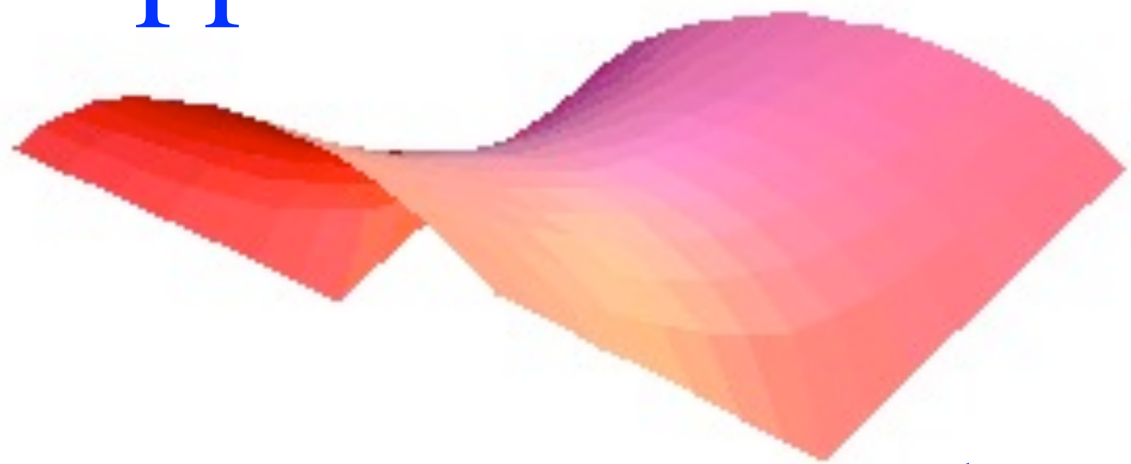
Présentation à l’Université Laval, février 2013

Modélisation de la variété (*manifold*) supportant les données



avec auto-encodeurs et densités non-normalisées

Pascal Vincent

LISA Laboratoire d’Informatique des Systèmes d’Apprentissage

Université de Montréal

Département d'informatique et de recherche opérationnelle

L'intelligence artificielle?

Historiquement deux approches en IA:

I.A. «Classique» : capacité de raisonnement logique

- Symbolique
- Ex: systèmes experts
- Ex: jeu d'échec

L'intelligence artificielle?

Historiquement deux approches en IA:

I.A. «Classique» : capacité de raisonnement logique

- Symbolique
- Ex: systèmes experts
- Ex: jeu d'échec

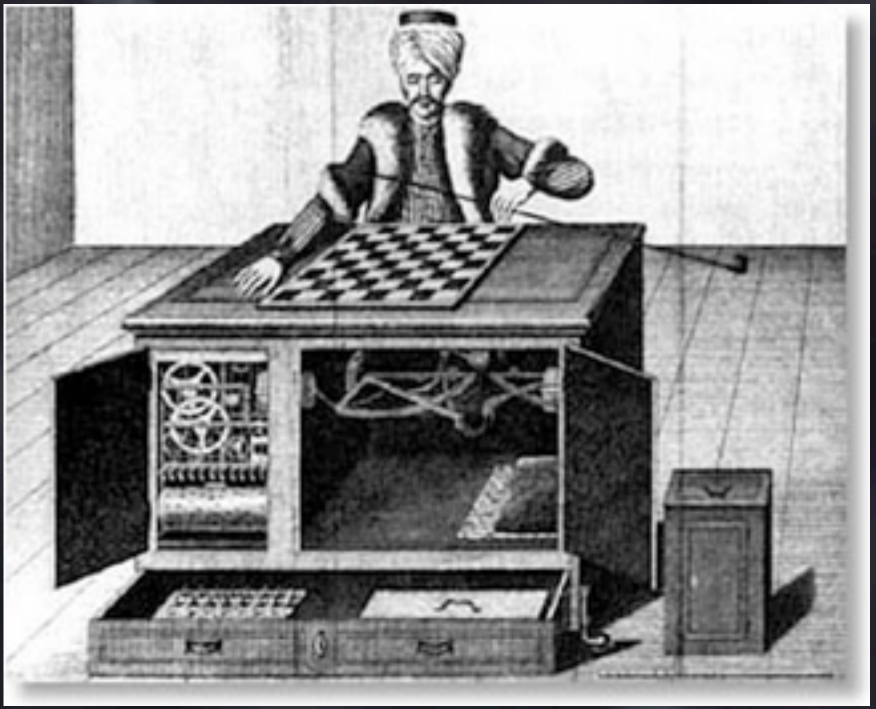
Apprentissage: capacité d'apprendre, de s'adapter à son environnement

- Sub-symbolique
- «Connexionniste»
- S'inspire du cerveau: réseaux de neurones
- Algorithmes **d'apprentissage** automatique

Ex: succès de l'IA classique

Aux échecs

1770: «Le Turc mécanique» automate joueur d'échec



A gagné contre Napoléon Bonaparte et Benjamin Franklin

Un canular!

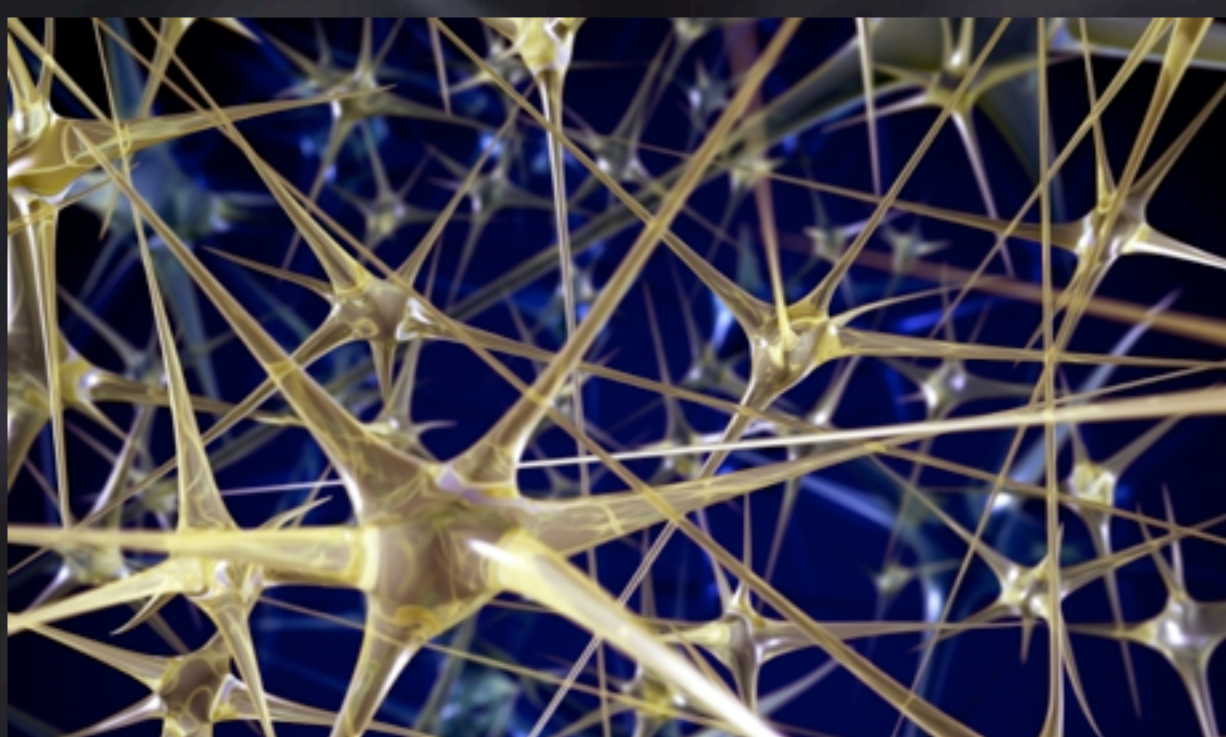
1997: Garry Kasparov contre «Deep Blue» d'IBM

Apprentissage automatique

Inspiration: cerveau qui apprend

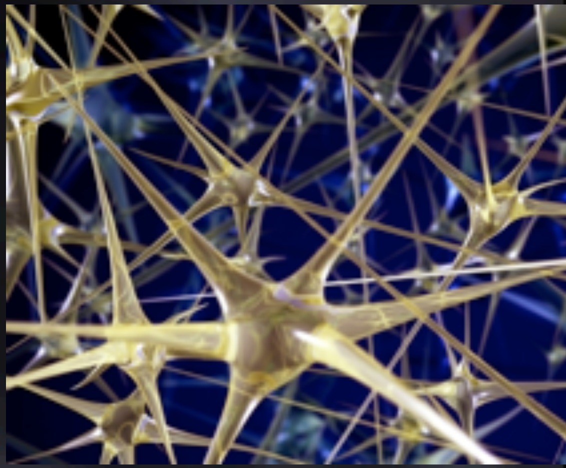
- 10^{11} neurones,
  10^{14} synapses

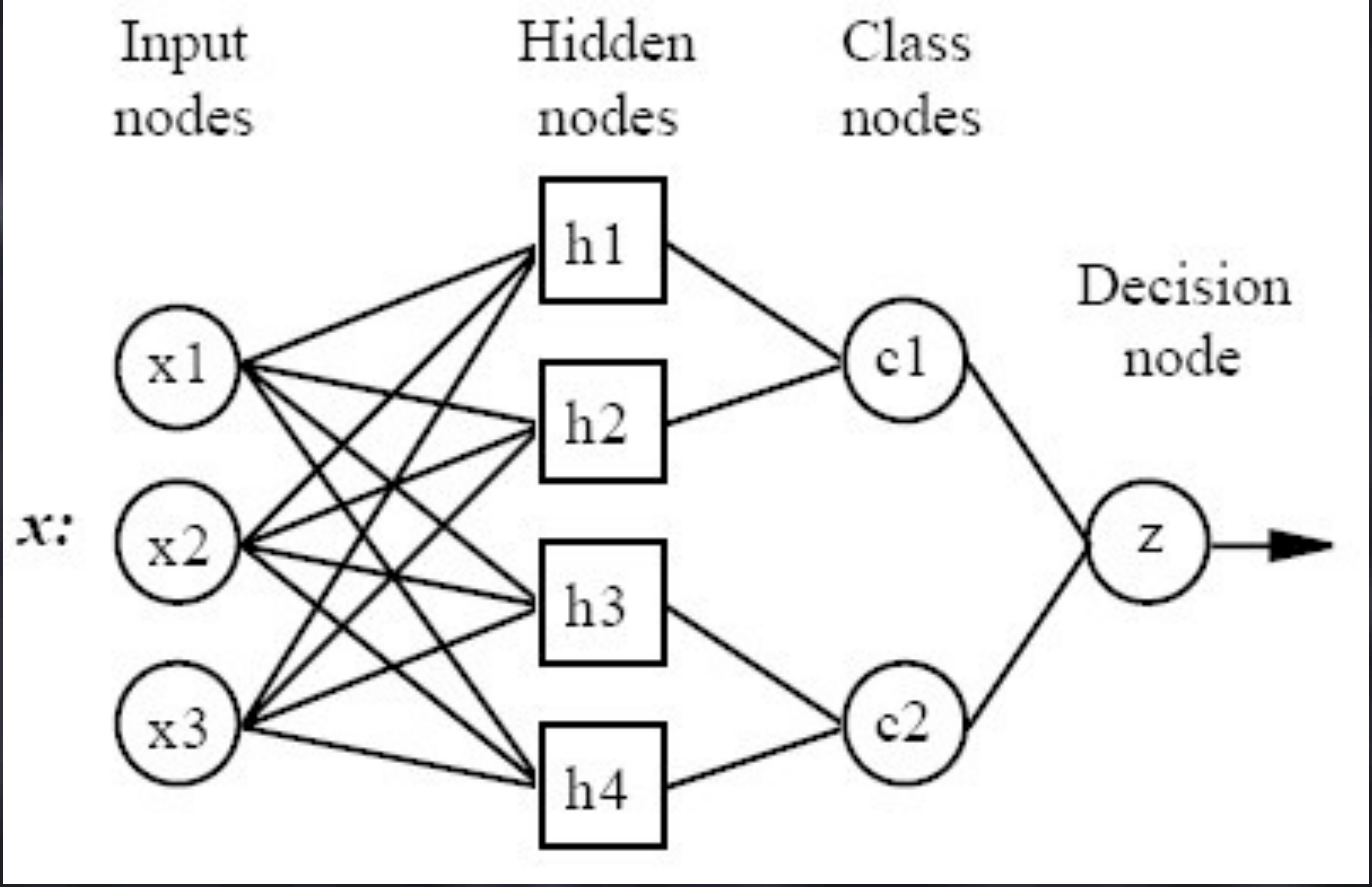
- Complexe réseau de neurones interconnectés



- Apprentissage: modifier synapses

Réseau de neurones artificiel

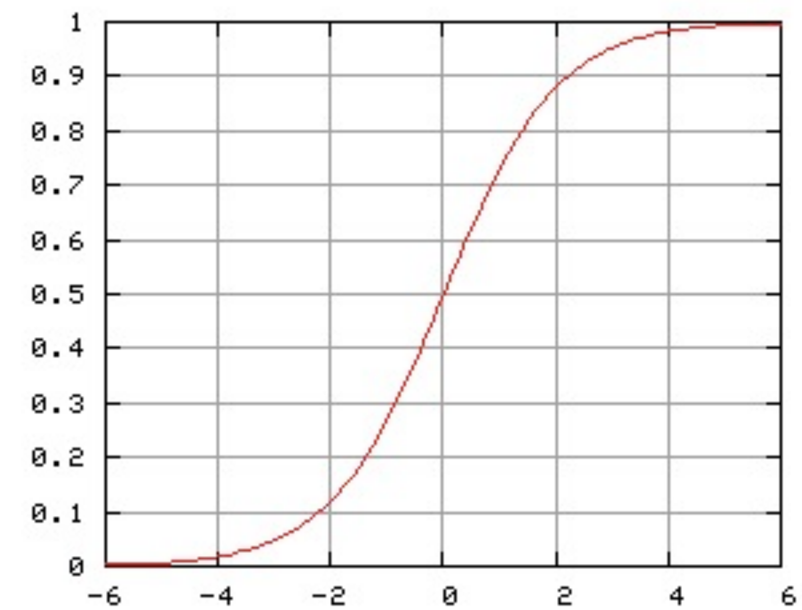


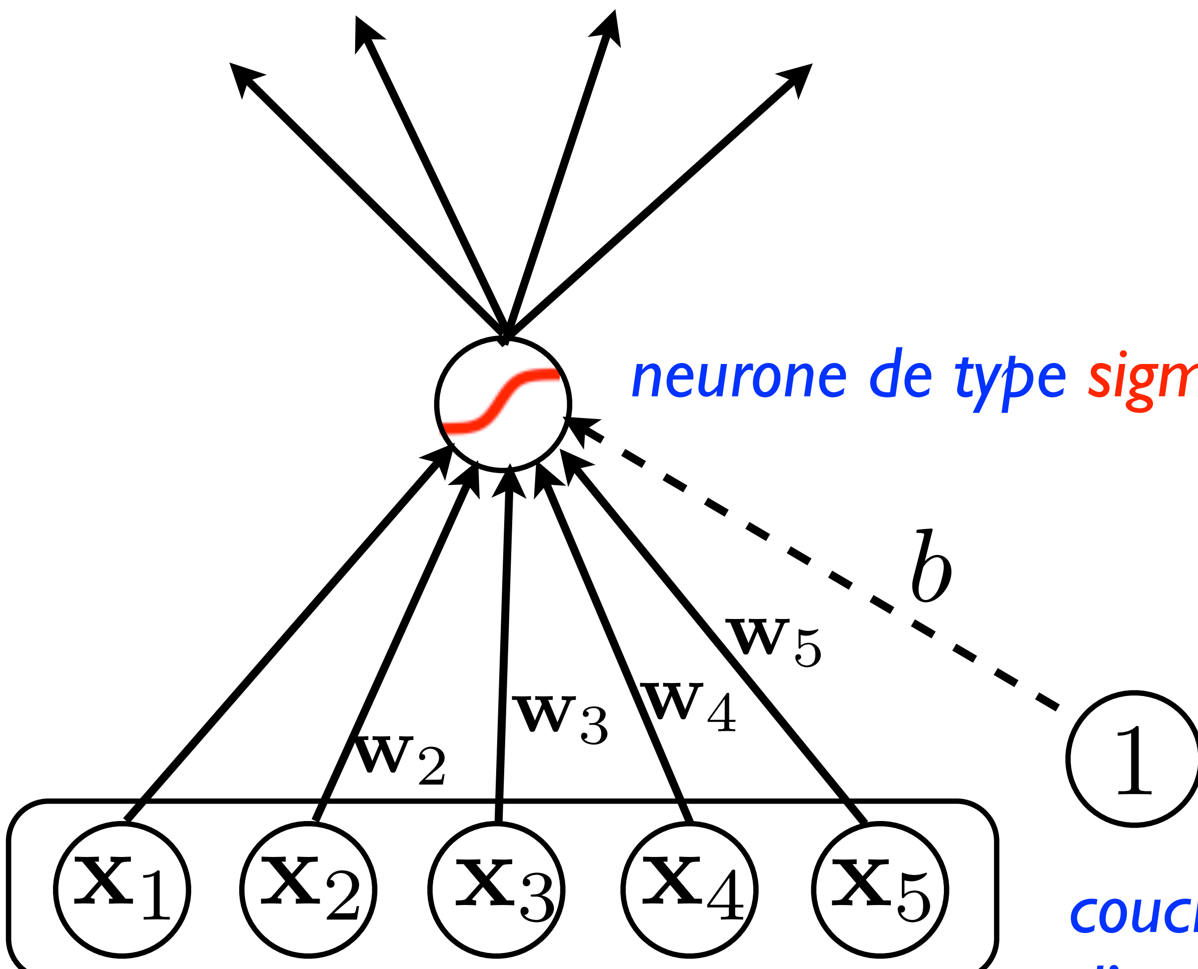


Modèle de neurone simplifié

$$\boxed{f_\theta(\mathbf{x}) = f_{\mathbf{w},b}(\mathbf{x}) = \underbrace{\mathrm{sigmoid}}_{\text{non-linéarité, fonction d'activation}}(\langle \mathbf{w}, \mathbf{x} \rangle + b)}$$

logistic $\mathrm{sigmoid}(x) = \dfrac{1}{1+e^{-x}}$





neurone de type sigmoide

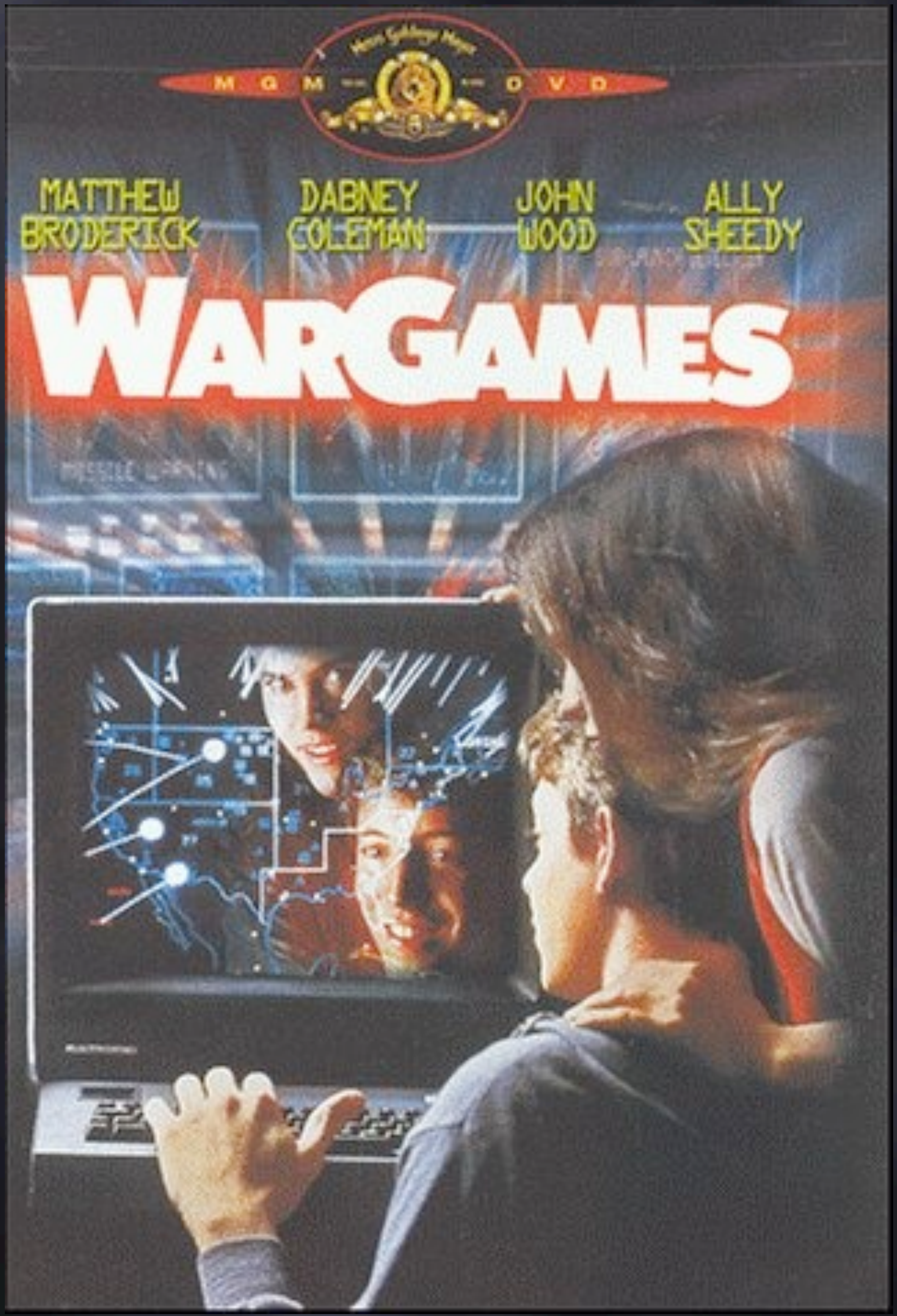
couche de neurones d'entrée

entrée (observation) $\mathbf{x}$

I.A. apprenante:

de la science-fiction...

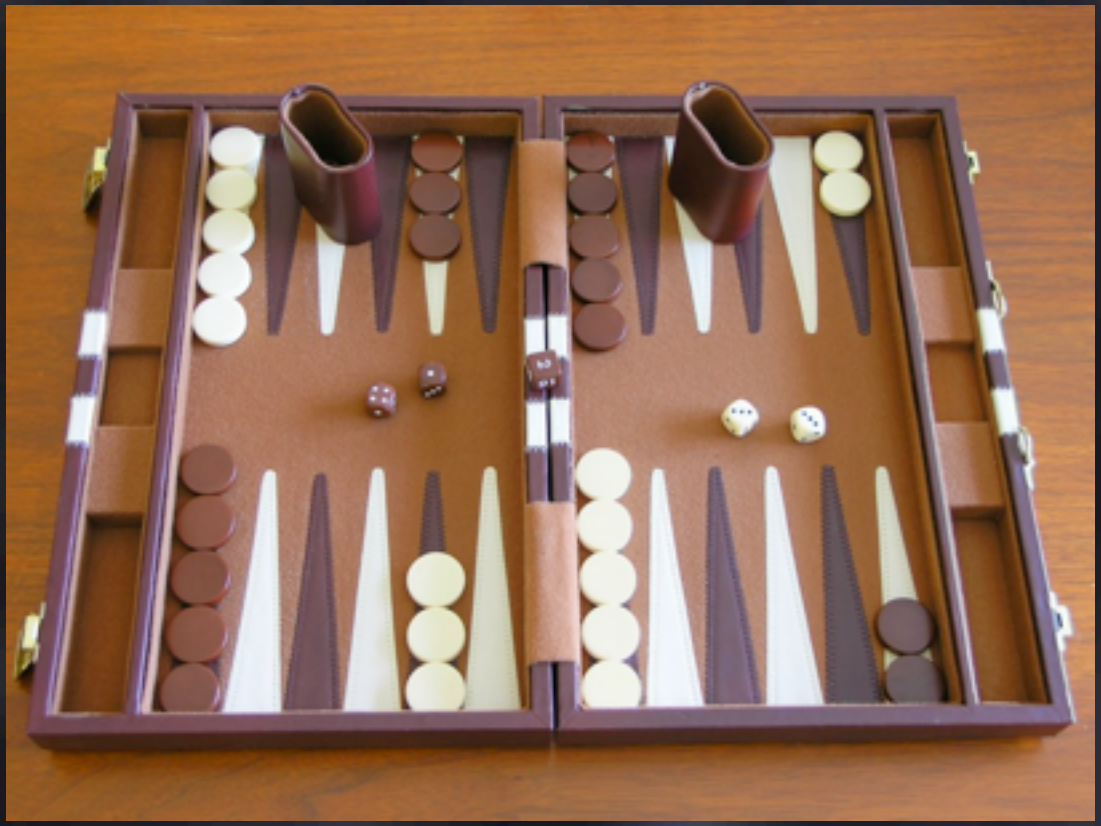
- 1983: dans WarGames, un ordinateur apprend en jouant contre lui-même à tic-tac-toe et “global thermonuclear war”.



à la réalité...

Au Backgammon

- 1995: TD-gammon, un **réseau de neurones artificiel** entraîné en jouant 200 000 parties de backgammon contre lui-même, joue à un niveau équivalent aux meilleurs joueurs mondiaux (Tesauro 1995).

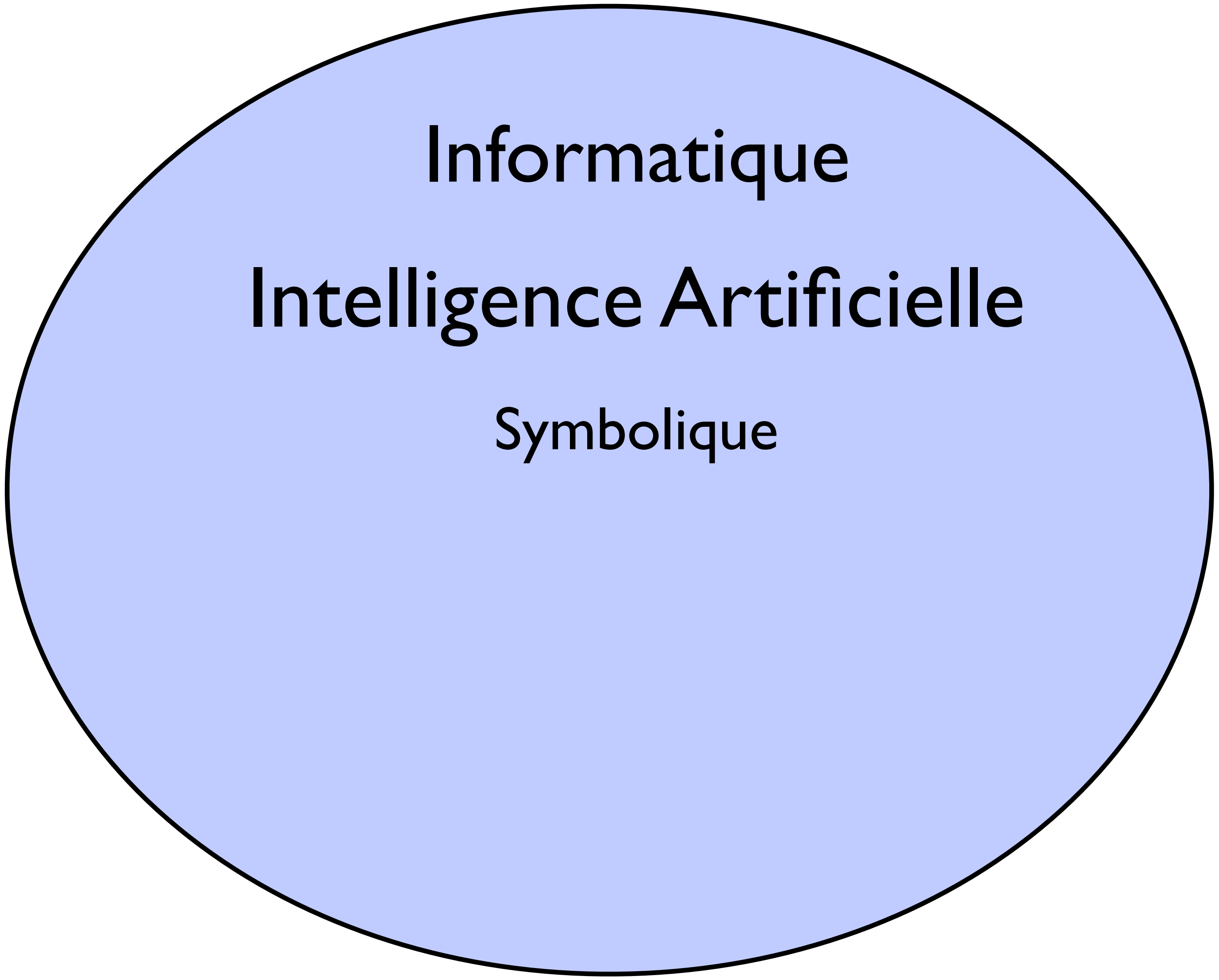


À Geopardy

- Février 2011: Watson, système d'IBM, bat les champions humains de Geopardy.
  Fondé sur l'apprentissage automatique à partir de données textuelles.

Vision de l'Intelligence Artificielle en **1957** (Rosenblatt, "Perceptron")

Vision de l'Intelligence Artificielle en **1957** (Rosenblatt, "Perceptron")



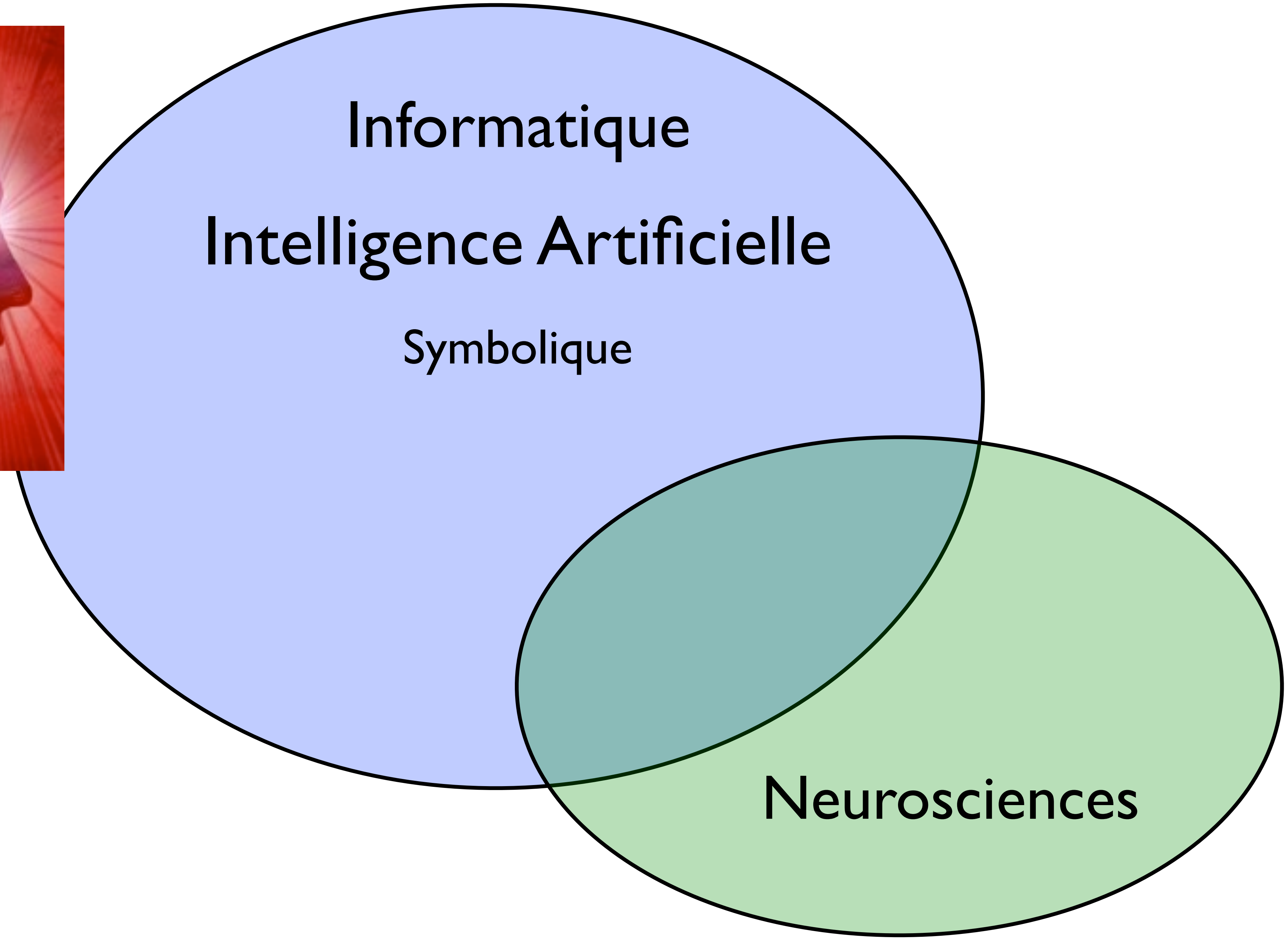
Vision de l'Intelligence Artificielle en **1957** (Rosenblatt, "Perceptron")

