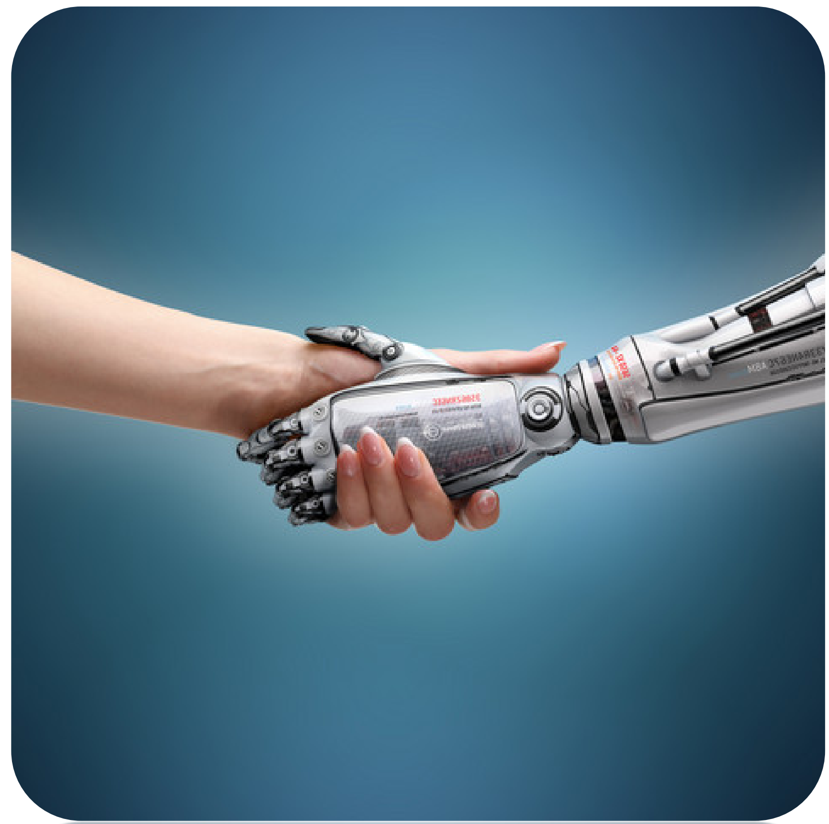
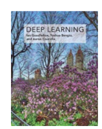
Zurich University
of Applied Sciences

Artificial Intelligence
V09: Unsupervised Learning with Autoencoders

Flavors of Unsupervised Learning
Autoencoders
Use cases for autoencoders



With material from Goodfellow et al., “Deep Learning”, ch. 14, 2016



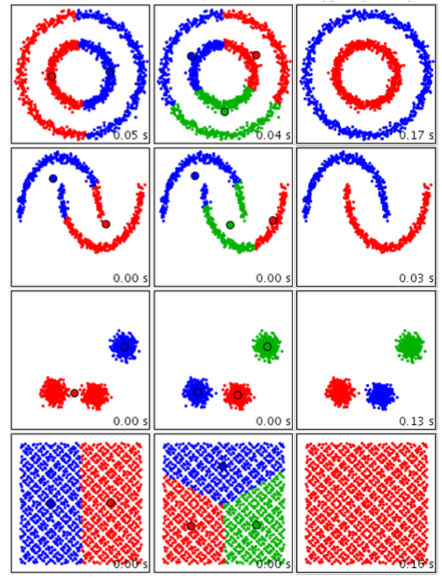
Zurich University of Applied Sciences and Arts
CAI Centre for Artificial Intelligence (stdm)

Educational objectives

- **Know** the breadth and **significance of unsupervised learning**
- **Understand how autoencoders learn** important facts about the structure of the **underlying** data-generating **distribution**
- **Be able to propose** unsupervised learning **schemes for** real-world problems like **predictive maintanance**

1. FLAVORS OF UNSUPERVISED LEARNING

Inductive unsupervised learning
Clustering and beyond



Left: Effect of density- vs. distance-based similarity. From left to right: K-Means (K=2), K-Means (K=3), DBSCAN (eps=.1, min=3)

Bottom: Problem of parameter choice in fitting a number of Gaussians to data. Top left to bottom right: True mixture (3), K=30, 14, 8, 3, 1

"Usual" task: Clustering

- N Examples are described by feature vectors $\vec{x}_i, \mathrm{i} = 1..\mathrm{N}$ alone (no labels)
- The examples naturally fall into K groups; K and the group membership function $f(x) = y, y \in 1..K$ are unknown

Challenges

a form of inductive bias!

