DATA MINING 2
Deep Neural Networks

Riccardo Guidotti

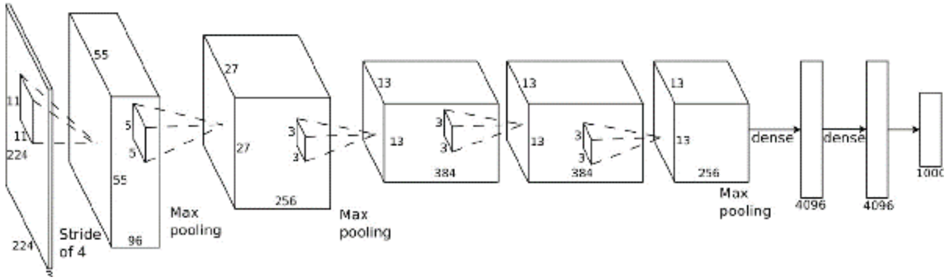
a.a. 2019/2020

UNIVERSITÀ DI PISA

Why Now?

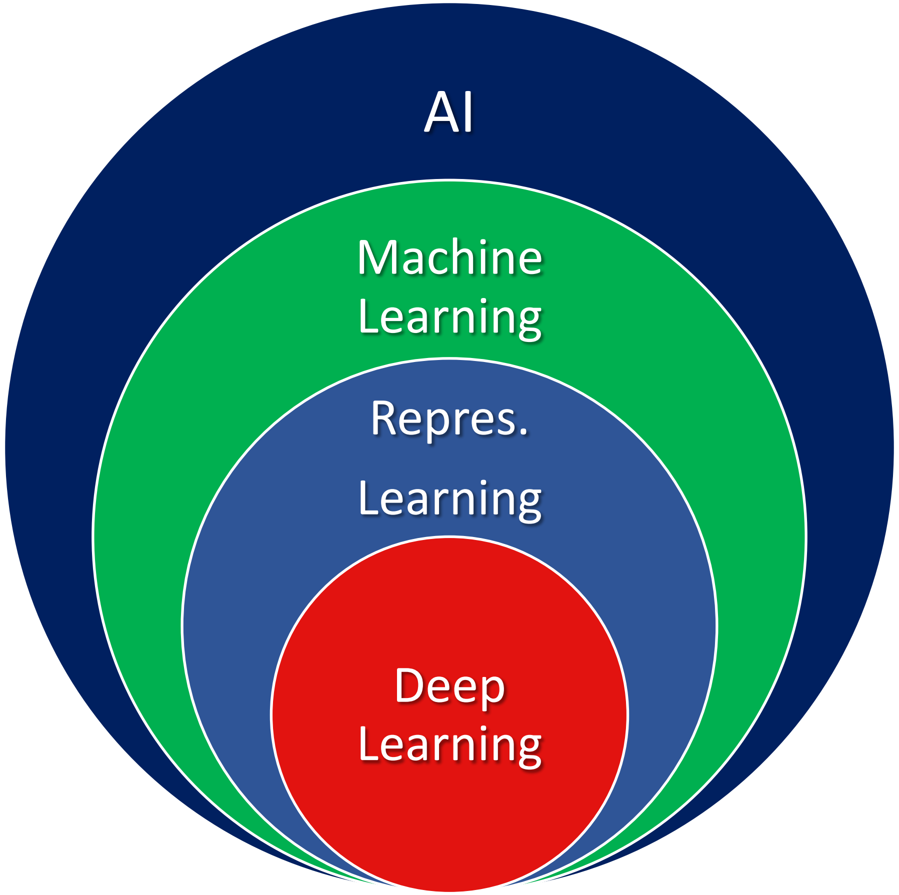
(Big) Data

GPU

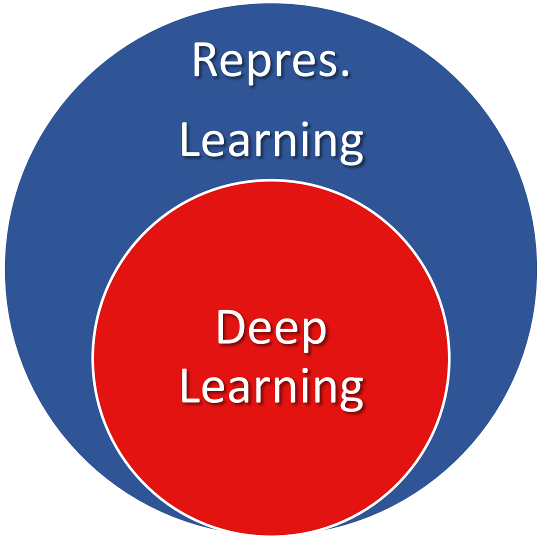


Theory

A quick look on Deep Learning



Deep learning

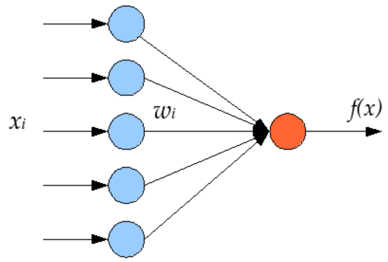


Representation learning methods that

- allow a machine to be fed with raw data and
- to automatically discover the representations needed for detection or classification.

Raw representation

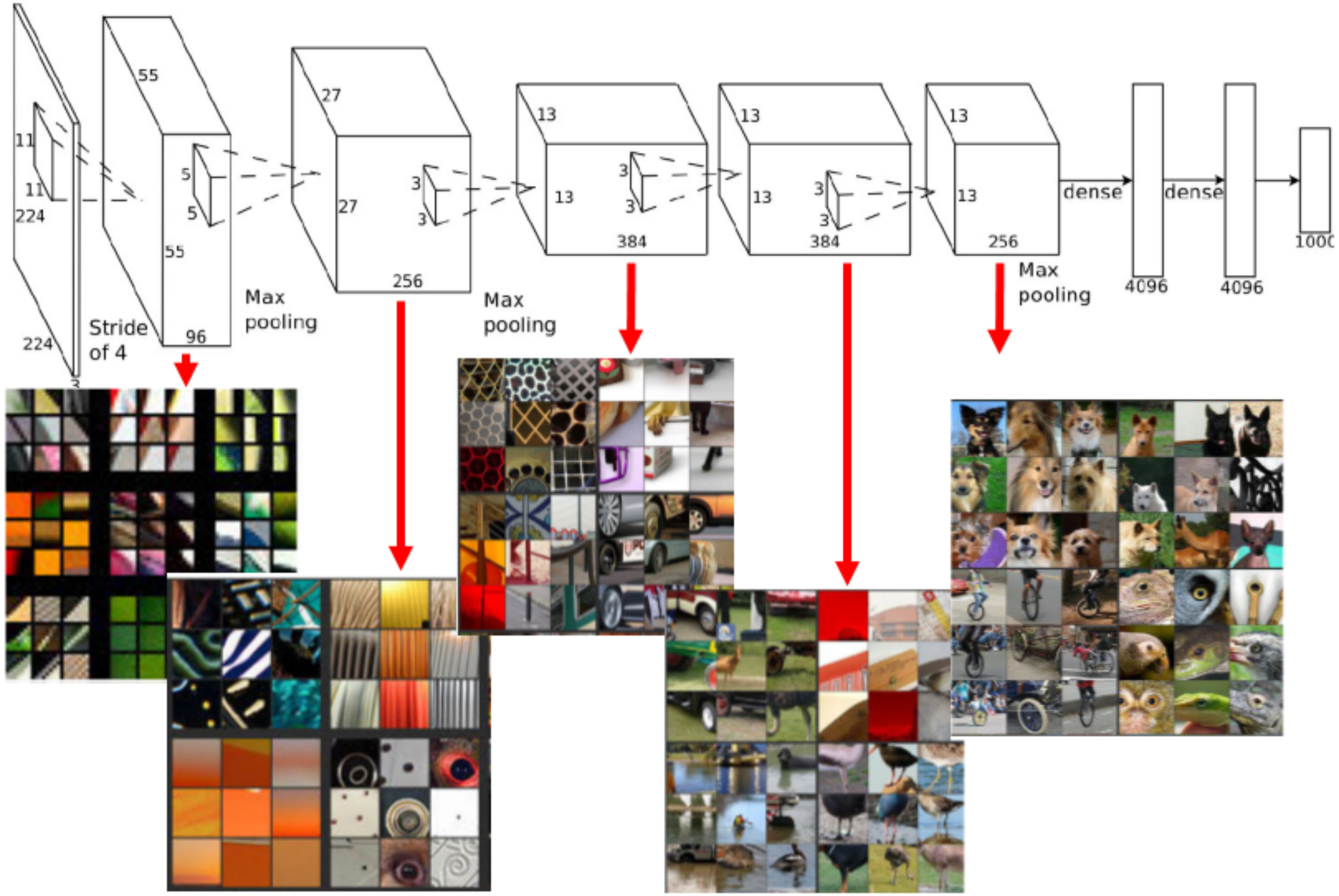
Age	35
Weight	65
Income	23 k€
Children	2
Likes sport	0.3
Likes reading	0.6
Education	high
…	…