Informatique

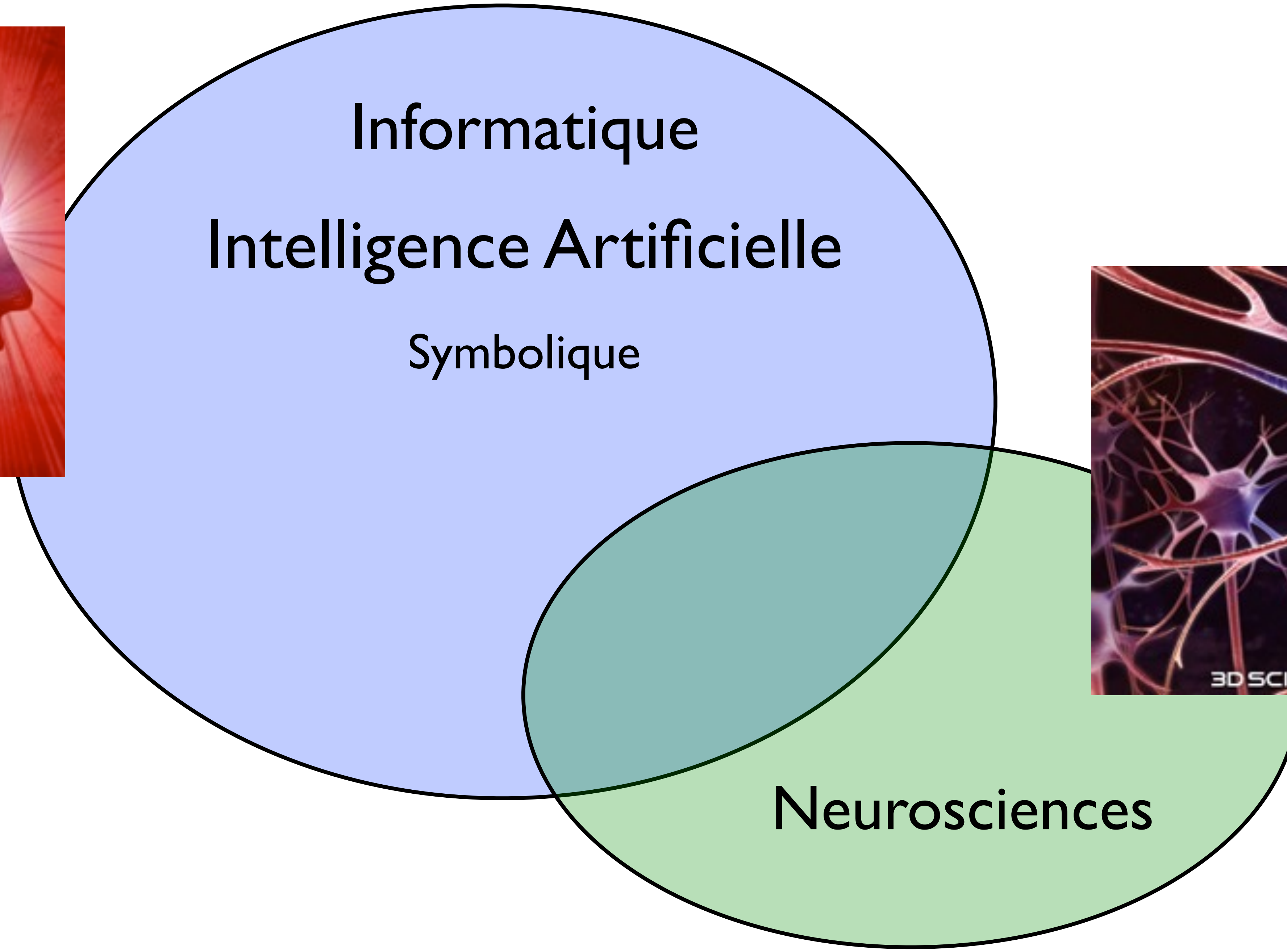
Intelligence Artificielle

Symbolique

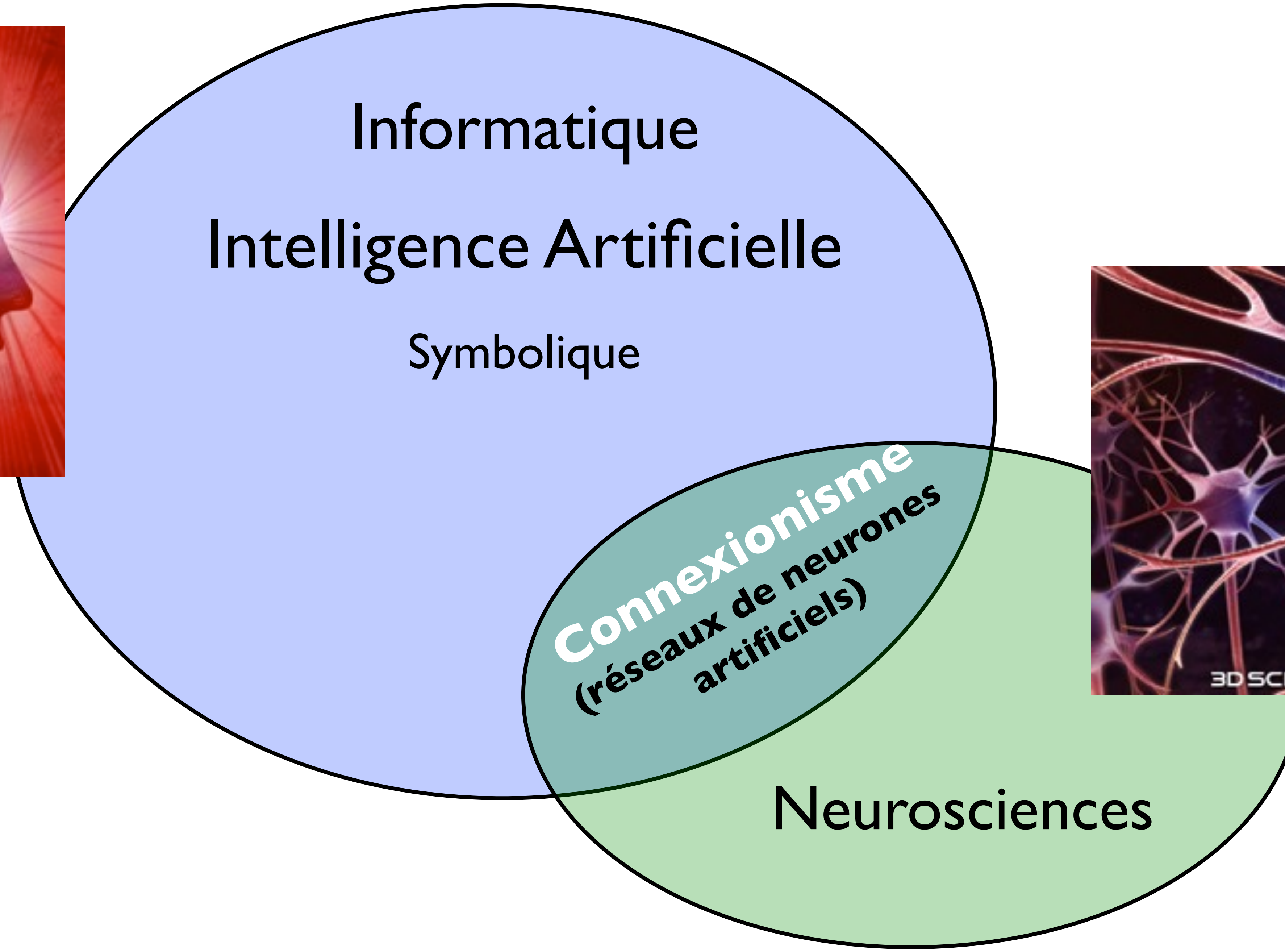
Vision de l'Intelligence Artificielle en **1957** (Rosenblatt, "Perceptron")



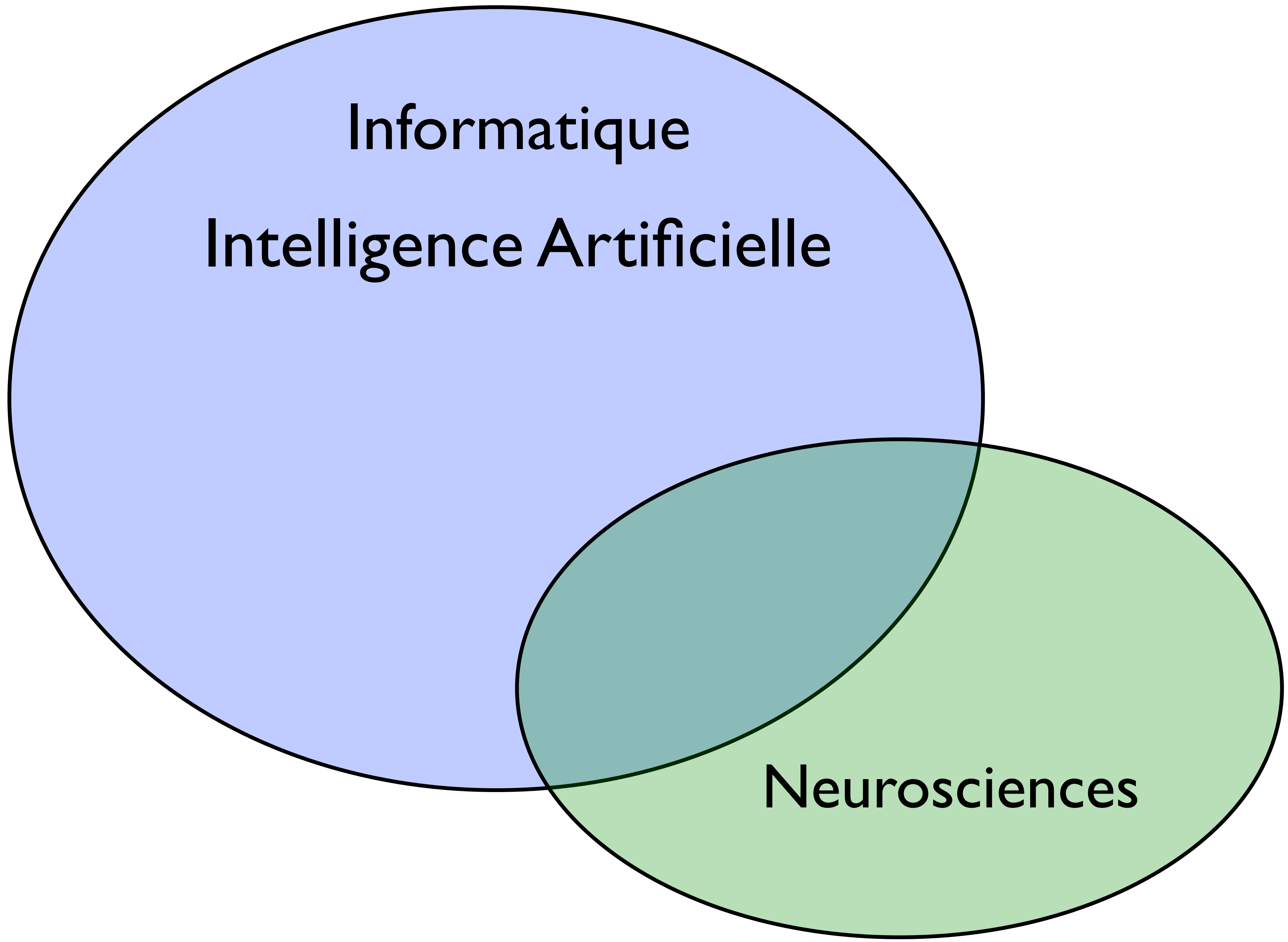
Vision de l'Intelligence Artificielle en **1957** (Rosenblatt, "Perceptron")



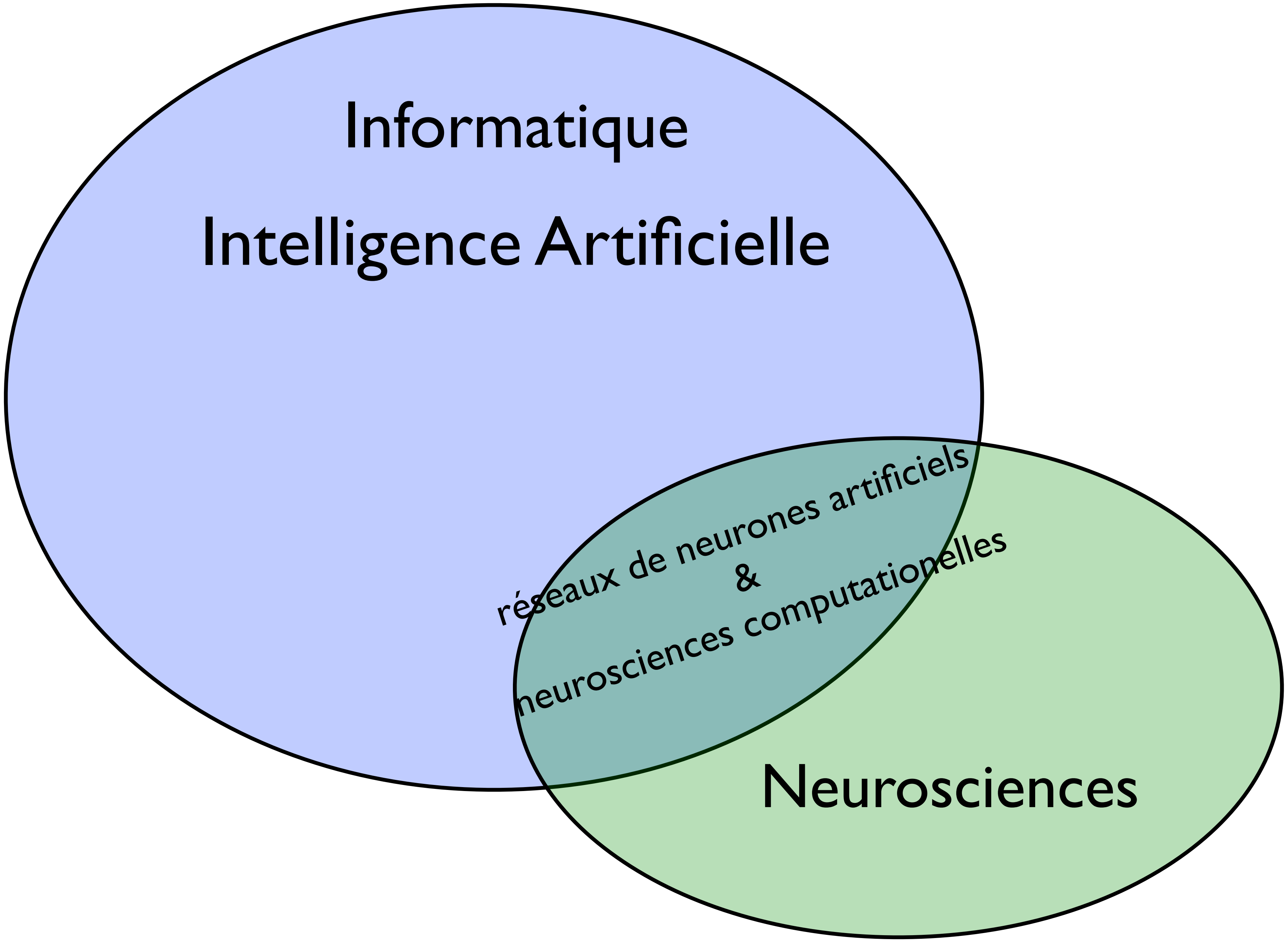
Vision de l'Intelligence Artificielle en **1957** (Rosenblatt, "Perceptron")



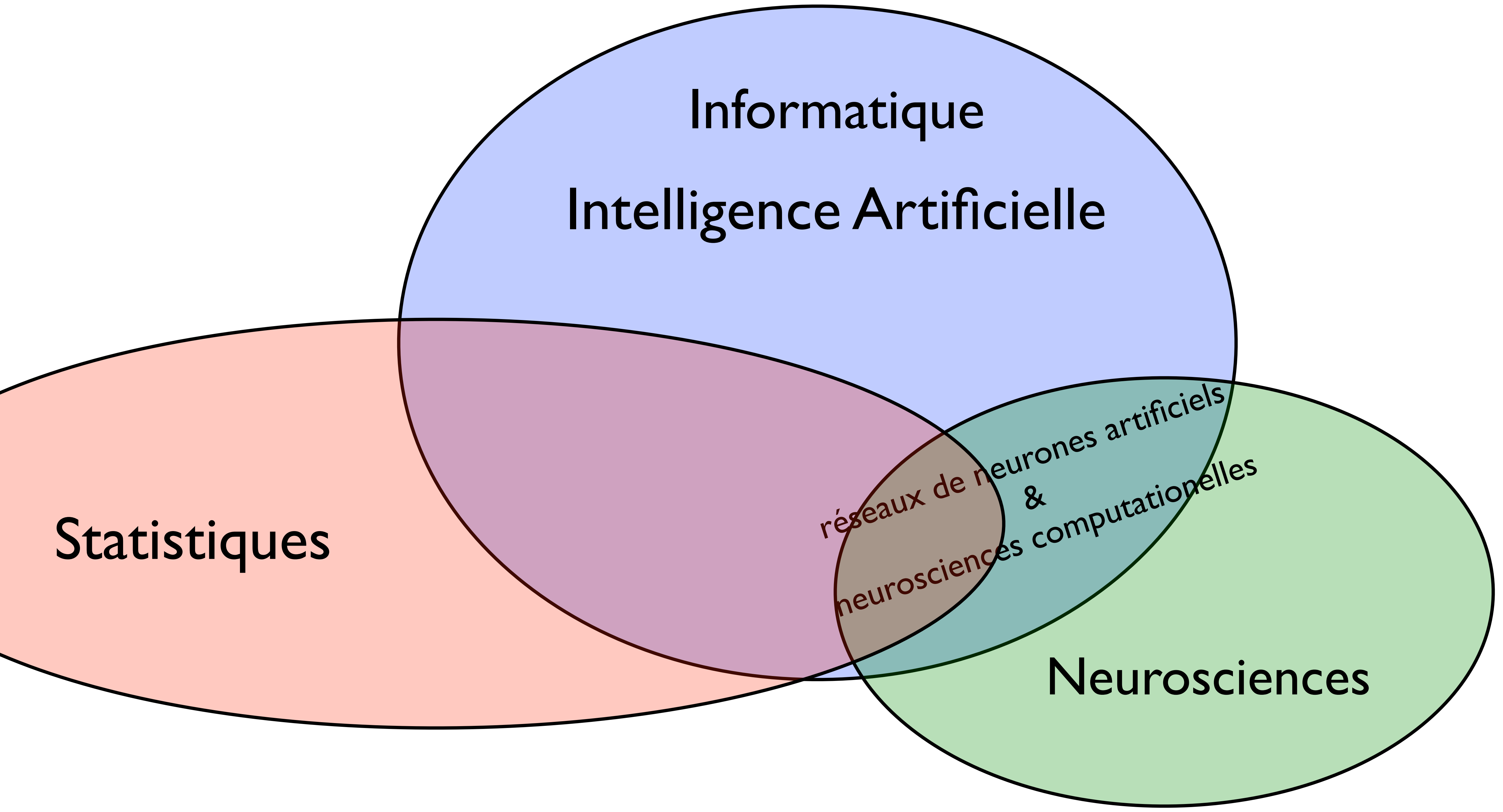
Vision actuelle des disciplines fondatrices



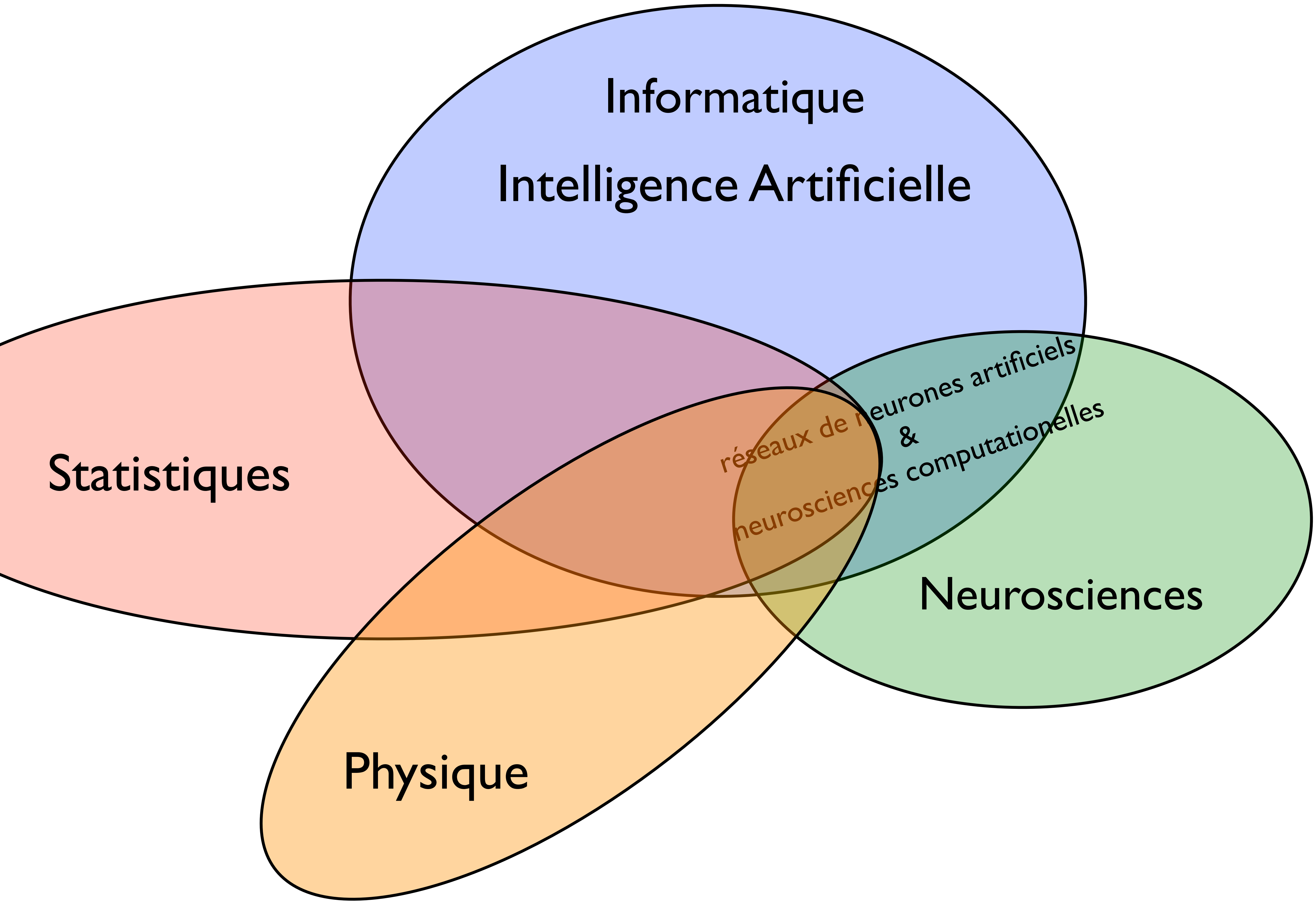
Vision actuelle des disciplines fondatrices



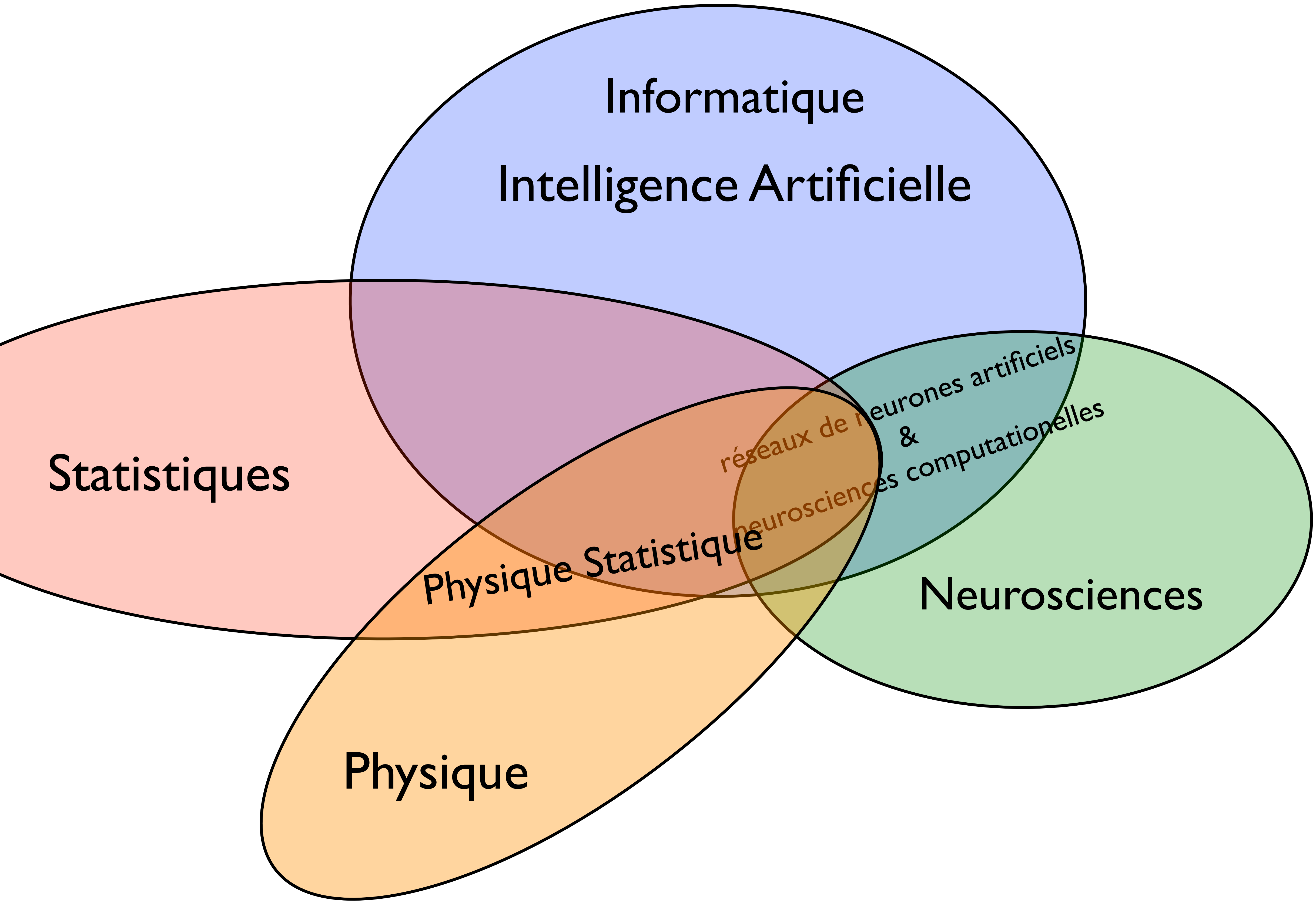
Vision actuelle des disciplines fondatrices



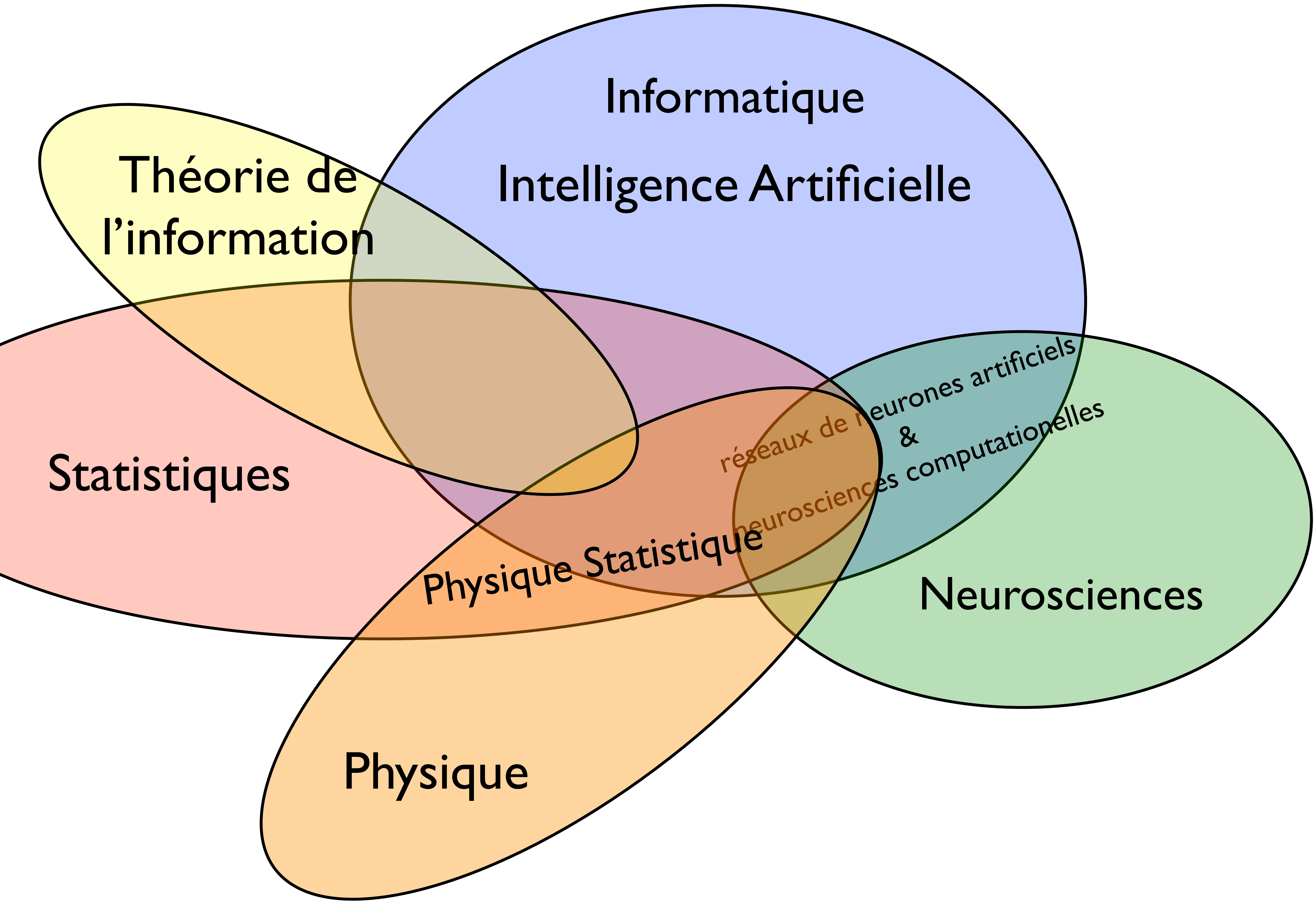
Vision actuelle des disciplines fondatrices



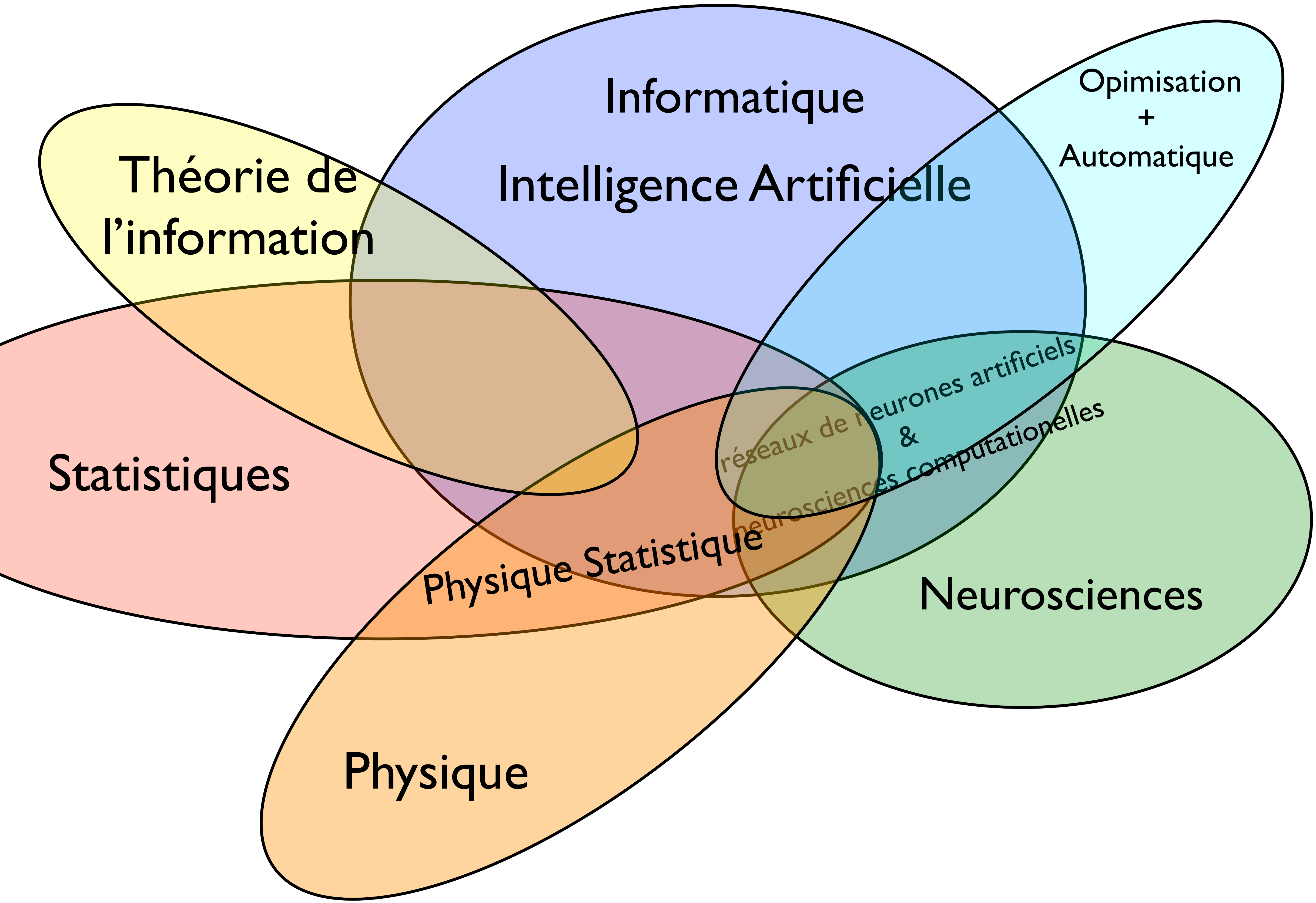
Vision actuelle des disciplines fondatrices



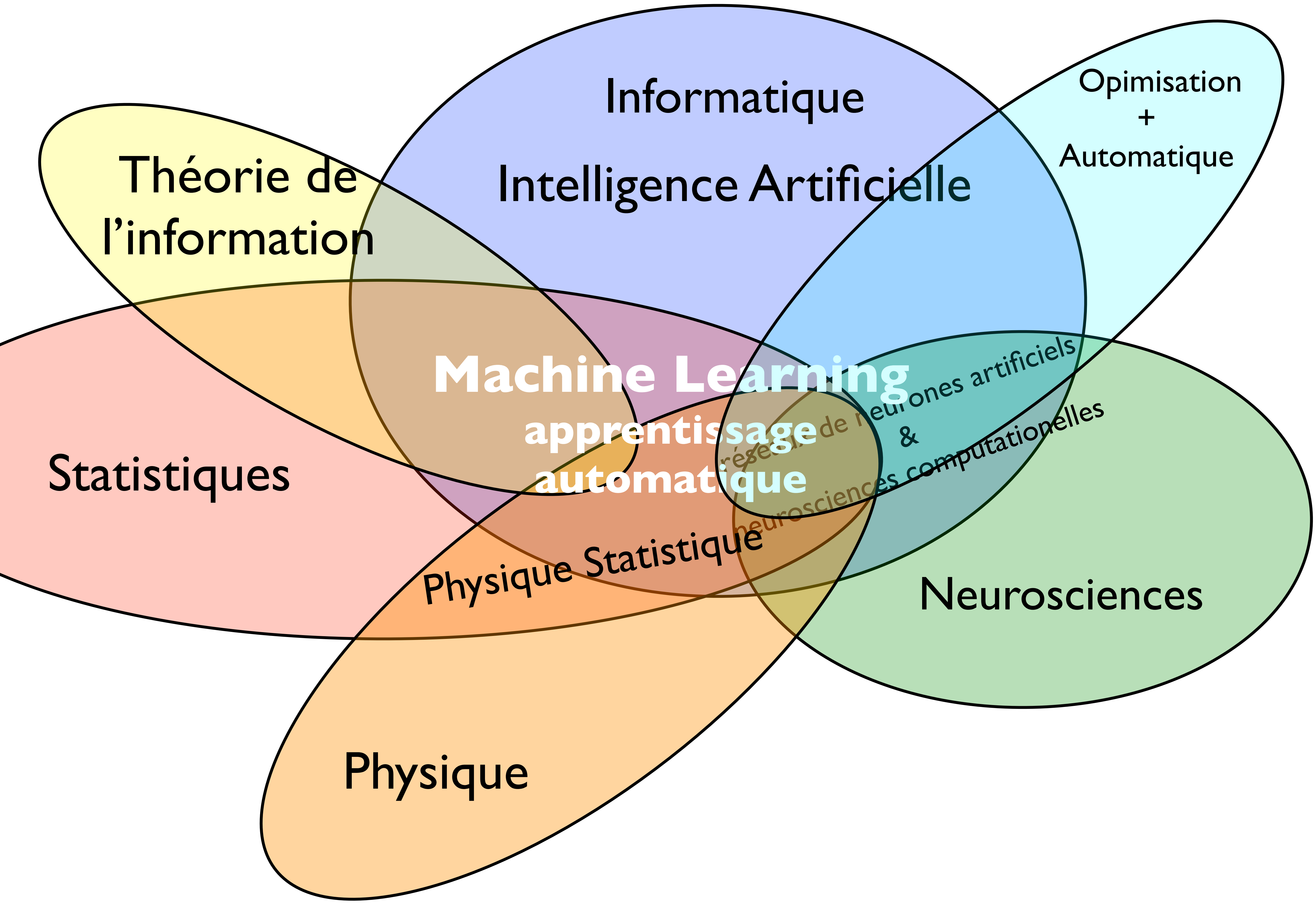
Vision actuelle des disciplines fondatrices



