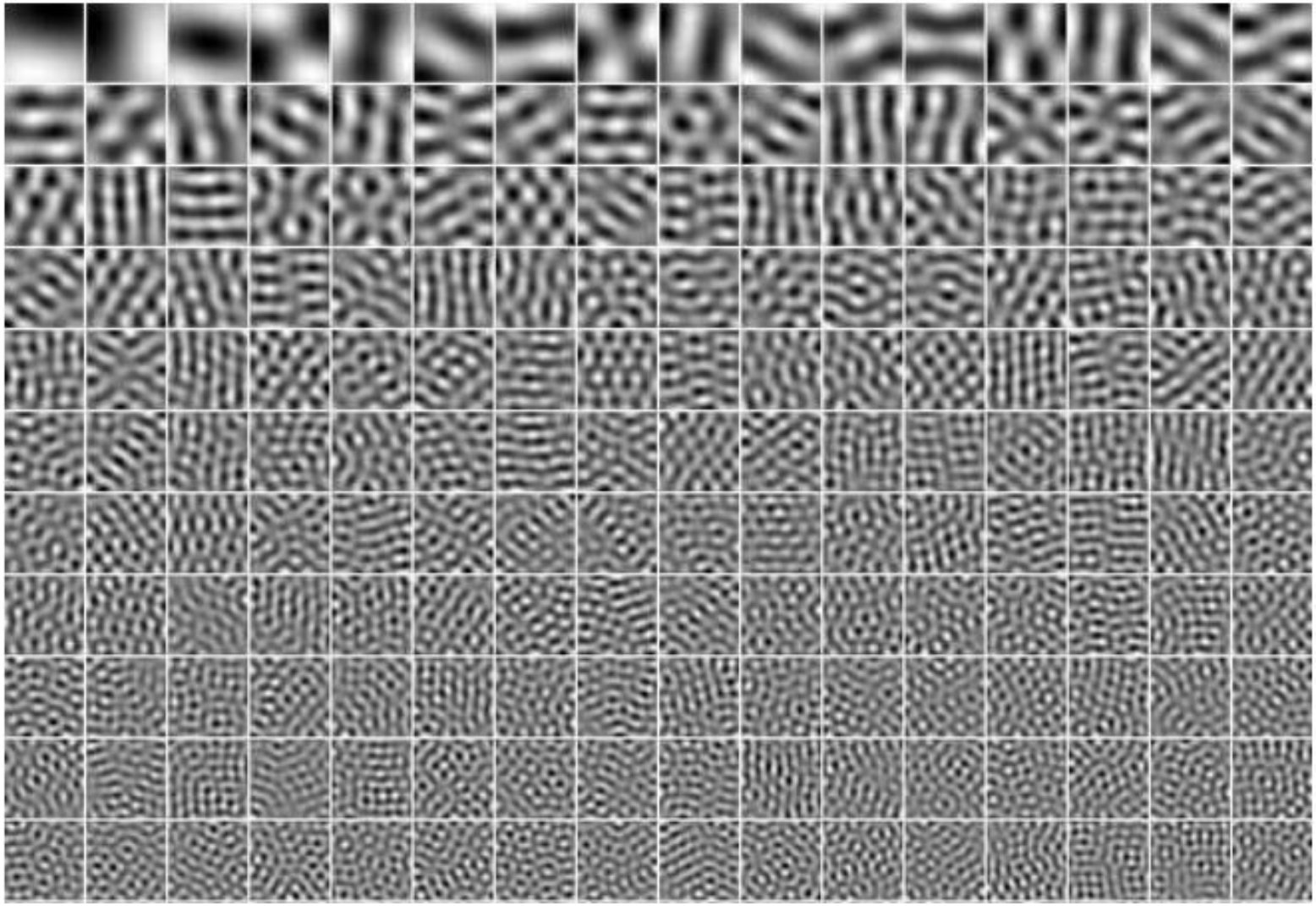
Independent Component Analysis (ICA)

$$I(x,y) = \sum_{i=1}^{m} A_i(x,y) s_i$$

- s_i are statistically independent
- s_i are non-gaussian (sparse coding)
- A_i is invertible

$$s_i = \sum_{x,y} W_i(x,y) I(x,y)$$

Principal Component Analysis (PCA)



Independent Component Analysis (ICA)