Higher-level representation

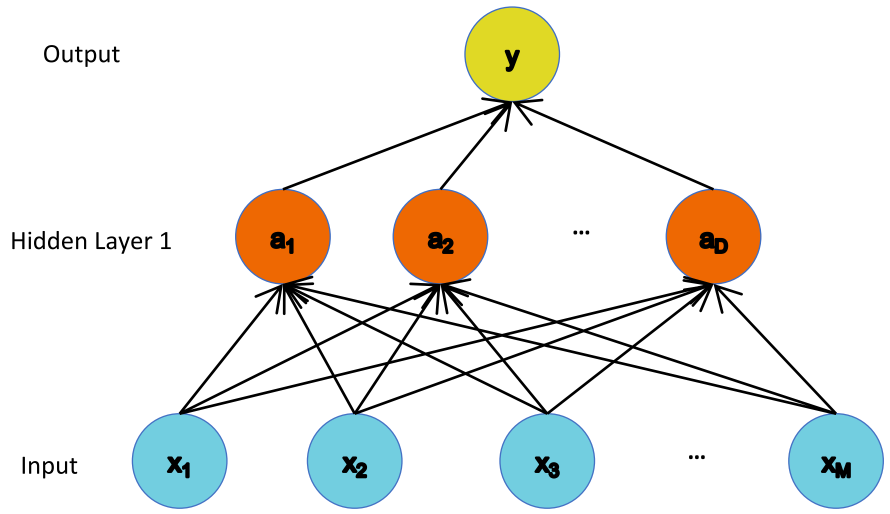
Young parent	0.9
Fit sportsman	0.1
High-educated reader	0.8
Rich obese	0.0
…	…

Multiple Levels Of Abstraction

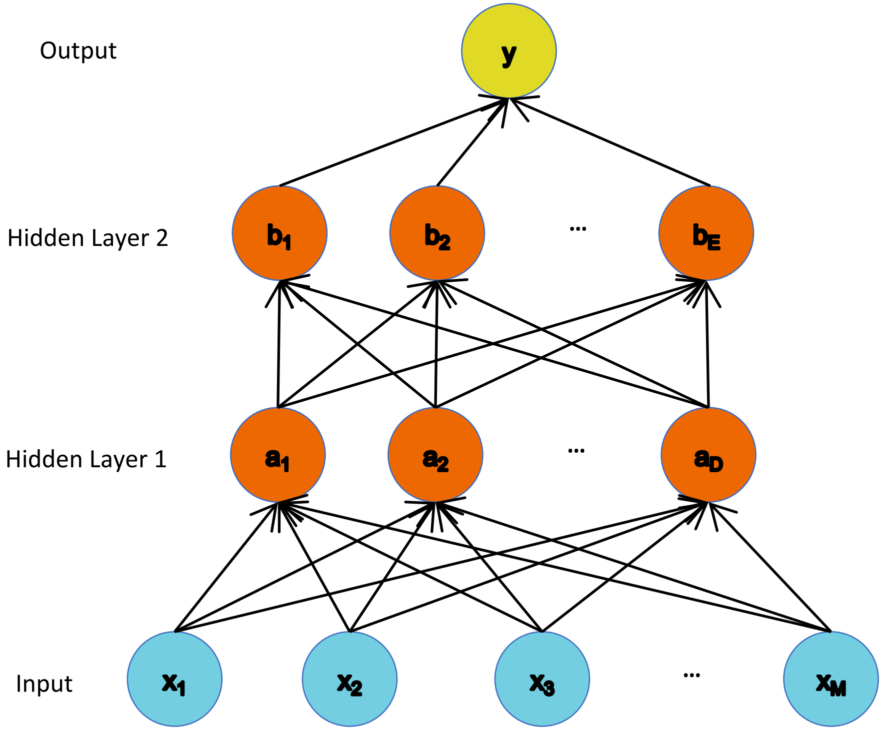


Deep Neural Networks

Deep Neural Networks

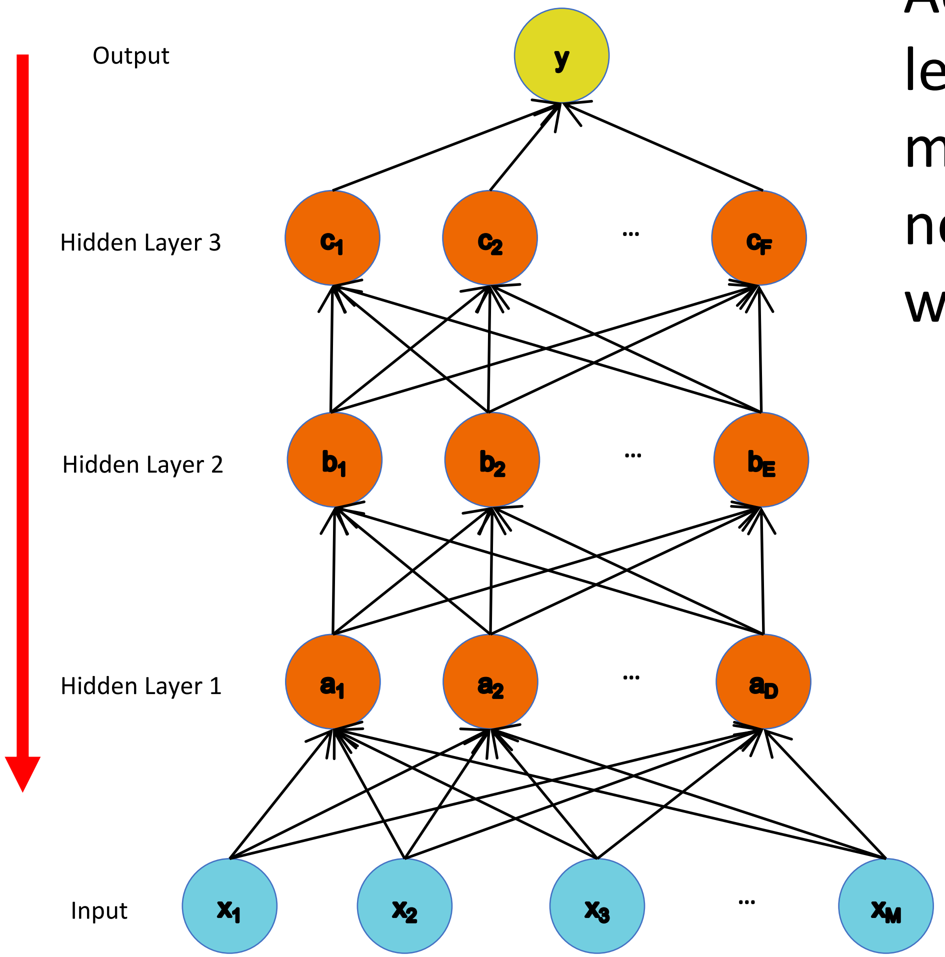


Deep Neural Networks



Deep Neural Networks

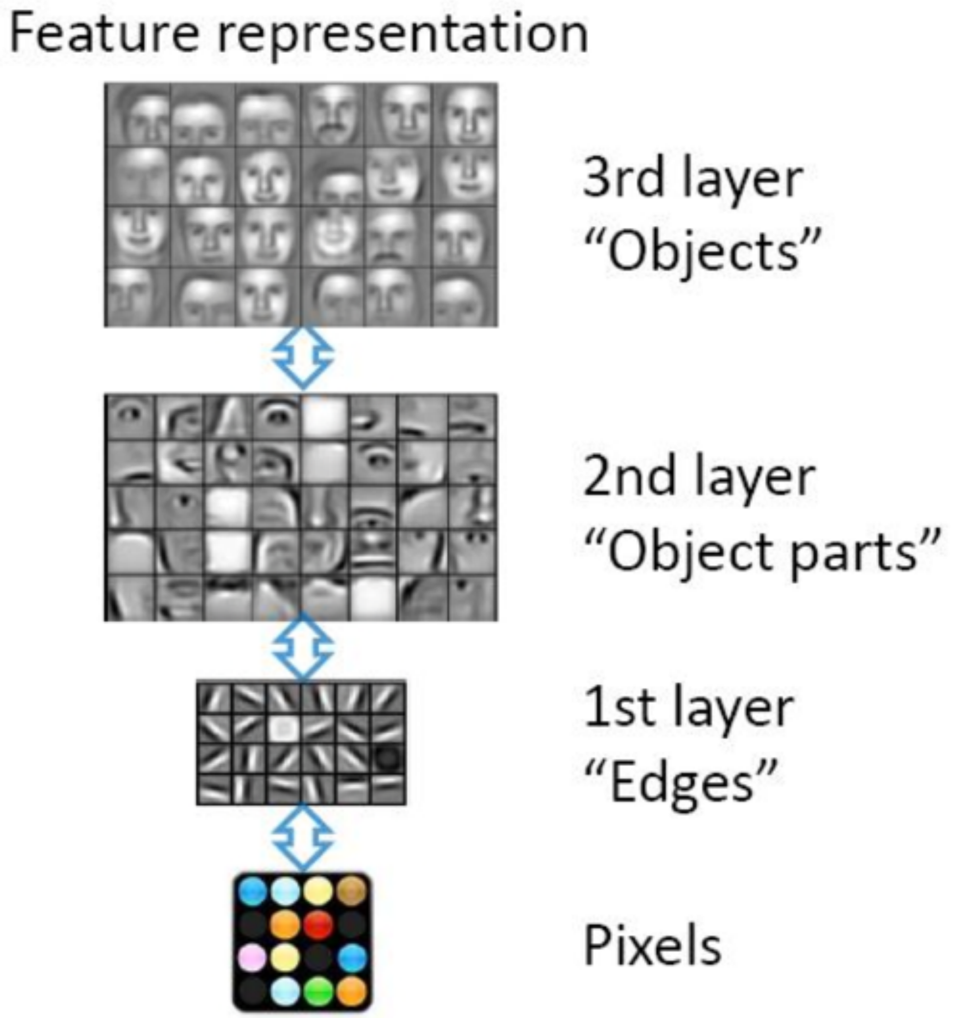
Backpropagation through many layers has numerical problems that makes learning not-straightforward (Gradient Vanish/Esplosion)



Actually deep learning is way more than having neural networks with a lot of layers

Representation learning

- We don't know the "right" levels of abstraction of information that is good for the machine
- So let the model figure it out!