Vision actuelle des disciplines fondatrices

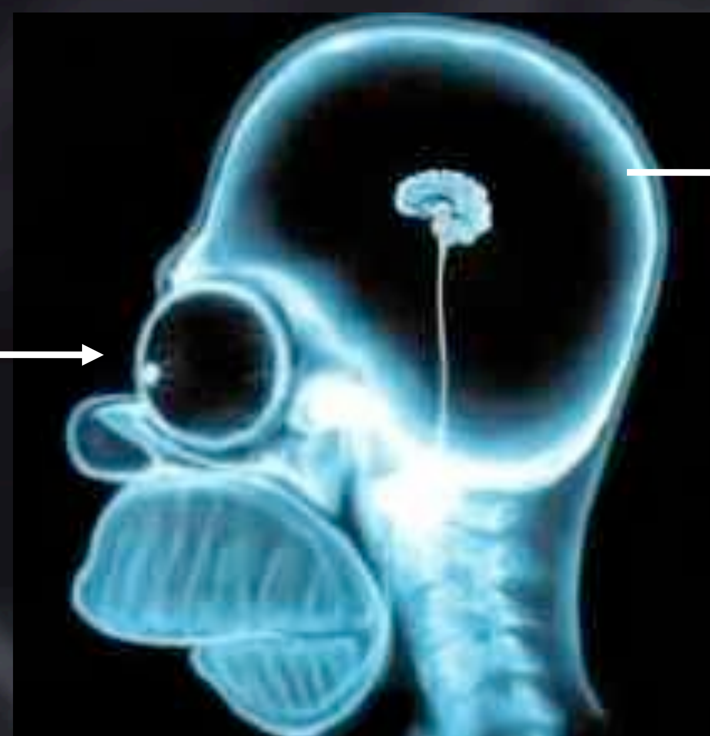


Vision actuelle des disciplines fondatrices



Exemple d'apprentissage
(supervisé)

Entrée X



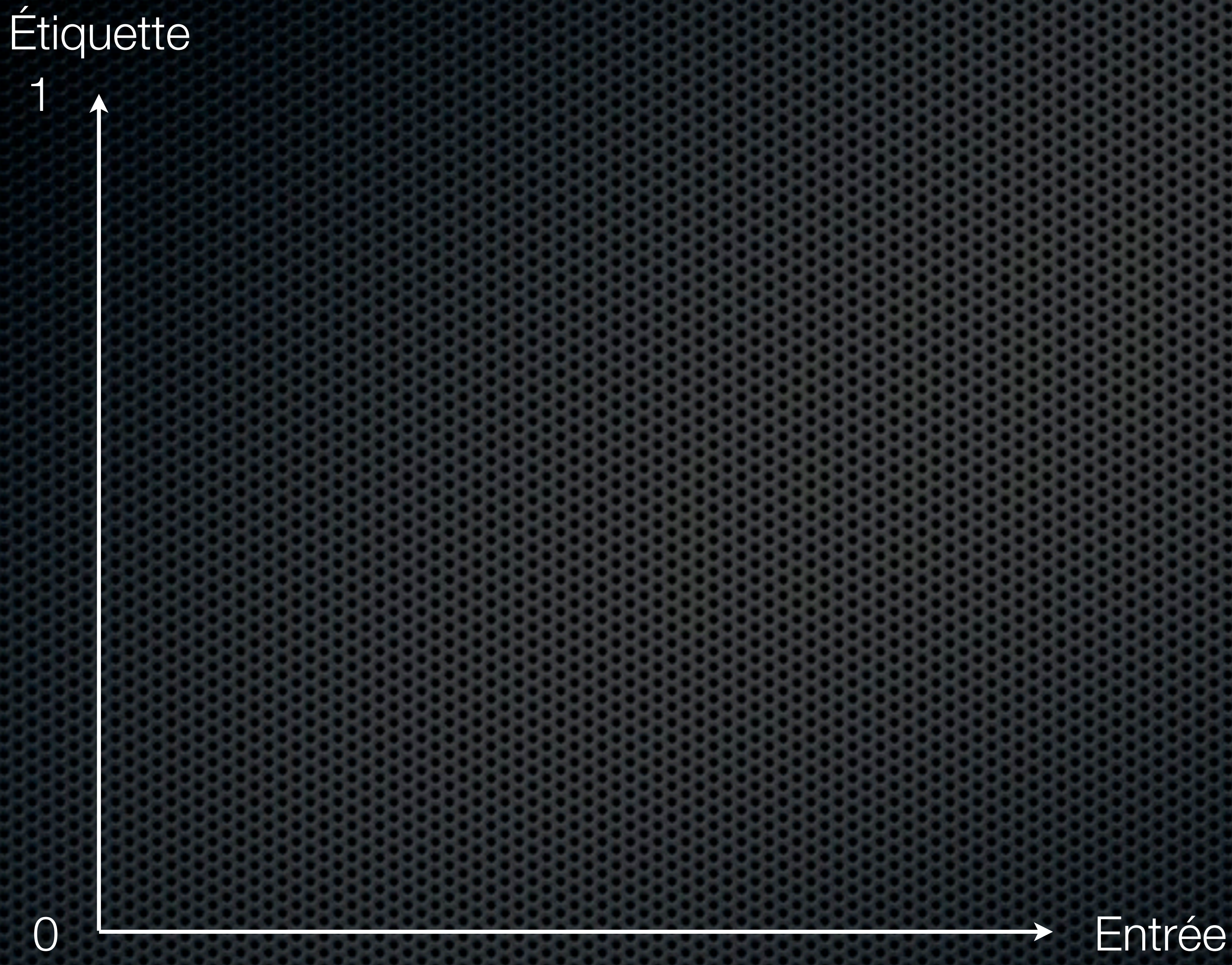
Sortie f(X)

six

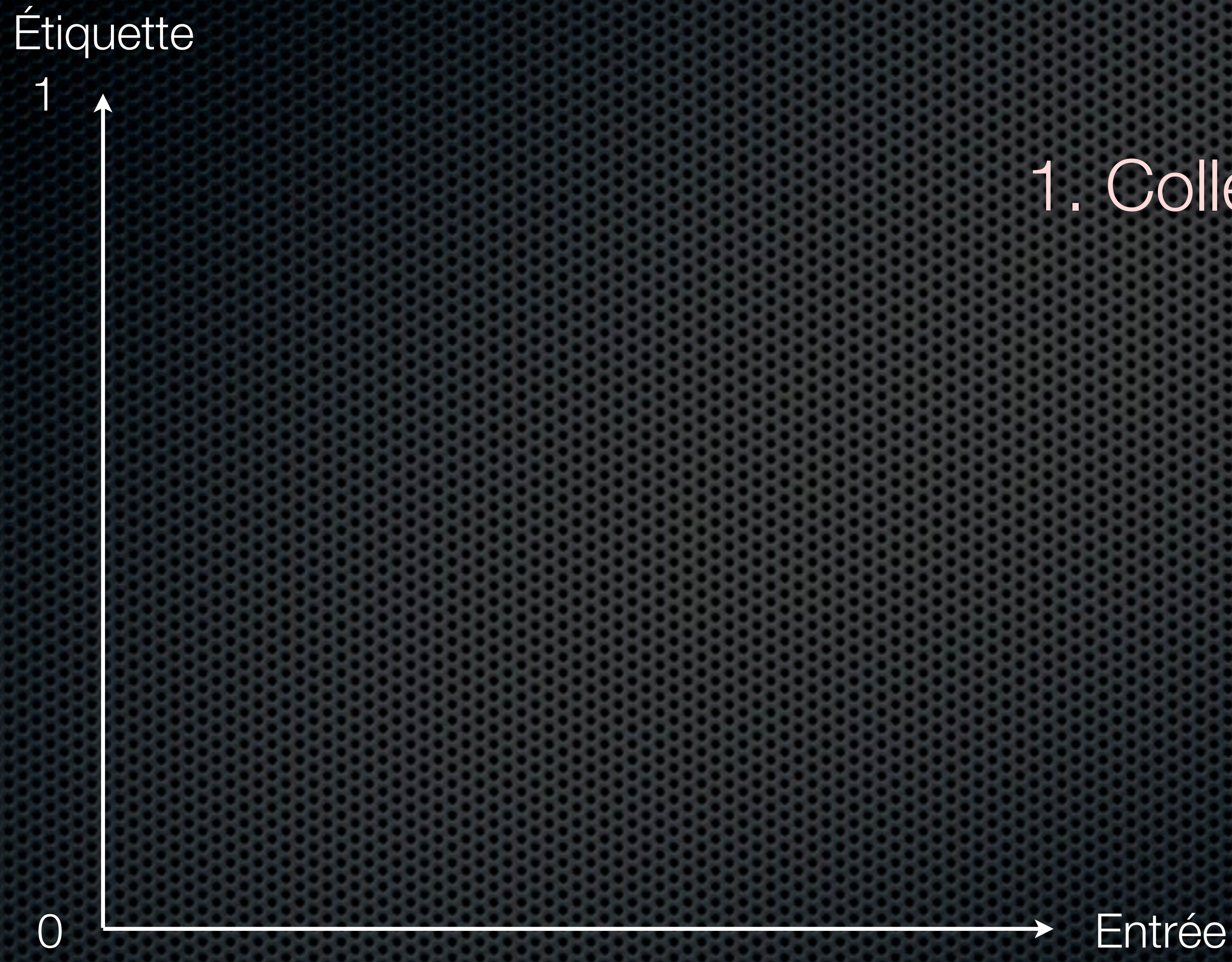
Cible Y

deux!

Ex: régression (1D)

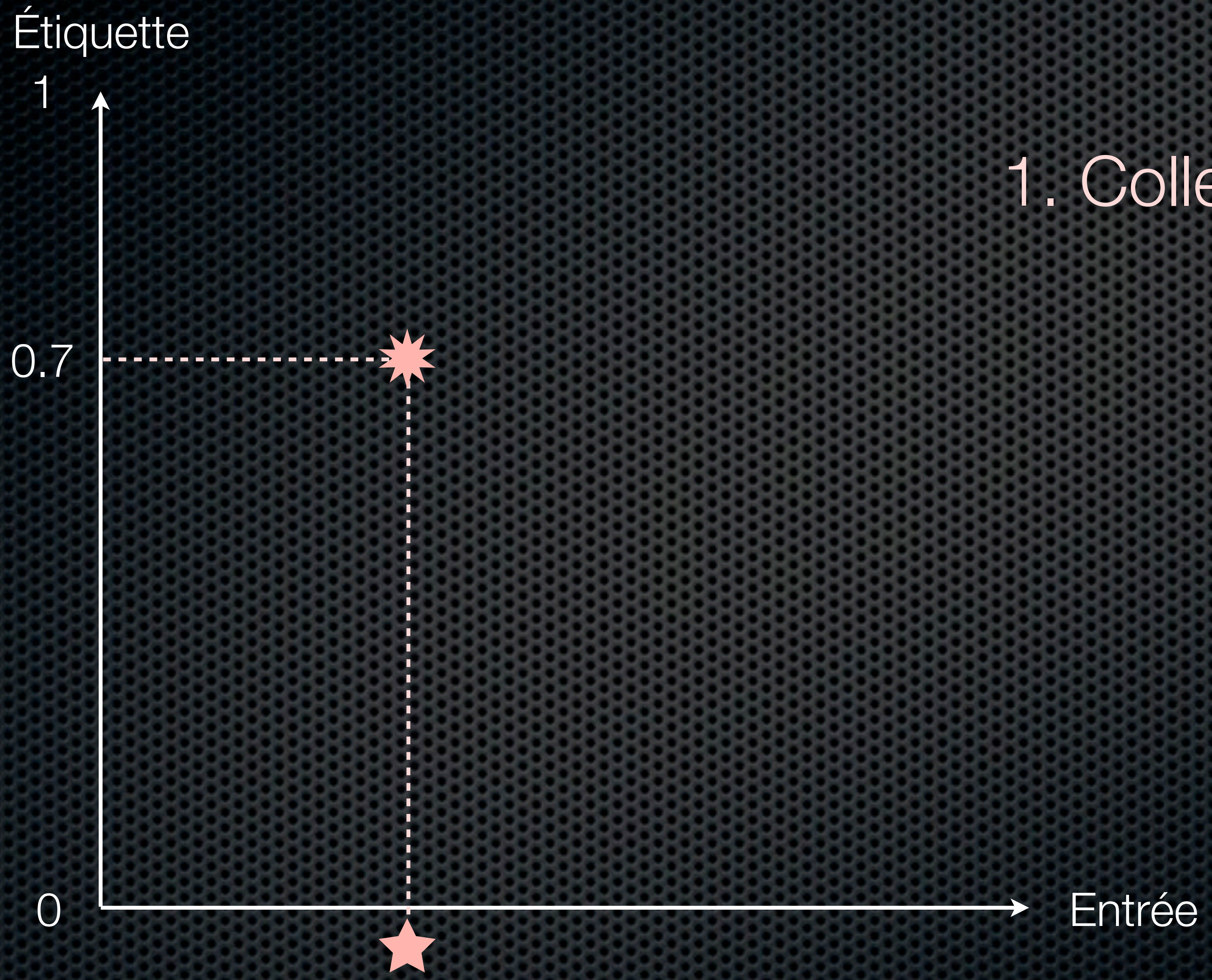


Ex: régression (1D)



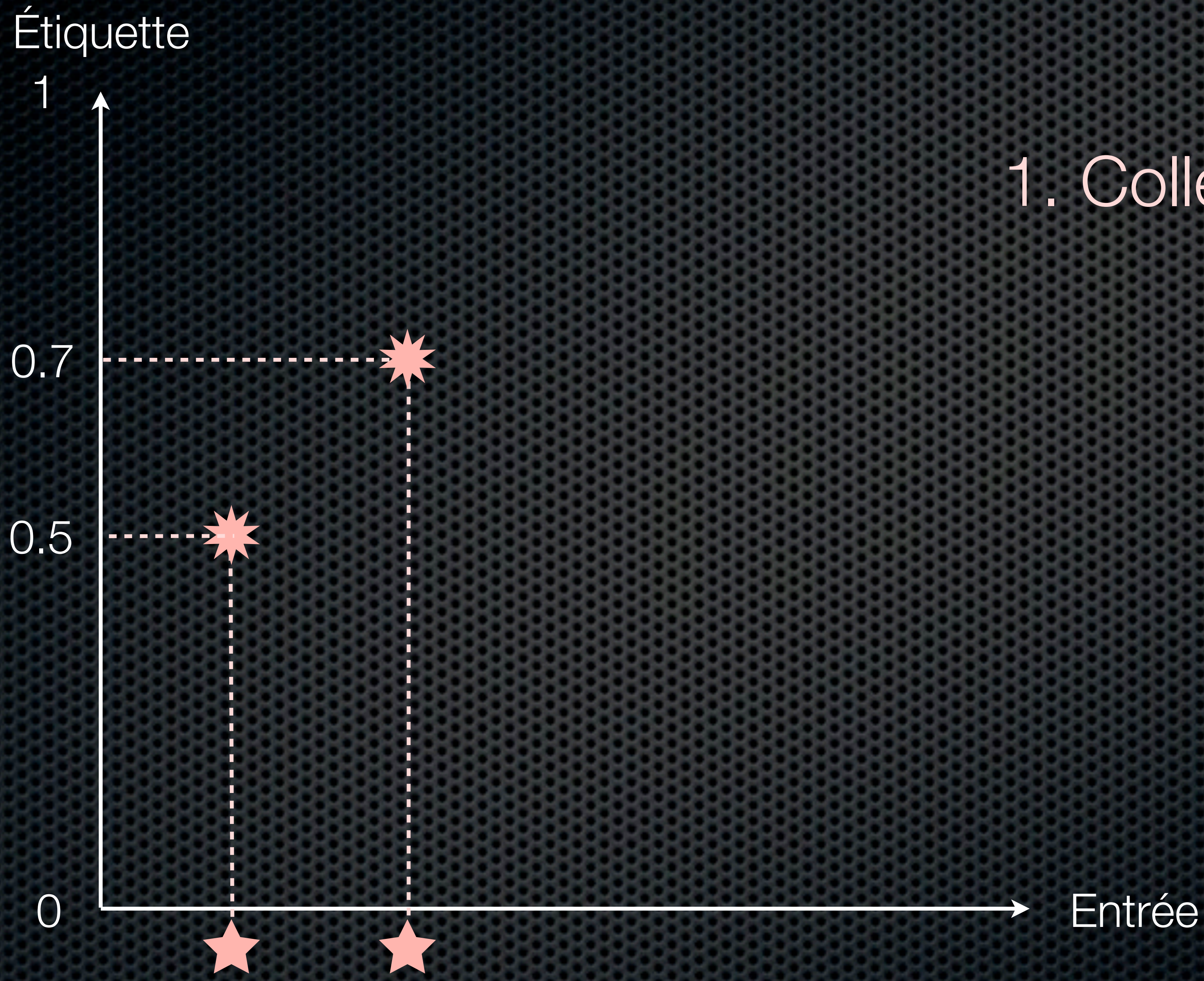
1. Collecter des données

Ex: régression (1D)



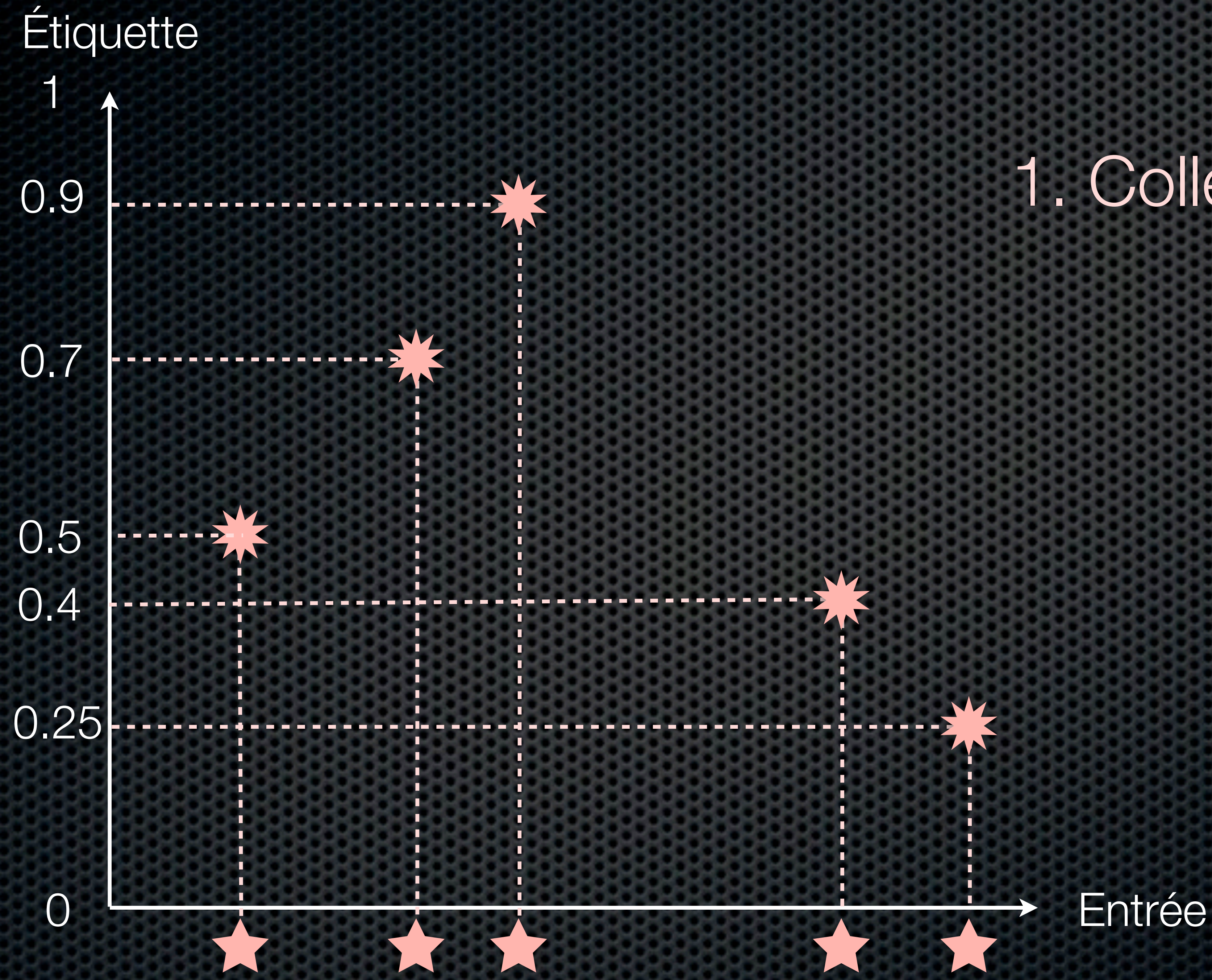
1. Collecter des données

Ex: régression (1D)



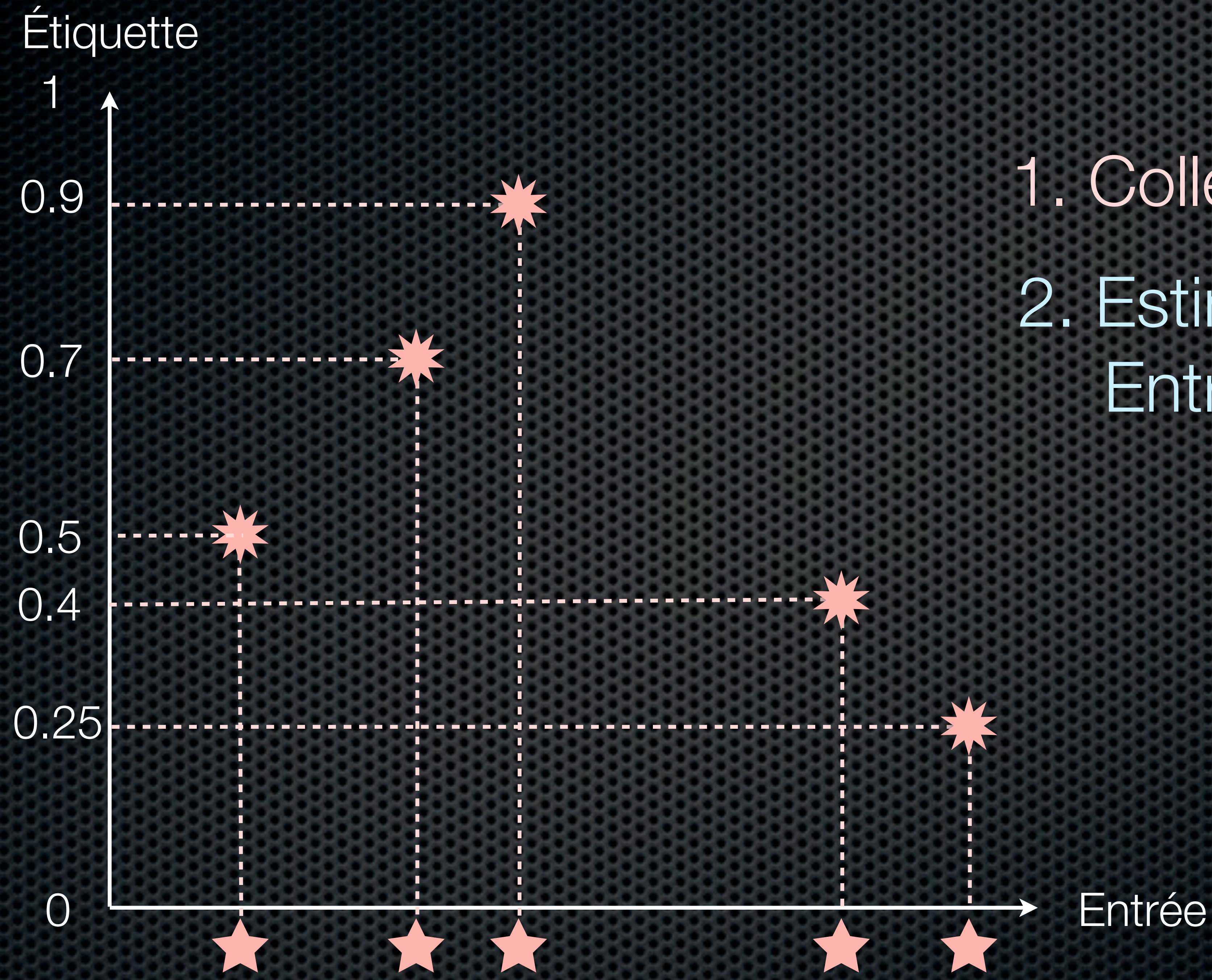
1. Collecter des données

Ex: régression (1D)



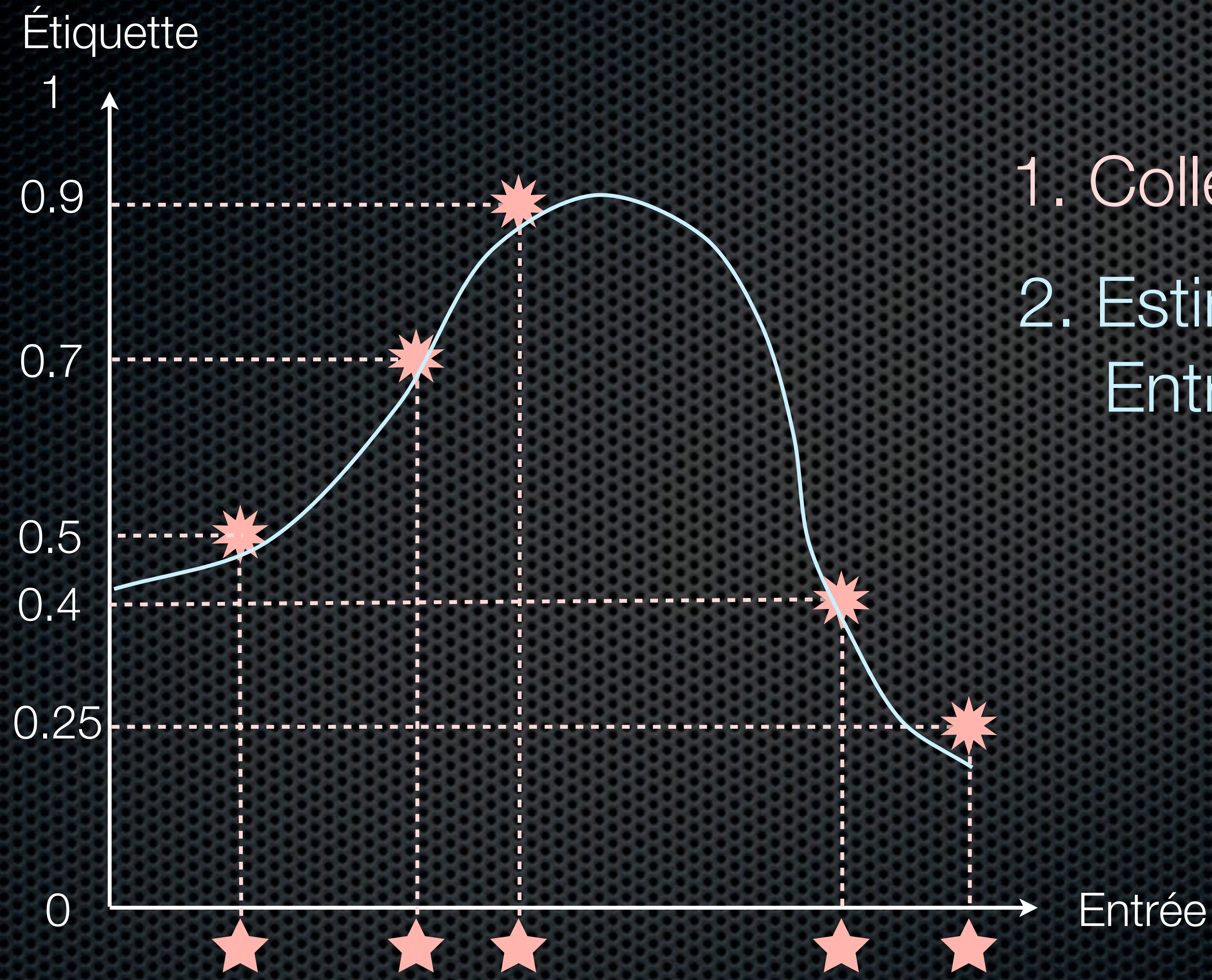
1. Collecter des données

Ex: régression (1D)



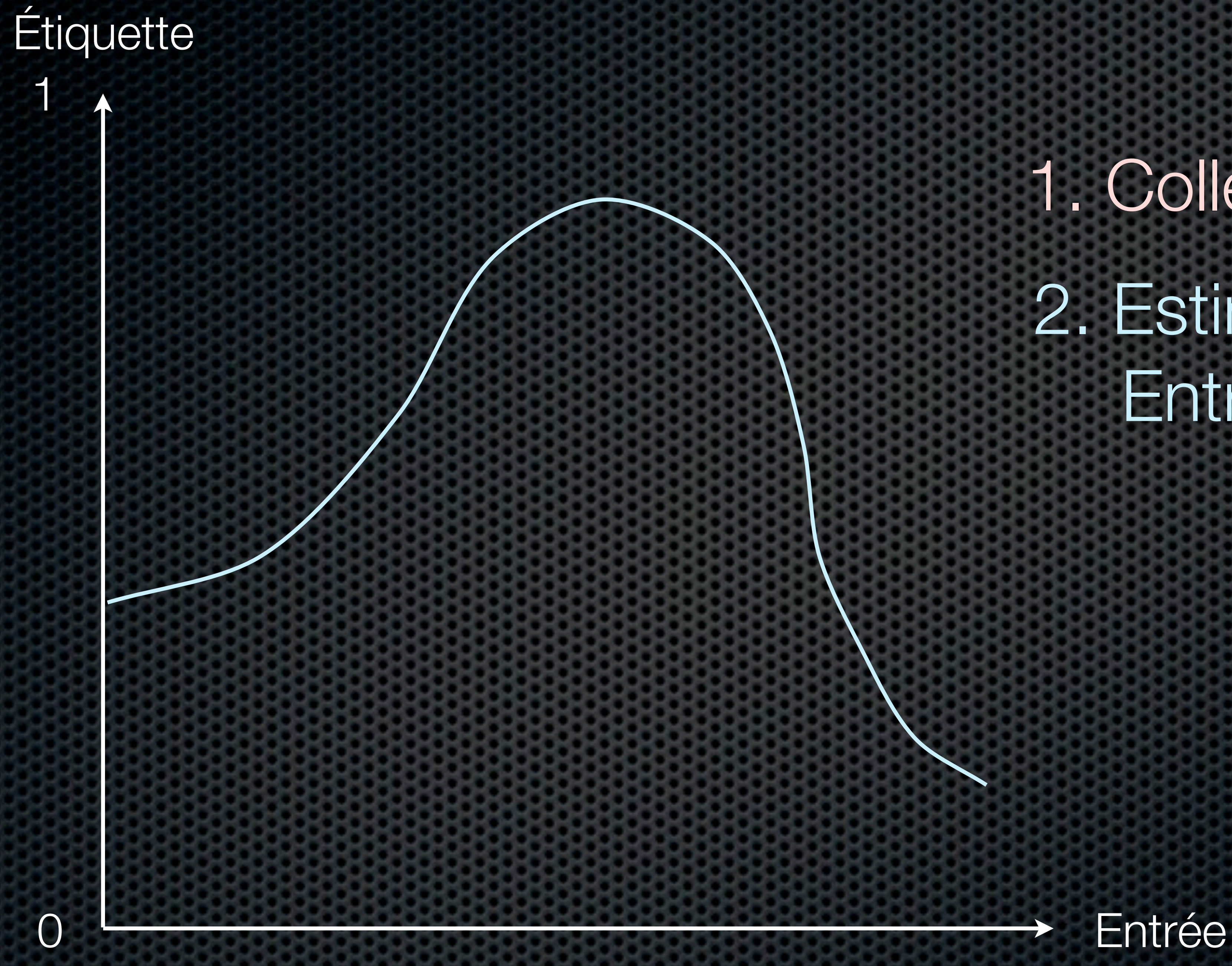
1. Collecter des données
2. Estimer la fonction
   Entrée $\longmapsto$ Étiquette

Ex: régression (1D)



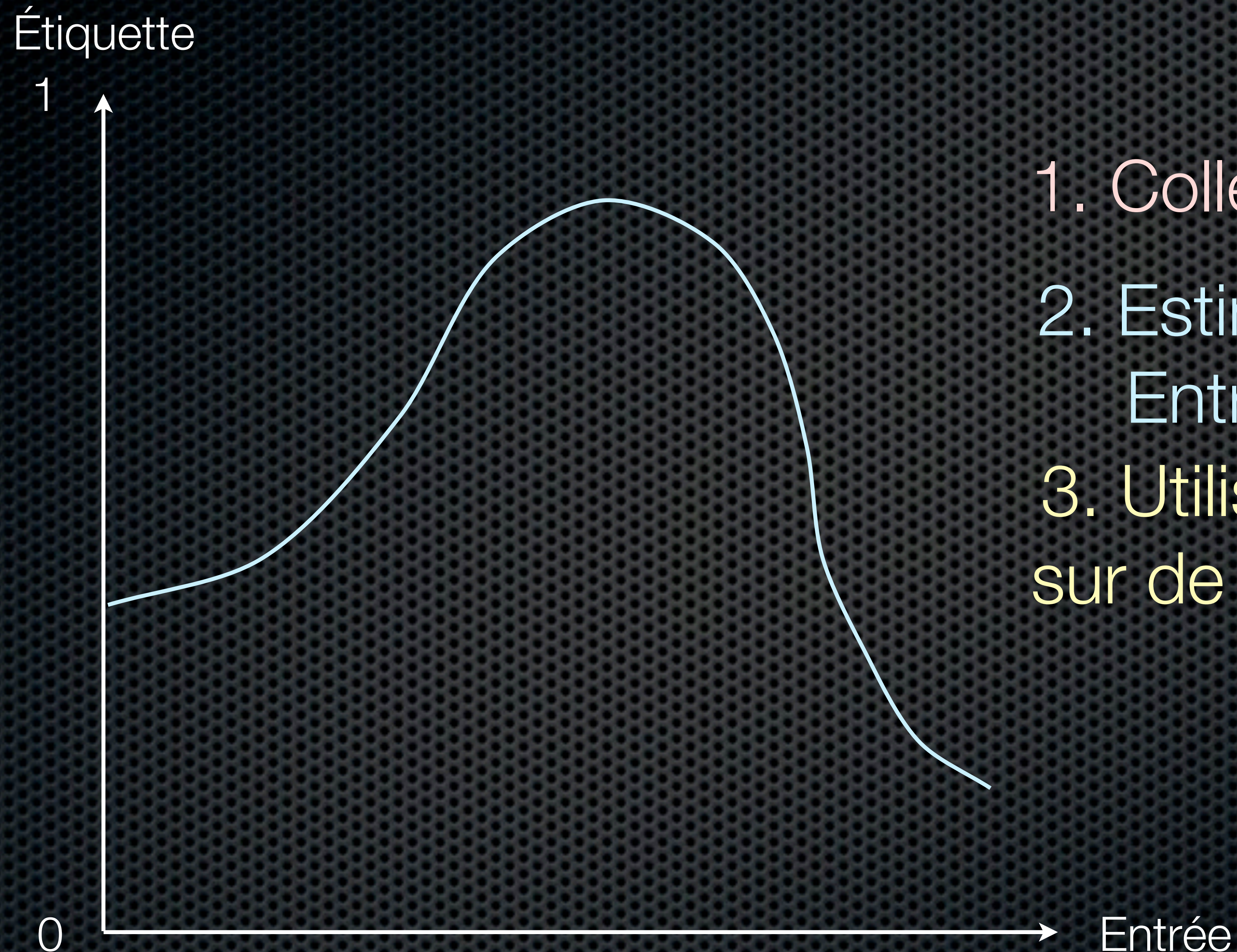
1. Collecter des données
2. Estimer la fonction
   Entrée $\longmapsto$ Étiquette