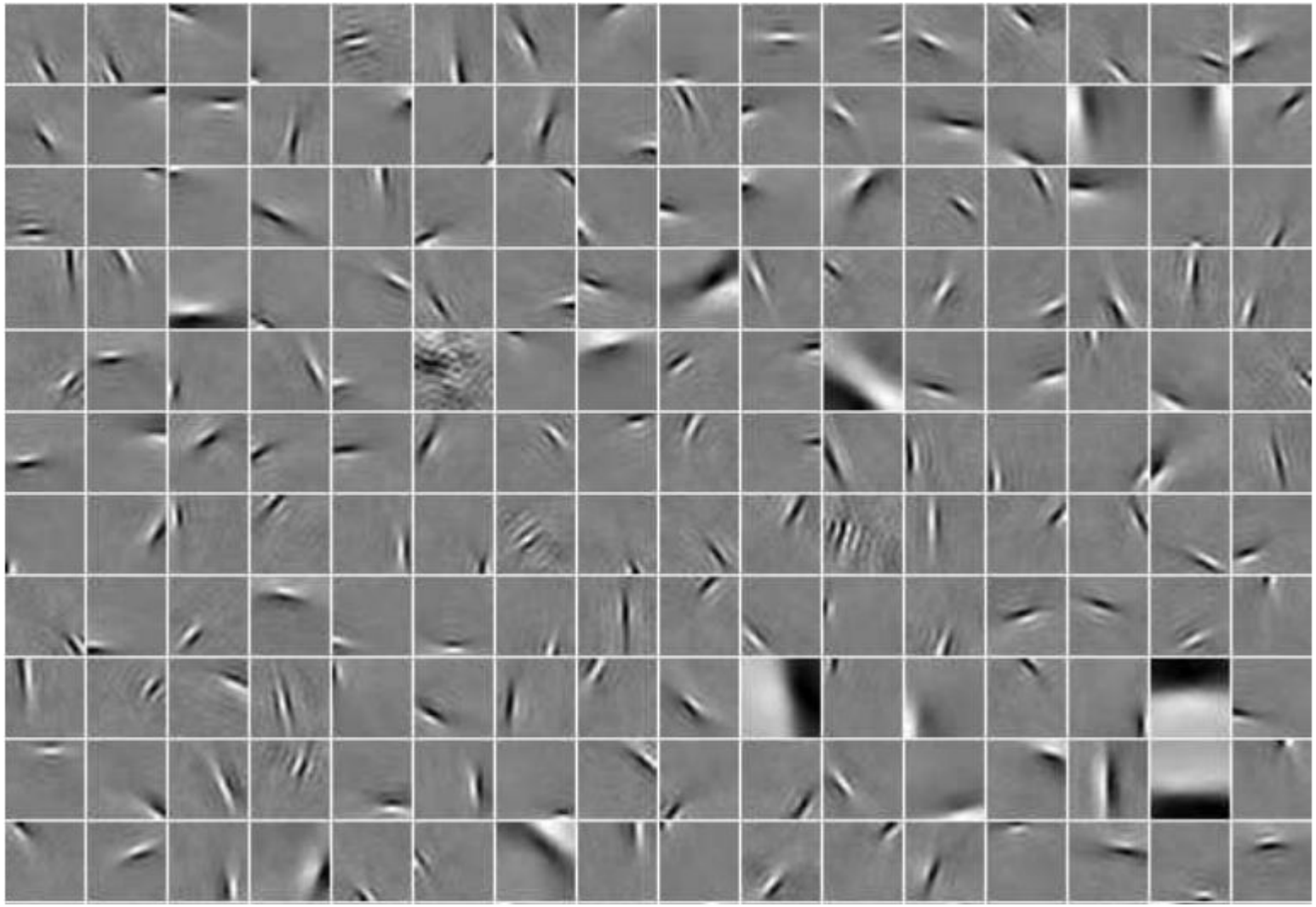
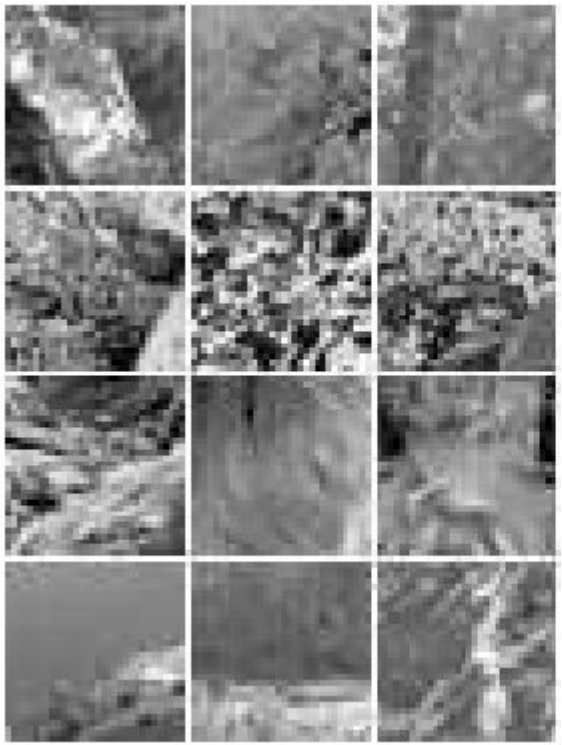


Image synthesis using PCA



Natural image patches

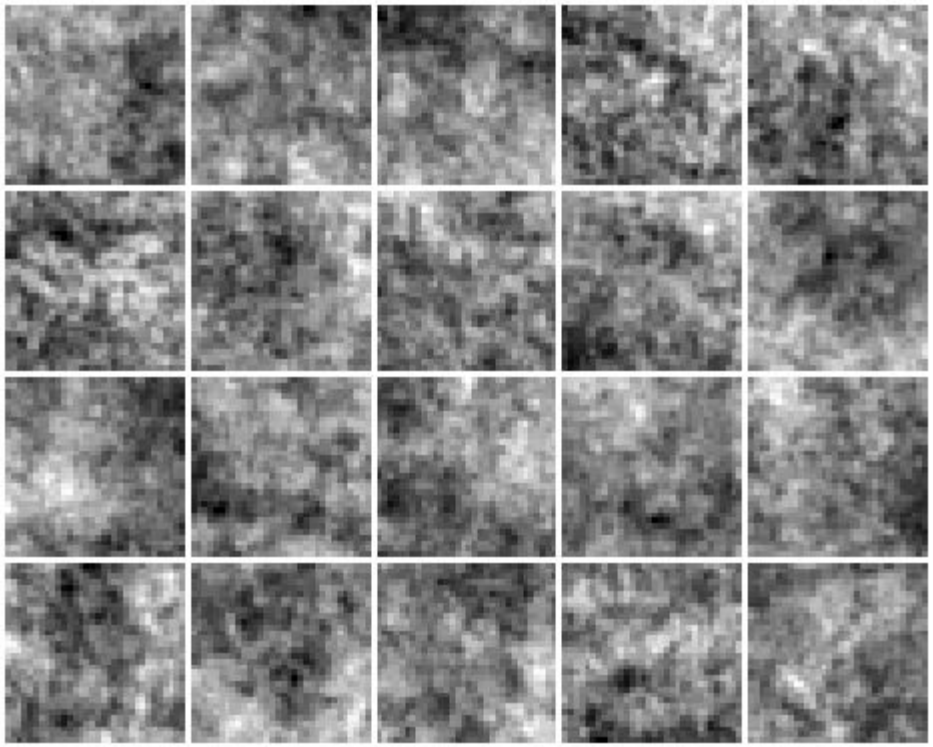
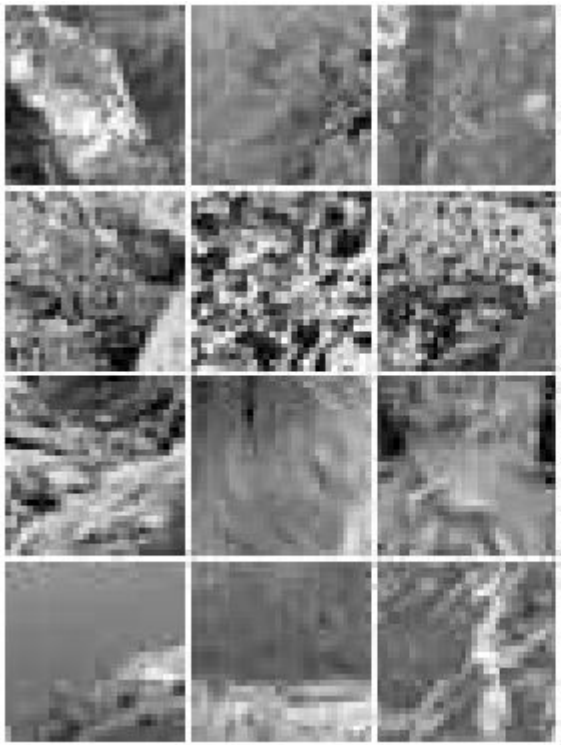