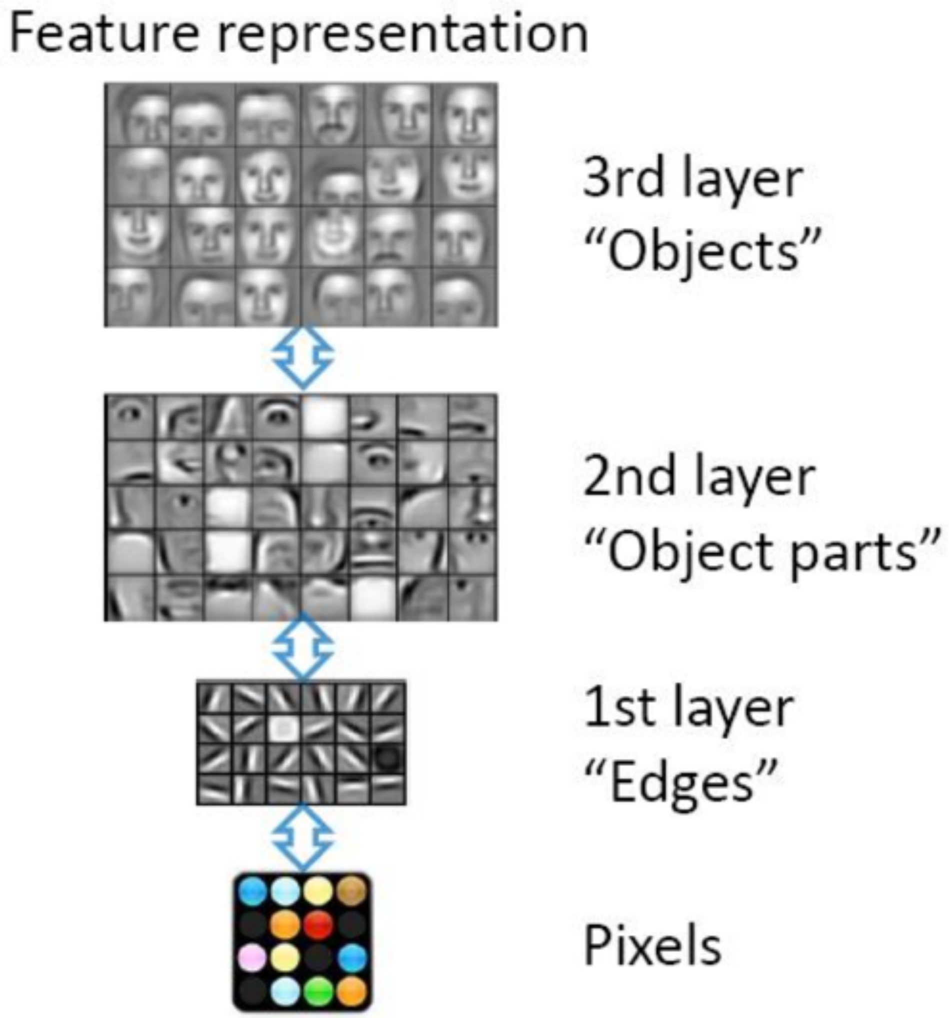


Example from Honglak Lee (NIPS 2010)

Representation learning

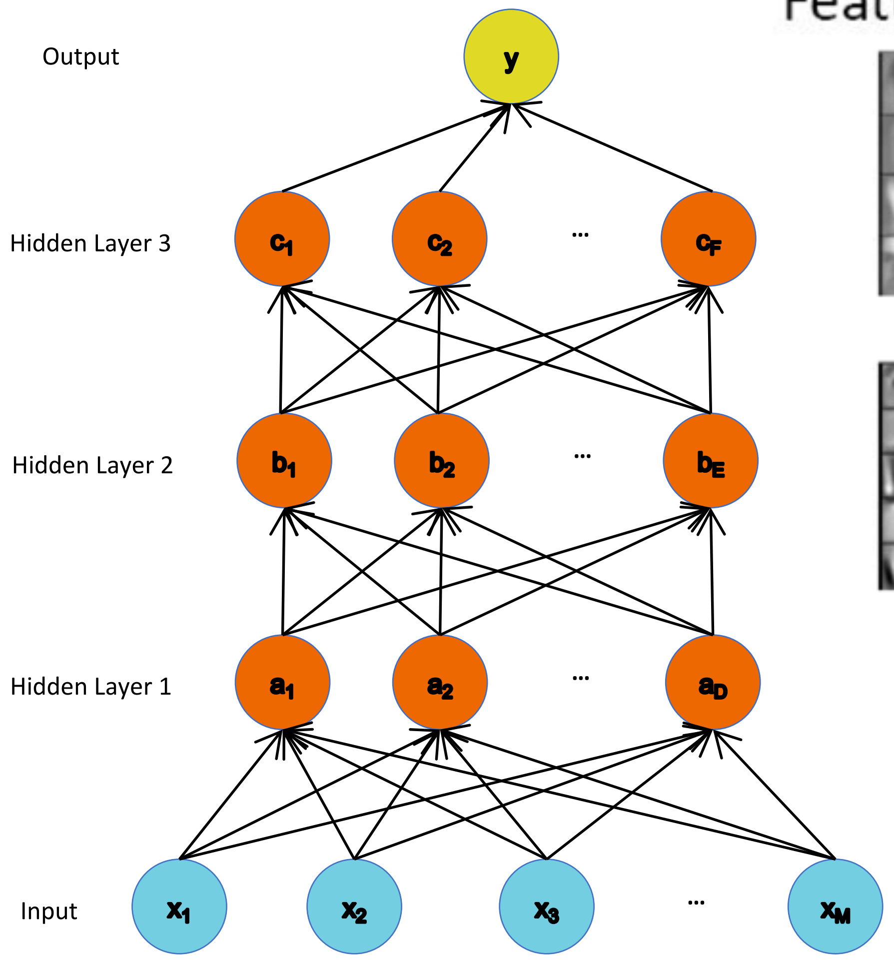
Face Recognition:

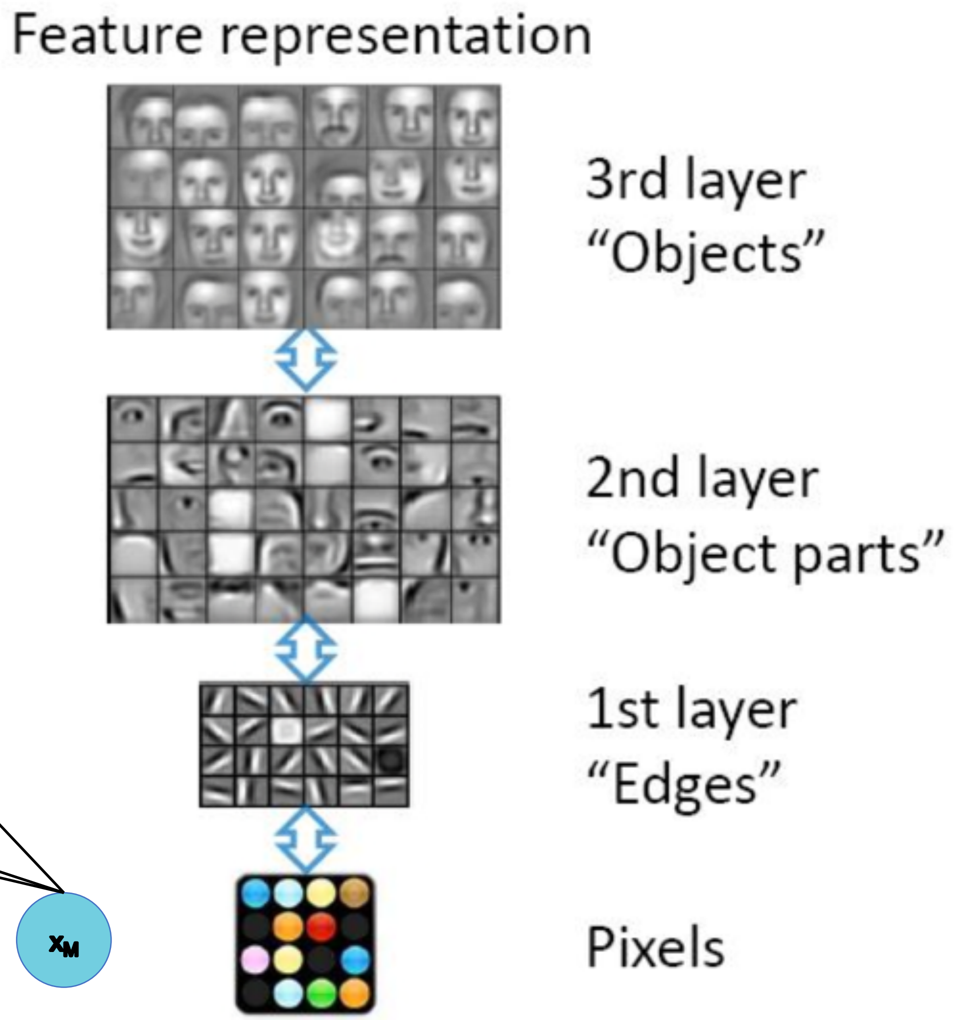
- Deep Network can build up increasingly higher levels of abstraction
- Lines, parts, regions