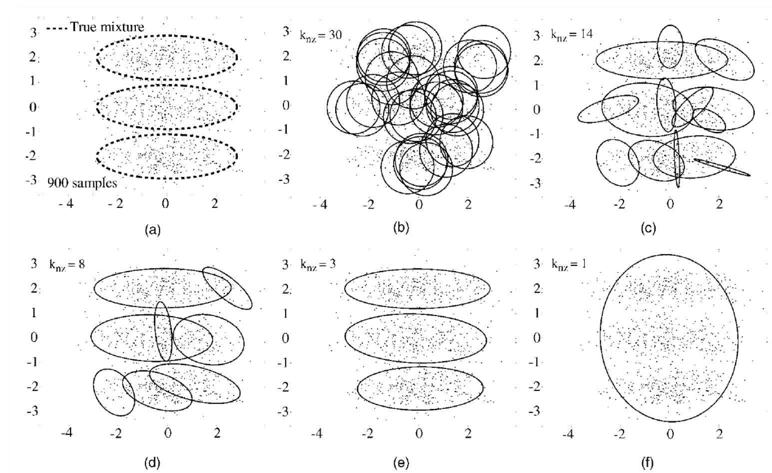
- Similarity by **distance** and/or **density**?
- Choice of **parameters** (i.e., range of K)

Other tasks

Also called «latent factors» or «hidden variables»

- Discovery of unobserved variables
- Dimensionality reduction
- **Feature learning (e.g. autoencoders)**
- Matrix completion (e.g. for recommendation)
- Discovery of dependency structure in features (graph analysis)

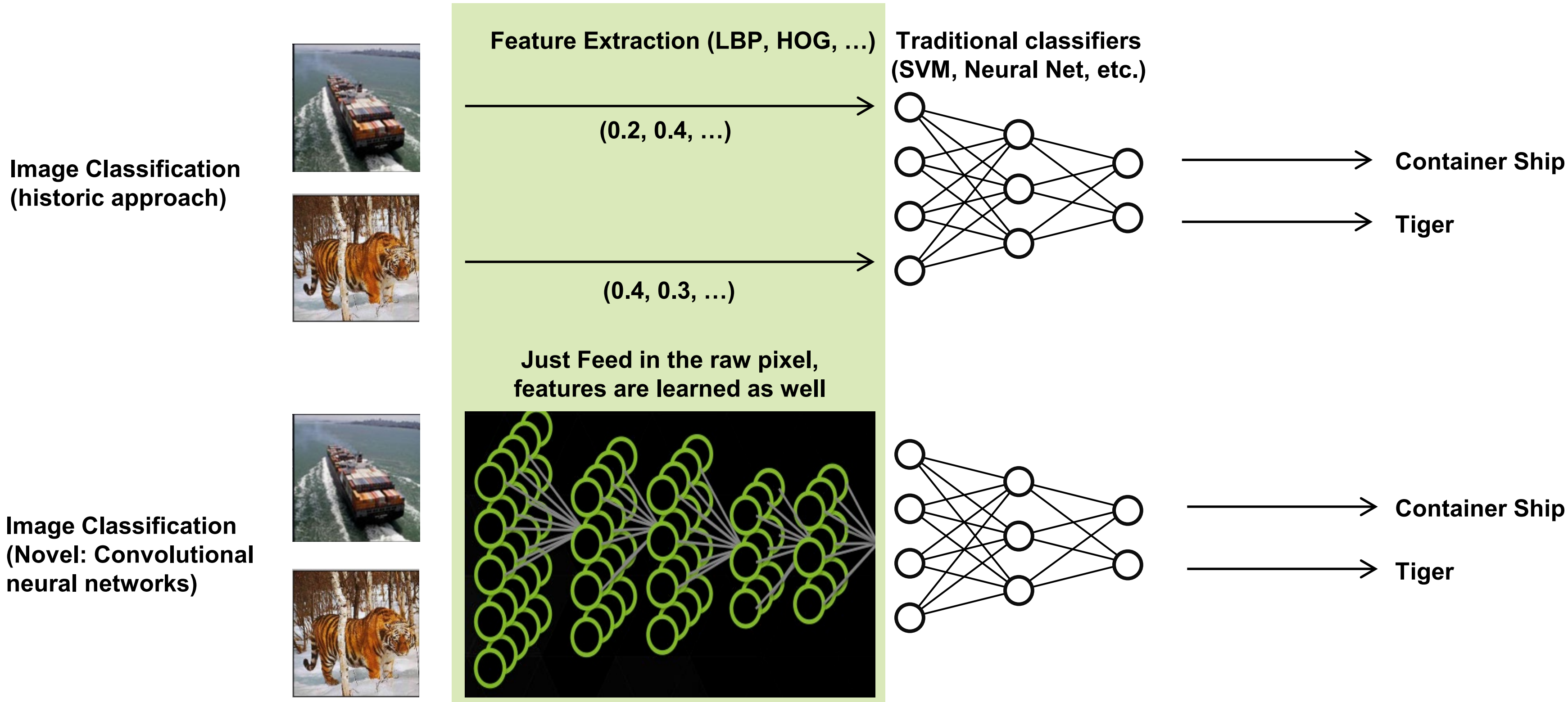
not graph**ics**!