Image patches synthesis using PCA

Image synthesis using ICA



Natural image patches

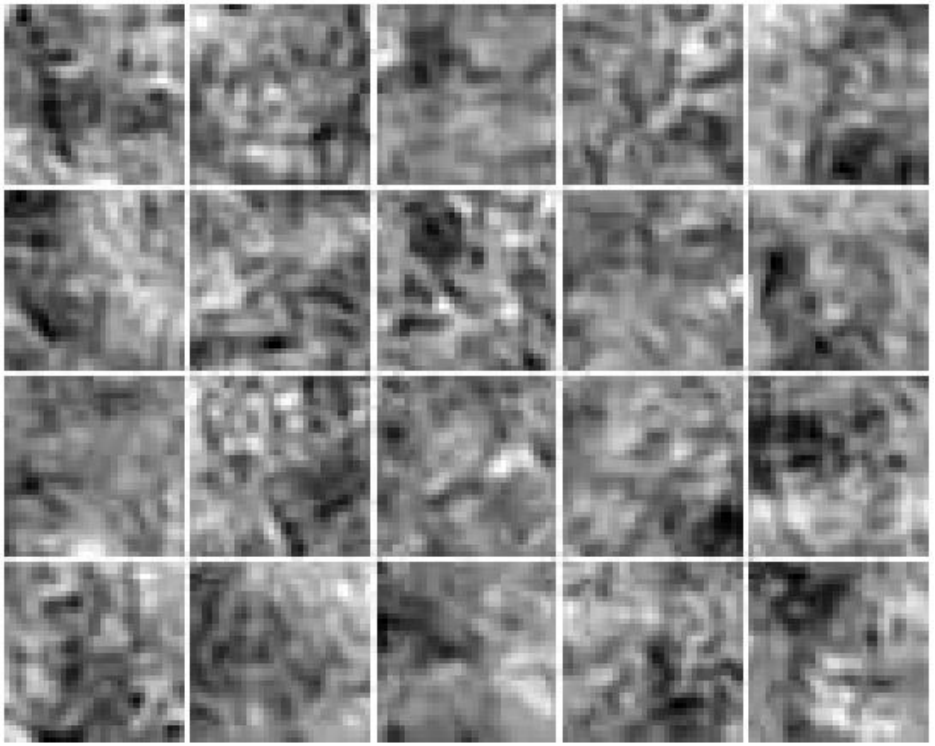
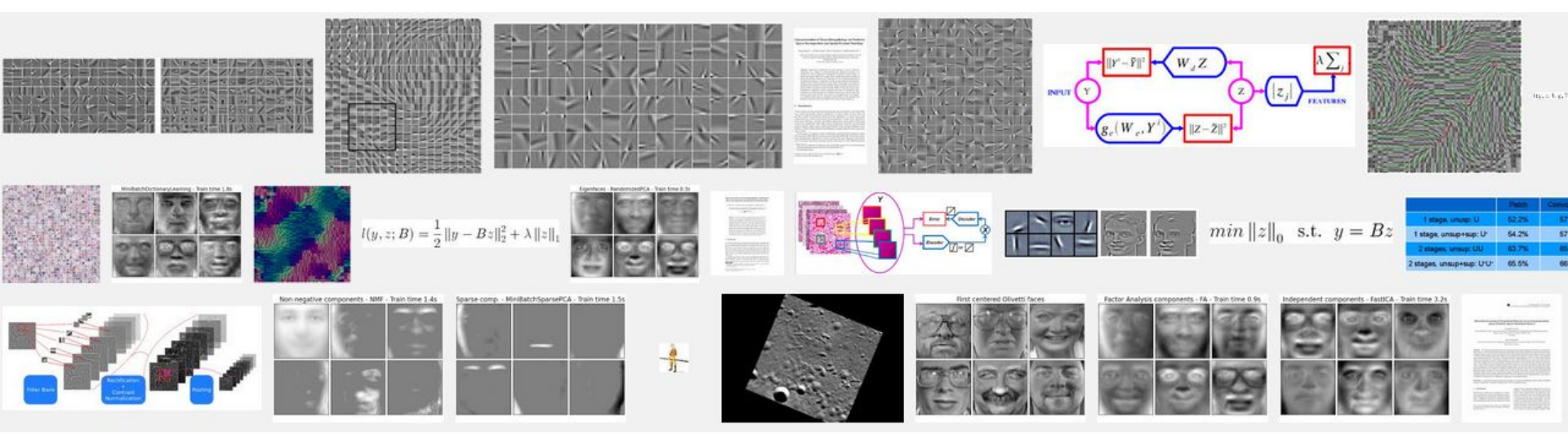
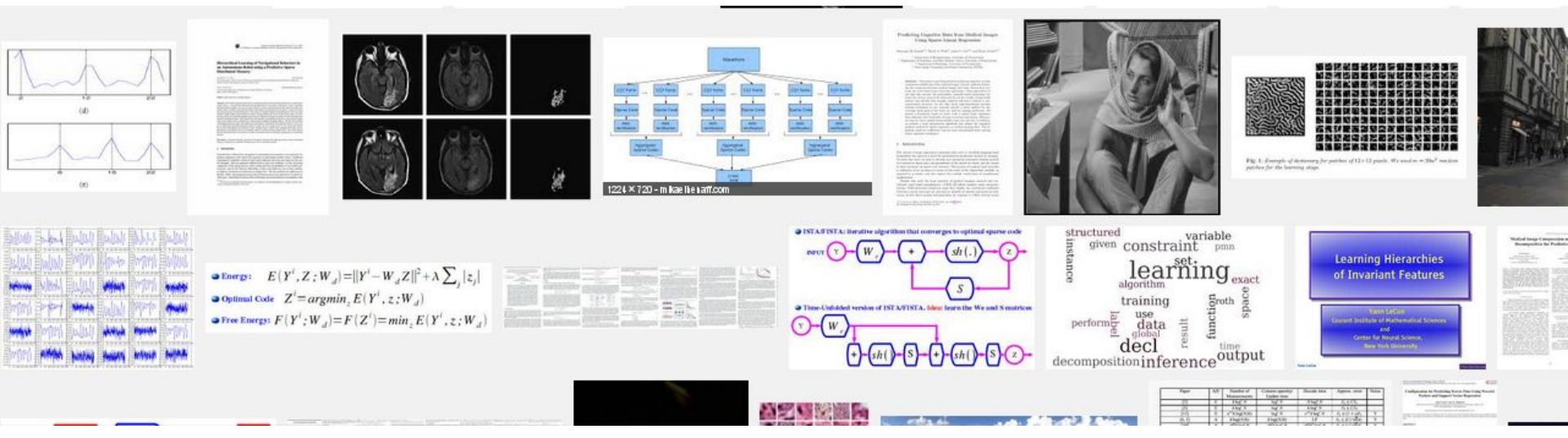