Example from Honglak Lee (NIPS 2010)

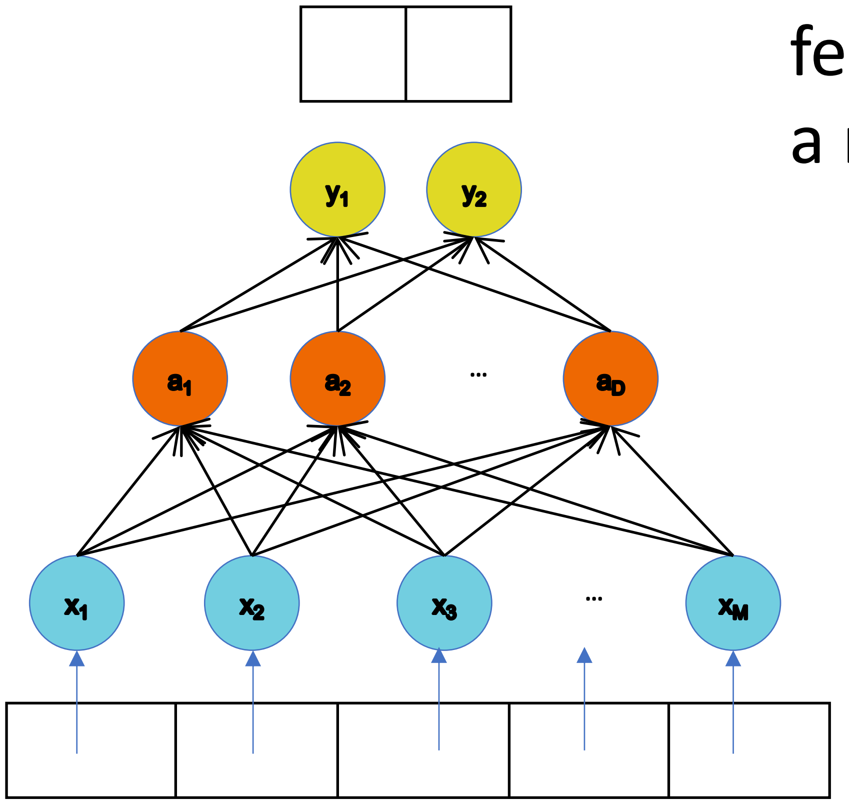
Representation learning





Example from Honglak Lee (NIPS 2010)

Vectorial Data Processing



It is (almost) trivial to feed vectorial data to a neural network

Neural networks can easily handle data of mixed type and distribution

However…

Preparing Vectorial Data for a Neural Network

Categorical Variables

- A categorical variable is a variable that **can belong to one of a number of k discrete categories**
- Categorical variables are usually encoded using **1-out-of k** coding (one hot)
- E.g. for three colors: red = (1 0 0), green = (0 1 0), Blue = (0 0 1)
- If we used red = 1, green = 2, blue = 3, then this type of encoding imposes a representational bias which is not semantically supported

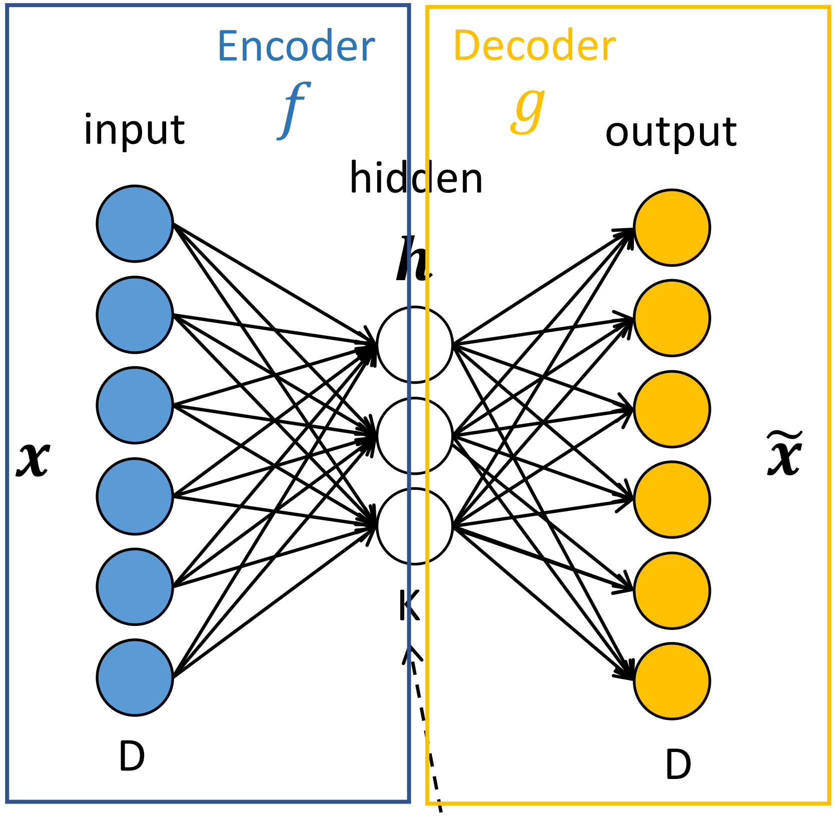
Preparing Vectorial Data for a Neural Network

Continuous Variables

- A continuous variable can be directly fed to a neural network.
- However, it is good practice to normalize data so that the dynamic range of inputs is limited
  - [0,1] normalization (min-max)
  - [-1,+1] normalization
  - Mean 0 and unitary variance (z-score)
  - Population normalization
  - Individual normalization

Autoencoders

Basic Autoencoder (AE)

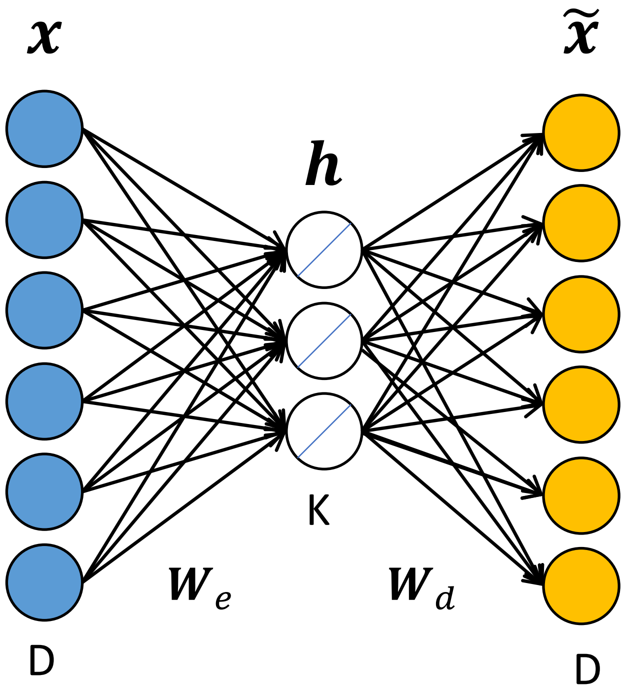


Latent space projection
(again)

- Train a model to **reconstruct the input**
- Passing through some form of **information bottleneck**
  - K << D, or?
  - **h** sparsely active
- Train by loss minimization

$$L(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = L(\boldsymbol{x}, g(f(\boldsymbol{x})))$$

A Very Well Known Autoencoder



Encoding-Decoding

$$\boldsymbol{h} = f(\boldsymbol{x}) = \boldsymbol{W}_e \boldsymbol{x}$$

$$\widetilde{\boldsymbol{x}} = g(\boldsymbol{h}) = \boldsymbol{W}_d \boldsymbol{W}_e \boldsymbol{x}$$

Tied weights (often, not always)

$$\boldsymbol{W}_d = \boldsymbol{W}_e^T = \boldsymbol{W}^T$$

Euclidean Loss

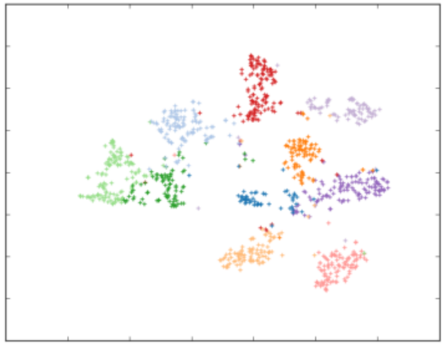
$$L(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) = \|\boldsymbol{x} - \boldsymbol{W}^T \boldsymbol{W} \boldsymbol{x}\|_2^2$$

What if we take f and g linear and K<<D?