Ex: régression (1D)



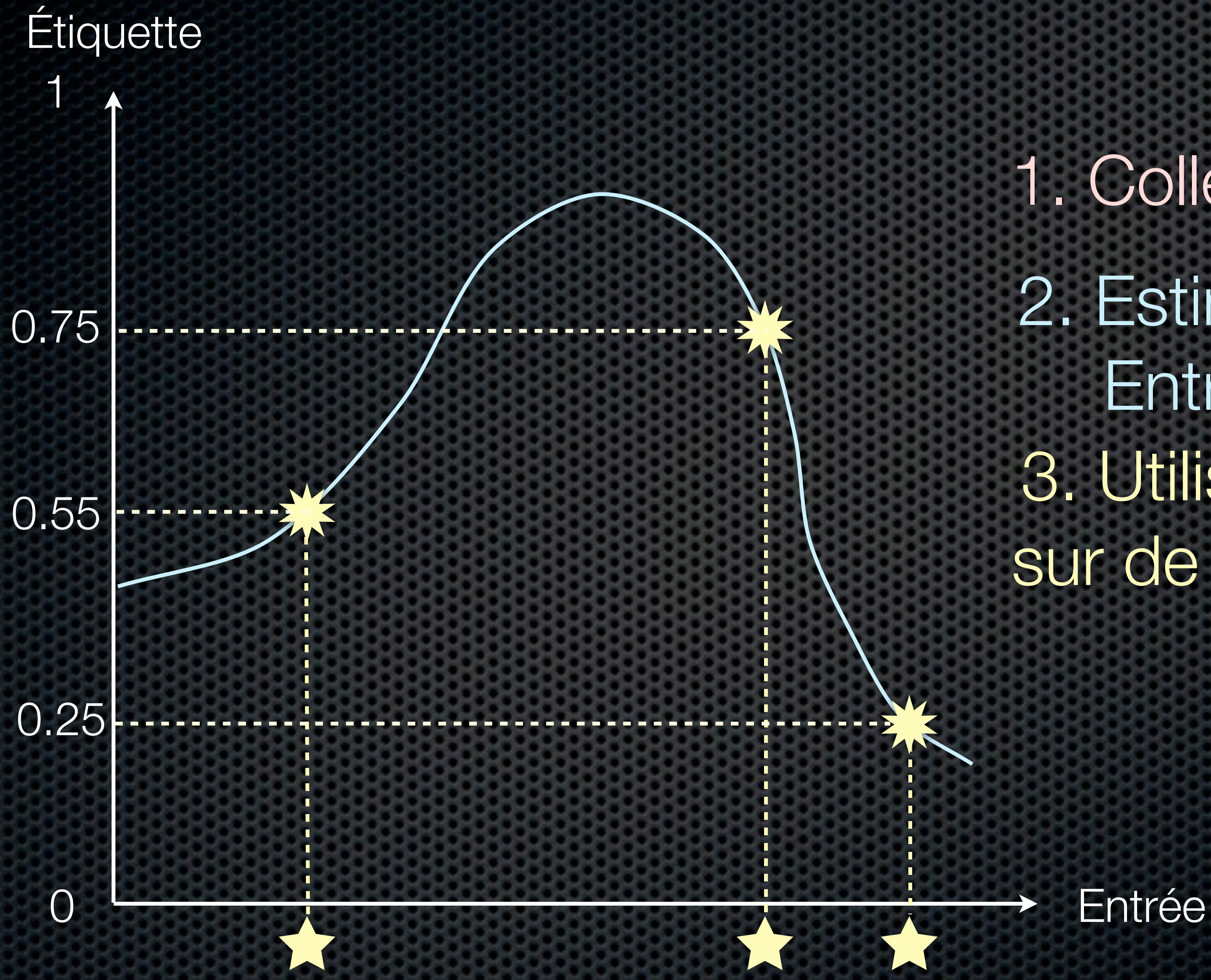
1. Collecter des données
2. Estimer la fonction
   Entrée $\longmapsto$ Étiquette

Ex: régression (1D)



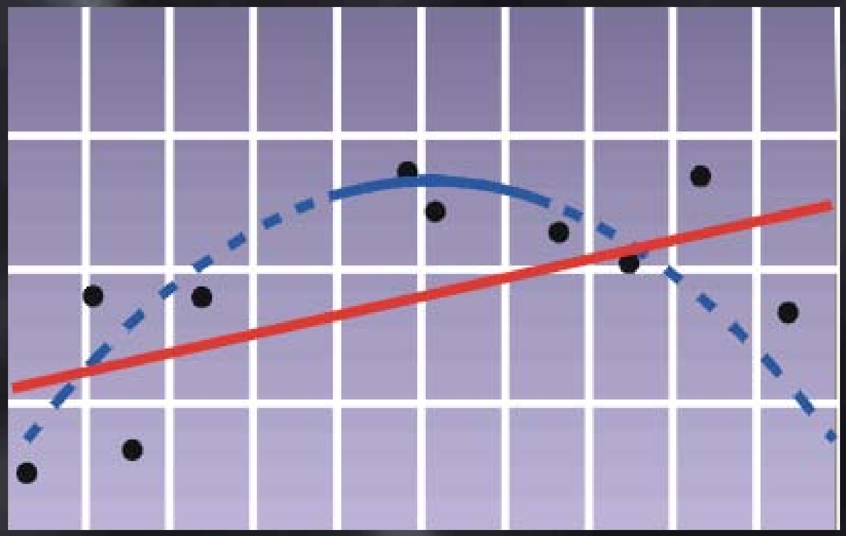
1. Collecter des données
2. Estimer la fonction
   Entrée ⟼ Étiquette
3. Utiliser cette fonction
sur de **nouvelles** données

Ex: régression (1D)

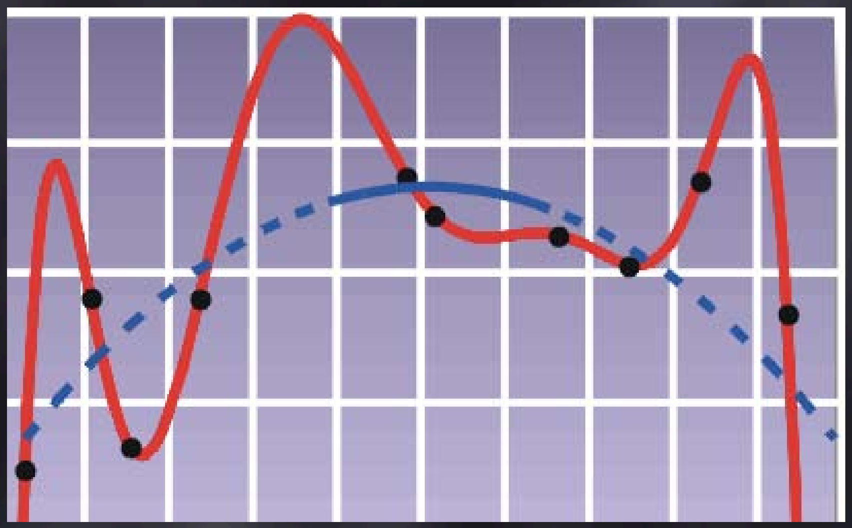


1. Collecter des données
2. Estimer la fonction
   Entrée ⟼ Étiquette
3. Utiliser cette fonction
   sur de **nouvelles** données

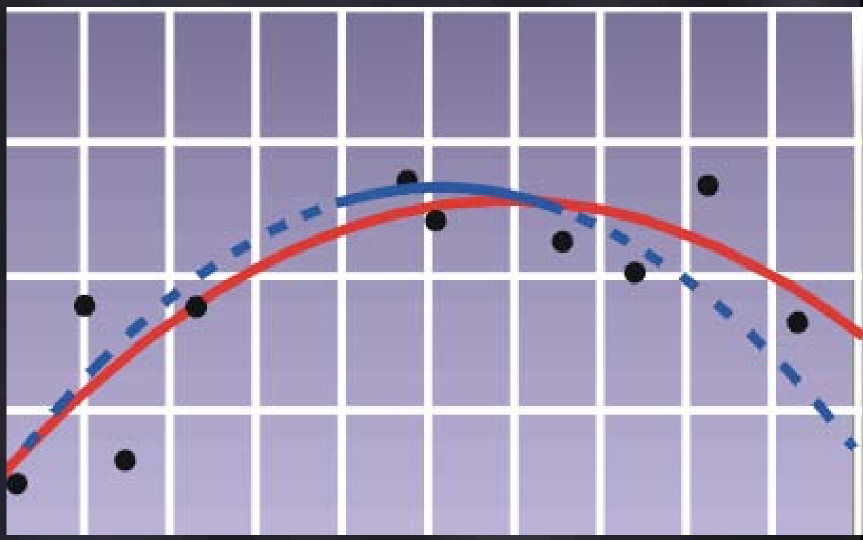
Contrôle de capacité et généralisation



capacité trop faible
⇨sous-apprentissage



capacité trop élevée
⇨sur-apprentissage

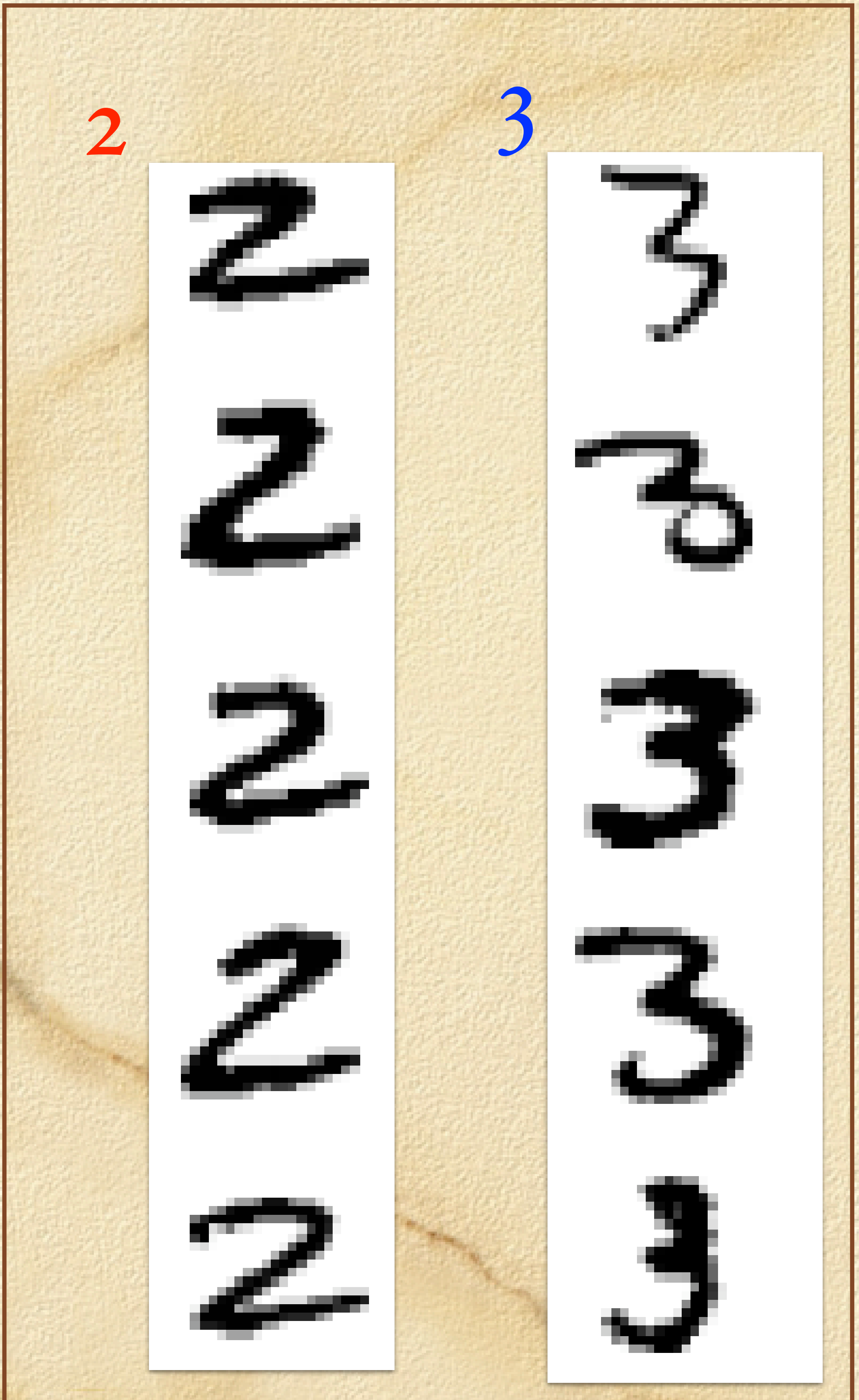


capacité optimale
⇨bonne généralisation

la performance sur l'ensemble d'entraînement n'est **pas** un bon estimé de la généralisation!

Données d'apprentissage:

Ensemble de **n exemples**

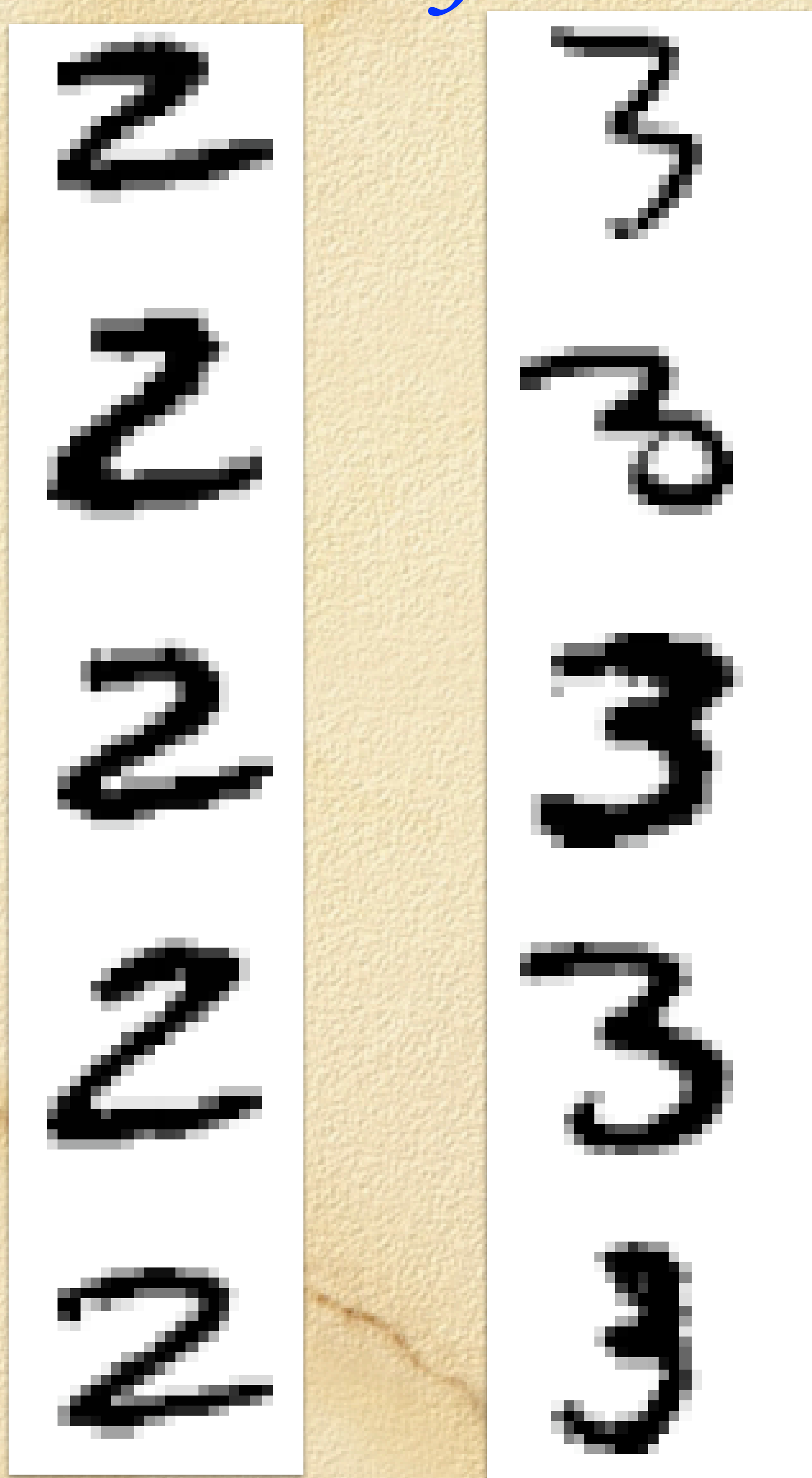


Ex: reconnaissance d'écriture: classification de chiffres manuscrits

Données d'apprentissage:

Ensemble de **n exemples**

2 3



Ex: reconnaissance d'écriture: classification de chiffres manuscrits

Point de test x

Données d'apprentissage:

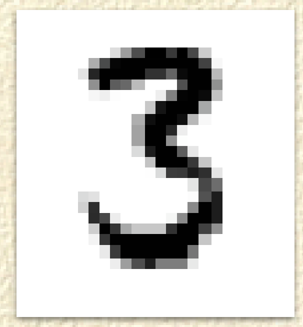
Ensemble de **n exemples**

2

3

Ex: reconnaissance d'écriture: classification de chiffres manuscrits

Point de test x



Données d'apprentissage:

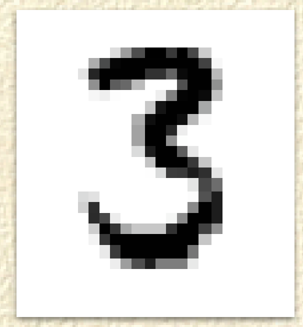
Ensemble de **n exemples**

2

3

Ex: reconnaissance d'écriture: <u>classification</u> de chiffres manuscrits

Point de test x



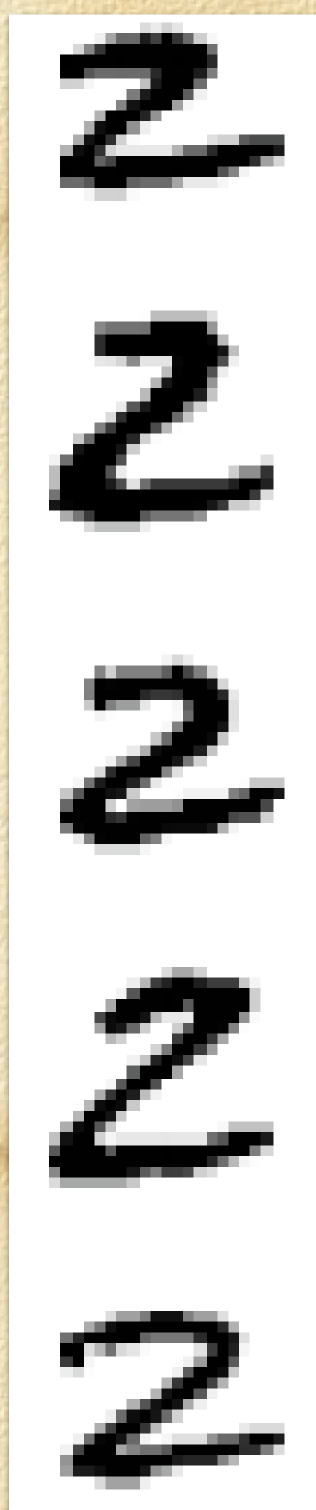
2 ou 3 ?

Ex: reconnaissance d'écriture: classification de chiffres manuscrits

Données d'apprentissage:

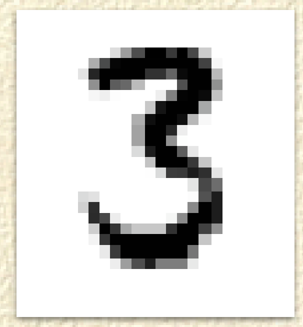
Ensemble de **n exemples**

2



3

Point de test x



2 ou 3 ?

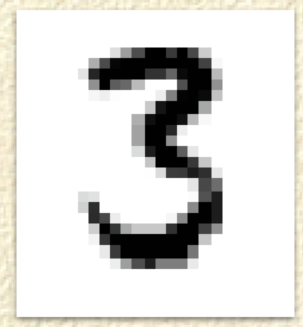
Apprendre n'est pas simplement mémoriser...

Données d'apprentissage:
Ensemble de **n exemples**

2 3


Ex: reconnaissance d'écriture: classification de chiffres manuscrits

Point de test x



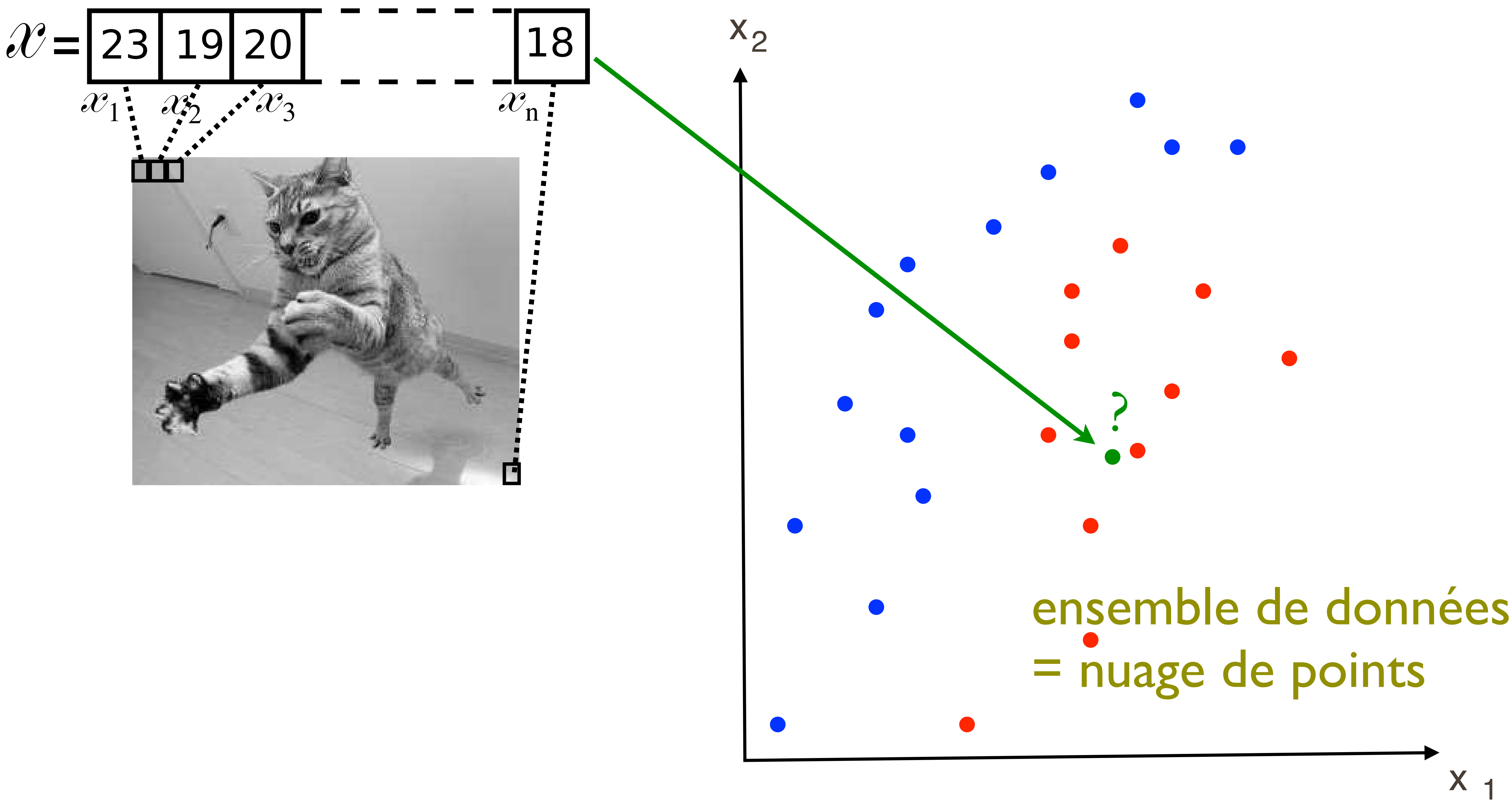
2 ou 3 ?

Apprendre n'est pas simplement mémoriser...

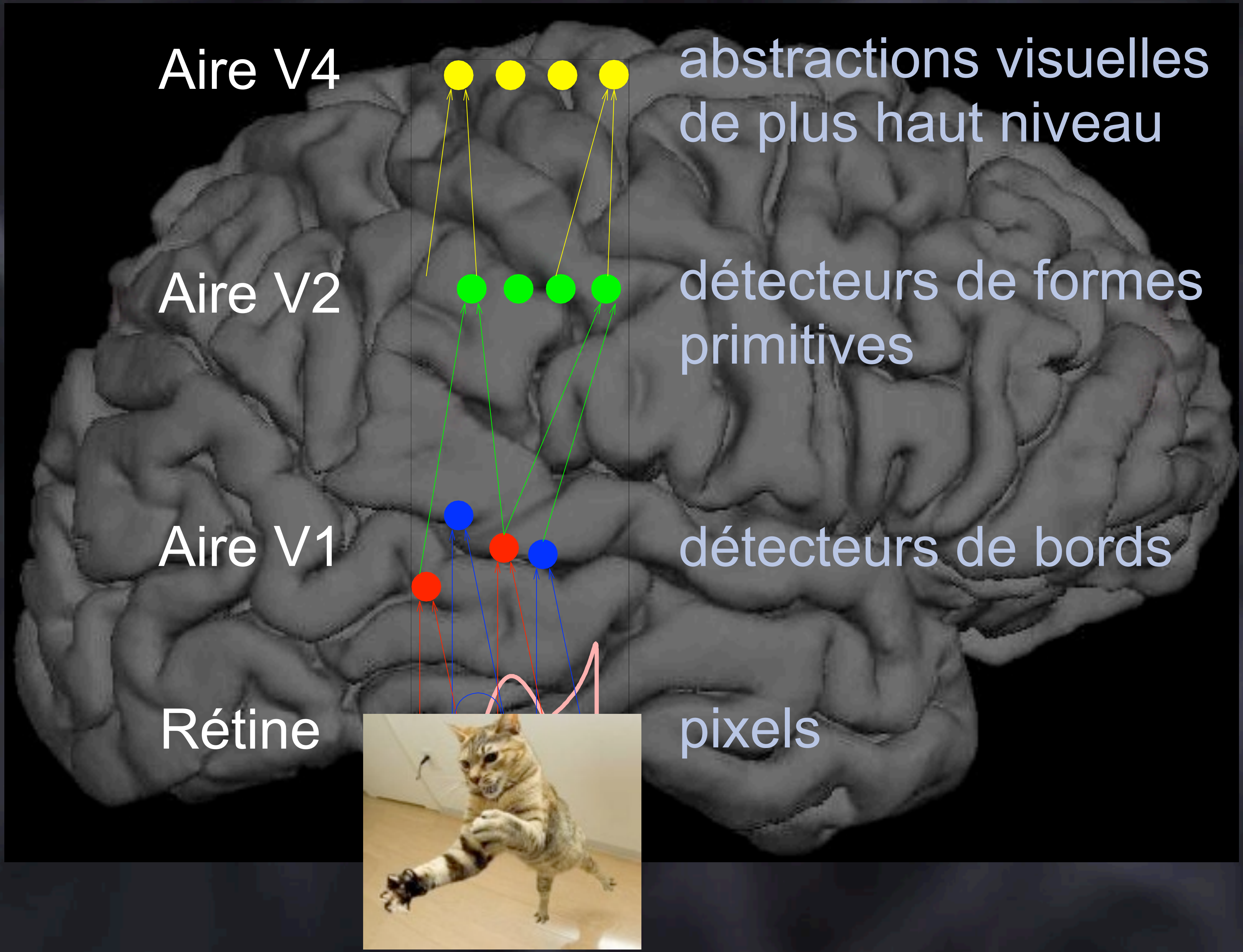
C'est être capable de généraliser!

Représentation des données

une image = un point dans un espace de haute dimension

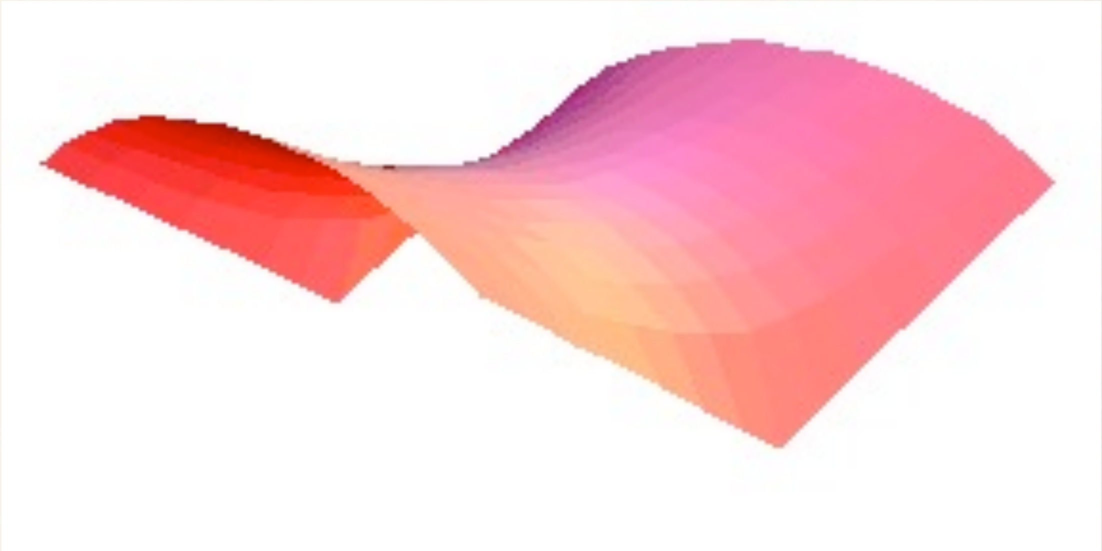


Niveau de représentation



Part I

Manifold modeling



high-dimensional data

The curse of dimensionality

There are 10^{96329} possible 200x200 RGB images.

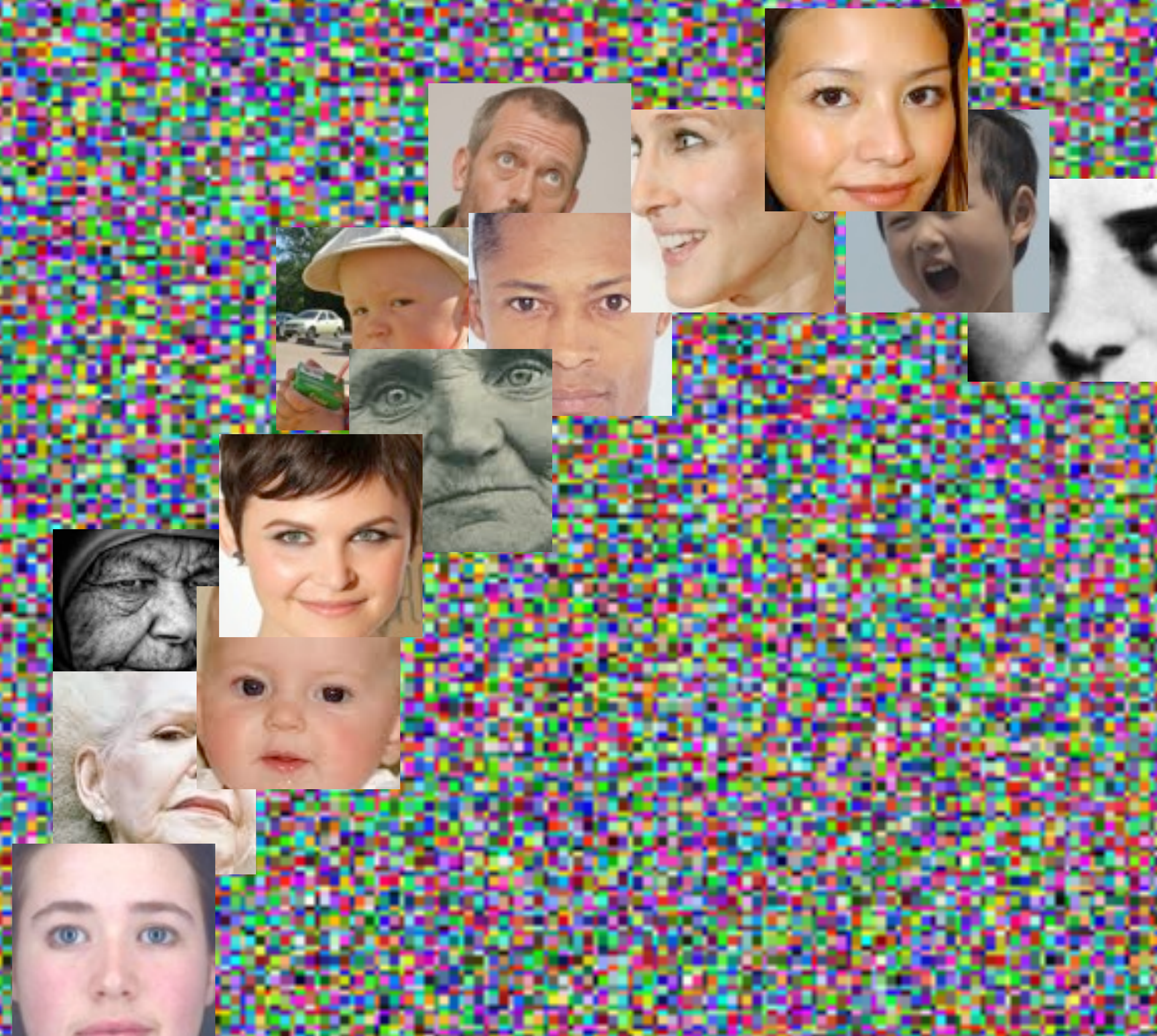
are we doomed?