Image patches synthesis using ICA

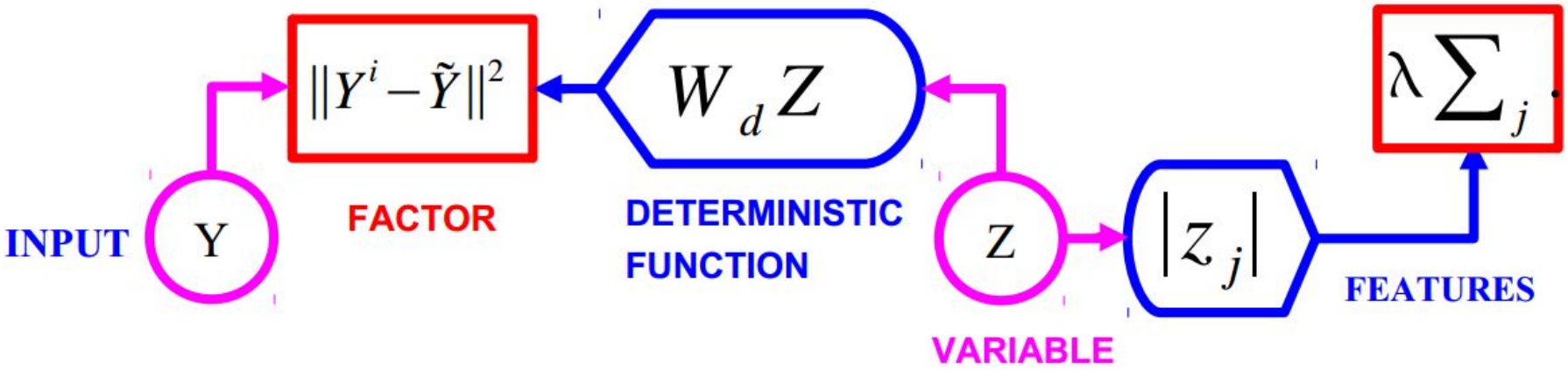


Predictive Sparse Decomposition (PSD)



Sparse Coding

$$E(Y^i, Z) = \|Y^i - W_d Z\|^2 + \lambda \sum_j |z_j|$$



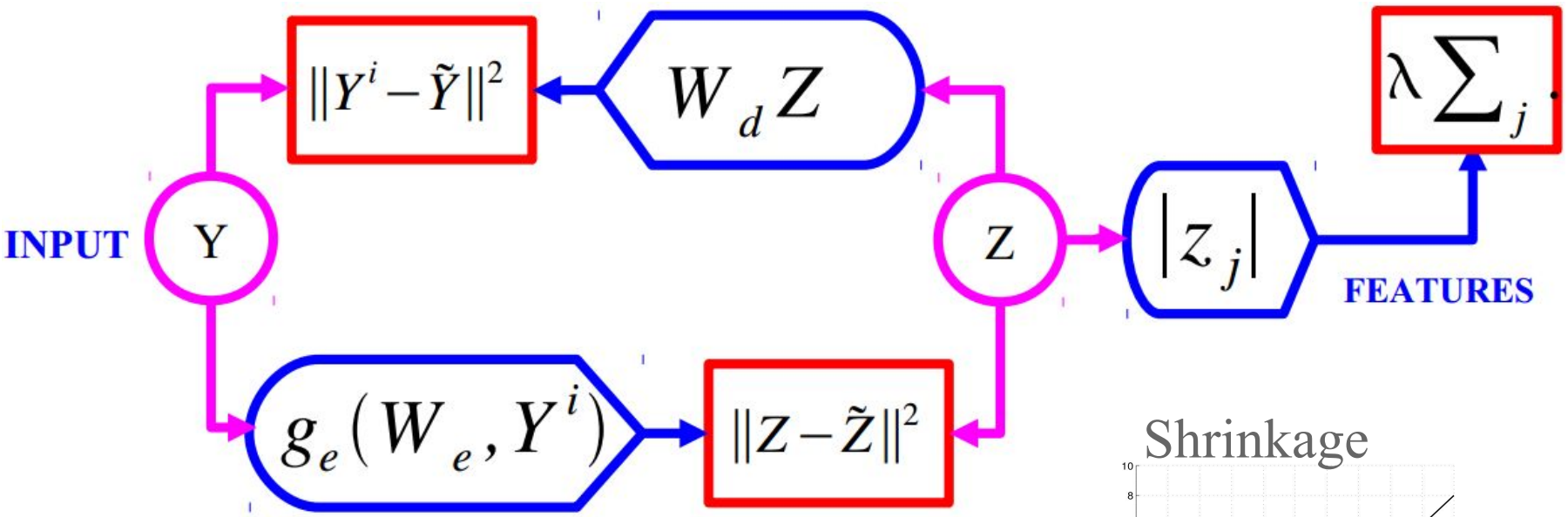
Inference is slow:

$$Y \rightarrow \hat{Z} = argmin_Z E(Y, Z)$$

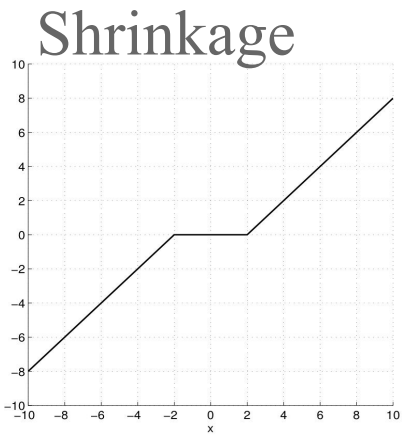
Olshausen, Bruno A, and David J Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?." *Vision research* 37.23 (1997): 3311-3325.