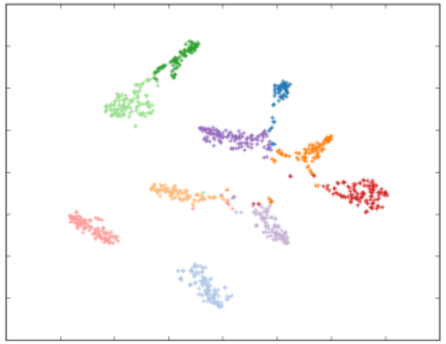
Learns the same subspace of PCA

AE Applications - Visualization

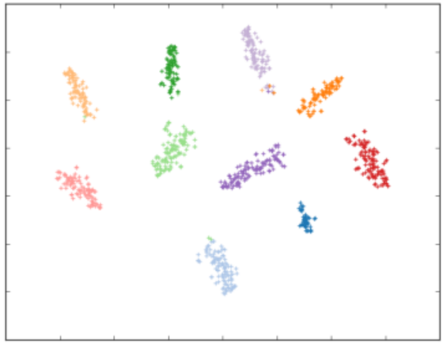
Visualizing complex data in learned latent space



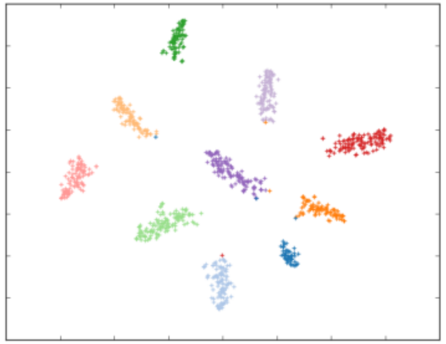
(a) Epoch 0



(b) Epoch 3

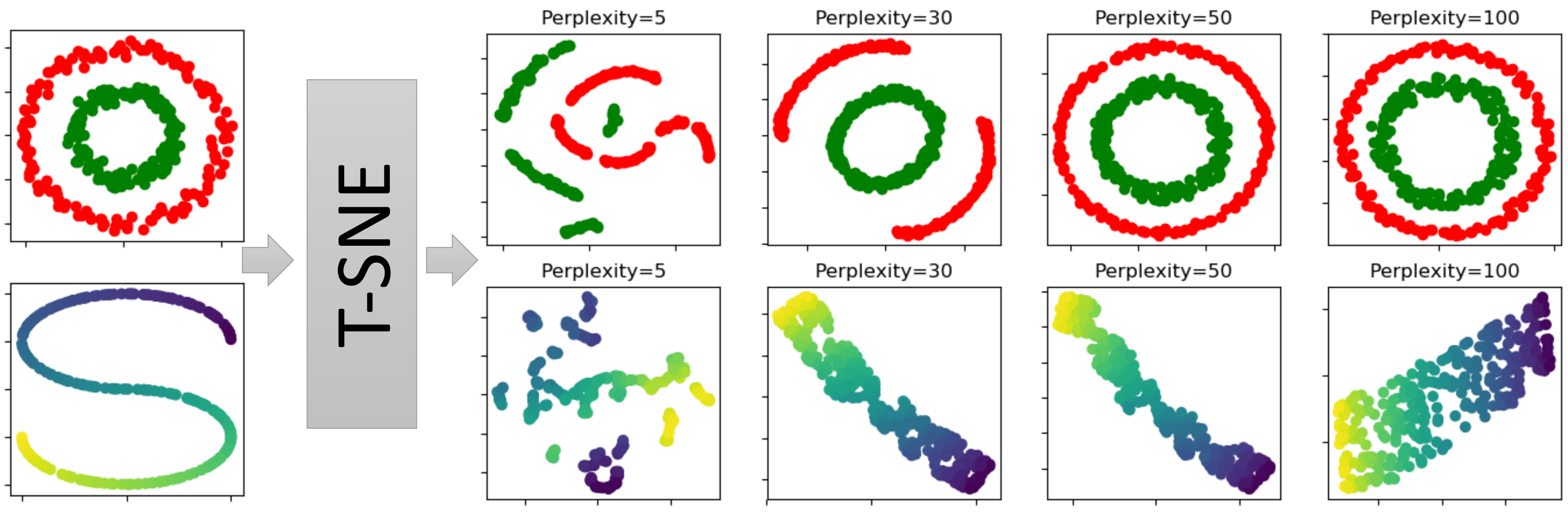


(d) Epoch 9



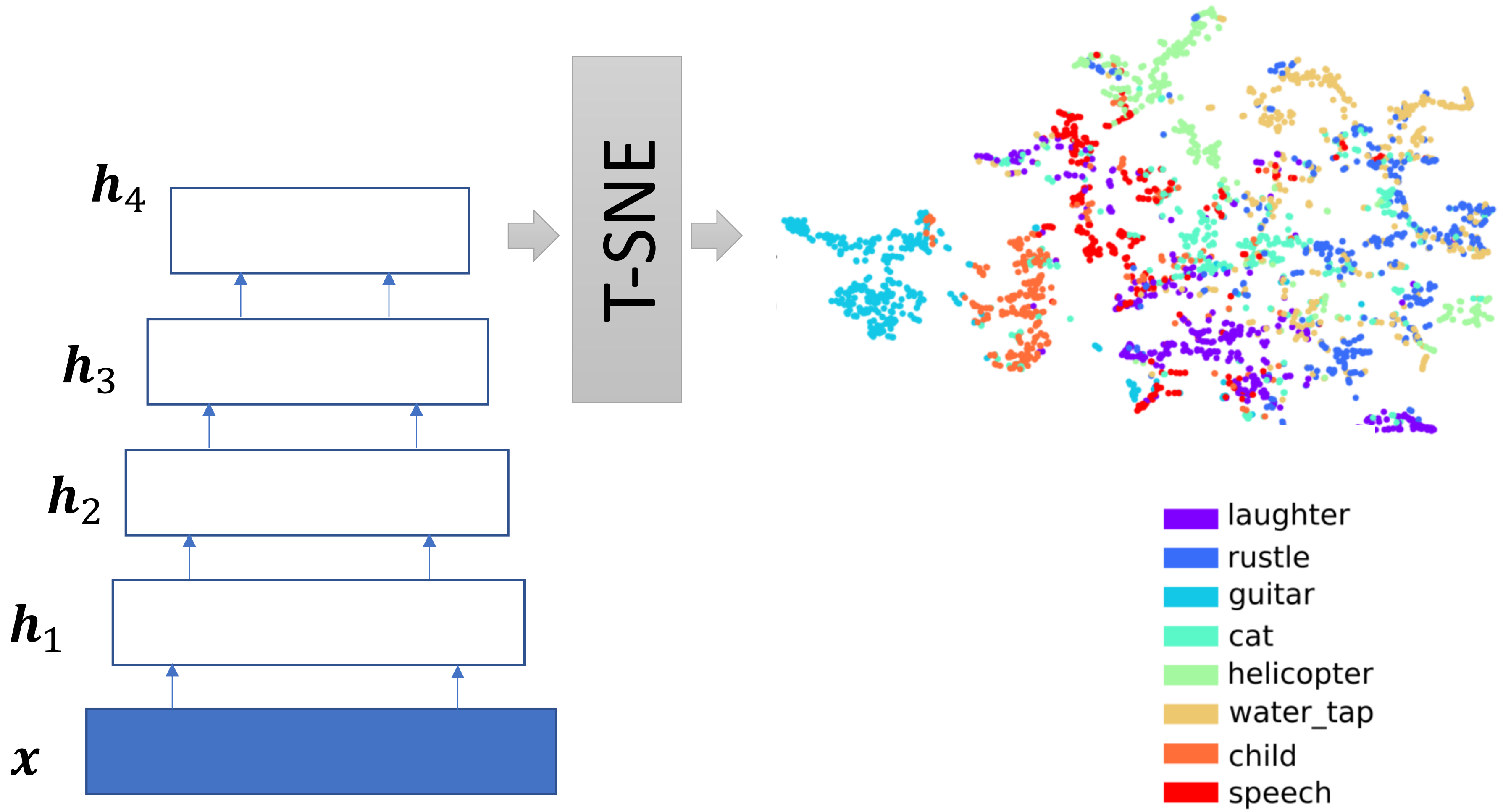
(e) Epoch 12

Visualizing learned neural encoding - t-SNE



https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

Visualizing learned neural encoding - t-SNE



Denoising Autoencoder (DAE)