Recap: Deep learning rationale
Key idea «feature learning» in CNNs

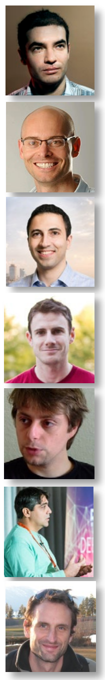
Towards learning representations rather than specific functions

- Convolutional/pooling layers are «feature extractor», dense layers are «classifier»



Unsupervised learning is trendy

What To Expect from Deep Learning in 2016 and Beyond?*



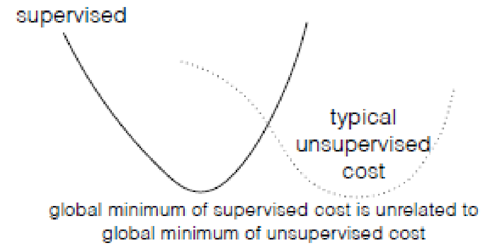
- Ilya Sutskever, OpenAI: «*…**substantial advances in unsupervised learning**.*»
- Pieter Abbeel, UC Berkeley: «*…significant advances in **deep unsupervised learning**…*»
- Eli David, Deep Instinct: «*Specifically, I think the most promising area will be unsupervised learning, as **most of the data in the world is unlabeled**, and **our own brain's neocortex** is primarily a very good unsupervised learning box.*»
- Daniel McDuff, Affectiva: «*I expect that more focus will be given to unsupervised training **and/or semi-supervised** training algorithms, as the amount of the data only continues to increase.*»
- Jörg Bornschein, CIFAR: «*I expect that unsupervised, semi-supervised and reinforcement-learning approaches will play much more prominent roles than today. When we consider machine learning as a component in larger systems, e.g., in robotic control systems or as parts that steer and focus the computational resources of larger systems, it just seems obvious that **purely supervised approaches are conceptually too limited** to appropriately solve these.*»
- Koray Kavukcuoglu & Alex Graves, **Google DeepMind**: «*We expect both unsupervised learning and reinforcement learning to become more prominent.*»

➔ 50% of interviewees expect nearby breakthroughs in unsupervised learning

*) from a 2016 KDNuggets interview: → see http://www.kdnuggets.com/2016/01/deep-learning-2016-beyond.html

Is unsupervised learning broken?

Insights from "Towards principled unsupervised learning" by Sutskever, Josefowicz, Gregor, Rezende, Lillicrap and Vinyals, 2016

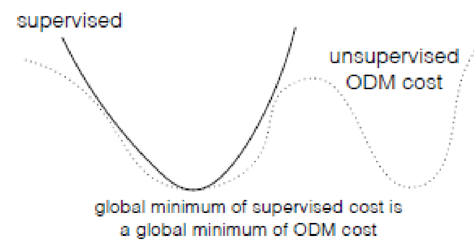


Observation

- Unsupervised learning (UL) is less employed than supervised (SL)
  ➔ because it is less successful?

Problem

- UL cost function unclear



Reason

- **UL often used to improve SL** in the absence of enough labeled data
- But without labels, UL cost **doesn't know which SL task to focus on**

Solution: **Output Distribution Matching (ODM) cost** function

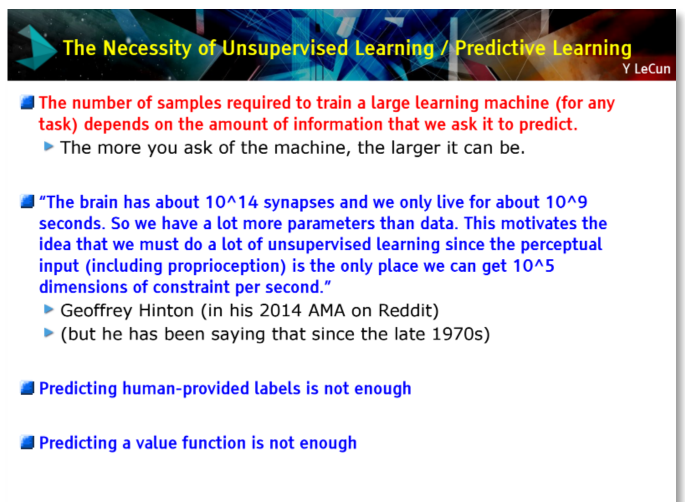
- SL maps data X to labels Y via $Y = F(X)$, $(X, Y) \sim D$
- Impose constraint on F using **uncorrelated** samples $x \sim D, y \sim D$: $Distr[F(x)] = Distr[y]$
- Use it as UL cost function: $KL(Distr[y] \,||\, Distr\,[F(x)])$