Predictive Sparse Decomposition (PSD)

$$E(Y^i, Z) = \|Y^i - W_d Z\|^2 + \|Z - g_e(W_e, Y^i)\|^2 + \lambda \sum_j |z_j|$$



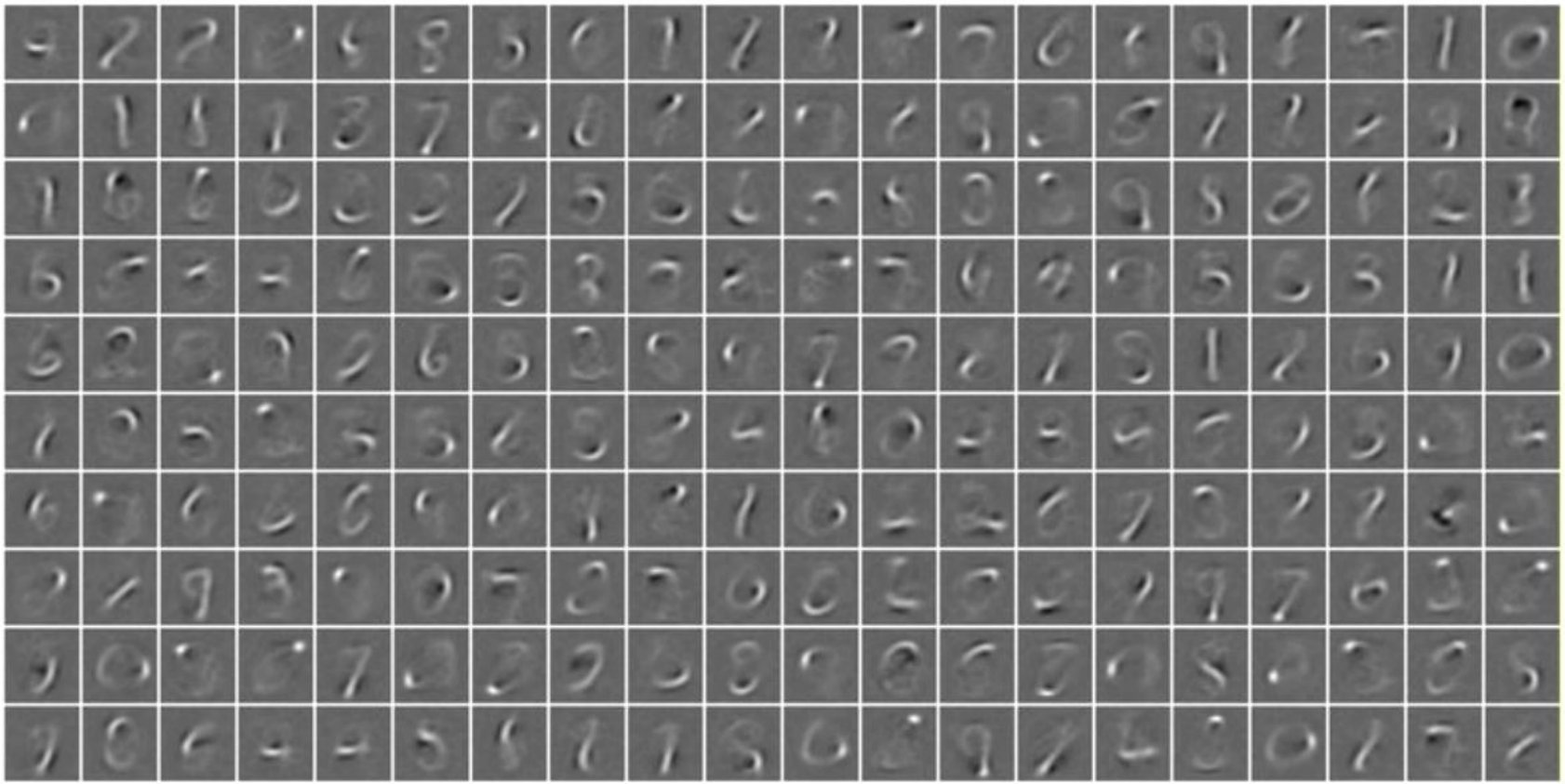
$$g_e(W_e, Y^i) = shrinkage(W_e Y^i)$$



Kavukcuoglu, Koray, Marc'Aurelio Ranzato, and Yann LeCun. "Fast inference in sparse coding algorithms with applications to object recognition." *arXiv preprint arXiv:1010.3467* (2008).

PSD: Basis Functions on MNIST

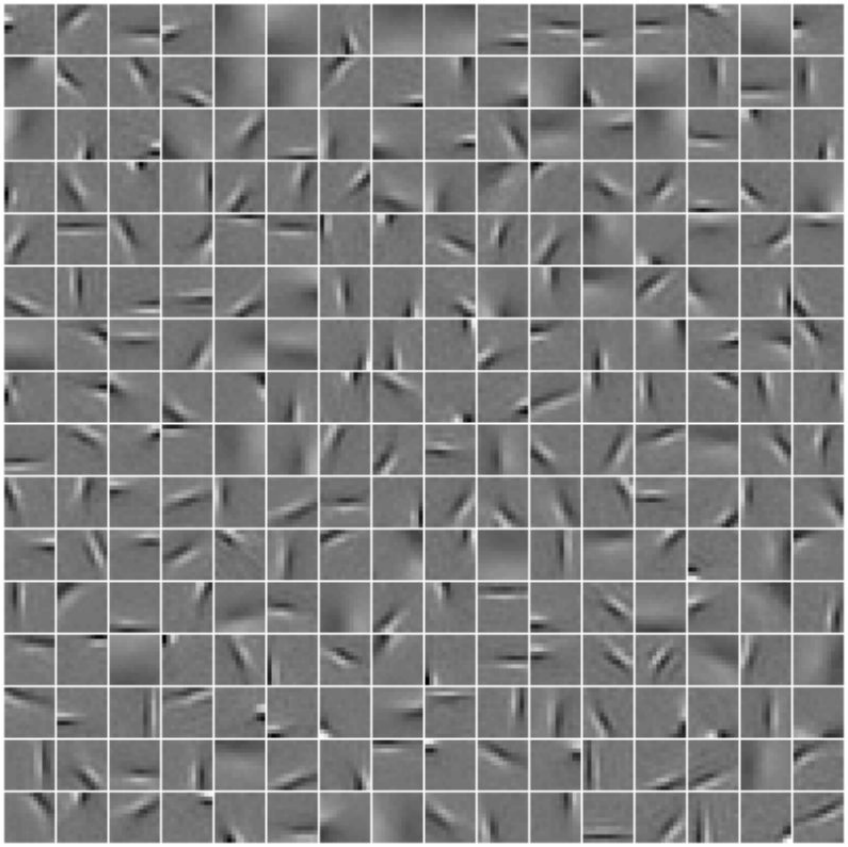
Basis functions (and encoder matrix) are digit parts