The manifold hypothesis

Data density contentrates near a lower dimensional manifold

The manifold hypothesis

Data density contentrates near a lower dimensional manifold



The manifold hypothesis

Data density contentrates near a lower dimensional manifold

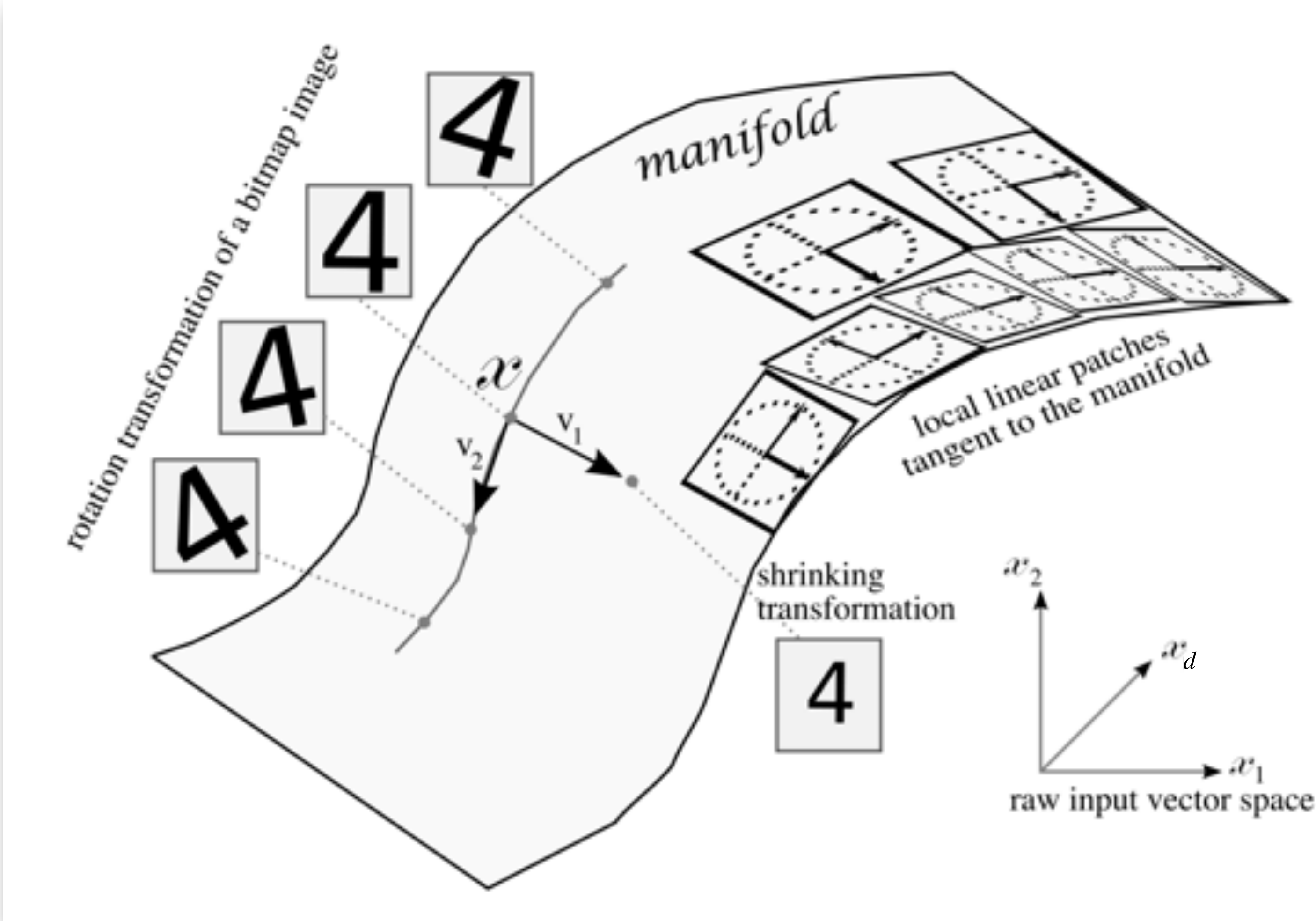
Can shift the curse from high d to $d_M \ll d$

Manifold follows naturally from continuous underlying factors (≈ intrinsic manifold coordinates)

Ex: pose parameters of a face

Image borrowed from University of Dayton Vision Lab website.

Ex: rotation, size of digits (+ line thickness, ...)

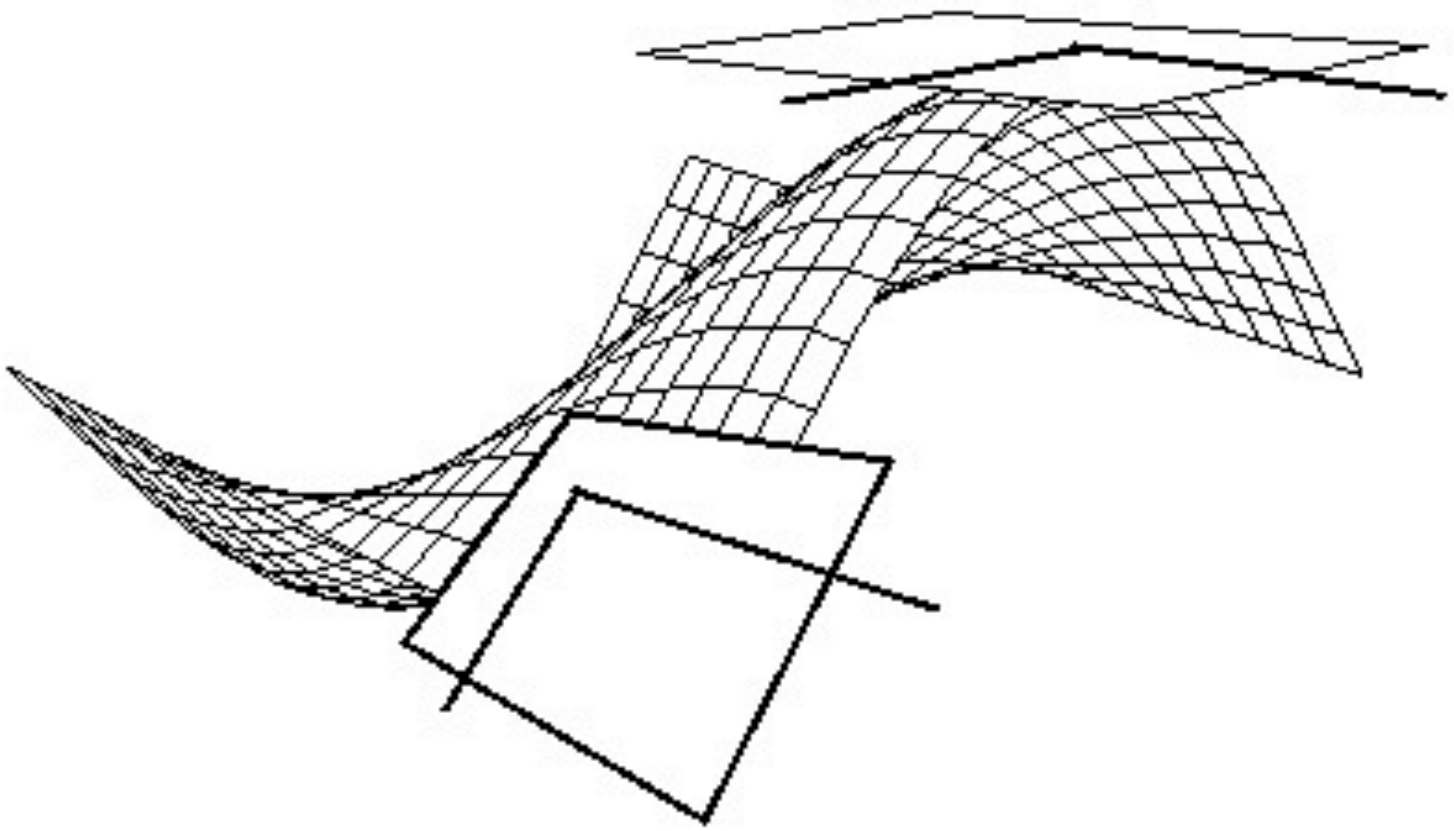


Such continuous factors are (part of) a meaningful represetation!

Modeling local tangent spaces

A non-linear manifold

- Can be represented by **patchwork** of **tangent spaces**
- Yields local linear coordinate systems (chart -> atlas)



Manifold Parzen windows
(Vincent and Bengio, NIPS 2003)

Manifold Parzen windows

(Vincent and Bengio, NIPS 2003)

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(x; x_i, C_i)$$

Classical Parzen Windows

density estimator

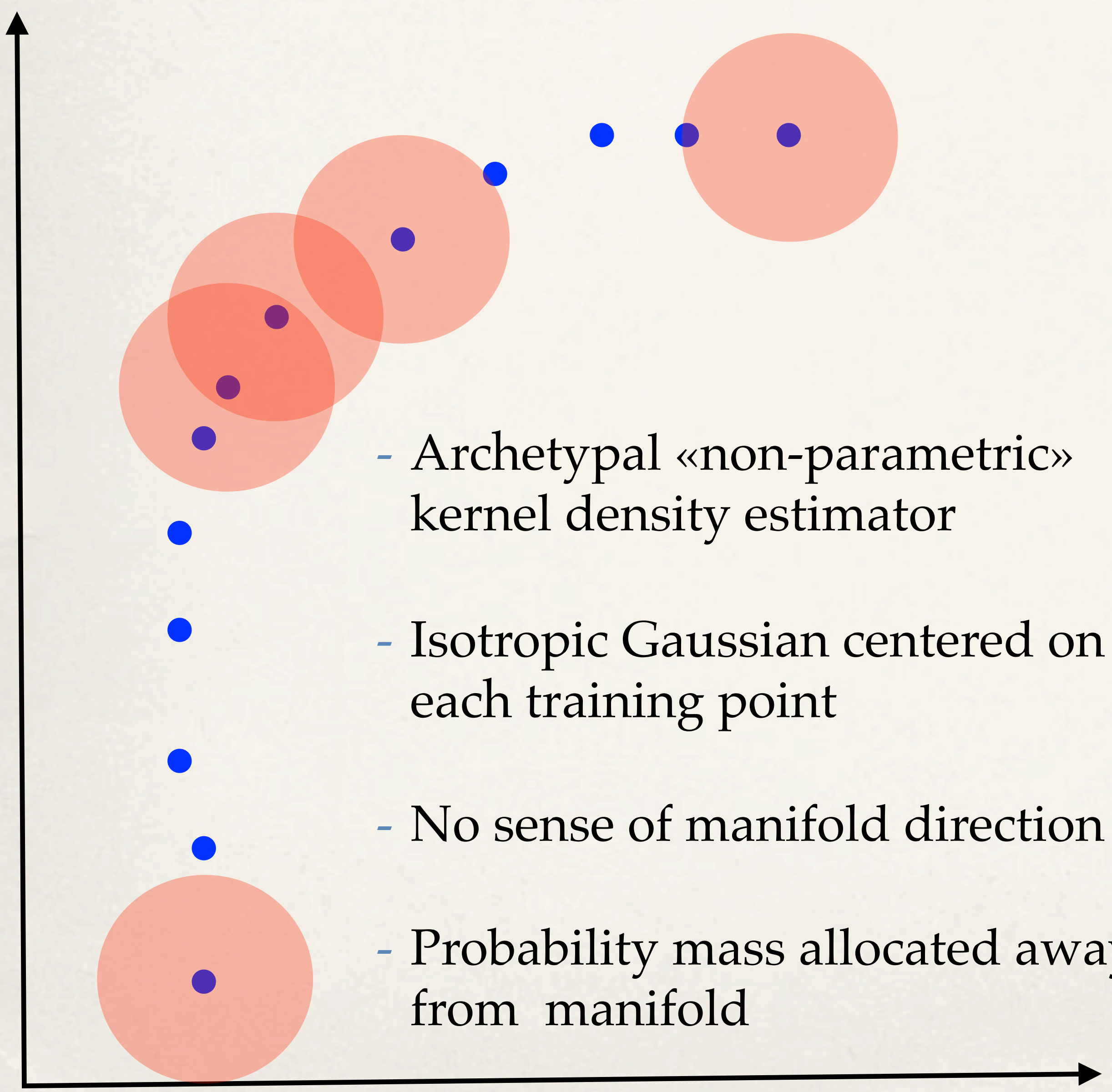
Manifold Parzen windows

(Vincent and Bengio, NIPS 2003)

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(x; x_i, C_i)$$

Classical Parzen Windows

density estimator



- Archetypal «non-parametric» kernel density estimator
- Isotropic Gaussian centered on each training point
- No sense of manifold direction
- Probability mass allocated away from manifold

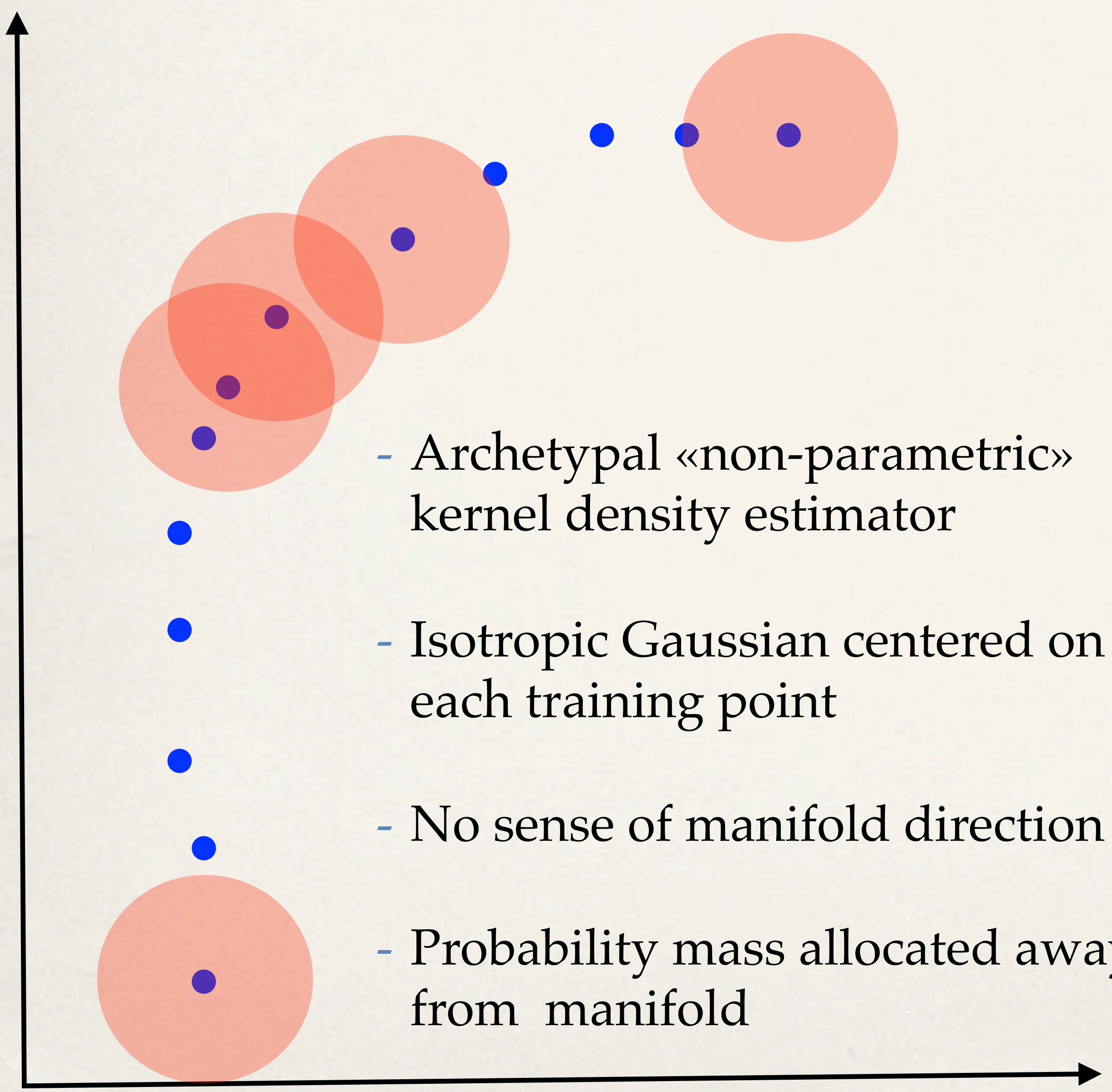
Manifold Parzen windows

(Vincent and Bengio, NIPS 2003)

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(x; x_i, C_i)$$

Classical Parzen Windows

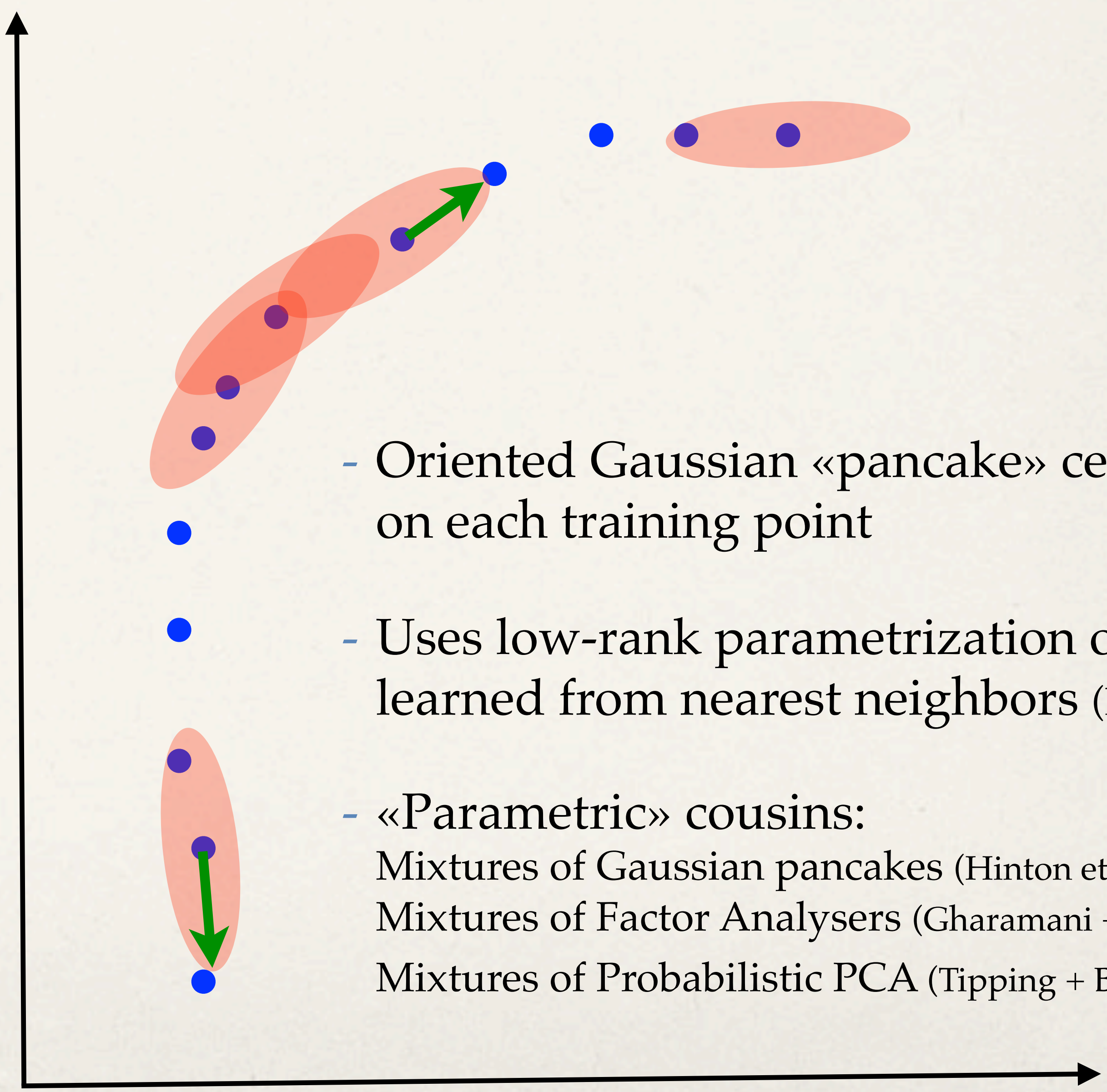
density estimator



- Archetypal «non-parametric» kernel density estimator
- Isotropic Gaussian centered on each training point
- No sense of manifold direction
- Probability mass allocated away from manifold

Manifold Parzen Windows

density estimator



- Oriented Gaussian «pancake» centered on each training point
- Uses low-rank parametrization of C_i, learned from nearest neighbors (local PCA)
- «Parametric» cousins:
  Mixtures of Gaussian pancakes (Hinton et al. 95)
  Mixtures of Factor Analysers (Gharamani + Hinton 96)
  Mixtures of Probabilistic PCA (Tipping + Bishop 99)

Non-local manifold Parzen windows

(Bengio, Larochelle, Vincent, NIPS 2006)

Isotropic Parzen:

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x; x_i, \underbrace{\sigma^2 I}_{\text{isotropic}})$$

Non-local manifold Parzen windows

(Bengio, Larochelle, Vincent, NIPS 2006)

Isotropic Parzen:

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x; x_i, \underbrace{\sigma^2 I}_{\text{isotropic}})$$

Manifold Parzen:

(Vincent and Bengio, NIPS 2003)

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x; x_i, \underbrace{C_i})$$

d_M high variance directions from PCA on k nearest neighbors

Non-local manifold Parzen windows

(Bengio, Larochelle, Vincent, NIPS 2006)

Isotropic Parzen:

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x; x_i, \underbrace{\sigma^2 I}_{\text{isotropic}})$$

Manifold Parzen:
(Vincent and Bengio, NIPS 2003)

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x; x_i, \underbrace{C_i})$$

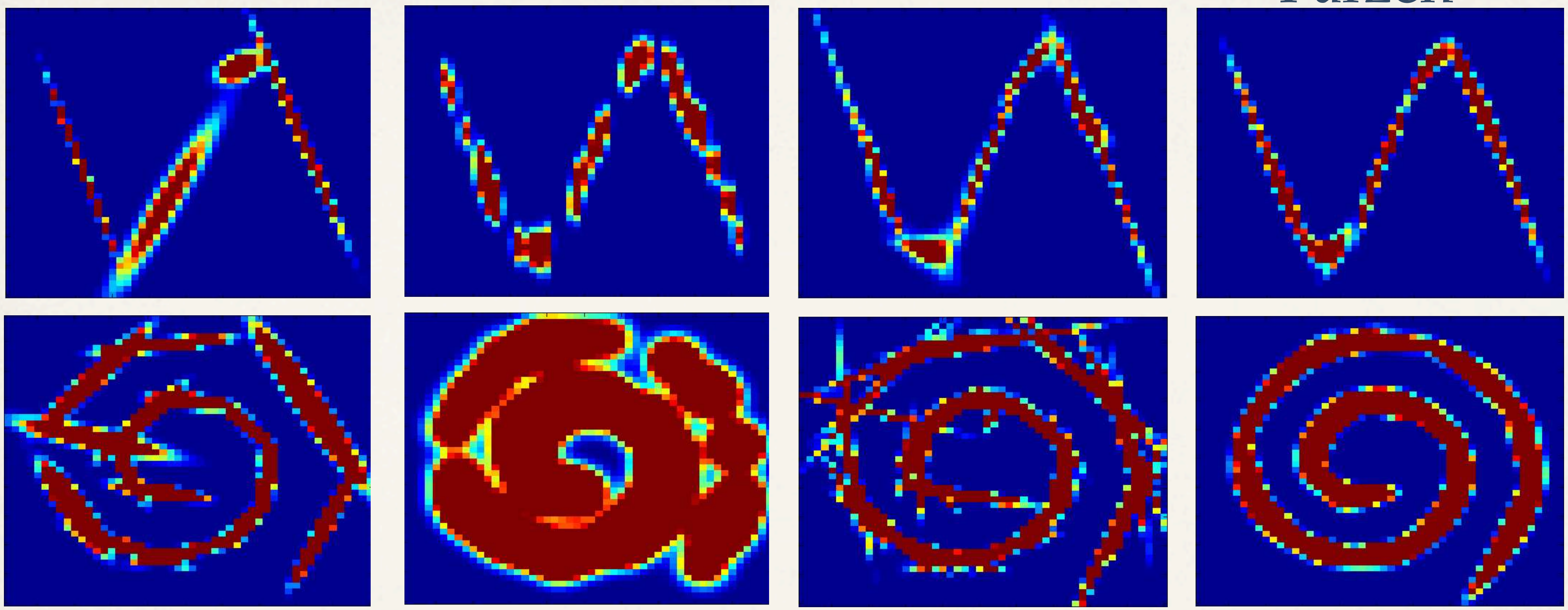
d_M high variance directions from PCA on k nearest neighbors

Non-local manifold Parzen:
(Bengio, Larochelle, Vincent, NIPS 2006)

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x; \underbrace{\mu(x_i), C(x_i)})$$

d_M high variance directions output by neural network
trained to maximize likelihood of k nearest neighbors

Mixture of k Gaussians | Parzen Windows | Manifold Parzen | Non-local Manifold Parzen



Use in Bayes classifier on USPS

Algorithm	Valid.	Test	Hyper-Parameters
SVM	1.2%	4.68%	$C = 100, \sigma = 8$
Parzen Windows	1.8%	5.08%	$\sigma = 0.8$
Manifold Parzen	0.9%	4.08%	$d = 11, k = 11, \sigma_0^2 = 0.1$
Non-local MP	**0.6%**	**3.64% (-1.5218)**	$d = 7, k = 10, k_\mu = 10,$ $\sigma_0^2 = 0.05, n_{hid} = 70$
Non-local MP*	**0.6%**	**3.54% (-1.9771)**	$d = 7, k = 10, k_\mu = 4,$ $\sigma_0^2 = 0.05, n_{hid} = 30$

What do most «manifold learning» approaches have in common ?

Purely non-parametric:

- Manifold Parzen, LLE, Isomap, Laplacian eigenmaps, t-SNE, ...

Learn parametrized function:

- Parametric t-SNE, semi-supervised embedding, **non-local** manifold Parzen, ...

?

Neighborhood-based training!

- Most explicitly use neighborhoods.
- Training with k-nearest neighbors, or pairs of points.
- Typically Euclidean neighbors
- But in **high** ***d***, your nearest Euclidean neighbor can be **very** different from you...

Neighborhood-based training!

Unreliable!

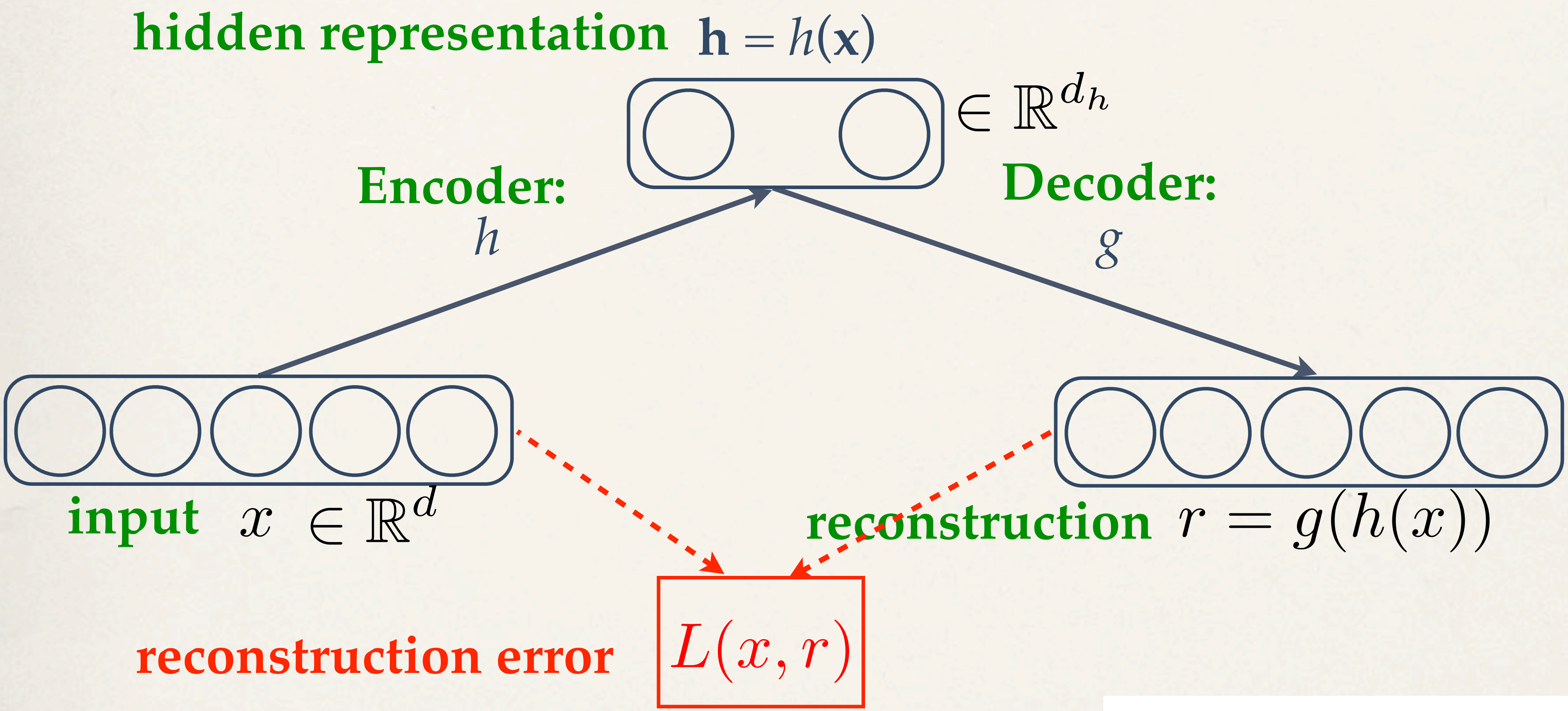
- Most explicitly use neighborhoods.
- Training with k nearest neighbors, or pairs of points.
- Typically Euclidean neighbors
- But in **high *d***, your nearest Euclidean neighbor can be **very** different from you...

PART II

On Auto-Encoders

their regularization,
and their link with manifolds.

Auto-Encoders (AE)

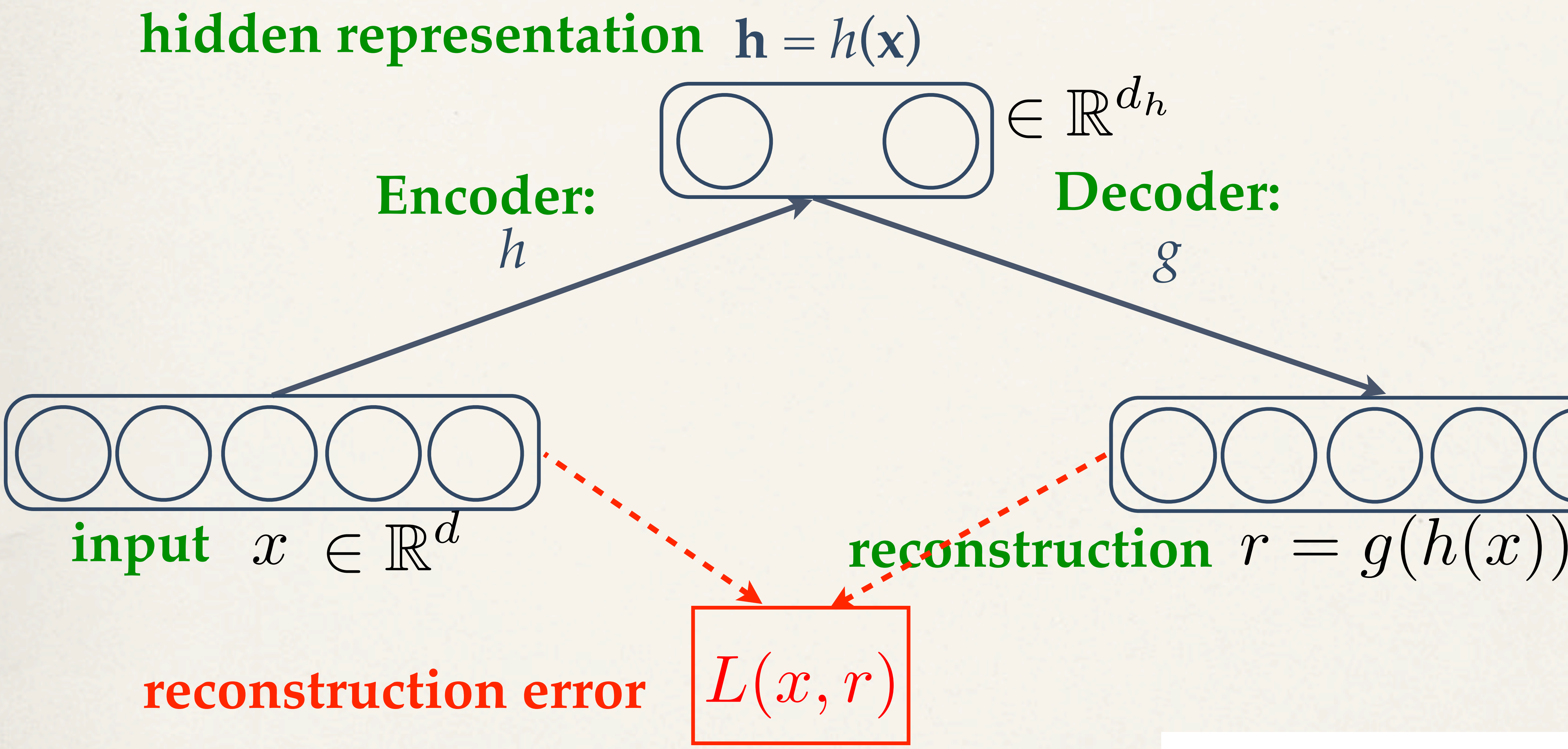


Minimize

$$\mathcal{J}_{\text{AE}} = \sum_{x \in D} L(x, g(h(x)))$$

Auto-Encoders (AE)

Typical form

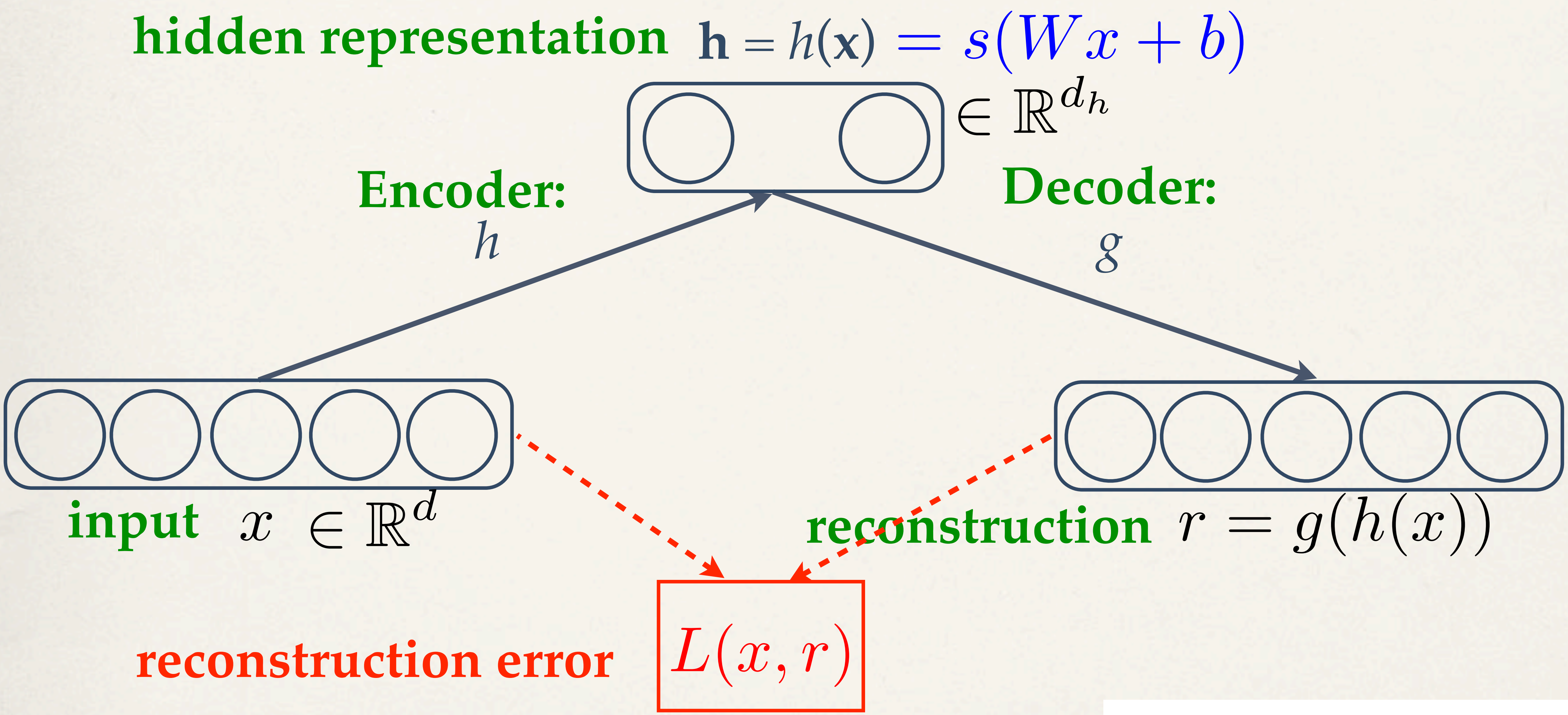


Minimize

$$\mathcal{J}_{\text{AE}} = \sum_{x \in D} L(x, g(h(x)))$$

Auto-Encoders (AE)