➔ cost works towards matching distribution of inferred labels to the one in known (x, y) pairs
➔ high chance of practically improving SL if ODM cost can be optimized

Beyond improving supervised learning
Unsupervised learning and it’s role towards AI

From Yann LeCun’s NIPS’2016 keynote (→ see also appendix)

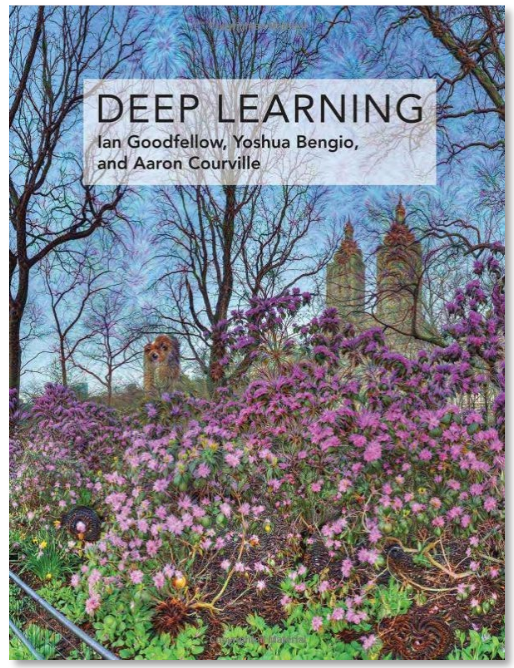
- Unsupervised learning fills the gap to **«common sense»** by implicitly learning «how the world works»
- It does so by **«self-supervised»** learning: predicting former/future/missing pieces of itself
- It is thus the **major workhorse of machine learning** («the cake»)



Zurich University
of Applied Sciences

2. AUTOENCODERS

Based on ch. 14 of «Deep Learning» by Goodfellow, Bengio and Courville 2016

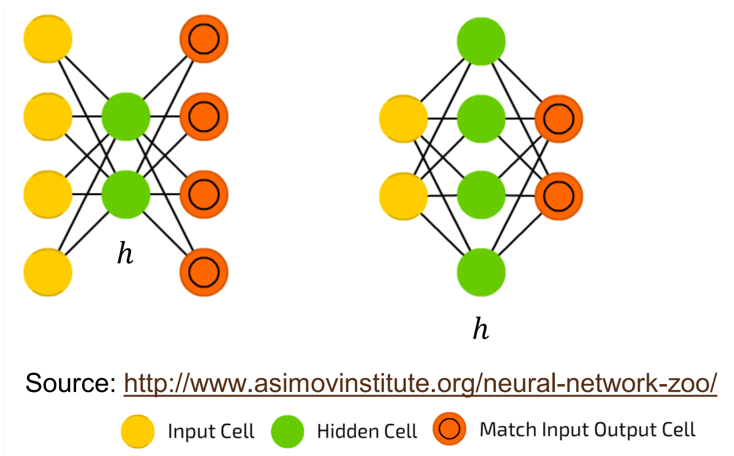


Autoencoders (AE)

«We hope that training the autoencoder to perform the input copying task will result in h taking on useful properties»

Definition

- A model (e.g., neural network) trained to **copy its input to its output**
- But: designed to be **unable** to learn **to copy perfectly**
  ➔ **forced to prioritize** which aspects to copy