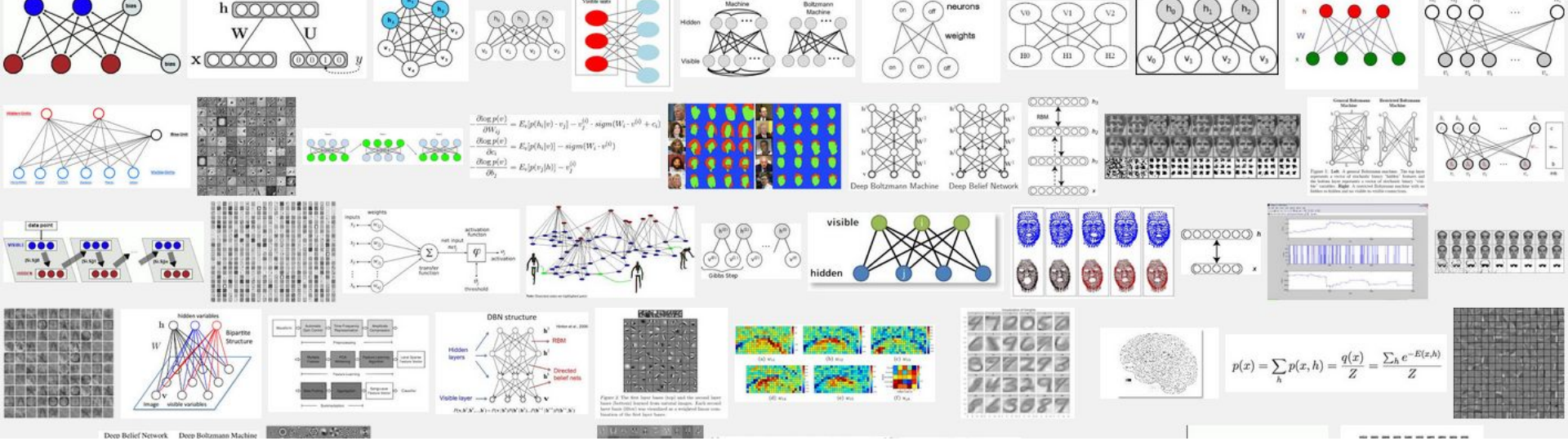
Kavukcuoglu, Koray, Marc'Aurelio Ranzato, and Yann LeCun. "Fast inference in sparse coding algorithms with applications to object recognition." *arXiv preprint arXiv:1010.3467* (2010).

PSD: Basis Functions on Natural Images Patches

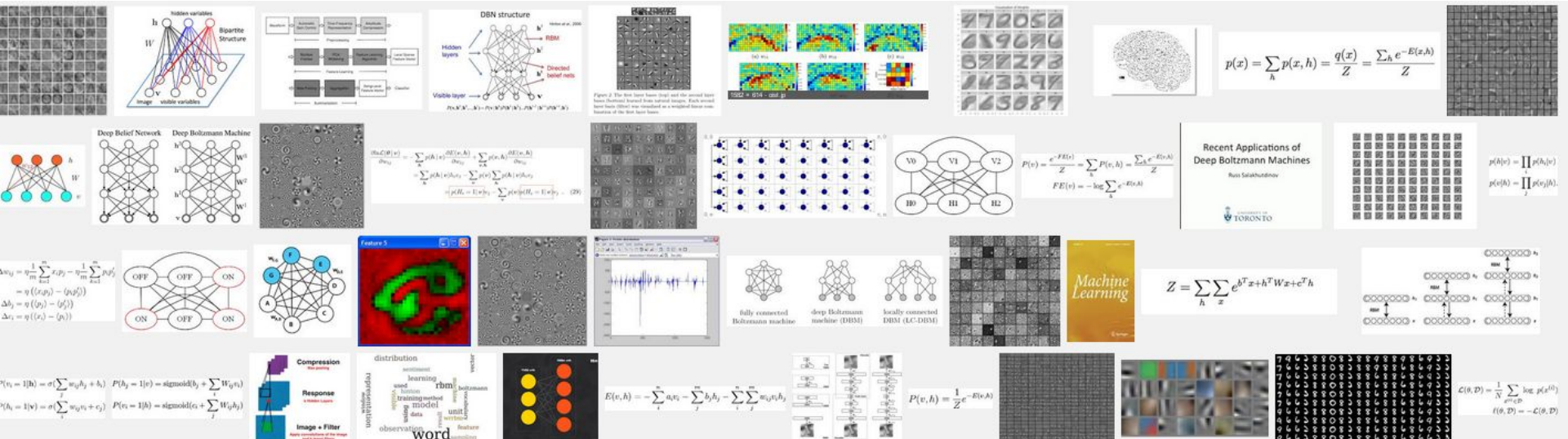
Basis functions are V1-like receptive fields



Kavukcuoglu, Koray, Marc'Aurelio Ranzato, and Yann LeCun. "Fast inference in sparse coding algorithms with applications to object recognition." *arXiv preprint arXiv:1010.3467* (2010).



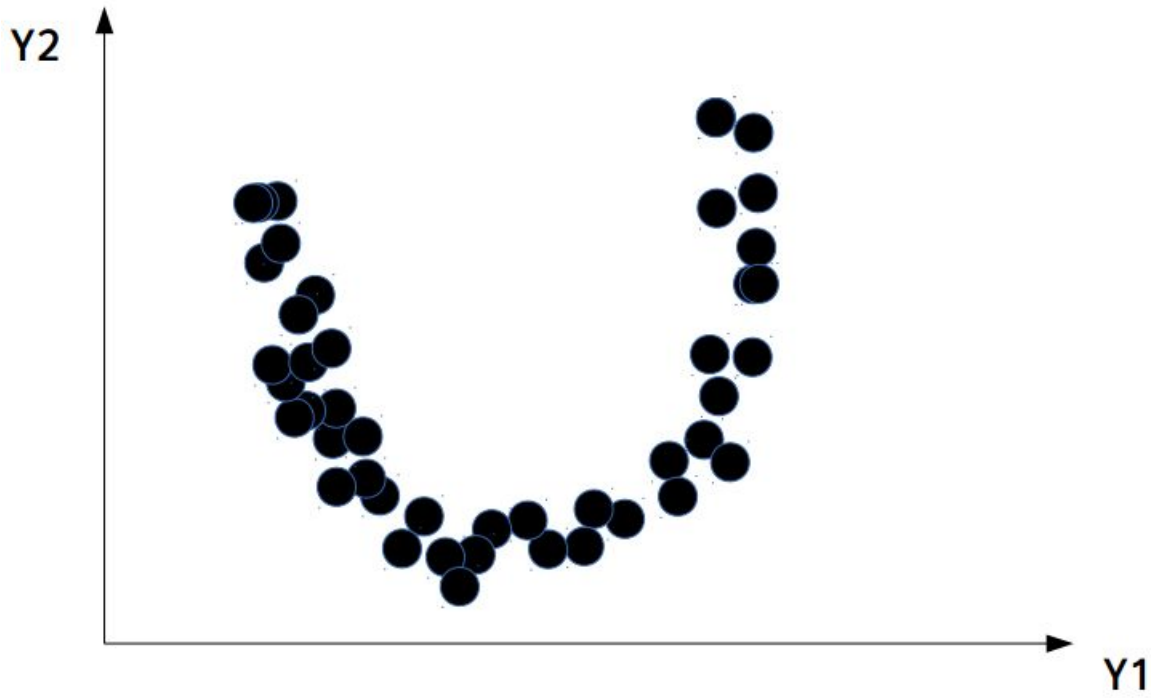
Energy-Based Unsupervised Learning



Energy-Based Unsupervised Learning

Learning an energy function that takes:

- low values on the data manifold
- higher values everywhere else

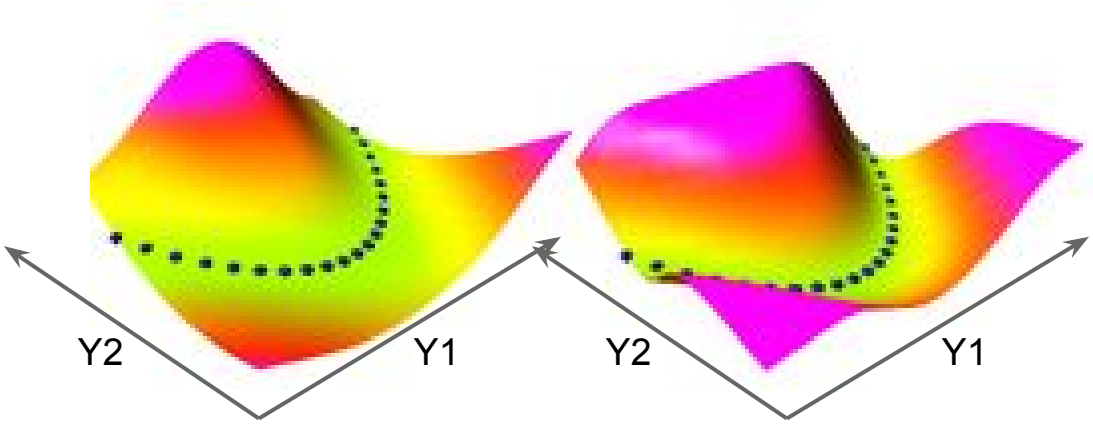


Source: Y LeCun & MA Ranzato. ICML 2013

Capturing Dependencies Between Variables with an Energy Function

The energy surface is a "contrast function" that takes low values on the data manifold, and higher values everywhere else

- Special case: energy = negative log density
- Example: the samples live in the manifold $Y2 = (Y1)^2$



Source: Y LeCun & MA Ranzato. ICML 2013

Capturing Dependencies Between Variables with an Energy Function

The energy can be interpreted as an unnormalized negative log density

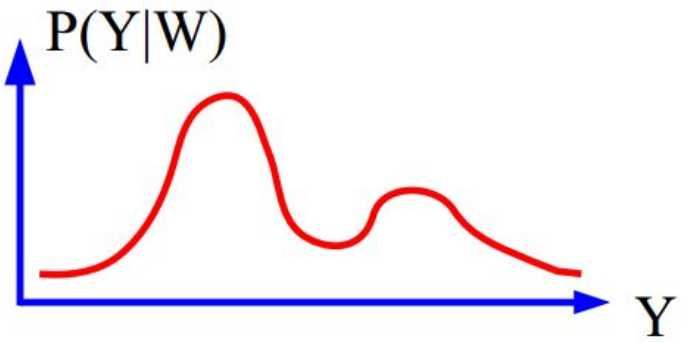
Gibbs distribution: Probability proportional to exp(-energy)

- **Beta parameter is akin to an inverse temperature**

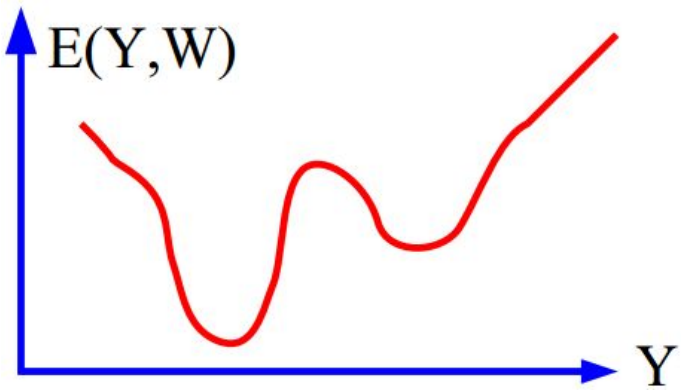
Don't compute probabilities unless you absolutely have to

- **Because the denominator is often intractable**

$$P(Y|W) = \frac{e^{-\beta E(Y,W)}}{\int_y e^{-\beta E(y,W)}}$$



$$E(Y,W) \propto -\log P(Y|W)$$

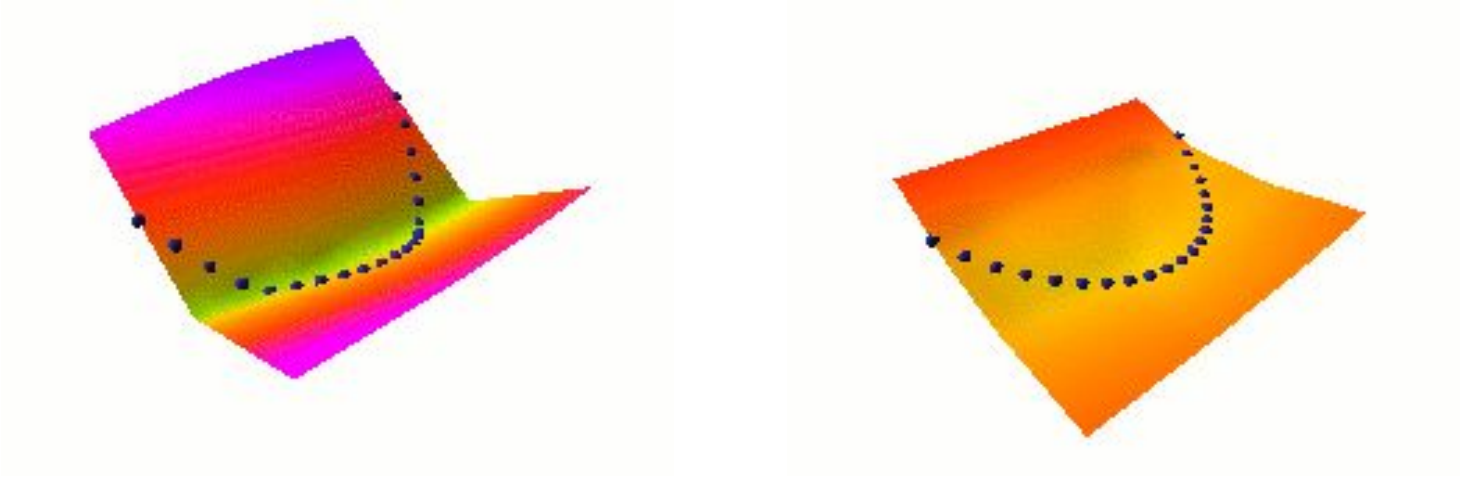


Source: Y LeCun & MA Ranzato. ICML 2013

Learning the Energy Function

parameterized energy function E(Y,W)