Typical form

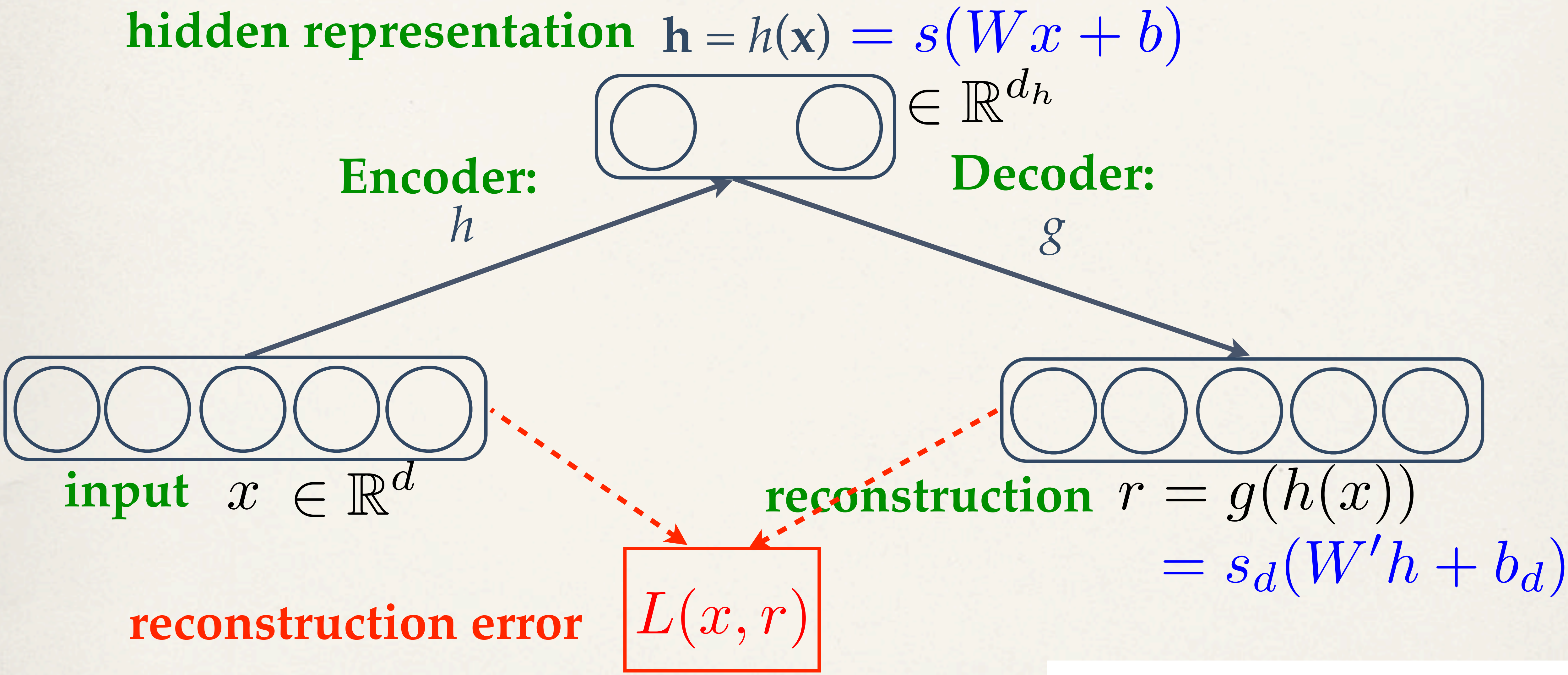


Minimize

$$\mathcal{J}_{\mathrm{AE}} = \sum_{x \in D} L(x, g(h(x)))$$

Auto-Encoders (AE)

Typical form



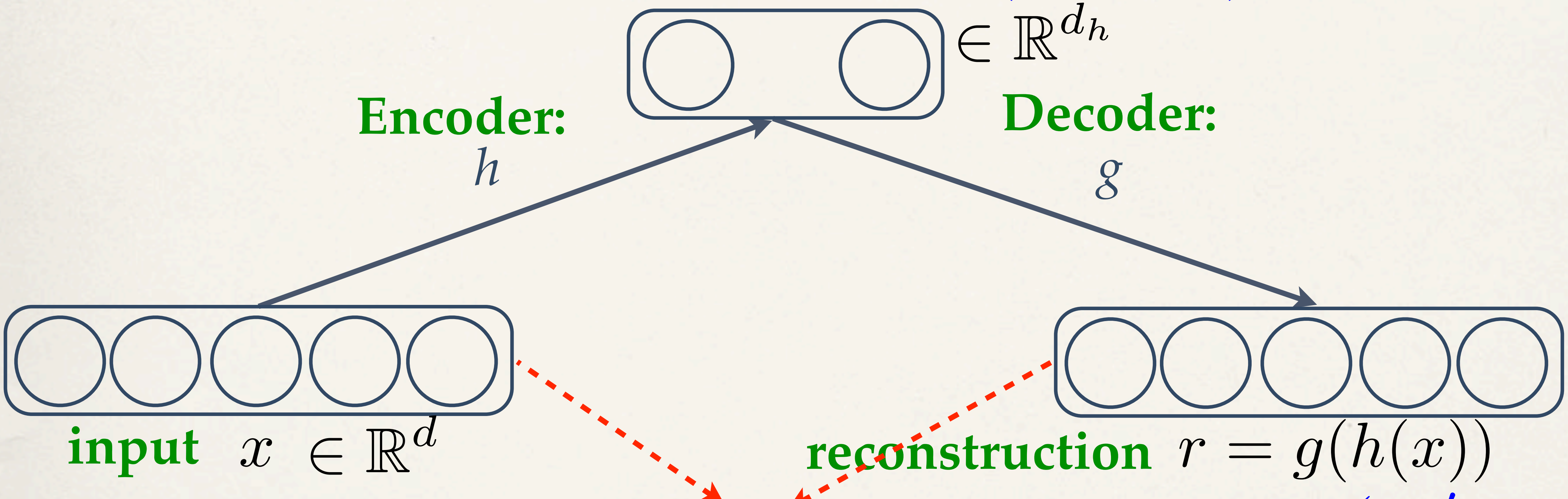
Minimize

$$\mathcal{J}_{\text{AE}} = \sum_{x \in D} L(x, g(h(x)))$$

Auto-Encoders (AE)

Typical form

hidden representation $\mathbf{h} = h(\mathbf{x}) = s(Wx + b)$ $\in \mathbb{R}^{d_h}$



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

reconstruction $r = g(h(x))$
$= s_d(W'h + b_d)$

reconstruction error $\boxed{L(x, r)}$

squared error: $\|x - r\|^2$

or Bernoulli cross-entropy

Minimize

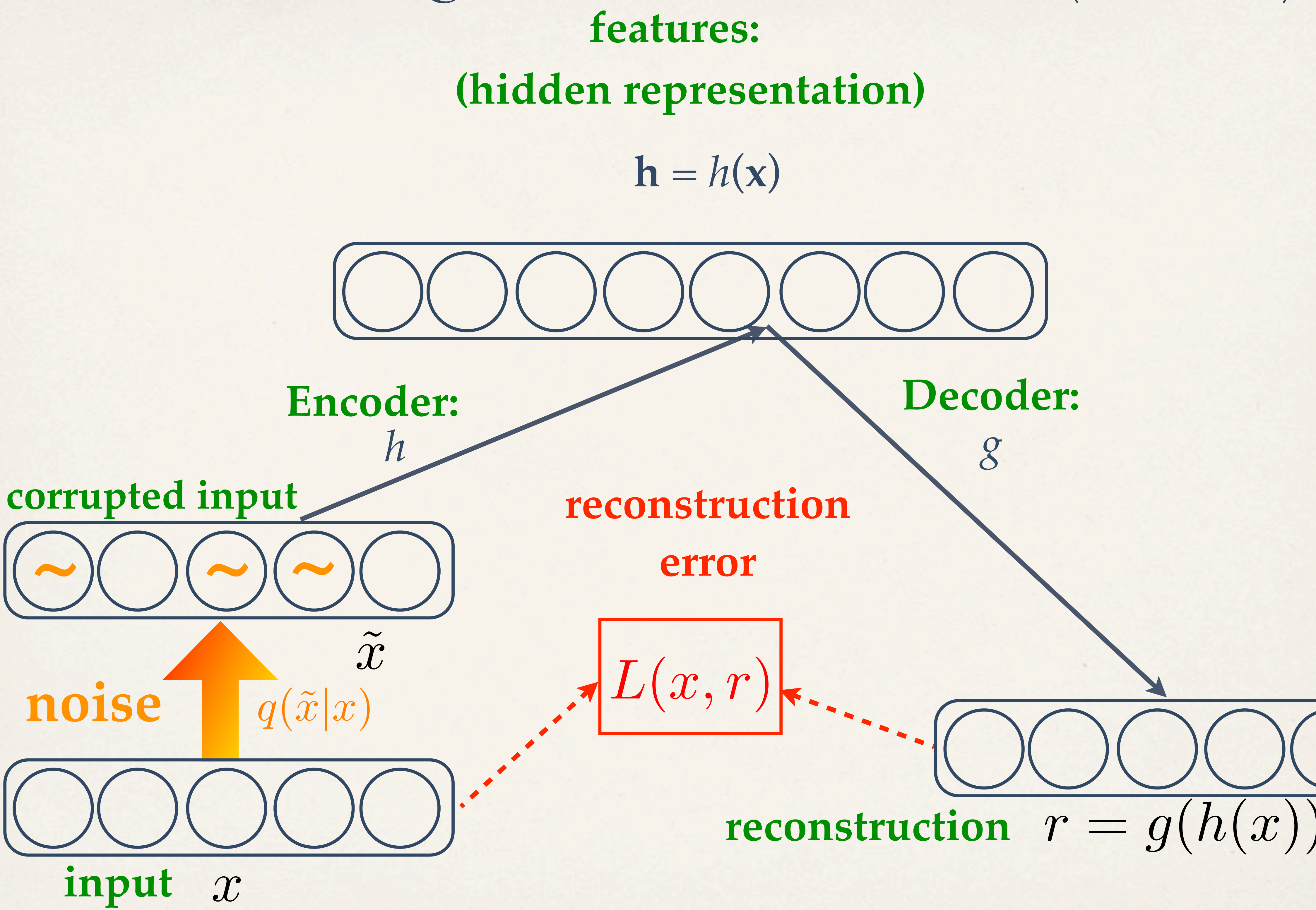
$$\mathcal{J}_{\text{AE}} = \sum_{x \in D} L(x, g(h(x)))$$

Regularized robust auto-encoders

Principle: reconstruction and representation should be **robust to input perturbations**

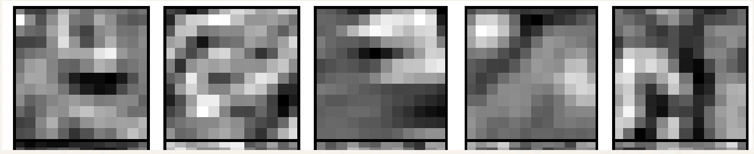
- **Denoising Auto-Encoder (DAE)**
  (Vincent, Larochelle, Bengio, Manzagol, ICML 2008)
  uses **stochastic input perturbations**
- **Contractive Auto-Encoder (CAE)**
  (Rifai, Vincent, Muller, Glorot, Bengio, ICML 2011)
  uses **analytic penalty** (penalizes input sensitivity)
- Both can learn over-complete representations ($d_h > d$)
- Related to training Gaussian RBM via *regularized score matching*

Denoising auto-encoder (DAE)

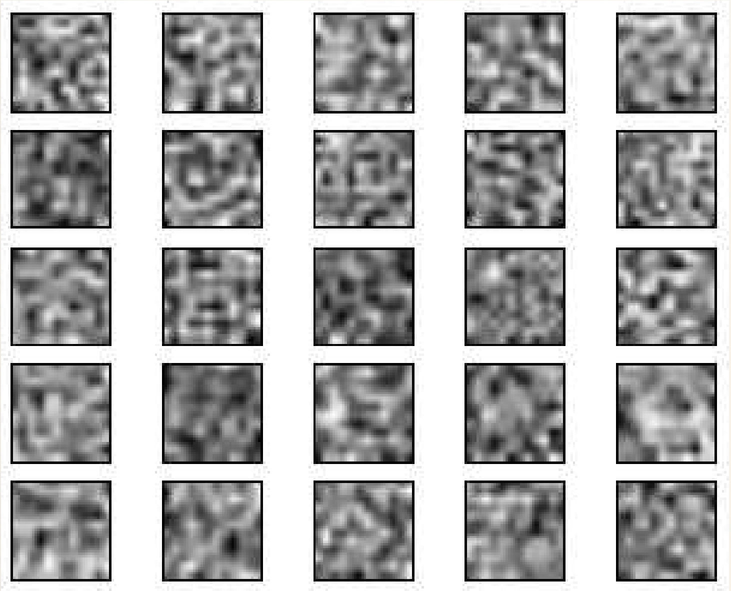


Learned filters

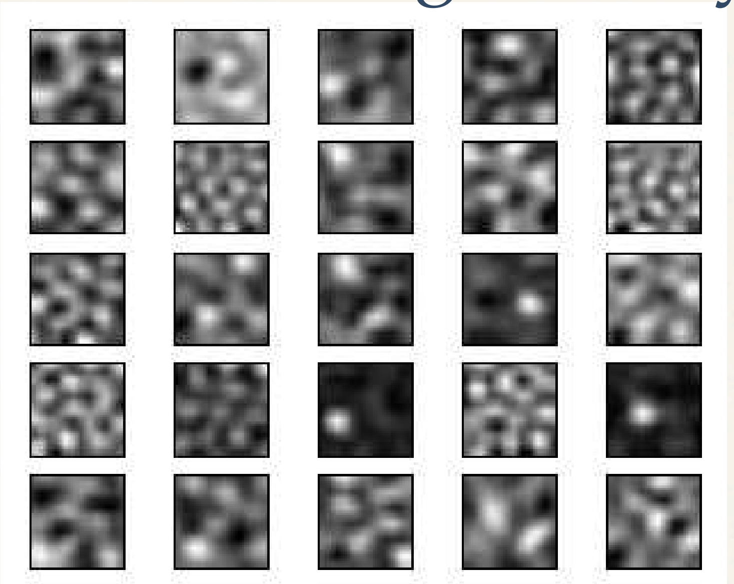
a) Natural image patches e.g.:



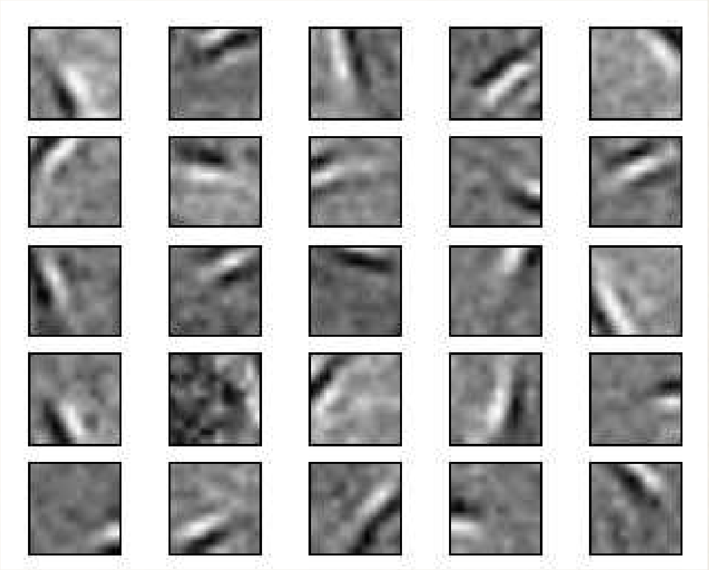
AE



AE with weight decay



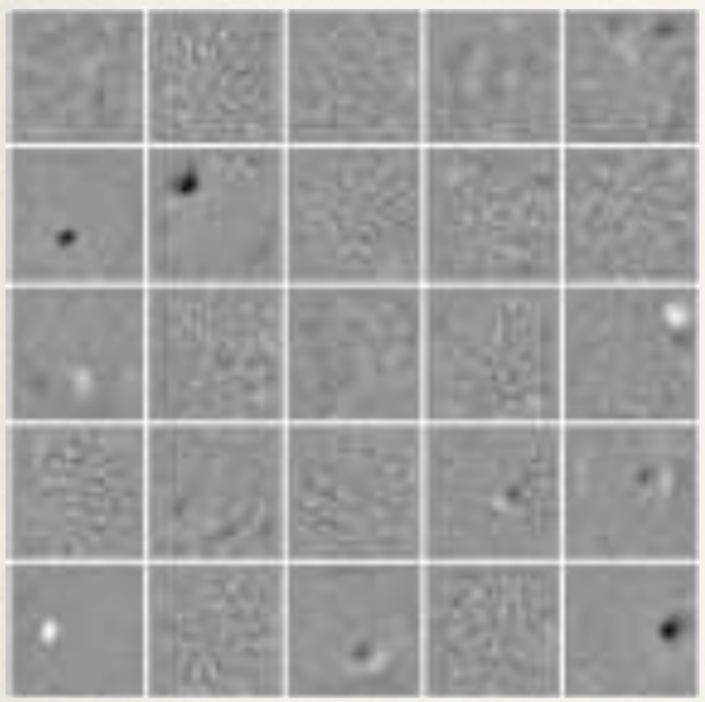
DAE



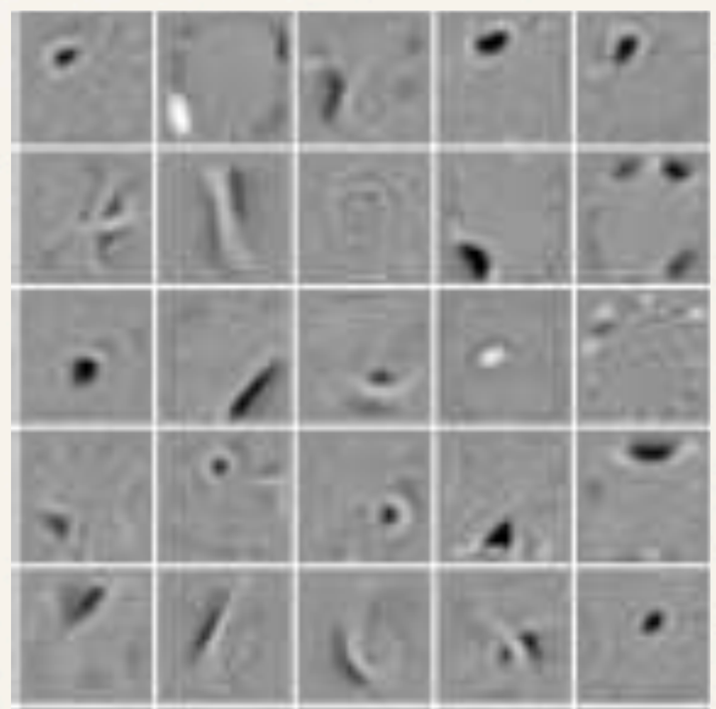
Learned filters

b) MNIST digits e.g.: 

AE



DAE



Increasing noise

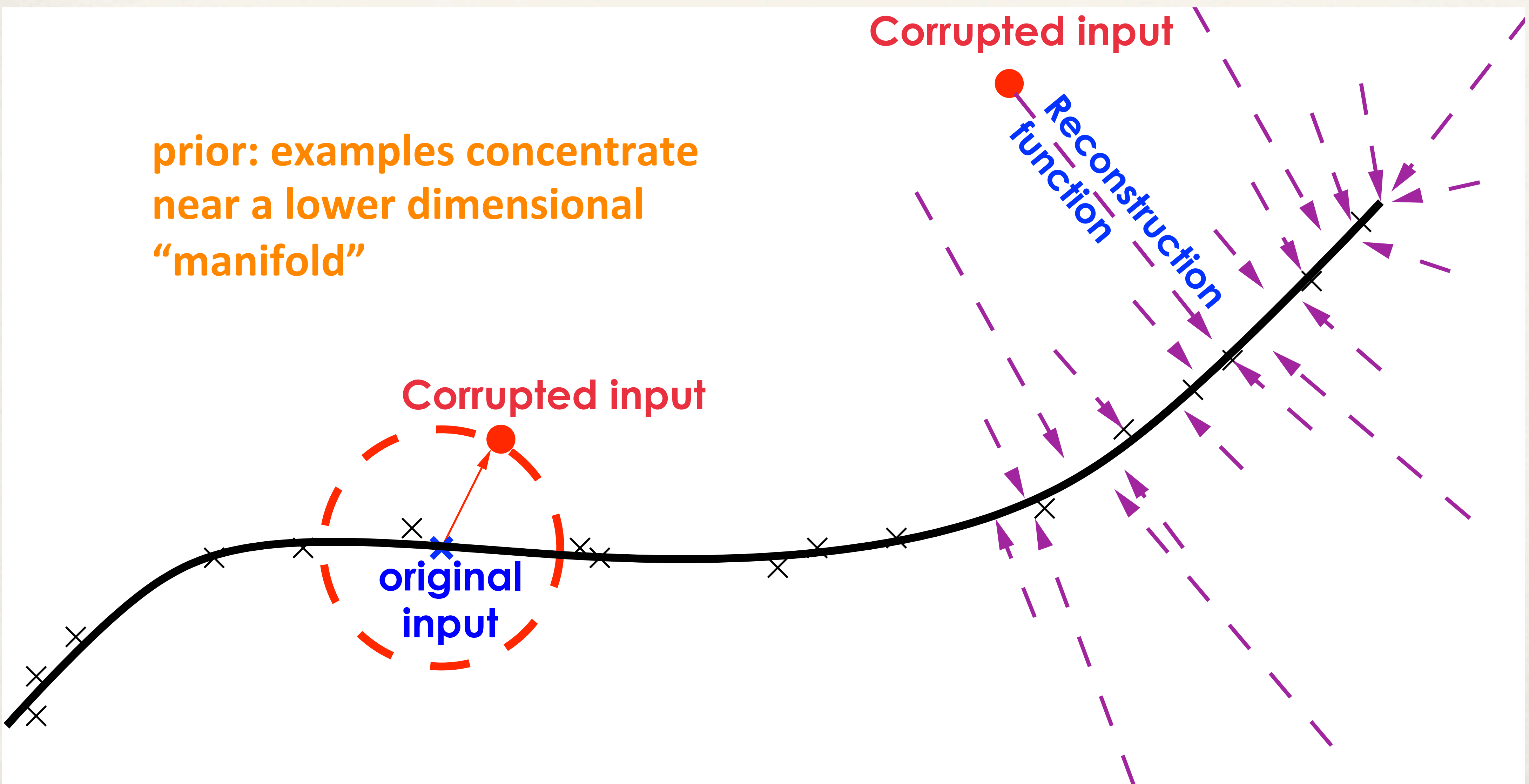
(d) Neuron A (0%, 10%, 20%, 50% corruption)

(e) Neuron B (0%, 10%, 20%, 50% corruption)

Denoising auto-encoder (DAE)

(Vincent, Larochelle, Bengio, Manzagol, ICML 2008)

- DAE learns to «project back» corrupted input onto manifold.
- Representation $h \approx$ location on the manifold



Contractive Auto-Encoder (CAE)

(Rifai, Vincent, Muller, Glorot, Bengio, ICML 2011)

- Minimize $$\mathcal{J}_{\text{CAE}} = \sum_{x \in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

Reconstruction Contraction

- For training examples, encourages both:
  - ➡ small reconstruction error
  - ➡ representation insensitive to small variations around example

With a sigmoid layer, penalty is easy and cheap to compute:

$$\frac{\partial h_j}{\partial x}(x) = h_j(x)(1 - h_j(x))W_j$$

CAE must capture manifold directions

$$\mathcal{J}_{\text{CAE}} = \sum_{x \in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

Reconstruction $\Leftarrow$ **tradeoff** $\Rightarrow$ **Contraction**

(warning: may require tied weights)

pressure to be insensitive to ***all*** directions

CAE must capture manifold directions

$$\mathcal{J}_{\text{CAE}} = \sum_{x \in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

Reconstruction

⇐ tradeoff ⇒

(warning: may require tied weights)

Contraction

pressure to be insensitive to ***all*** directions

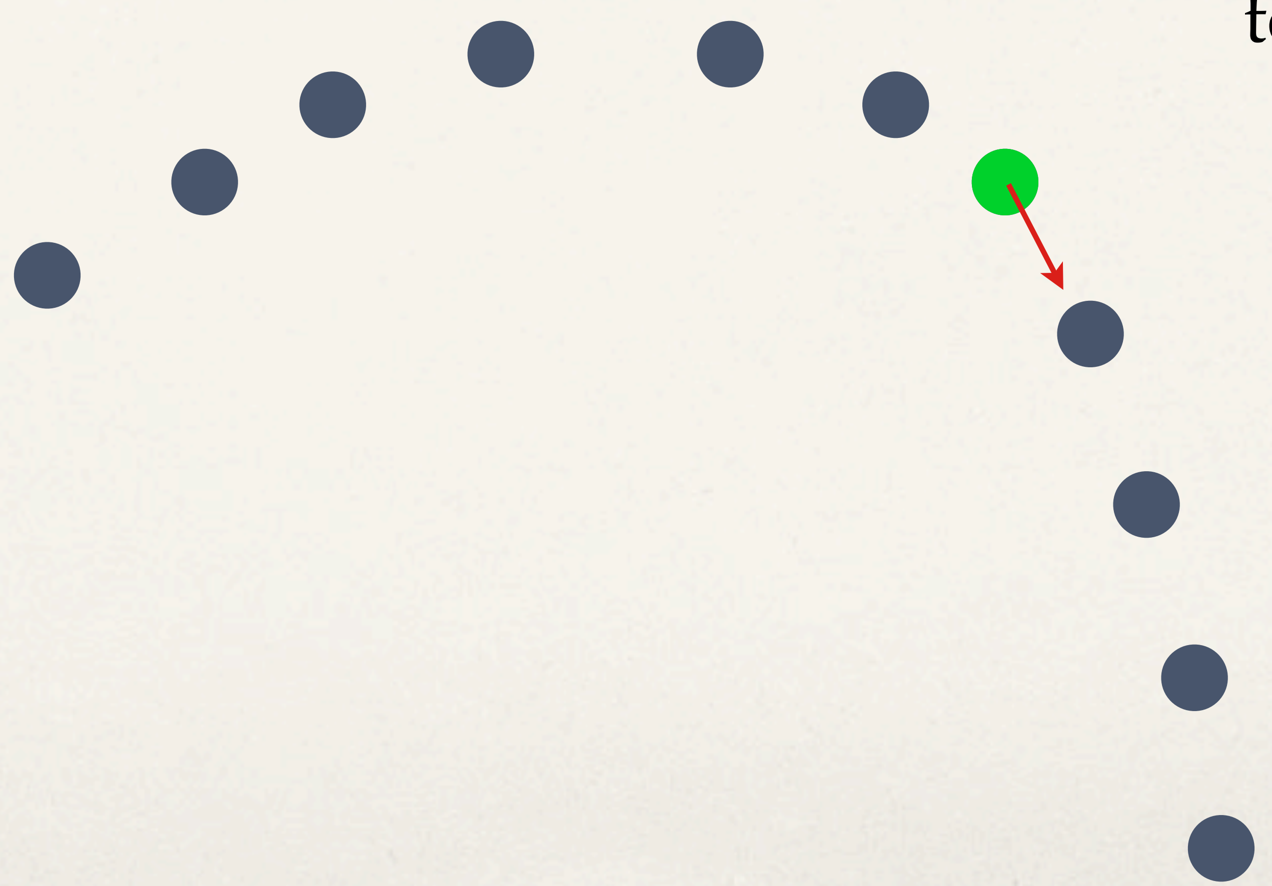
CAE must capture manifold directions

$$\mathcal{J}_{\text{CAE}} = \sum_{x\in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

Reconstruction $\Leftarrow$ **tradeoff** $\Rightarrow$ **Contraction**

(warning: may require tied weights)

pressure to be insensitive to ***all*** directions



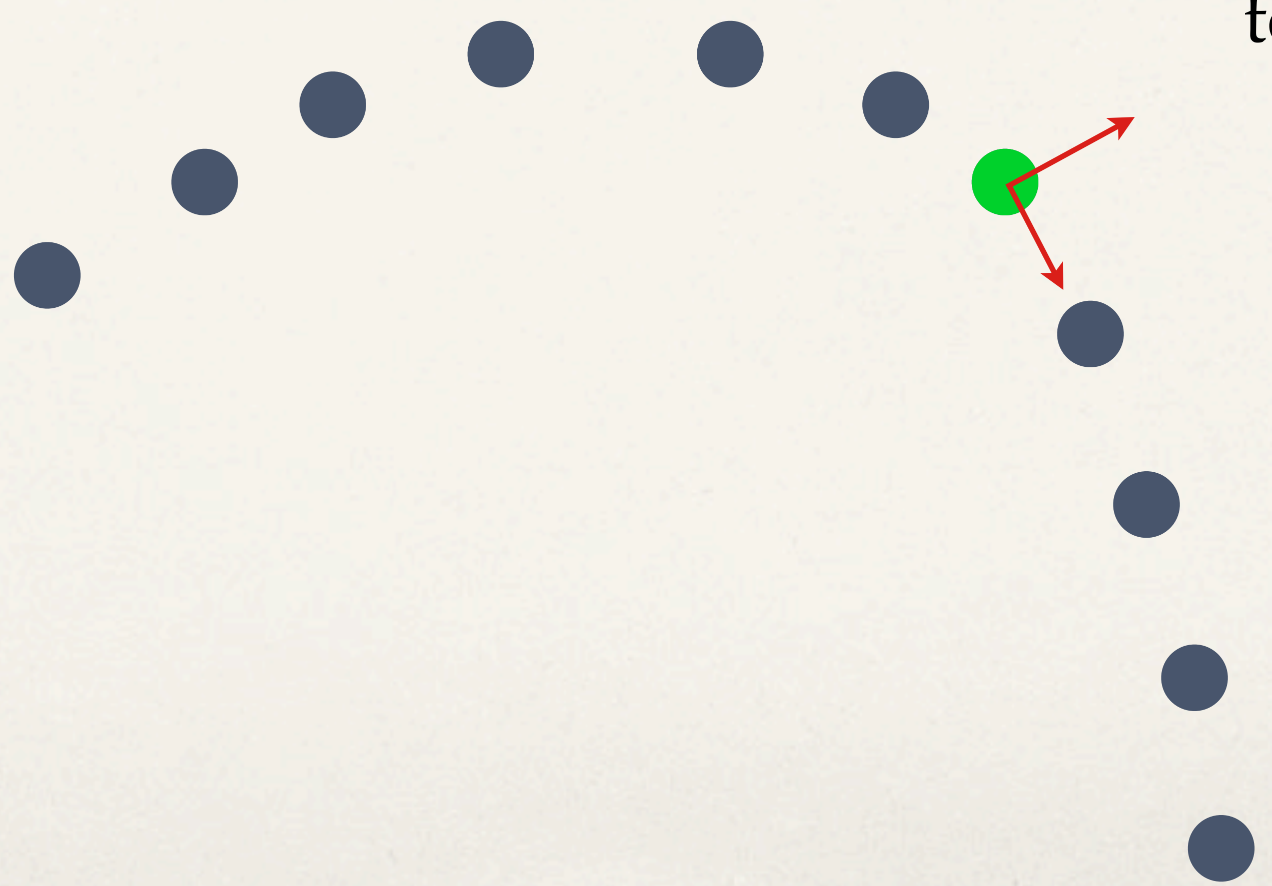
CAE must capture manifold directions

$$\mathcal{J}_{\mathrm{CAE}} = \sum_{x \in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

Reconstruction $\Leftarrow$ **tradeoff** $\Rightarrow$ **Contraction**

(warning: may require tied weights)

pressure to be insensitive to ***all*** directions



CAE must capture manifold directions

$$\mathcal{J}_{\text{CAE}} = \sum_{x \in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

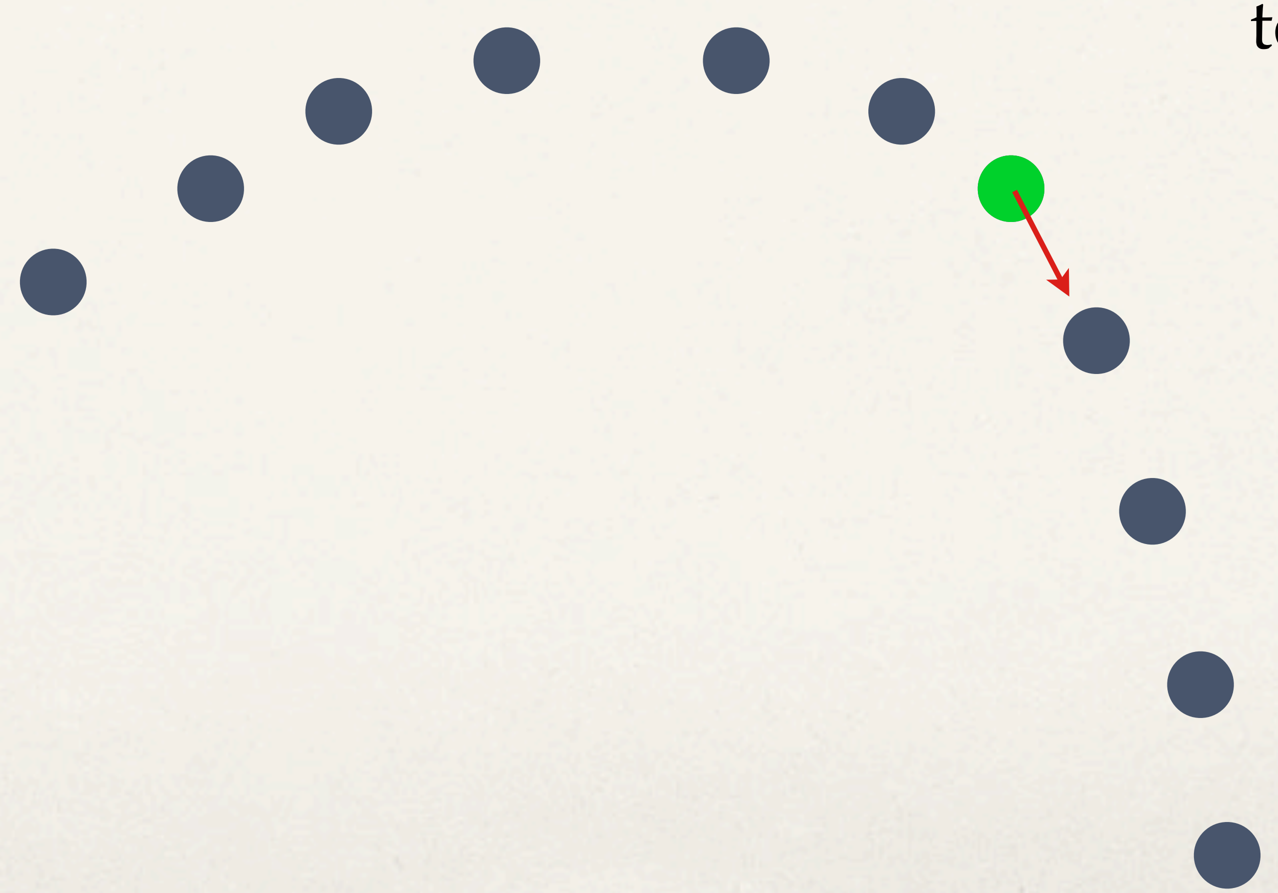
Reconstruction

⇐ tradeoff ⇒

(warning: may require tied weights)