Source: http://www.asimovinstitute.org/neural-network-zoo/

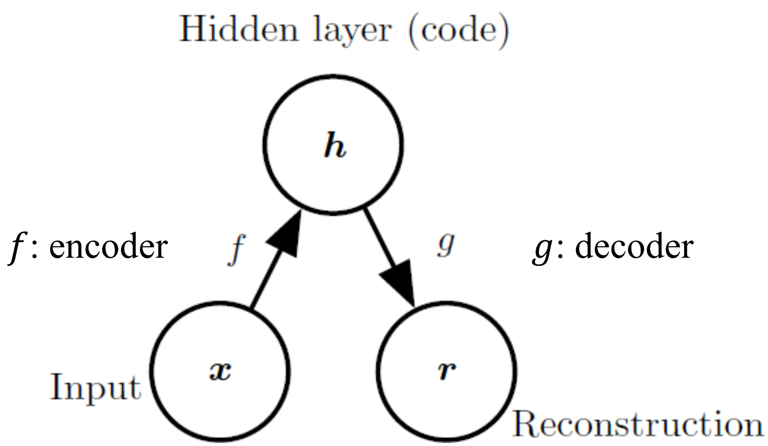
Desired effect

- **Learn useful properties** of the data

Application scenarios

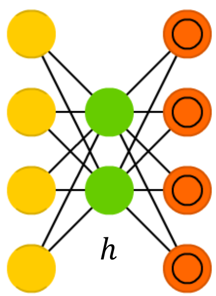
- Traditionally: dimensionality reduction, feature learning
- Recently: generative modeling (VAE, GAN)

Avoiding Trivial Identity



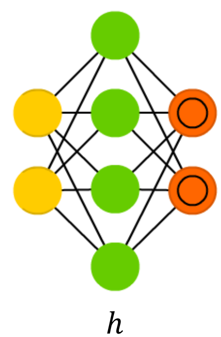
Undercomplete autoencoders
- h has lower dimension than x

➔ Must discard/compress some information in h



Overcomplete autoencoders
- h has higher dimension than x

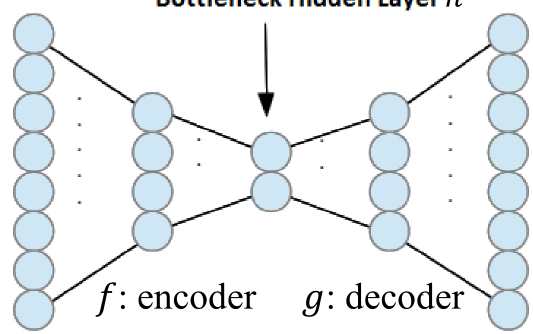
➔ Must be regularized



Undercomplete (compressing) autoencoders

Basic setup

- Constrain h to have smaller dimension than x
- Minimize reconstruction loss $L\left(x, g\left(f(x)\right)\right)$



Problem

- High-capacity f and g can learn to en-/decode each x_i to/from the single integer i
  ➔ f or g **needs** to have **low capacity** (e.g., linear g)

If additionally $\mathrm{L} = E_{MSE}$, the compressing autoencoder learns the PCA subspace

Prospect

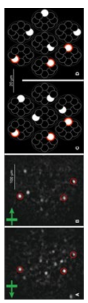
- Learn powerful **nonlinear** generalizations of **PCA**
- Find **salient features** in the data, represented in h

Overcomplete (regularized) autoencoders
→ regularization by sparsity

Basic setup
- Loss function encourages models with additional properties:
  e.g. sparsity of representation (this slide); robustness to noise or missing inputs (later)
- Sparse AE loss function: $\underbrace{L}_{\text{reconstruction loss}}\left(x, g\left(f(x)\right)\right) + \underbrace{\beta}_{\text{weight (a hyper parameter)}} \cdot \underbrace{\Omega_p(h)}_{\text{sparsity constraint on activations in hidden layer } h}$

Sparsity constraint $\Omega_p(h)$

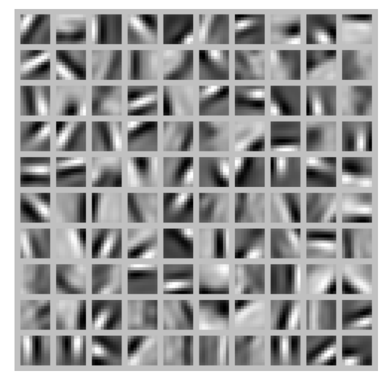
A neuron/unit is active or «fires» when it has activation close to 1; it is inactive with activation close to 0



- $\Omega_p(h) = \sum_{j=1}^{|h|} KL\left(B(p)||B(\hat{p}_j)\right)$: Enforces firing probability p for all of layer h's units
- $\mathrm{KL}\left(B(\,)||B(\,)\right)$: KL divergence between 2 Bernoulli distributions (0/1 distribution)
- p: sparsity parameter (target probability of any unit of h firing over all training data)
- $\hat{p}_j$: average activation of unit j in layer h ($[0..1]$ for the sigmoid activation function)

learn sparse, distributed representations

Prospect
- **Unlimited** model **capacity** (code size, depth of encoder/decoder)
- **Approximate** way of training a **generative model** (e.g., VAE)
- Remember V08? Sparsity as a more general inductive bias