- Make the energy low on the samples
- Make the energy higher everywhere else
- Making the energy low on the samples is easy
- **But how do we make it higher everywhere else?**



Source: Y LeCun & MA Ranzato. ICML 2013

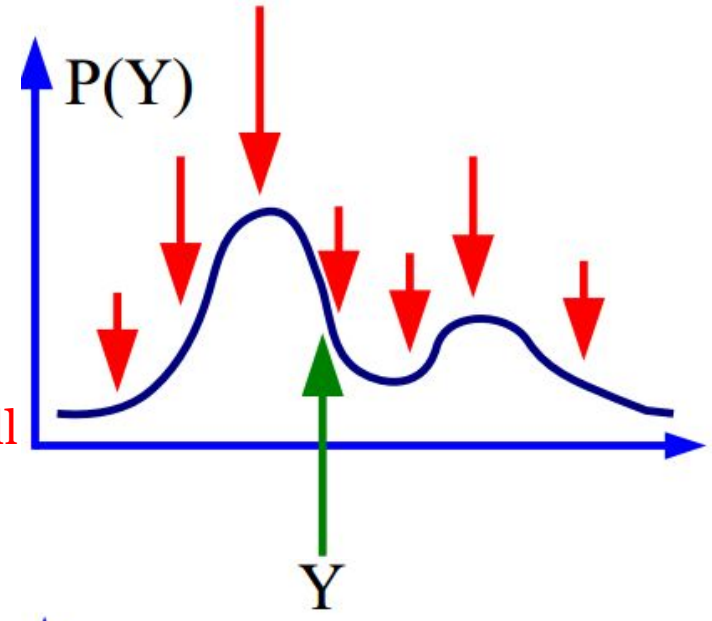
Max Likelihood

Maximizing P(Y|W)
on training samples

$$P(Y|W) = \frac{e^{-\beta E(Y,W)}}{\int_y e^{-\beta E(y,W)}}$$

make this big

make this small

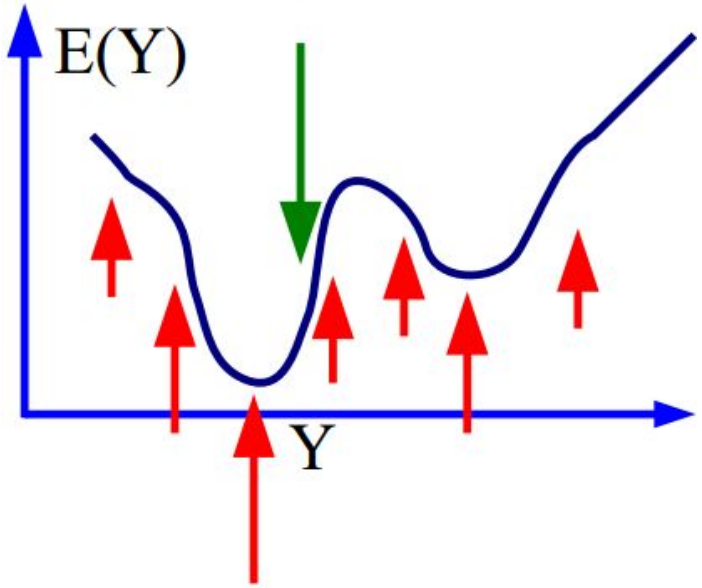


Minimizing -log P(Y,W) on
training samples

$$L(Y,W) = E(Y,W) + \frac{1}{\beta} \log \int_y e^{-\beta E(y,W)}$$

make this small

make this big

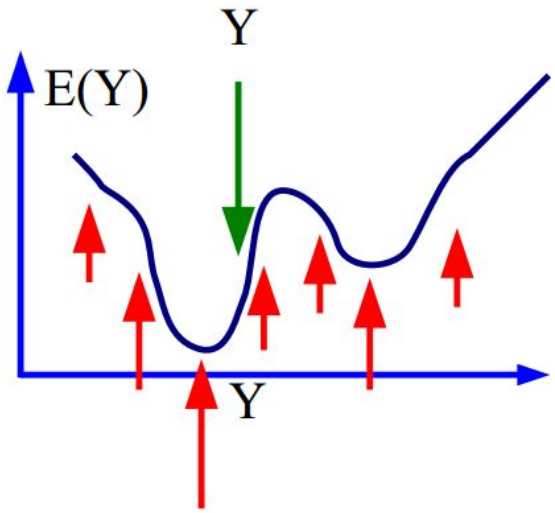


Source: Y LeCun & MA Ranzato. ICML 2013

Max Likelihood

Gradient of the negative log-likelihood loss for one sample Y:

$$\frac{\partial L(Y,W)}{\partial W} = \frac{\partial E(Y,W)}{\partial W} - \int_y P(y|W) \frac{\partial E(y,W)}{\partial W}$$



Gradient descent:

$$W \leftarrow W - \eta \frac{\partial L(Y,W)}{\partial W}$$

Pushes down on the energy of the samples

Pulls up on the energy of low-energy Y's

$$W \leftarrow W - \eta \frac{\partial E(Y,W)}{\partial W} + \eta \int_y P(y|W) \frac{\partial E(y,W)}{\partial W}$$

Source: Y LeCun & MA Ranzato. ICML 2013

Contrastive Divergence (CD)

Basic Idea:

- Pick a training sample, lower the energy at that point
- From the sample, move down in the energy surface with noise
- Stop after a while
- Push up on the energy of the point where we stopped
- This creates grooves in the energy surface around data manifolds

Persistent CD: use a bunch of "particles" and remember their positions

- Make them roll down the energy surface with noise
- Push up on the energy wherever they are
- Faster than CD

Source: Y LeCun & MA Ranzato ICML 2013