Contraction

pressure to be insensitive to ***all*** directions



CAE must capture manifold directions

$$\mathcal{J}_{\mathrm{CAE}} = \sum_{x\in D}^{n} L(x, g(h(x))) + \lambda \left\| \frac{\partial h(x)}{\partial x} \right\|^2$$

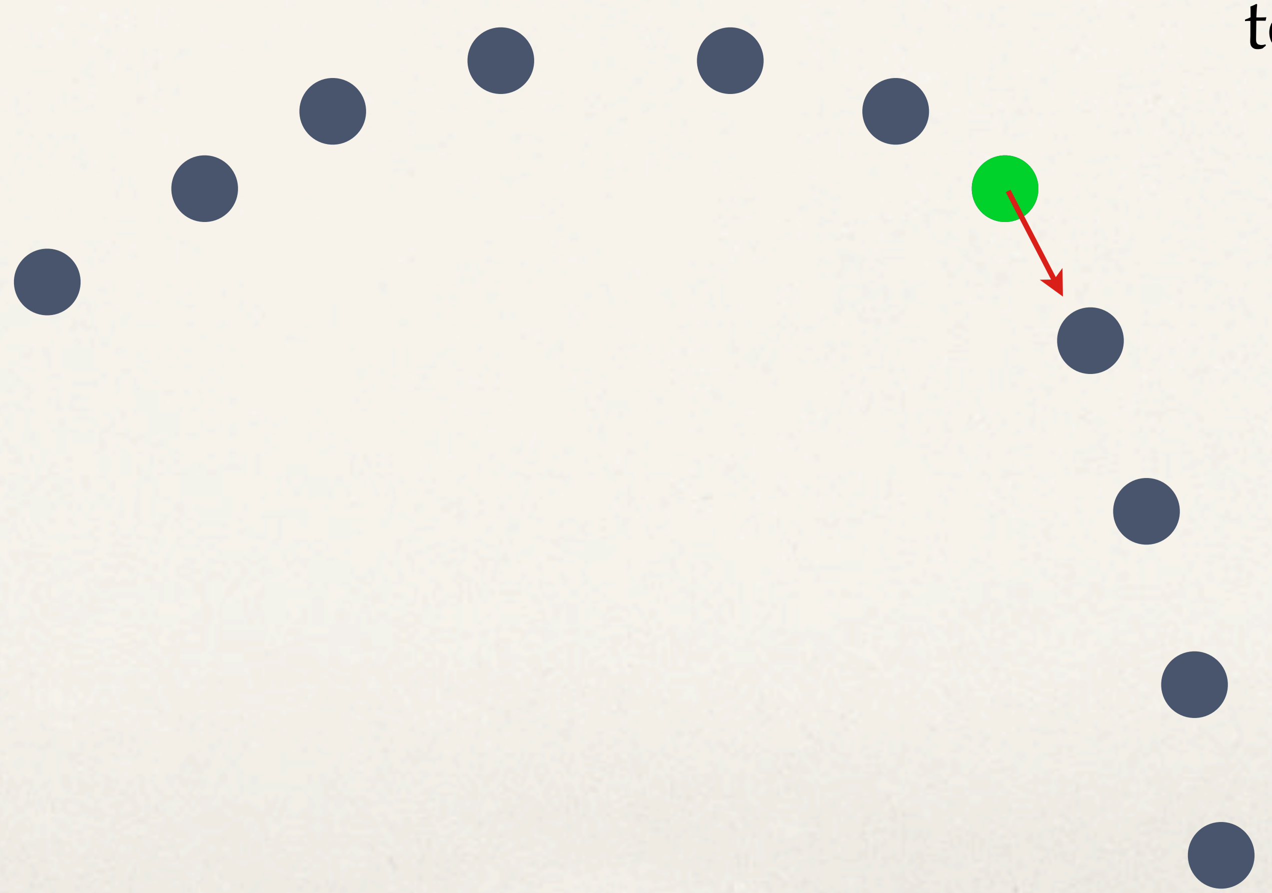
Reconstruction

$\Leftarrow$ **tradeoff** $\Rightarrow$

(warning: may require tied weights)

Contraction

pressure to be insensitive to ***all*** directions



Learned tangent space

- Jacobian $J_h(x) = \frac{\partial h}{\partial x}(x)$ measures sensitivity of h locally around x

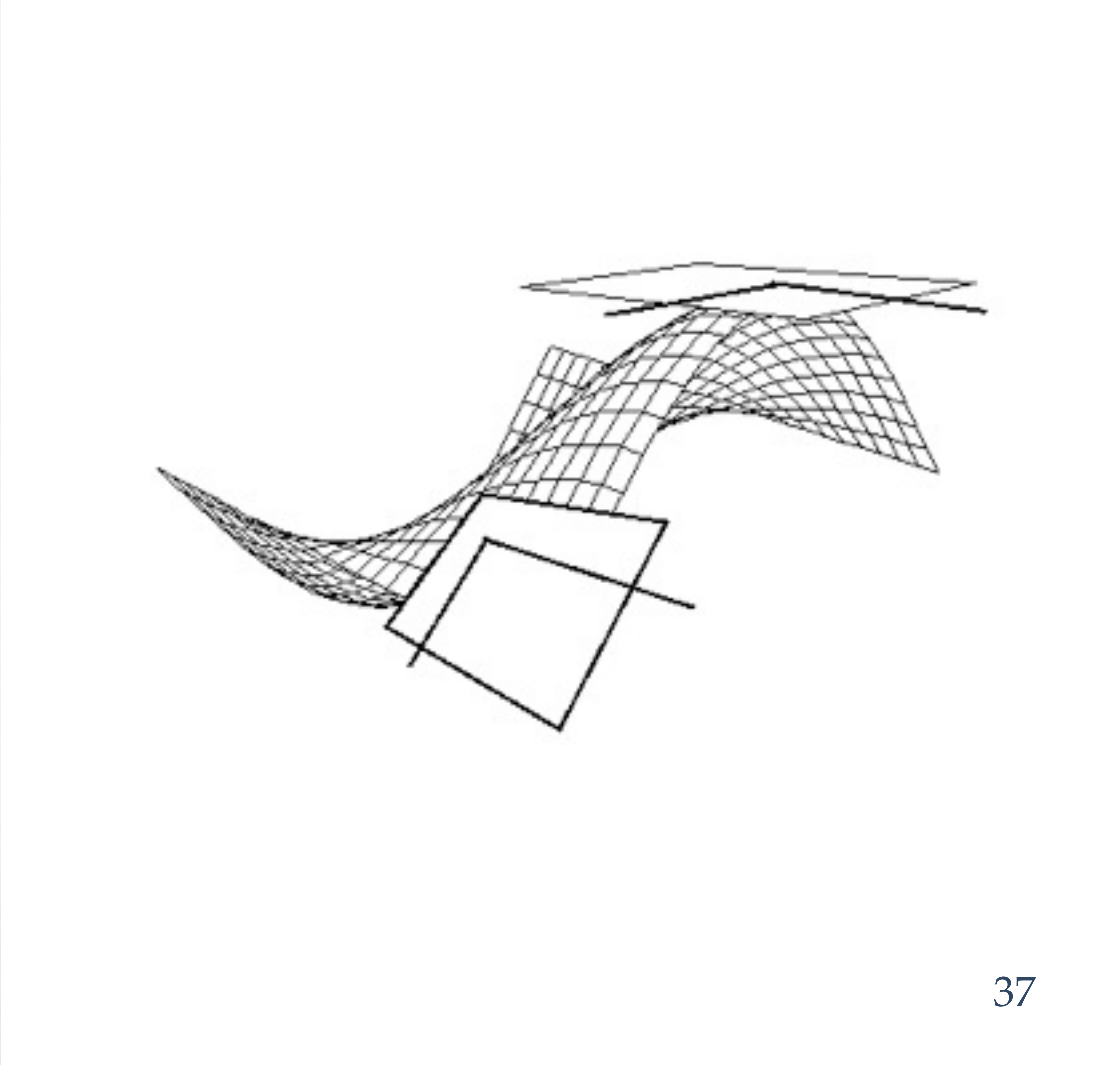
SVD:

$$\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

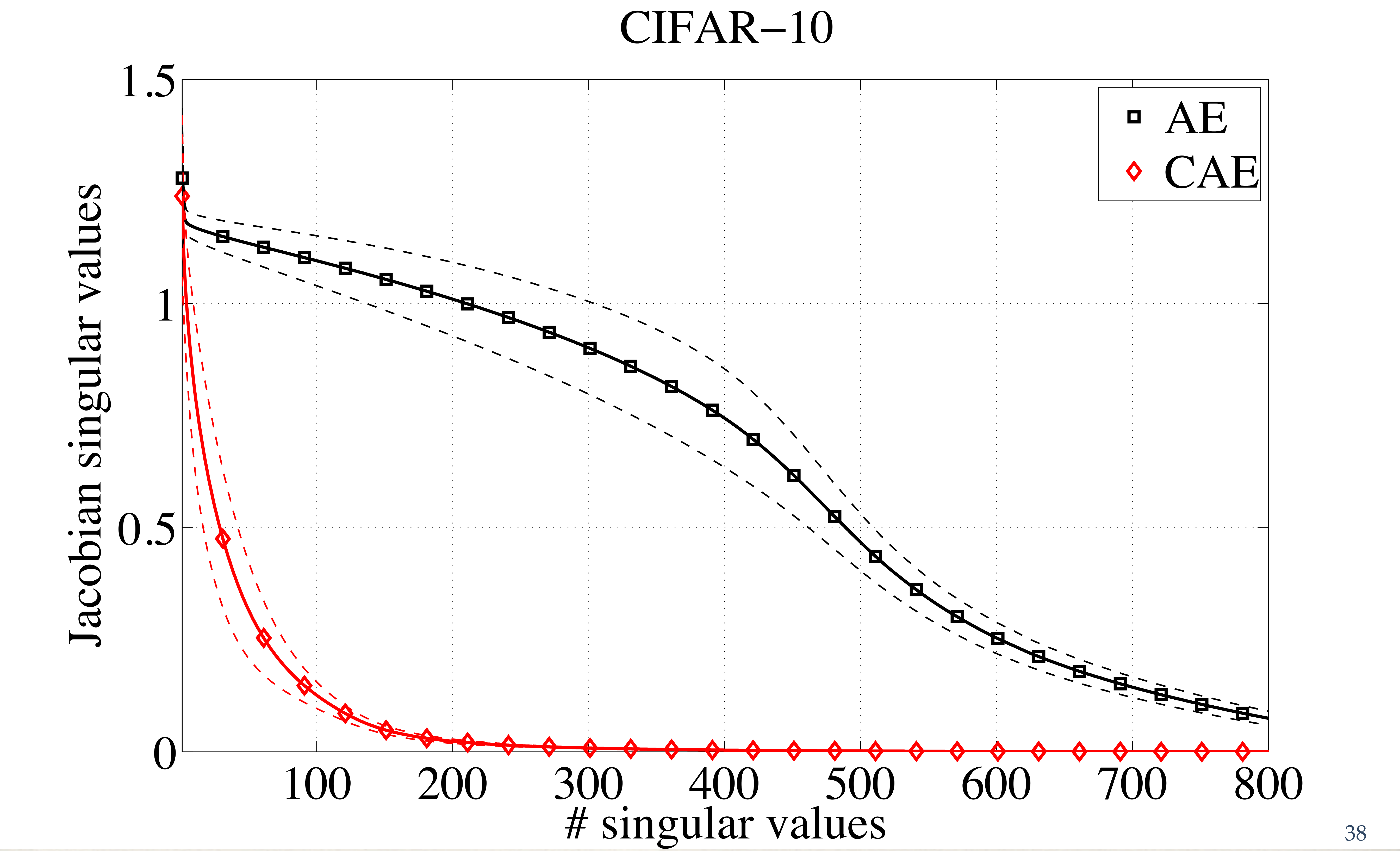
Top **singular vectors** are **tangent** directions to which *h* is most sensitive.

$$\mathbf{T}_x = \{\mathbf{U}_{\cdot k} | \mathbf{S}_{kk} > \epsilon\}$$

- CAE captures the structure of the manifold by defining an atlas of charts.



SVD of $J_h(x) = \frac{\partial h}{\partial x}(x)$

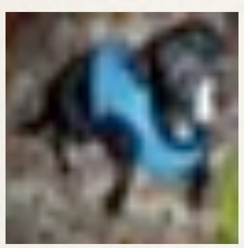


Learned tangents CIFAR-10

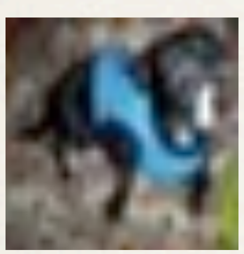
Input Point

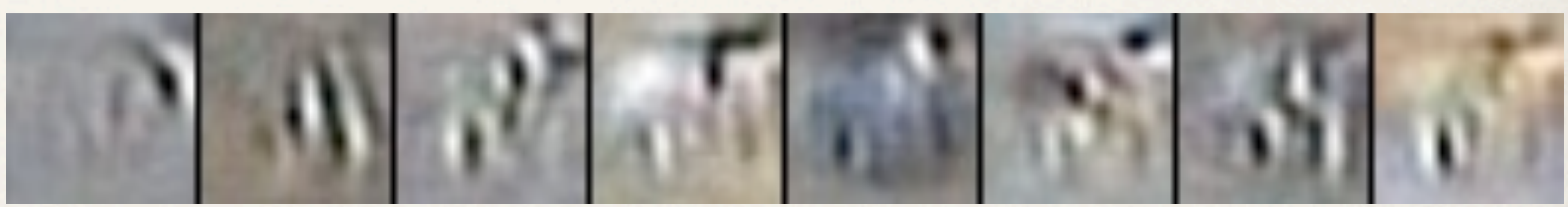
Tangents

Local PCA (as e.g. in Manifold Parzen)



Contractive Auto-Encoder (singular vectors of $J_h(x)$)



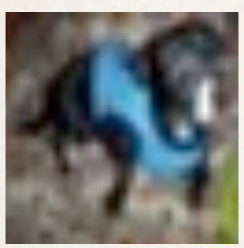


Not based on explicit neighbors or pairs of points!

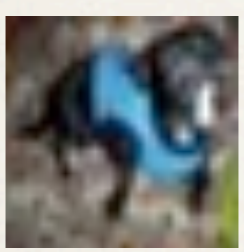
Learned tange

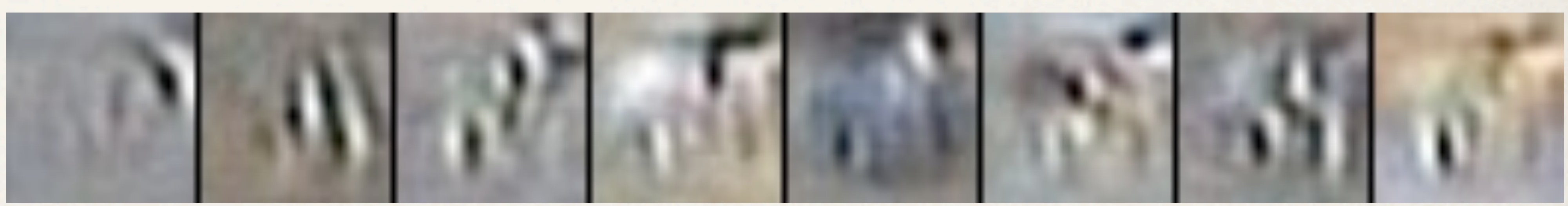
Input Point

Local PCA (as e.g. in



Contractive Auto-Encoder (singular vectors of $J_h(x)$)





Not based on explicit neighbors or pairs of points!

How to exploit the learned tangents

- **Simard et al, 1993** exploited tangents derived from prior-knowledge of image deformations **we can use our learned tangents instead**.

- Use them to define **tangent distance** to use in your favorite distance (k-NN) or kernel-based classifier...

- Use them with **tangent propagation** when fine-tuning a deep-net classifier to make class prediction insensitive to tangent directions. (*Manifold Tangent Classifier*, Rifai et al. NIPS 2011) 0.81% on MNIST

- Moving preferably along tangents allows **efficient quality sampling** (Rifai et al. ICML 2012)

How to exploit the learned tangents

- **Simard et al, 1993** exploited tangents derived from prior-knowledge of image deformations **we can use our learned tangents instead**.

- Use them to define **tangent distance** to use in your favorite distance (k-NN) or kernel-based classifier...

- Use them with **tangent propagation** when fine-tuning a deep-net classifier to make class prediction insensitive to tangent directions. (*Manifold Tangent Classifier*, Rifai et al. NIPS 2011) 0.81% on MNIST

- Moving preferably along tangents allows **efficient quality sampling** (Rifai et al. ICML 2012)

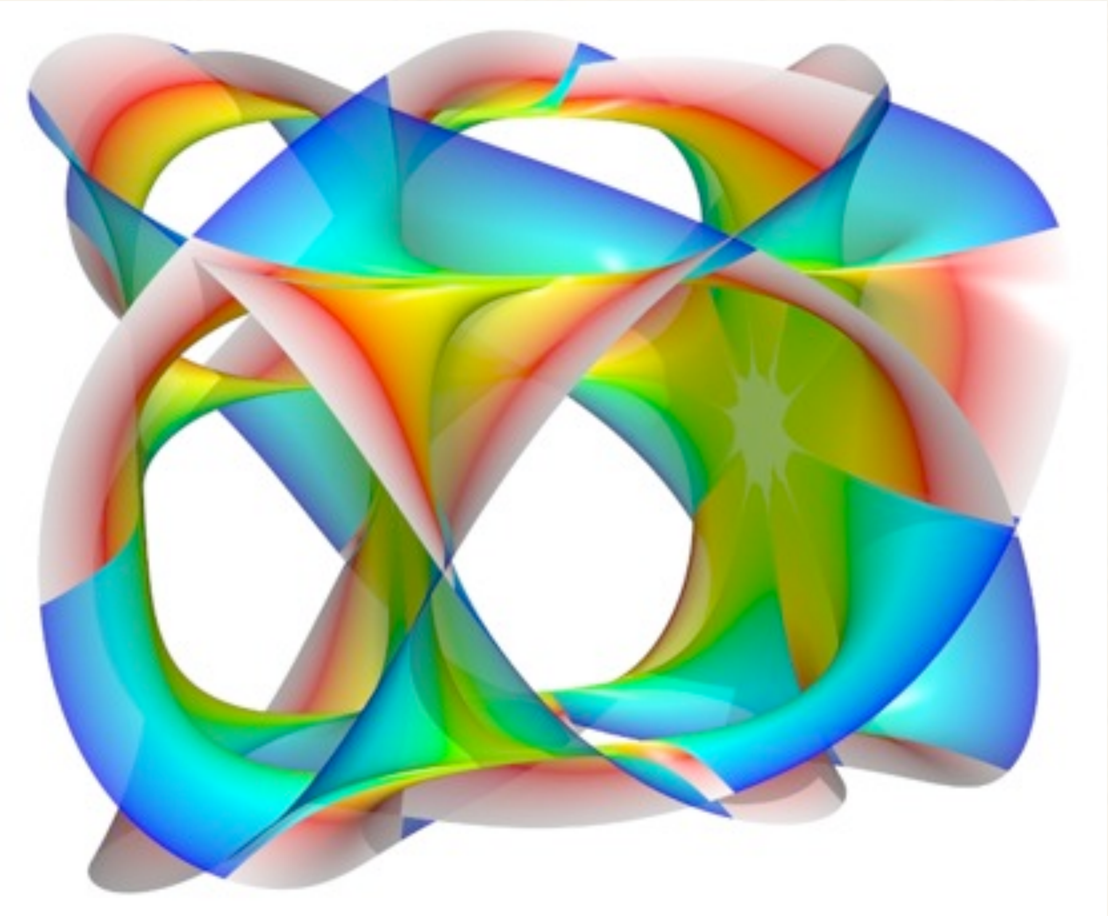
Results: MNIST

(standard split)

Models invariant to random feature permutation
i.e. no domain knowledge, except *LeNet*

K-NN	NN	SVM	DBN	CAE	DBM	*LeNet* CNN	**MTC**
3,09%	1,60%	1,40%	1,17%	1,04%	0,95%	0,95%	**0,81%**

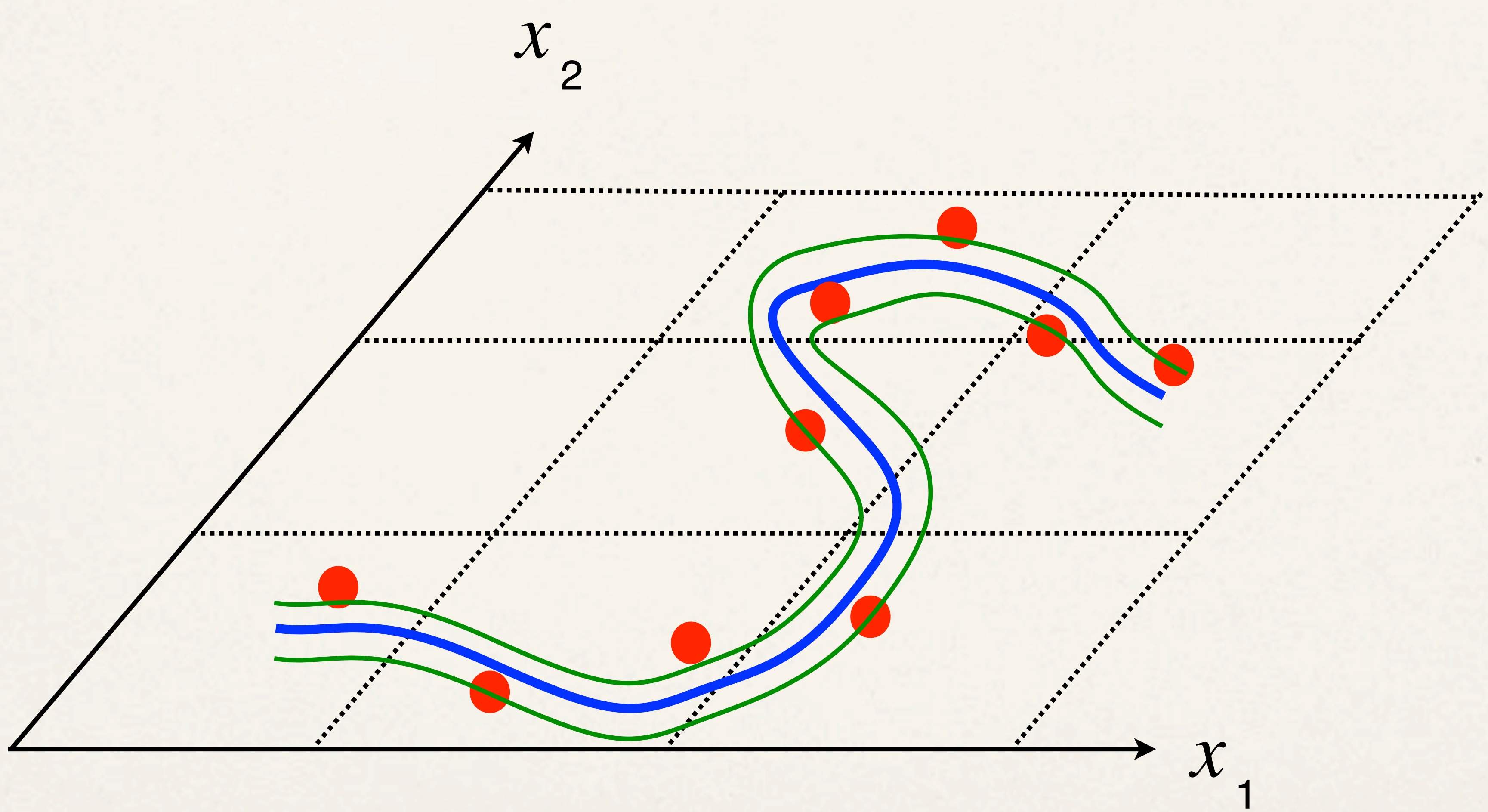
PART III



Ongoing work,
thoughts and speculation

Probability density modeling
of data that follows the manifold hypothesis

Example:

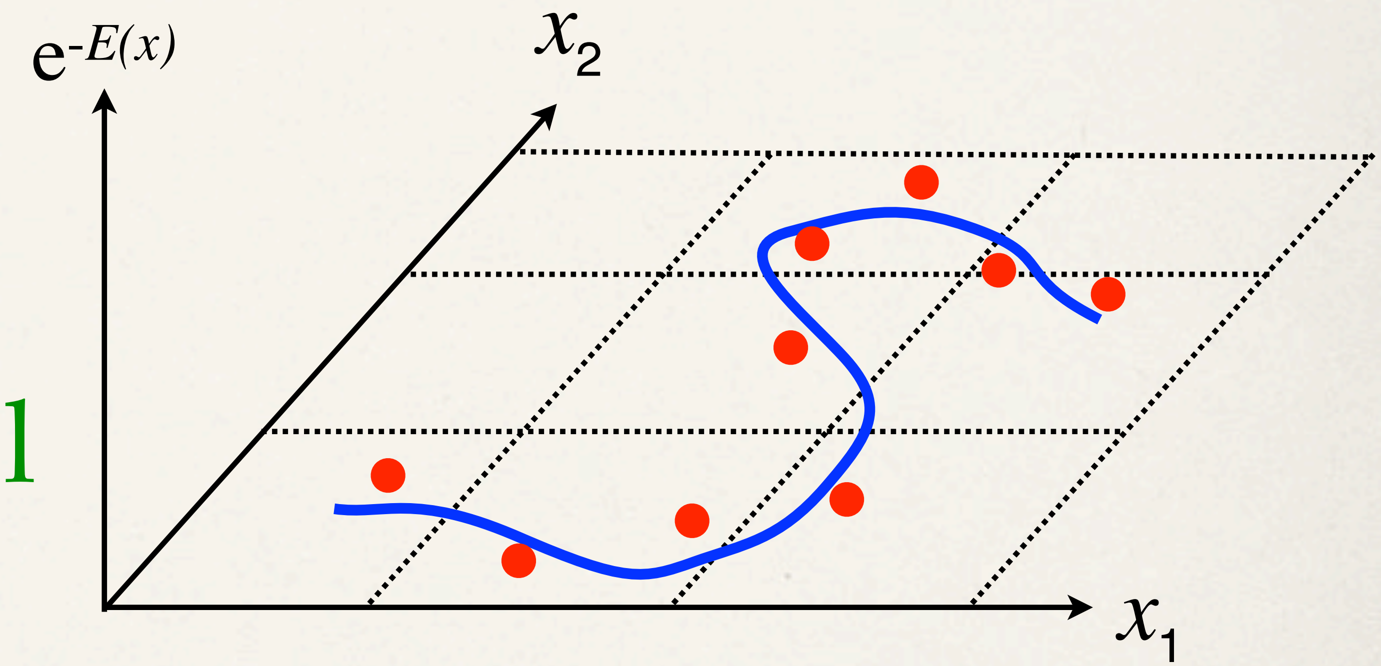


What do these inductive principles yield

- Maximum likelihood
- Contrastive Divergence
- Score Matching

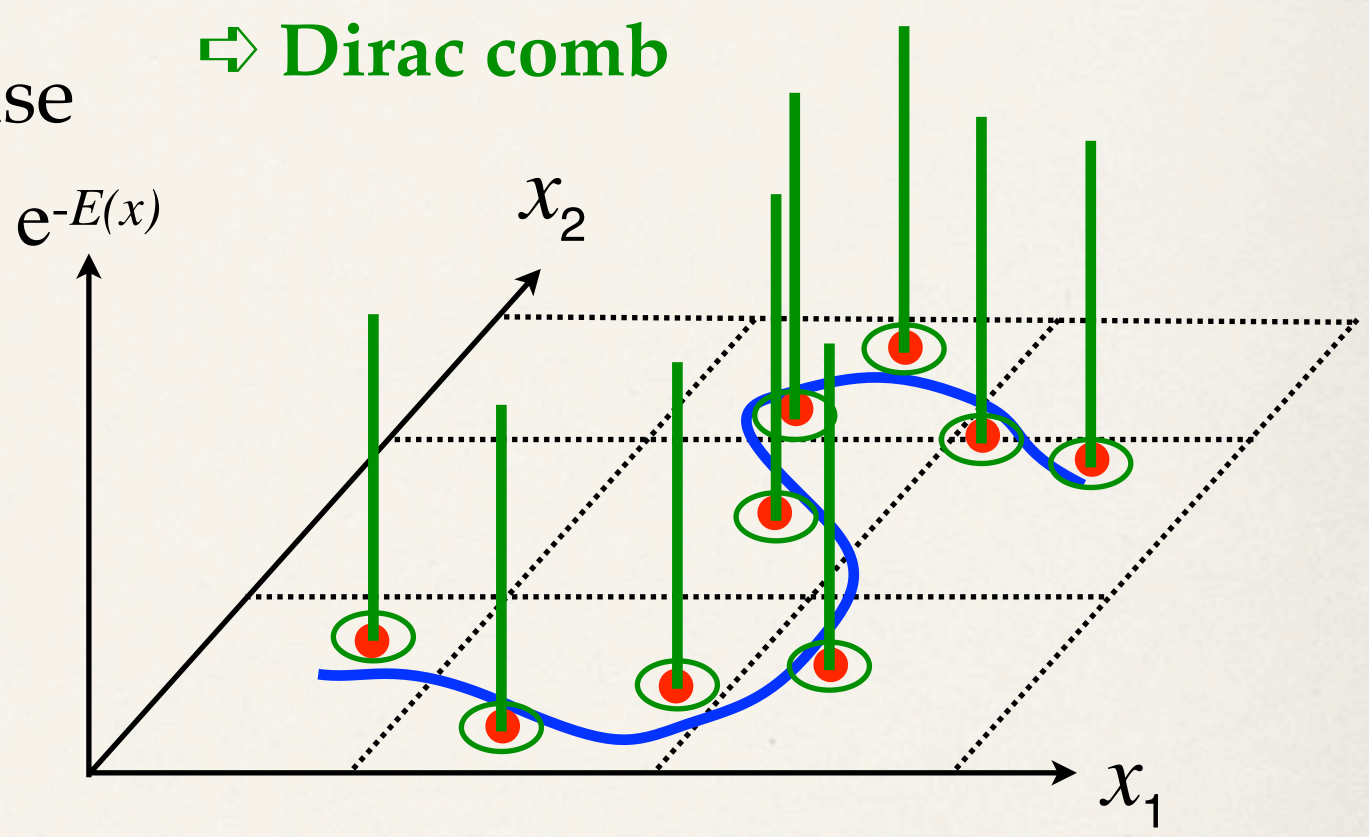
when applied to a model of very high capacity

?



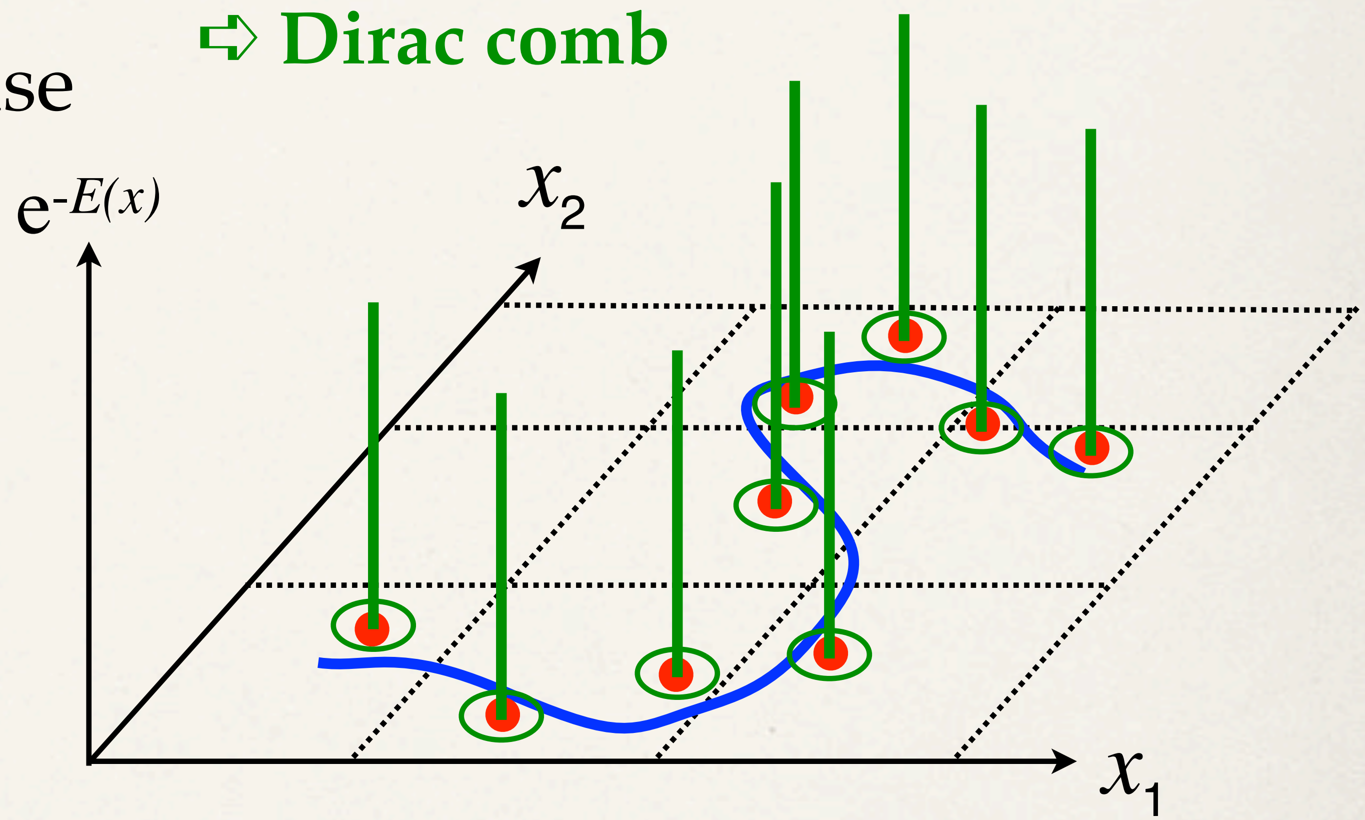
They tend to a Dirac comb!

Common fitting procedures increase probability at training points lower it everywhere else...



They tend to a Dirac comb!