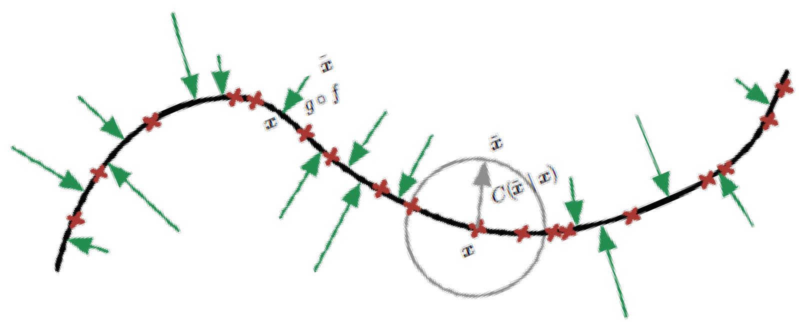


Synthesized inputs that maximize the activations of the hidden layer (100 units) of a sparse AE trained on 10x10 pixel images: Each unit learns a different edge. Source: A. Ng, «CS294A Lecture Notes: Sparse Autoencoder», 2011.

Overcomplete (regularized) autoencoders
→ regularization through denoising

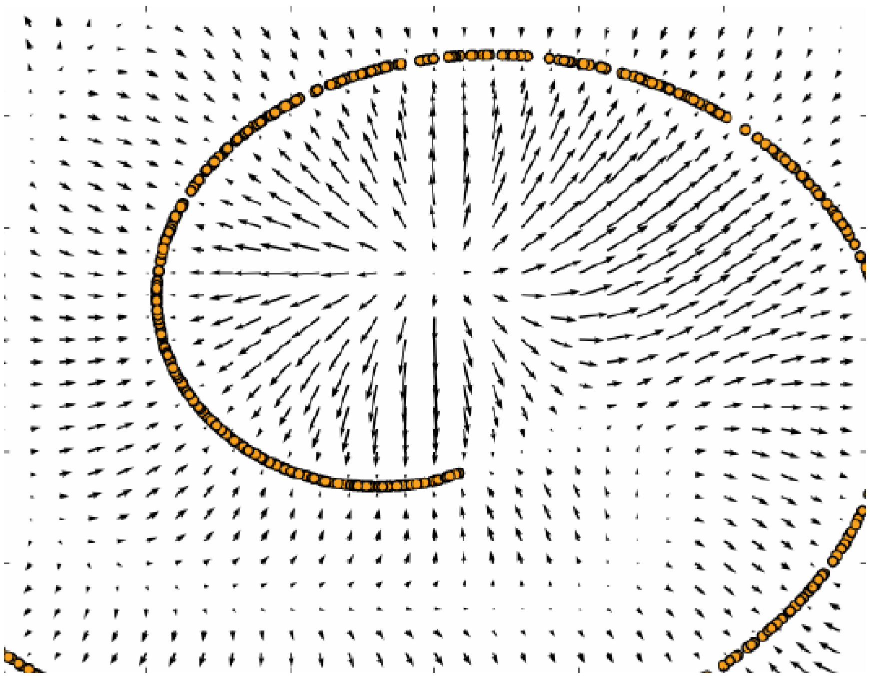
Basic setup
- Instead of minimizing $L\left(x, g\left(f(x)\right)\right)$ (with e.g., L being the Euclidean distance $= L^2$ norm)
- …minimize $L\left(x, g\left(f(\tilde{x})\right)\right)$ (with $\tilde{x}$ being a noisy copy of x → thus *denoising* (D) AE)

Illustration
- A DAE maps corrupted data points $\tilde{x}$ back to the original x
  ➔ it learns to map $\tilde{x}$ to the nearest point on the lower-dimensional manifold where x concentrates on
  ➔ it **learns a vector field** (green arrows)



x: red crosses lying near a low-dimensional manifold (black line)
$C(\tilde{x}|x)$: grey circle of equiprobable corruptions (grey arrow)

Overcomplete (regularized) autoencoders
→ regularization through denoising (contd.)



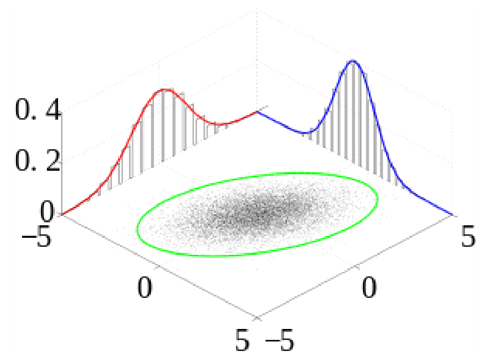
Arrows length: proportional to $g\left(f(\tilde{x})\right) - x$
(reconstruction minus input)

Example: Vector field learned by DAE

- 2D training data concentrates on 1-D curved manifold
- **Vectors point towards highly probably region of training data occurrence**

➔ **DAE** implicitly **estimates** the **probability distribution** of the data

The distribution (or its density function, PDF) describes everything there is to know about a basically random phenomenon; it is the essence of its structure (→ see V10).