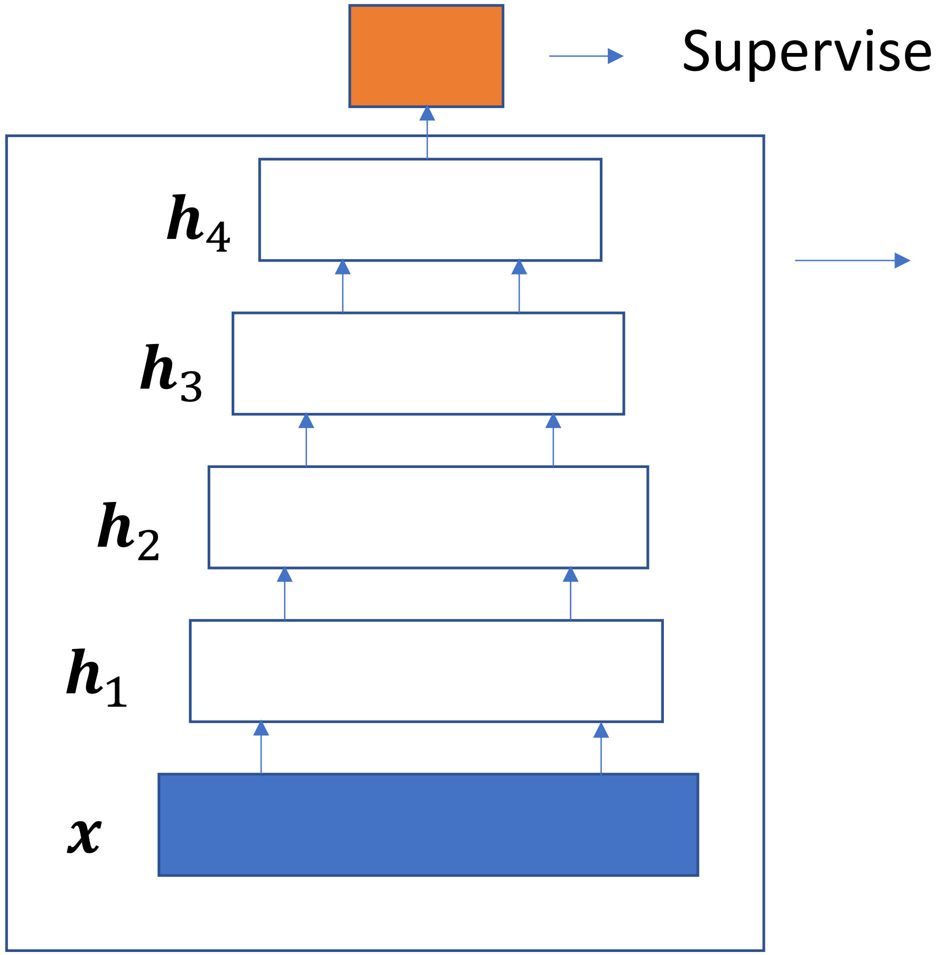
Train the AE to minimize the function

$$L(\boldsymbol{x}, g(f(\widehat{\boldsymbol{x}})))$$

where $\widehat{\boldsymbol{x}}$ **is a version of original input** $\boldsymbol{x}$ **corrupted by some noise** process $\mathrm{C}(\widehat{\boldsymbol{x}}|\boldsymbol{x})$

Key Intuition - Learned **representations should be robust to partial destruction** of the input

Deep Autoencoder



Supervised learning

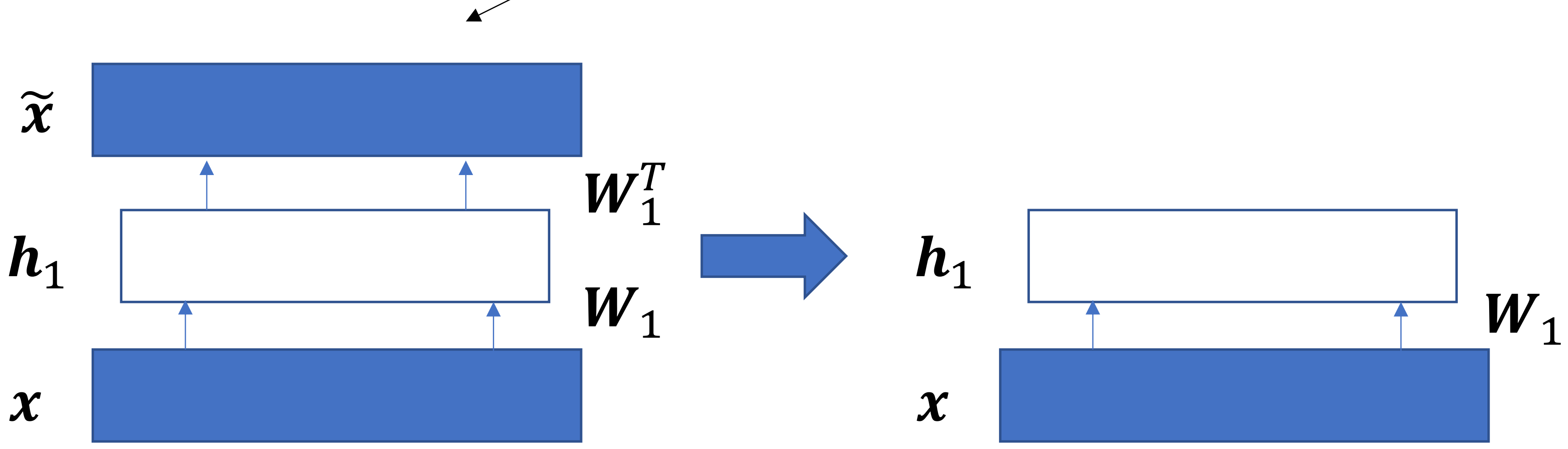
- Unsupervised training
- Hierarchical autoencoder
- Extracts a **representation of inputs** that facilitates
  - Data **visualization**, exploration, indexing,…
  - Realization of a **supervised** task

Tips and Tricks

Unsupervised Layerwise Pretraining

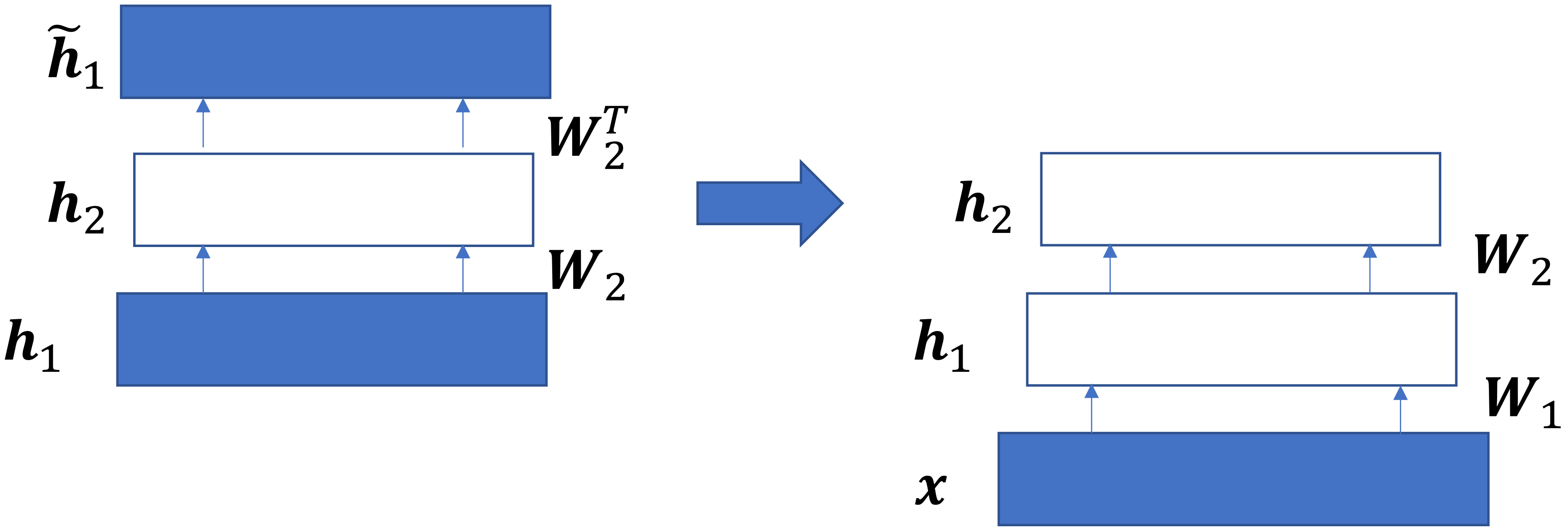
Incremental unsupervised construction of the Deep AE