Common fitting procedures increase probability at training points lower it everywhere else...



The Porcupine effect!

Ex: score matching
(Hyvärinen 2005)

Learn energy function by minimizing:

$$J_{SM}(\theta) = \sum_{x \in D} \left(\left\| \frac{\partial E}{\partial x}(x) \right\|^2 - \sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x) \right)$$

Ex: score matching
(Hyvärinen 2005)

Learn energy function by minimizing:

$$J_{SM}(\theta) = \sum_{x \in D} \left(\left\| \frac{\partial E}{\partial x}(x) \right\|^2 - \sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x) \right)$$

$$\|J_E(x)\|^2$$

Ex: score matching

(Hyvärinen 2005)

Learn energy function by minimizing:

$$J_{SM}(\theta) = \sum_{x \in D} \left(\left\| \frac{\partial E}{\partial x}(x) \right\|^2 - \sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x) \right)$$

$\|J_E(x)\|^2$

First derivative encouraged to be small: ensures training points stay close to local minima of E

Ex: score matching
(Hyvärinen 2005)

Learn energy function by minimizing:

$$J_{SM}(\theta) = \sum_{x \in D} \left(\underbrace{\left\| \frac{\partial E}{\partial x}(x) \right\|^2}_{\|J_E(x)\|^2} - \underbrace{\sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x)}_{\substack{\text{Laplacian} \\ \mathrm{Tr}(H_E(x))}} \right)$$

First derivative encouraged to be small: ensures training points stay close to local minima of E

Ex: score matching
(Hyvärinen 2005)

Learn energy function by minimizing:

$$J_{SM}(\theta) = \sum_{x \in D} \left(\left\| \frac{\partial E}{\partial x}(x) \right\|^2 - \sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x) \right)$$

$\|J_E(x)\|^2$

Laplacian

$\mathrm{Tr}(H_E(x))$

First derivative encouraged to be small: ensures training points stay close to local minima of E

Encourage large positive curvature in all directions

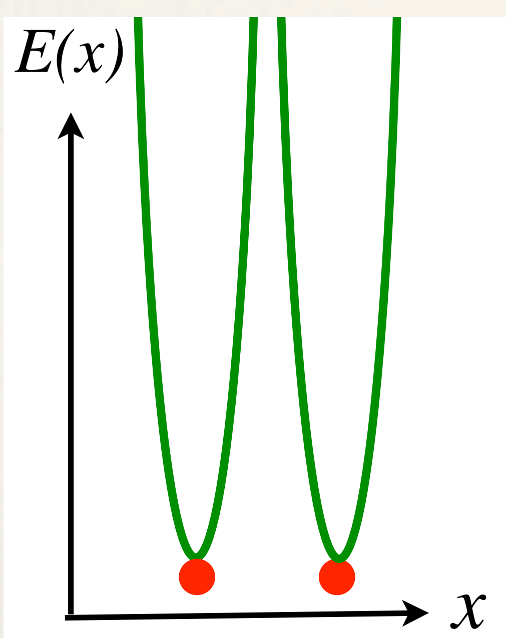
Ex: score matching
(Hyvärinen 2005)

Learn energy function by minimizing:

$$J_{SM}(\theta) = \sum_{x \in D} \left(\underbrace{\left\| \frac{\partial E}{\partial x}(x) \right\|^2}_{\|J_E(x)\|^2} - \underbrace{\sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x)}_{\substack{\text{Laplacian} \\ \mathrm{Tr}(H_E(x))}} \right)$$

First derivative encouraged to be small: ensures training points stay close to local minima of E

Encourage large positive curvature in all directions



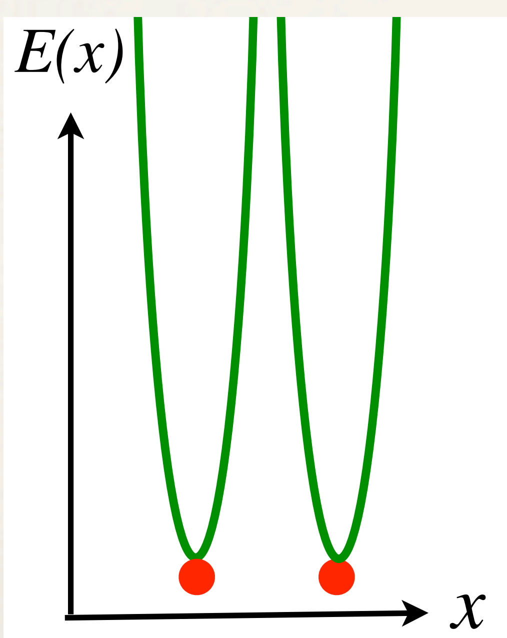
Ex: score matching
(Hyvärinen 2005)

Learn energy function by minimizing:

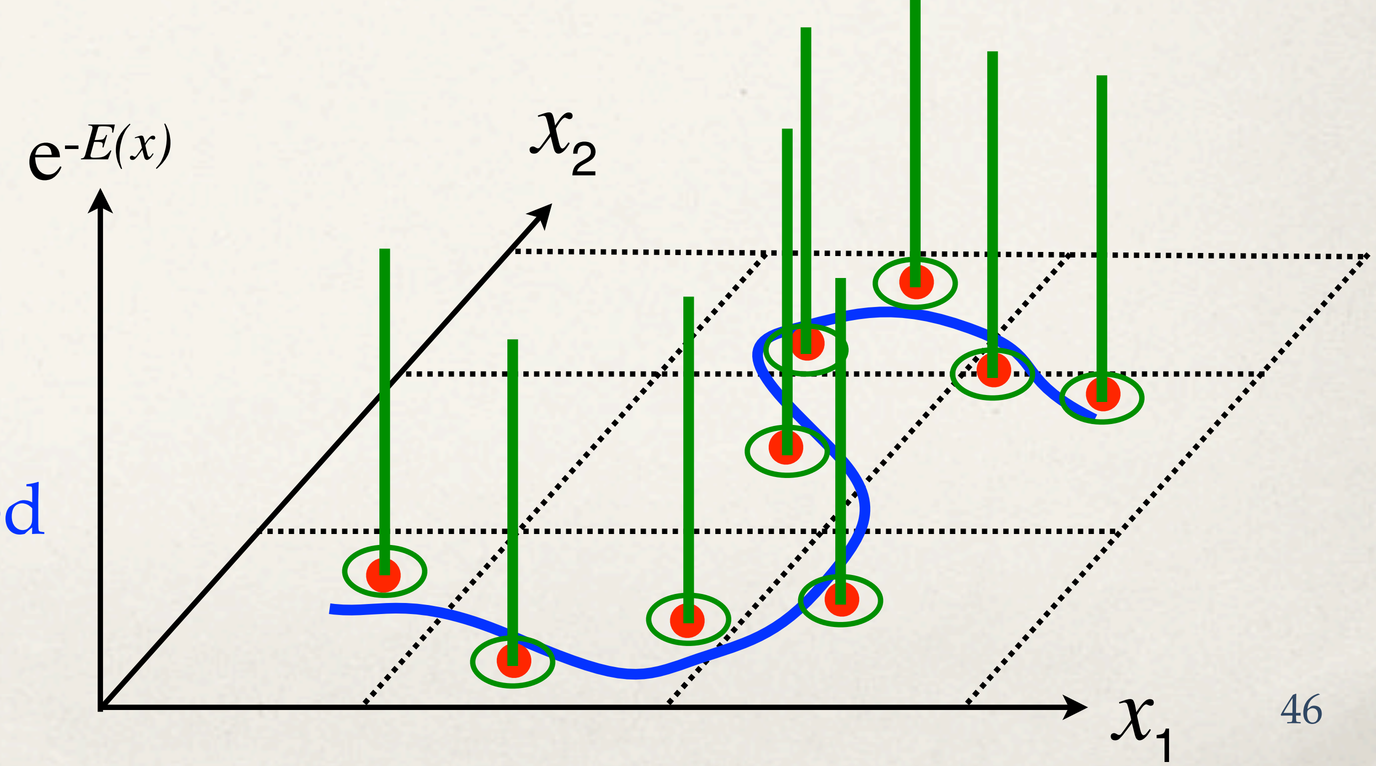
$$J_{SM}(\theta) = \sum_{x \in D} \left(\underbrace{\left\| \frac{\partial E}{\partial x}(x) \right\|^2}_{\|J_E(x)\|^2} - \underbrace{\sum_{i=1}^{d} \frac{\partial^2 E}{\partial x_i^2}(x)}_{\substack{\text{Laplacian} \\ \mathrm{Tr}(H_E(x))}} \right)$$

First derivative encouraged to be small: ensures training points stay close to local minima of E

Encourage large positive curvature in all directions



sharply peaked density



Unseen porcupine ?

- Porcupine corresponds to **many *0d* manifolds**.
- We don't usually see him because **parametrized models** are very constrained.
- He may be *lurking* for when we increase capacity...

Could we define a **non-parametric** prior preference towards **>0**-dimensional manifolds ?

A manifold inductive bias

for learning an Energy function E

Can we bias the training criteria towards modeling a manifold structure?

Score matching

Manifold bias

A manifold inductive bias

for learning an Energy function E

Can we bias the training criteria towards modeling a manifold structure?

Score matching

- Encourages large «curvature» of E in **all** directions:
- By maximizing Laplacian

$$\mathrm{Tr}(H_E(x)) = \sum_{i=1}^{d} \lambda_i$$

$$H_E(x) = \frac{\partial^2 E}{\partial x^2}(x)$$

Manifold bias

A manifold inductive bias

for learning an Energy function E

Can we bias the training criteria towards modeling a manifold structure?

Score matching

- Encourages large «curvature» of E in **all** directions:
- By maximizing Laplacian

$$\mathrm{Tr}(H_E(x)) = \sum_{i=1}^{d} \lambda_i$$

eigenspectrum of $H_E\,(x)$

$$H_E(x) = \frac{\partial^2 E}{\partial x^2}(x)$$

Manifold bias

A manifold inductive bias

for learning an Energy function E

Can we bias the training criteria towards modeling a manifold structure?

Score matching

- Encourages large «curvature» of E in **all** directions:
- By maximizing Laplacian

$$\mathrm{Tr}(H_E(x)) = \sum_{i=1}^{d} \lambda_i$$

eigenspectrum of $H_E(x)$

Isotropic

$$H_E(x) = \frac{\partial^2 E}{\partial x^2}(x)$$

Manifold bias

A manifold inductive bias
for learning an Energy function E

Can we bias the training criteria towards modeling a manifold structure?

Score matching

- Encourages large «curvature» of E in **all** directions:
- By maximizing Laplacian

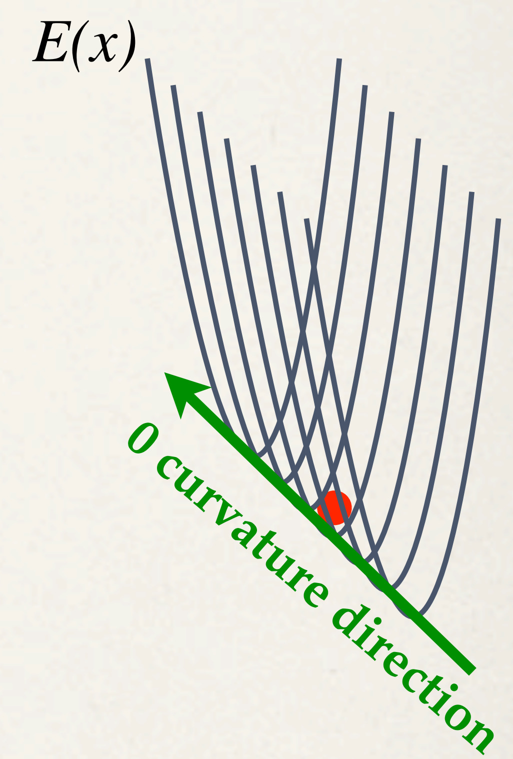
$$\mathrm{Tr}(H_E(x)) = \sum_{i=1}^{d} \lambda_i$$

eigenspectrum of $H_E(x)$

Isotropic

$$H_E(x) = \frac{\partial^2 E}{\partial x^2}(x)$$

Manifold bias



A manifold inductive bias

for learning an Energy function E

Can we bias the training criteria towards modeling a manifold structure?

Score matching

- Encourages large «curvature» of E in **all** directions:
- By maximizing Laplacian

$$\mathrm{Tr}(H_E(x)) = \sum_{i=1}^{d} \lambda_i$$

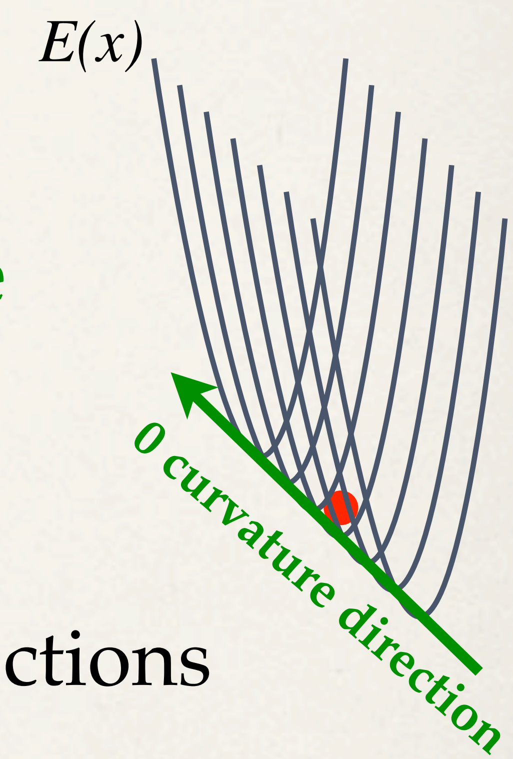
eigenspectrum of $H_E(x)$

Isotropic

$$H_E(x) = \frac{\partial^2 E}{\partial x^2}(x)$$

Manifold bias

- Instead encourage
  - near **zero curvature** in **≈d_M** directions
  - target positive curvature C in remaining d-d_M directions



- by encouraging

$$\lambda \approx (\underbrace{C, \ldots, C}_{d-d_M}, \underbrace{0, \ldots, 0}_{d_M})$$

A manifold inductive bias

for learning a representation *h*

Can we bias the training criteria towards modeling a manifold structure?

Contractive autoencoder penalty

Manifold bias

A manifold inductive bias

for learning a representation h

Can we bias the training criteria towards modeling a manifold structure?

Contractive autoencoder penalty

- Encourages insensitivity of representation in **all** directions:
- By penalizing Jacobian norm

$$\|J_h(x)\|_F^2 = \mathrm{Tr}(J_h J_h^T) = \sum_{i=1}^{d} \lambda_i$$

$$J_h(x) = \frac{\partial h}{\partial x}(x)$$

Manifold bias

A manifold inductive bias

for learning a representation h

Can we bias the training criteria towards modeling a manifold structure?

Contractive autoencoder penalty

- Encourages insensitivity of representation in **all** directions:
- By penalizing Jacobian norm

$$\|J_h(x)\|_F^2 = \mathrm{Tr}(J_h J_h^T) = \sum_{i=1}^{d} \lambda_i$$

$$J_h(x) = \frac{\partial h}{\partial x}(x)$$

squared singular value spectrum of $J_h\,(x)$

Manifold bias

A manifold inductive bias

for learning a representation h

Can we bias the training criteria towards modeling a manifold structure?

Contractive autoencoder penalty

- Encourages insensitivity of representation in **all** directions:
- By penalizing Jacobian norm

$$\|J_h(x)\|_F^2 = \mathrm{Tr}(J_h J_h^T) = \sum_{i=1}^{d} \lambda_i$$

$$J_h(x) = \frac{\partial h}{\partial x}(x)$$

squared singular value spectrum of $J_h\,(x)$

Isotropic

Manifold bias

A manifold inductive bias

for learning a representation h

Can we bias the training criteria towards modeling a manifold structure?

Contractive autoencoder penalty

- Encourages insensitivity of representation in **all** directions:
- By penalizing Jacobian norm

$$\|J_h(x)\|_F^2 = \mathrm{Tr}(J_h J_h^T) = \sum_{i=1}^{d} \lambda_i$$

$$J_h(x) = \frac{\partial h}{\partial x}(x)$$

squared singular value spectrum of $J_h\,(x)$

Isotropic

Encourages 0 dim. «manifolds»

Manifold bias

A manifold inductive bias

for learning a representation h

Can we bias the training criteria towards modeling a manifold structure?

Contractive autoencoder penalty

- Encourages insensitivity of representation in **all** directions:
- By penalizing Jacobian norm

$$\|J_h(x)\|_F^2 = \mathrm{Tr}(J_h J_h^T) = \sum_{i=1}^{d} \lambda_i$$

$$J_h(x) = \frac{\partial h}{\partial x}(x)$$

squared singular value spectrum of $J_h(x)$

Isotropic

Encourages 0 dim. «manifolds»

Manifold bias

- Instead encourage
  - representation sensitive to $\approx d_M$ input directions
  - insensitive to remaining d-d_M directions
- by encouraging

$$\lambda \approx (\underbrace{1,\ldots,1}_{d_M},\underbrace{0,\ldots,0}_{d-d_M})$$

How to encourage this kind of spectrum $\lambda \approx (\underbrace{1, \ldots, 1}_{d_M}, \underbrace{0, \ldots, 0}_{d-d_M})$ in practice

Manifold Autoencoder:

- Encourage *J* to be a **partial isometry**: $J_h J_h^T J_h \approx J_h$ will ensure that singular values are close to either 1 or 0.
- Encourage sum of squared singular values to be close to target ***d_M***

$$\sum_{i=1}^{d} \lambda_i = \mathrm{Tr}(J_h J_h^T) = \|J_h(x)\|_F^2 \approx d_M$$

How to encourage this kind of spectrum $\lambda \approx (\underbrace{1, \ldots, 1}_{d_M}, \underbrace{0, \ldots, 0}_{d-d_M})$ in practice

Manifold Autoencoder:

- Encourage *J* to be a **partial isometry**: $J_h J_h^T J_h \approx J_h$
  will ensure that singular values are close to either 1 or 0.

- Encourage sum of squared singular values to be close to target ***d_M***

$$\sum_{i=1}^{d} \lambda_i = \mathrm{Tr}(J_h J_h^T) = \|J_h(x)\|_F^2 \approx d_M$$

Penalty, added to reconstruction error:

Contractive Autoencoder:	Manifold Autoencoder:

How to encourage this kind of spectrum $\lambda \approx (\underbrace{1,\ldots,1}_{d_M},\underbrace{0,\ldots,0}_{d-d_M})$ in practice

Manifold Autoencoder:

- Encourage J to be a **partial isometry**: $J_h J_h^T J_h \approx J_h$ will ensure that singular values are close to either 1 or 0.
- Encourage sum of squared singular values to be close to target ***d_M***

$$\sum_{i=1}^{d} \lambda_i = \mathrm{Tr}(J_h J_h^T) = \|J_h(x)\|_F^2 \approx d_M$$

Penalty, added to reconstruction error:

Contractive Autoencoder:

$$\gamma \|J_h(x)\|_F^2$$

Isotropic

Manifold Autoencoder:

How to encourage this kind of spectrum $\lambda \approx (\underbrace{1,\ldots,1}_{d_M},\underbrace{0,\ldots,0}_{d-d_M})$ in practice

Manifold Autoencoder:

- Encourage *J* to be a **partial isometry**: $J_h J_h^T J_h \approx J_h$ will ensure that singular values are close to either 1 or 0.
- Encourage sum of squared singular values to be close to target $\boldsymbol{d_M}$

$$\sum_{i=1}^{d} \lambda_i = \mathrm{Tr}(J_h J_h^T) = \|J_h(x)\|_F^2 \approx d_M$$

Penalty, added to reconstruction error:

Contractive Autoencoder:

$$\gamma \|J_h(x)\|_F^2$$

Isotropic

Manifold Autoencoder:

$$\gamma_1 \left(\|J_h(x)\|_F^2 - d_M\right)^2 + \gamma_2 \|J_h J_h^T J_h - J_h\|_F^2$$

How to encourage this kind of spectrum $\lambda \approx (\underbrace{1,\ldots,1}_{d_M},\underbrace{0,\ldots,0}_{d-d_M})$ in practice

Manifold Autoencoder:

- Encourage J to be a **partial isometry**: $J_h J_h^T J_h \approx J_h$ will ensure that singular values are close to either 1 or 0.
- Encourage sum of squared singular values to be close to target $\boldsymbol{d_M}$

$$\sum_{i=1}^{d} \lambda_i = \mathrm{Tr}(J_h J_h^T) = \|J_h(x)\|_F^2 \approx d_M$$

Penalty, added to reconstruction error:

Contractive Autoencoder:

$$\gamma \|J_h(x)\|_F^2$$

Isotropic

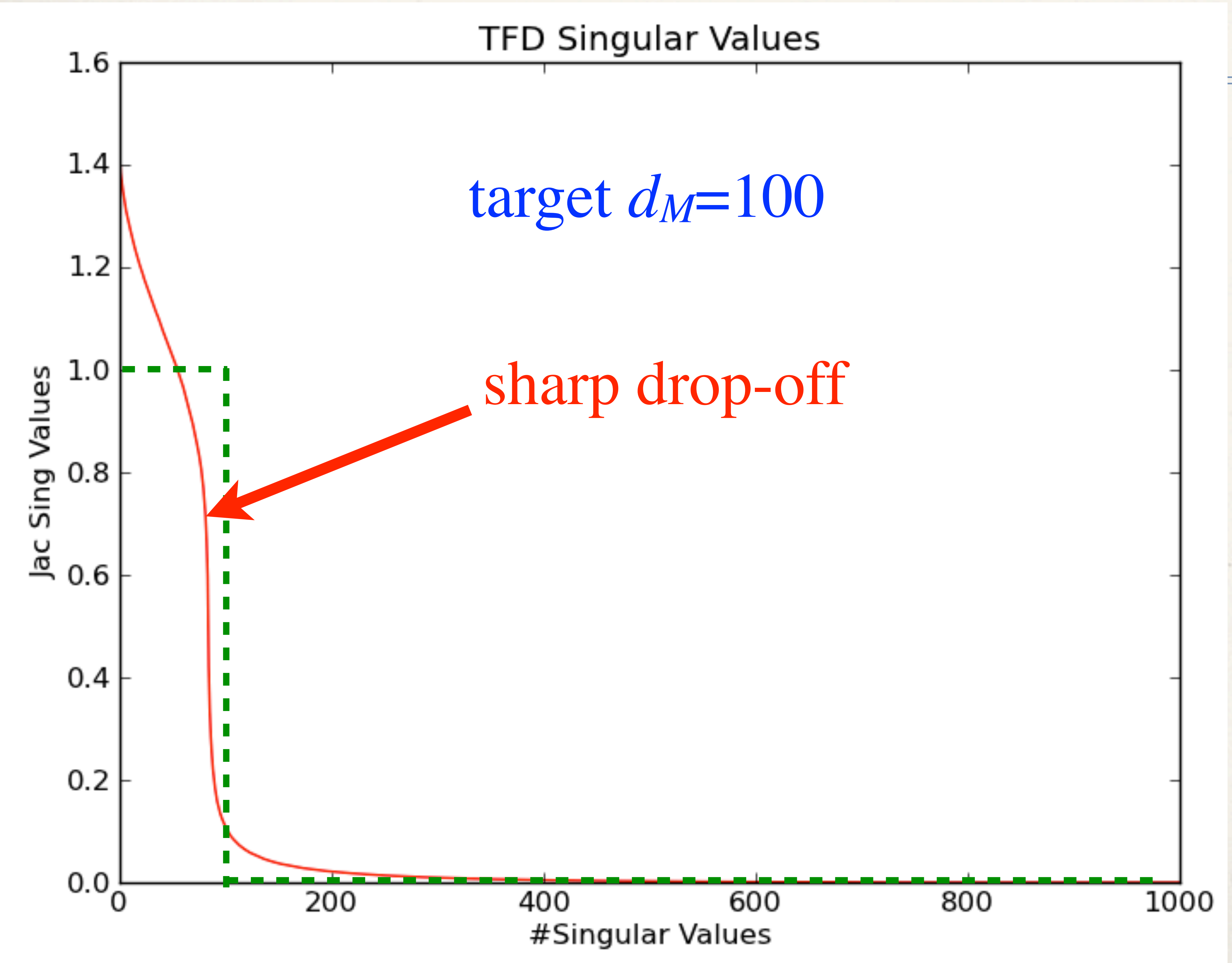
Manifold Autoencoder:

$$\gamma_1 \left(\|J_h(x)\|_F^2 - d_M\right)^2 + \gamma_2 \underbrace{\|J_h J_h^T J_h - J_h\|_F^2}$$

or computationally more efficient stochastic probe: $\|J_h J_h^T J_h \epsilon - J_h \epsilon\|_F^2$

ϵ random vector (or matrix)

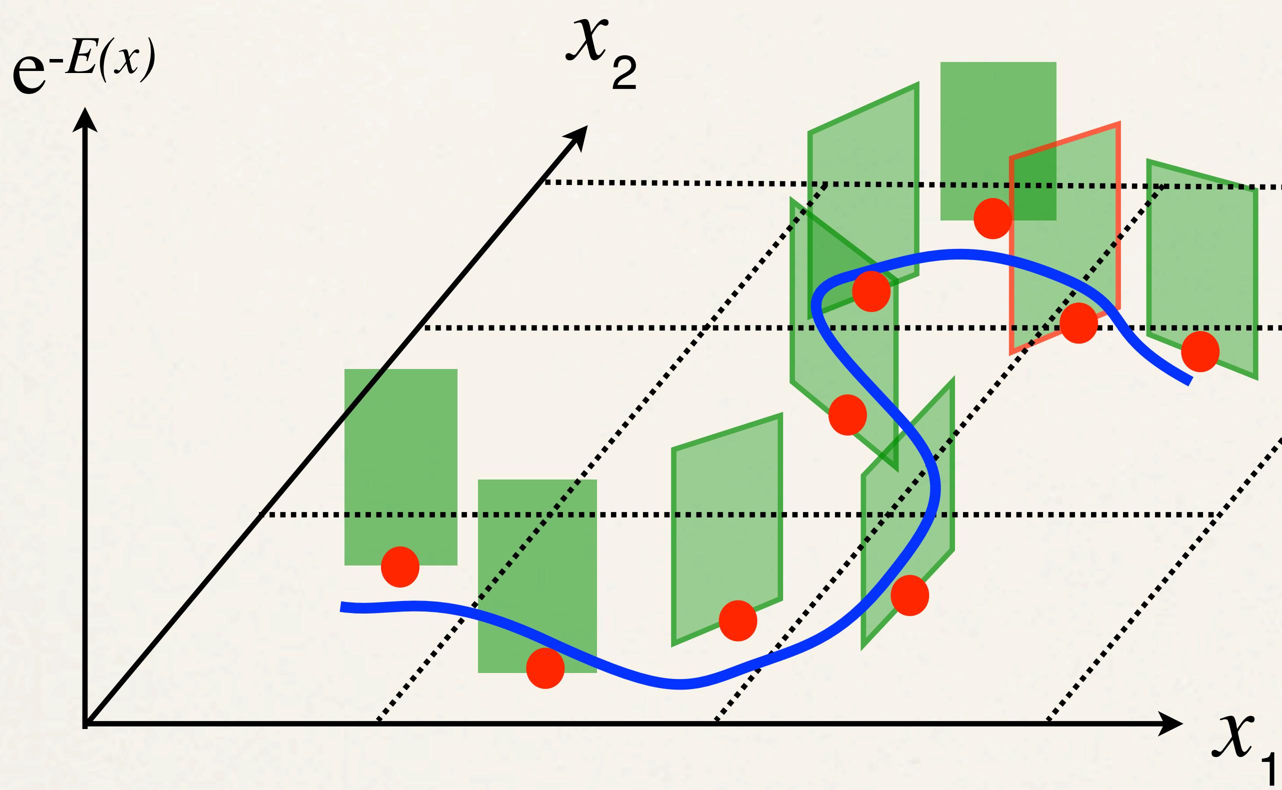
Preliminary proof of concept result



student:
Hani Almousli

Is this enough?

- Now instead of a Porcupine (dimension 0 peaks) we can get a **Pangolin** (d_M dimensional scales) !!!

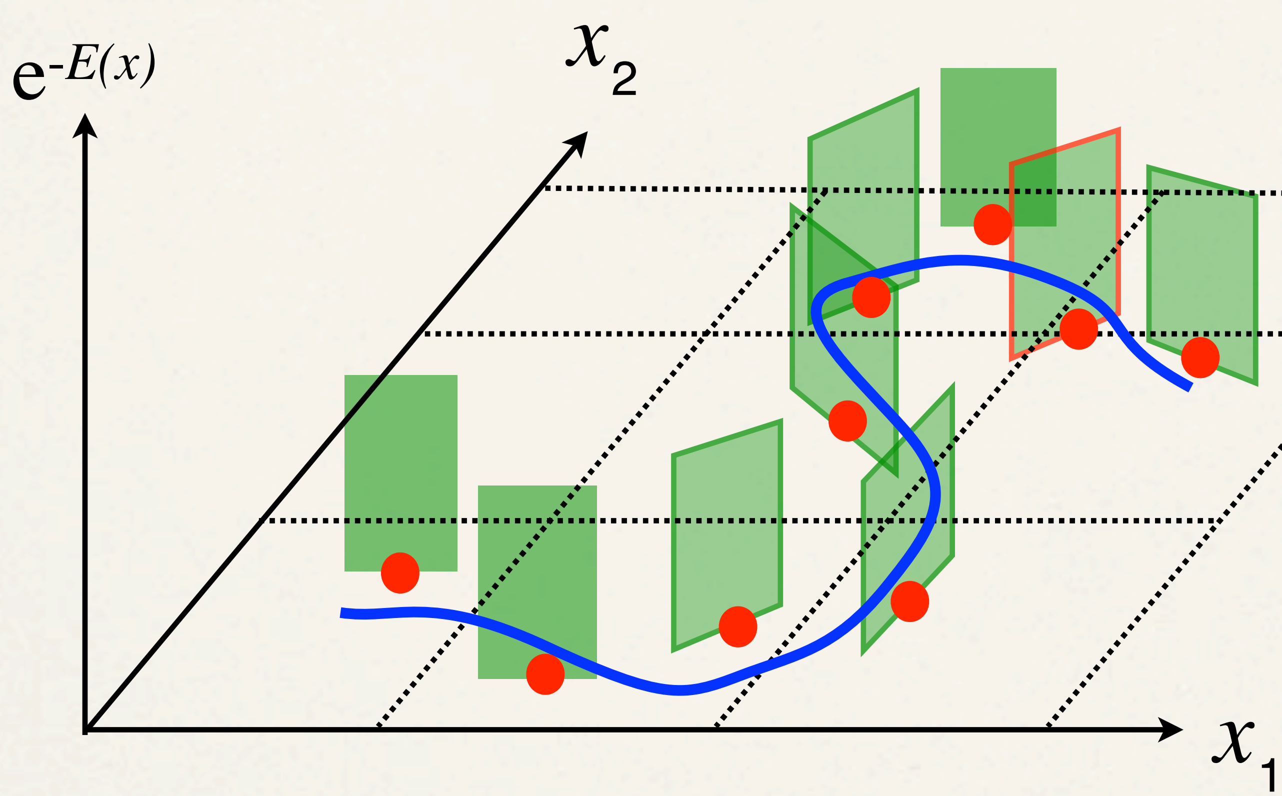


Ex: $\boldsymbol{d_M = 1}$

- Not guaranteed to be oriented along manifold!
- How can this be fixed?

Is this enough?

- Now instead of a Porcupine (dimension 0 peaks) we can get a **Pangolin** (d_M dimensional scales) !!!



Ex: $\boldsymbol{d_M}$**=1**

- Not guaranteed to be oriented along manifold!
- How can this be fixed?

Additional requirement: smoothness

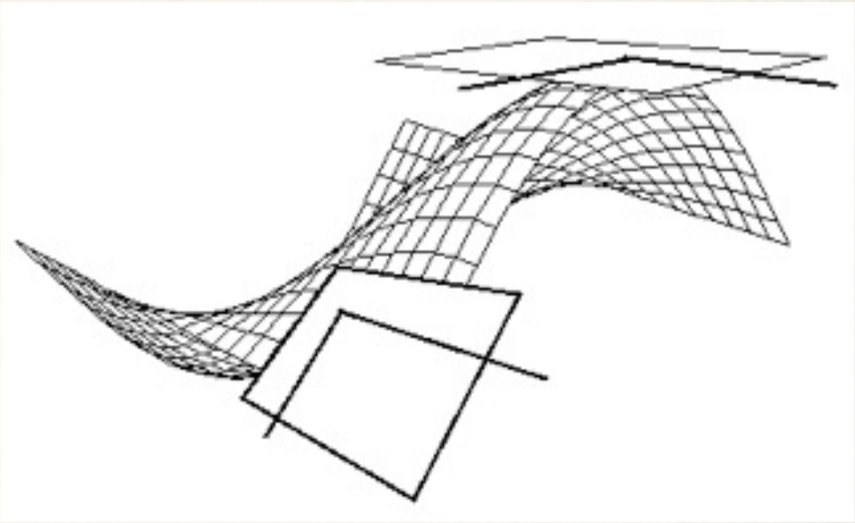
- **Additional requirement:** smoothness of $J_h(x)$ or $H_E(x)$
  - may be (partially?) induced by model parametrization
  - can be added to criterion as in e.g.
    Higher Order Contractive Autoencoders (CAE+H)
    (Rifai, Mesnil, Vincent, Muller, Bengio, Dauphin, Glorot; ECML 2011)

$$\mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2)}\left[\|J_h(x) - J_h(\epsilon)\|^2\right]$$

Additional smoothness penalty

Efficient parametrization of huge tangent spaces patchwork

- Modeling tangent spaces of real world data may require a **huge** patchwork
- Ex: Combinatorial explosion of tangent spaces due to combinatorial possibilities of movements of objects in a scene
- Rather than a big mixture of PPCA or FA (like manifold Parzen) with indep. params. Use a **product of mixtures**!
- These yield gigantic combinatorial mixture with shared parameters.
- => RBMs with low-rank Gaussian covariance parametrizations.



And then what?

Stitching together the patchwork

- Large representations = distributed representations of local tangent spaces.
- Having a good model of tangent spaces...
- How can we go beyond *local* coordinate systems?
- To synthesize more global continuous coordinates?

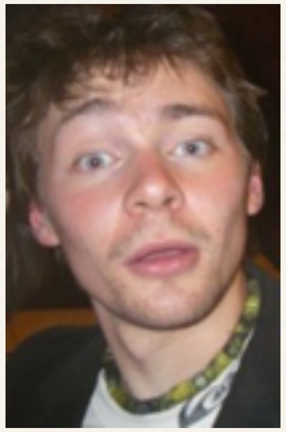
Thank you to past and current students
who did most of the hard work


Hugo Larochelle

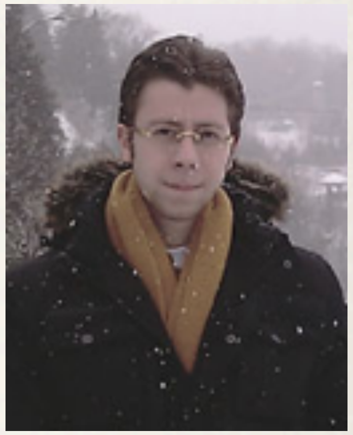

Isabelle Lajoie


Salah Rifai


Xavier Muller


Grégoire Mesnil


Yann Dauphin


Hani Almousli

and to my colleague and mentor


Yoshua Bengio

Questions ?