Example: Training data points from a joint Gaussian probability distribution, with marginal densities. Source: https://en.wikipedia.org/wiki/Joint_probability_distribution

Overcomplete (regularized) autoencoders
→ regularizing the magnitude of the encoder's derivatives

Contractive AEs

- Goal: code does not change much when x changes slightly (i.e., **smoothness**)
- Approach: encourage derivatives of $f()$ be as small as possible
- Effect & relation to the name: CAEs resist perturbations in the input
  ➔ map **a larger x neighborhood to a smaller $f(x)$ neighborhood**
  ➔ local contraction of the space, hence *"contraction"*

- (CAE loss function: $L\left(x, g\left(f(x)\right)\right) + \beta \cdot \Omega(h, x)$ with $\Omega(h, x) = \left\lVert \frac{\partial f(x)}{\partial x} \right\rVert_F^2$
  (squared sum of squared elements (Frobenius norm) of the encoder's (Jacobian) matrix of partial derivatives))

➔ Another way to **learn** (a) the underlying manifold structure or (b) **a probabilistic model**!

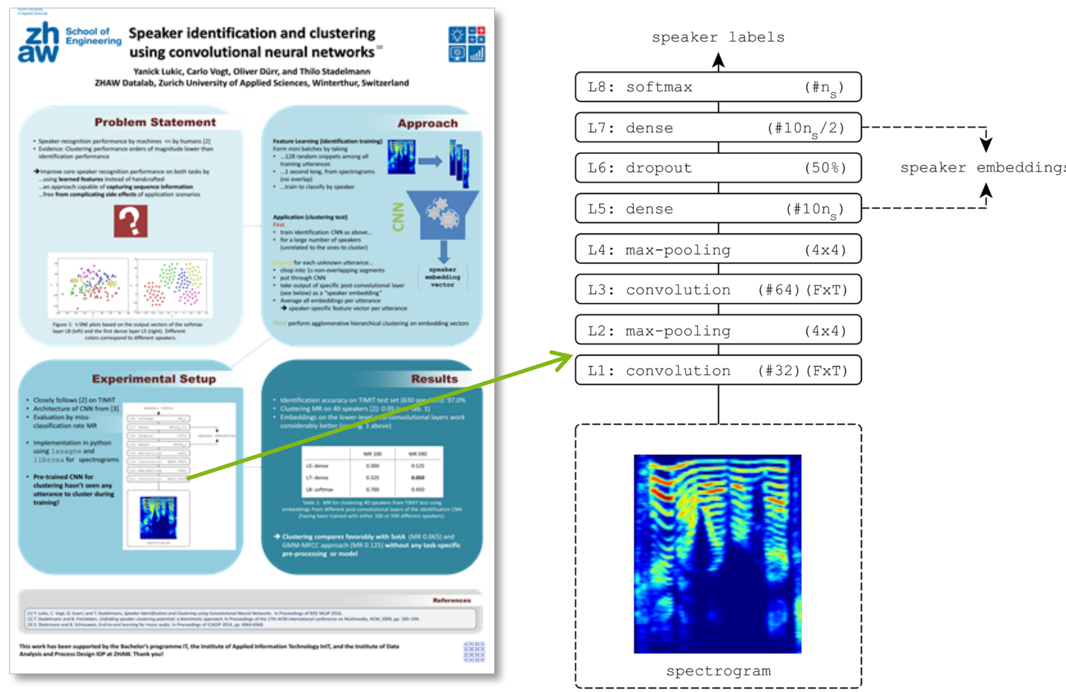
3. USE CASES FOR AUTOENCODERS

Use case 1: Learning embeddings

Embedding := Lower-dimensional representation in an “embedded subspace” (manifold)

Applications

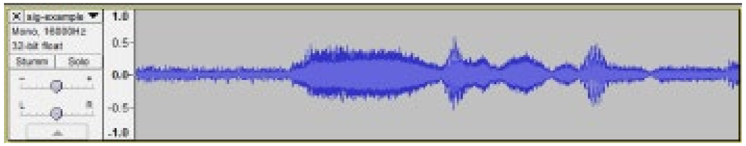
- Unsupervised pre-training ✓
- Feature learning ✓
- Dimensionality reduction ✓



Lukic, Vogt, Dürr, and Stadelmann, *“Speaker Identification and Clustering using Convolutional Neural Networks”*, MLSP’16.