Any form of AE, e.g. those shown in previous slides



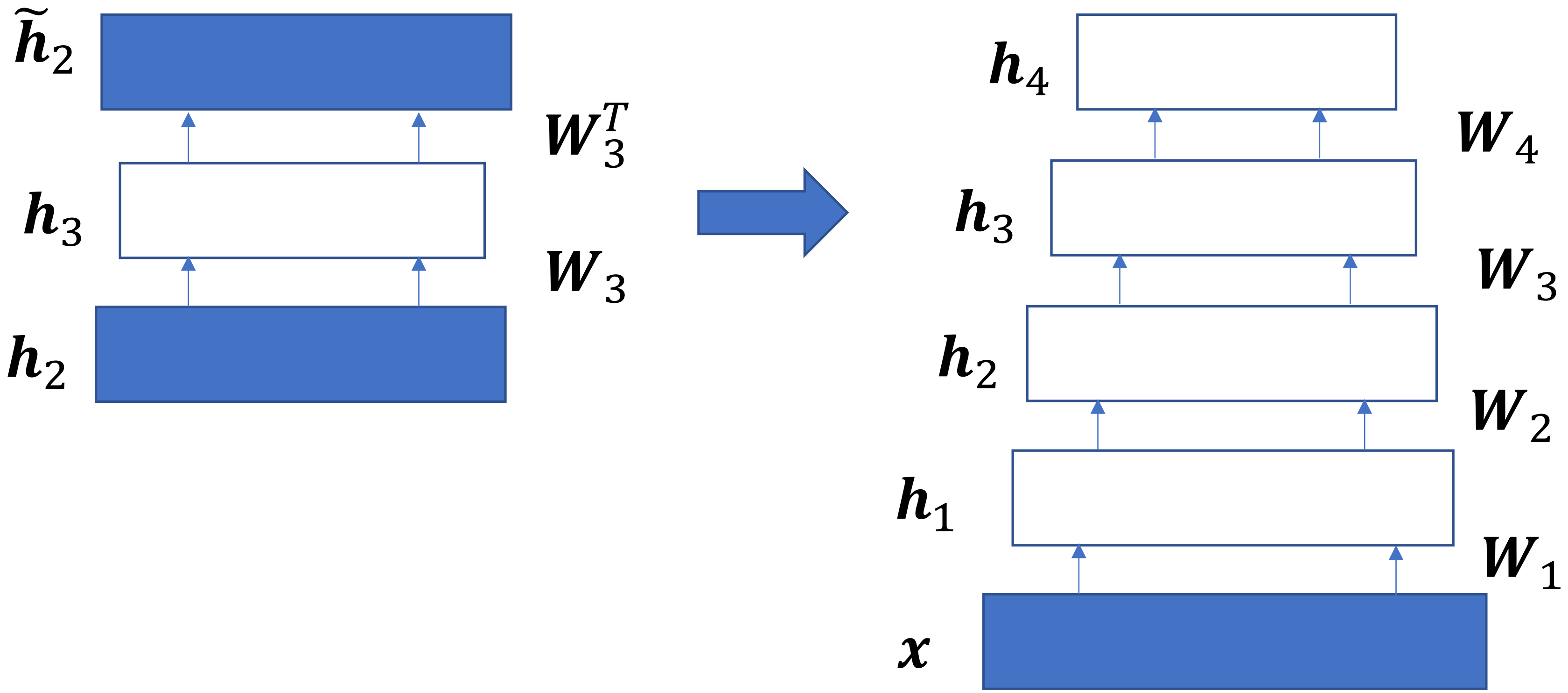
Unsupervised Layerwise Pretraining

Incremental unsupervised construction of the Deep AE



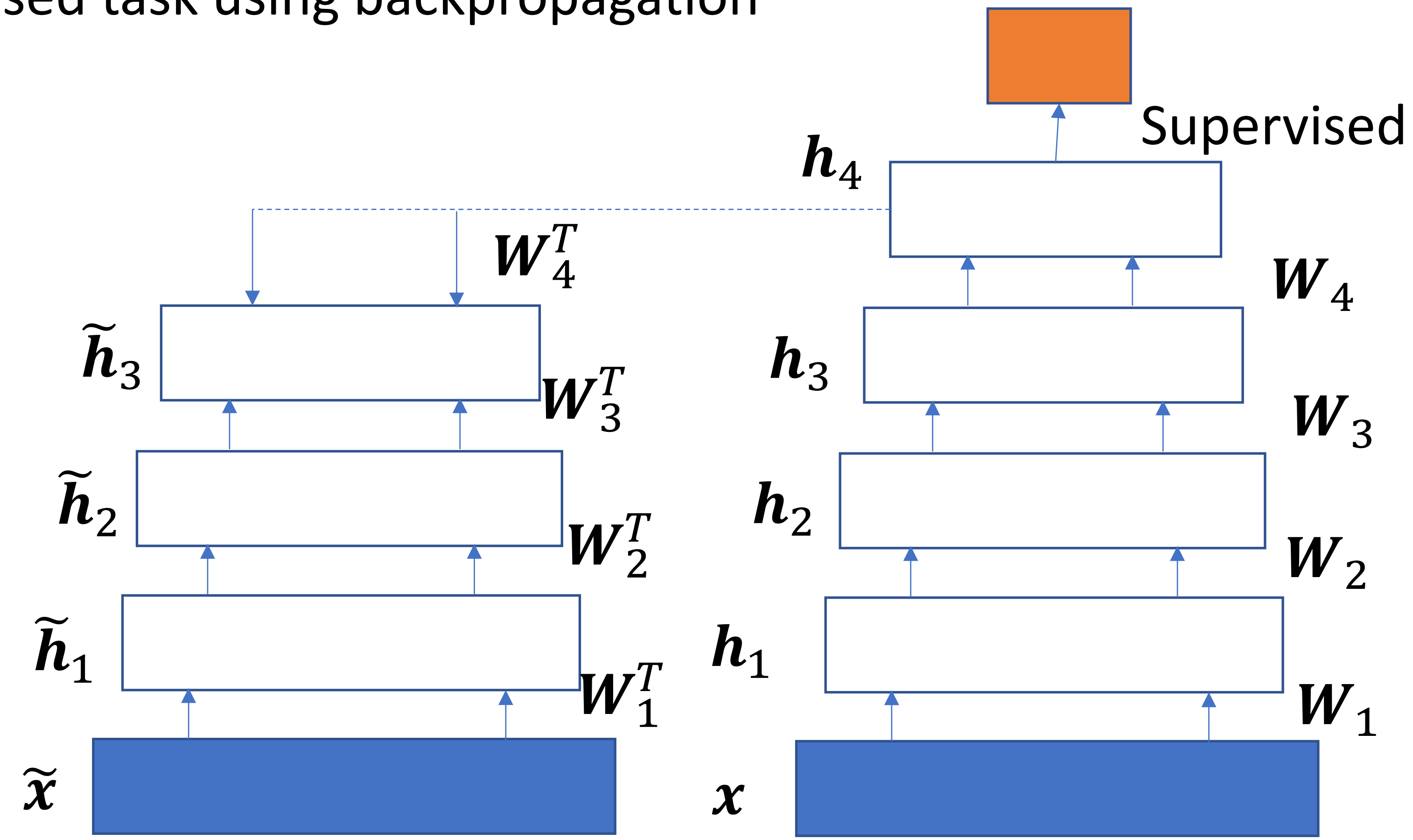
Unsupervised Layerwise Pretraining

Incremental unsupervised construction of the Deep AE



Optional Fine Tuning

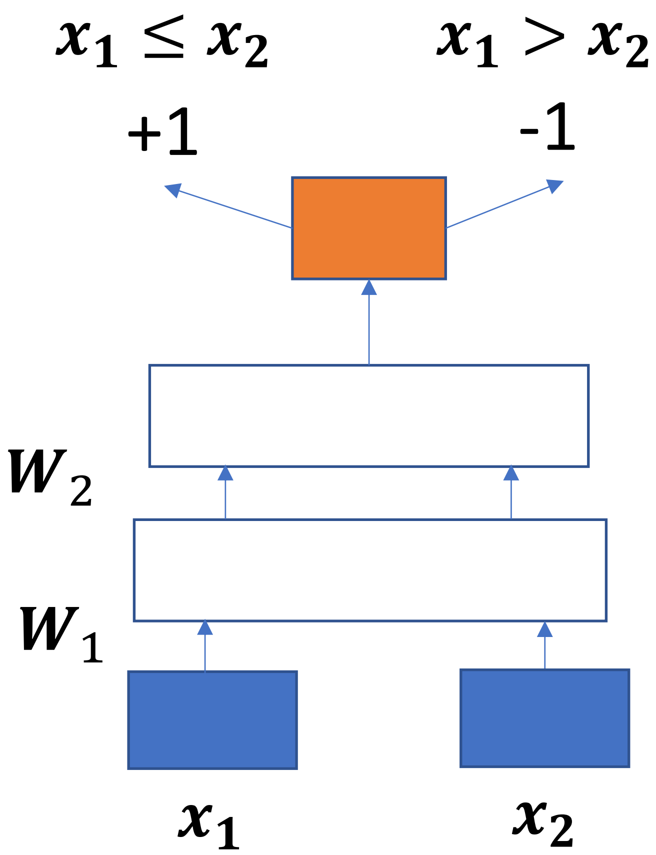
Fine **tune the whole autoencoder** to optimize input reconstruction or a supervised task using backpropagation



Weight Sharing

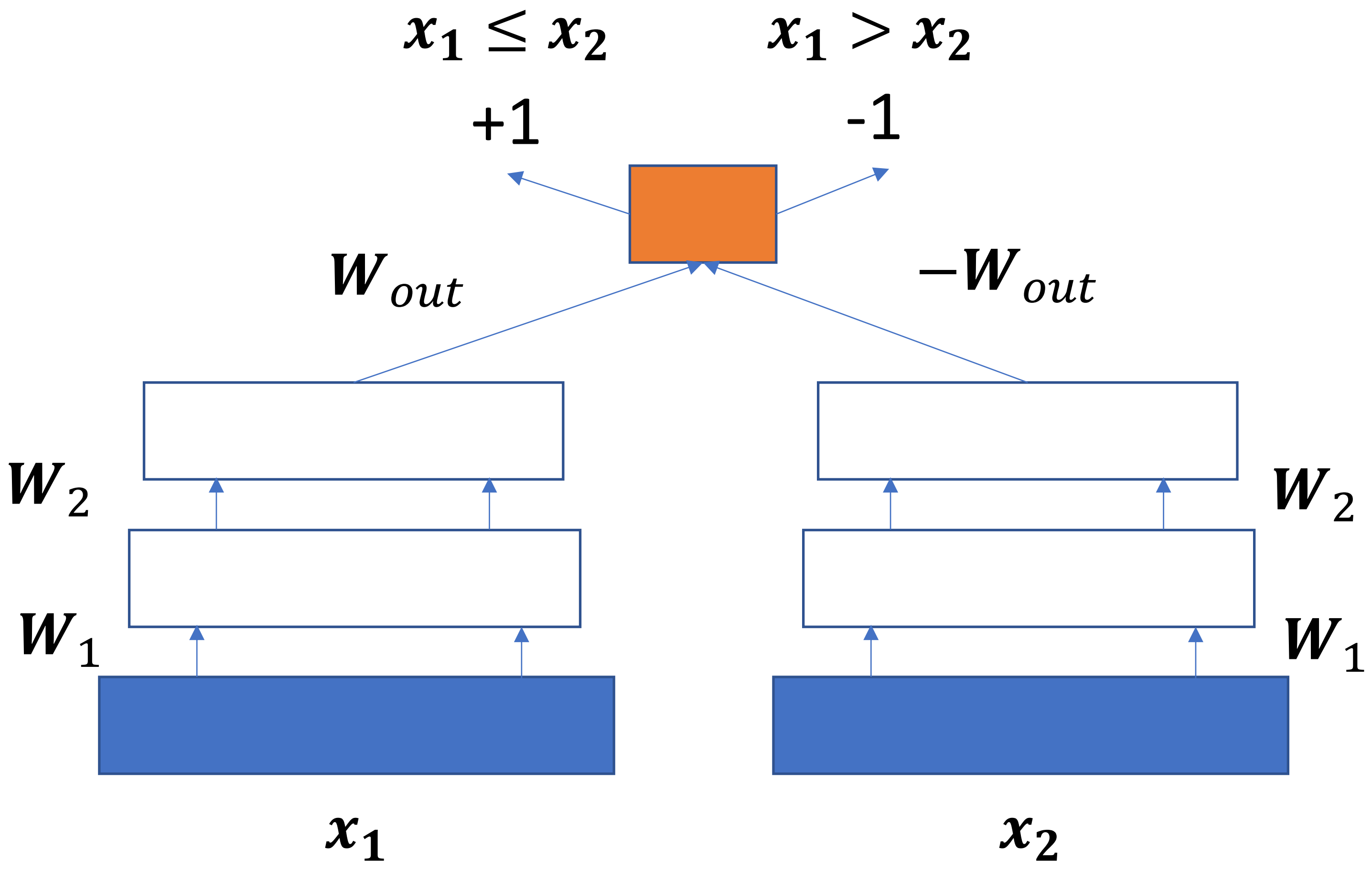
Let us consider a learning to rank task

Possible solution to pairwise ranking



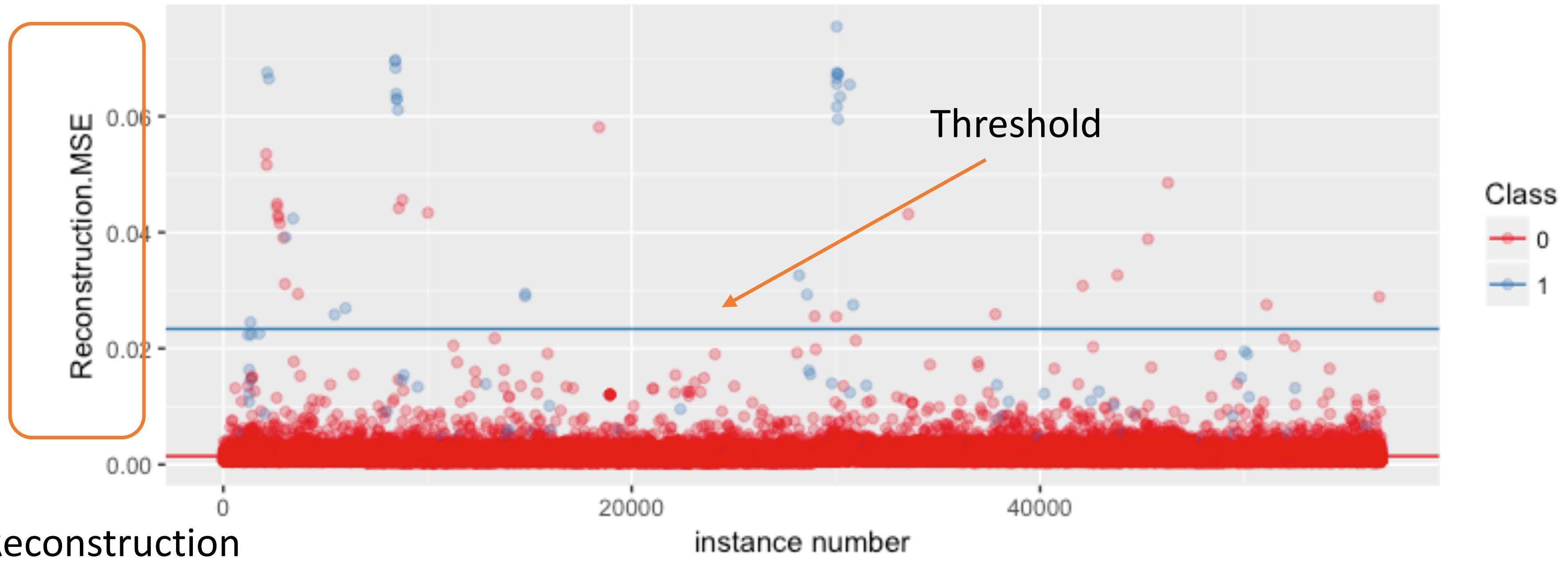
Weight Sharing

Weight sharing architecture inspired by prior knowledge on the task (i.e. simmetry)



Applications

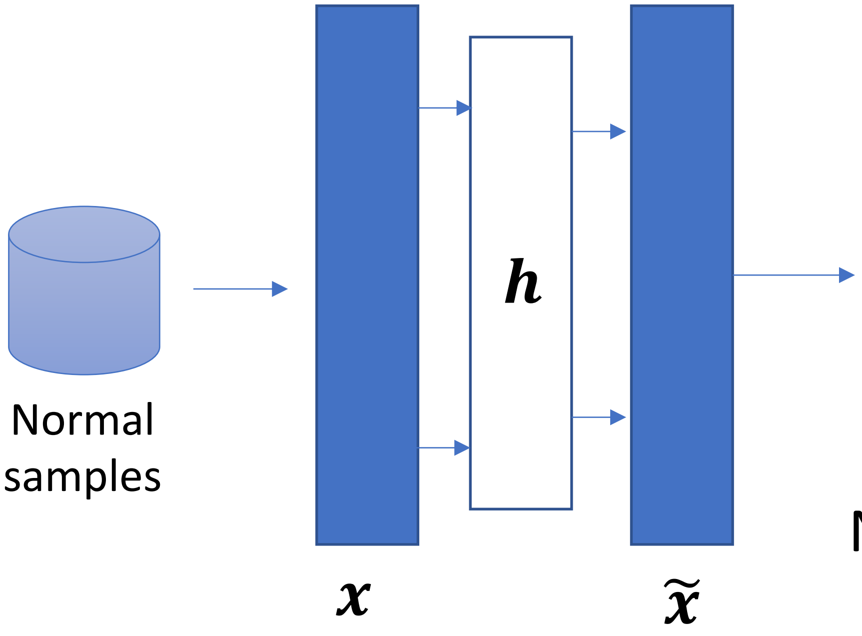
Anomaly Detection



Reconstruction

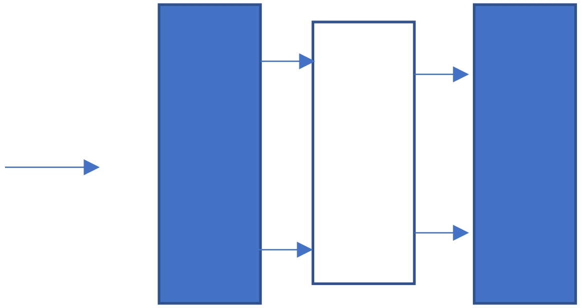
So, how do we realize it?

Using autoencoders of course…



Statistics on the «normal» differences between $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$

Normal discrepancy threshold $\boldsymbol{\theta}$



$d(\boldsymbol{x}, \tilde{\boldsymbol{x}}) > \theta$ WARNING

$d(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \leq \theta$ OK

References

- Artificial Neural Network. Chapter 5.4 and 5.5. Introduction to Data Mining.
- Hands-on Machine Learning with Scikit-Learn, Keras & Tensorflow. A practical handbook to start wrestling with Machine Learning models (2nd ed).
- Deep Learning. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. The reference book for deep learning models.