Background on speech: The audio signal

The waveform $s[n]$ (a 1D array of N integer samples)

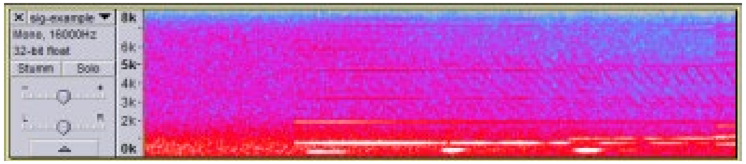


Time domain information (2D: time, amplitude):

- Energy (~loudness): $NRG = \frac{1}{N}\sum_n s[n]^2$
- Zero crossing rate (~prominent frequency for monophonic signals): $ZCR = \frac{1}{N}\sum_n I(s[n] \cdot s[n-1] < 0)$

Frequency domain information (3D: time, frequency, amplitude):

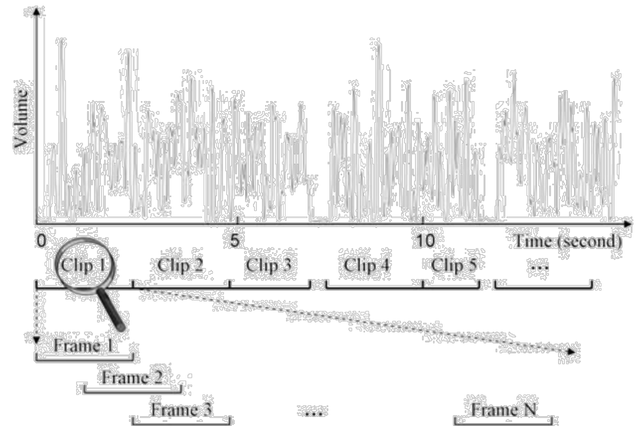
- Time frequency representations via FFT or DWT (phase information typically discarded)



More on signal processing: Smith, *"Digital Signal Processing - A Practical Guide for Engineers and Scientists"*, 2003

Background on speech: Frame-based processing
From signal to features

Feature extraction in general
- **Reduction** in **overall** information
- …**while** maintaining or even **emphasizing** the **useful** information



Source: http://what-when-how.com/video-search-engines/audio-features-audio-processing-video-search-engines/

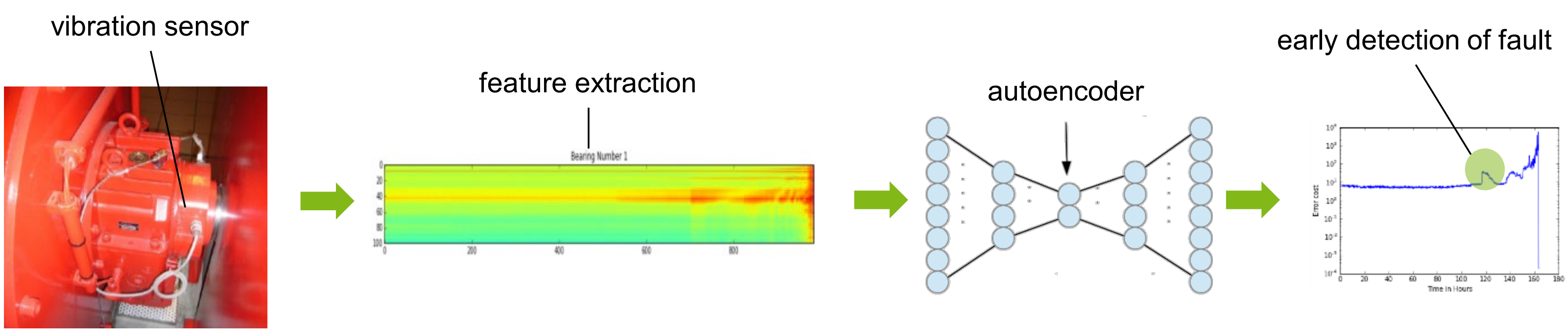
Challenging audio signal properties
- Neither stationary (i.e., statistical figures change over time)
  ➔ **problem** with transformations like Fourier transform **when analyzed in whole**
- …nor conveys its meaning in single samples
  ➔ **problem** when analyzing **per sample**
- Speech frames convey multiple information → **fractal structure**
  (Linguistic such as phonemes, syllables, words, sentences, phrases; identity, gender, dialect, …)

Solution
- **Chop into** short, usually overlapping chunks called frames
  ➔ extract basic acoustic features per frame
- Let a **representation learner** learn useful **higher-level features** related to the task at hand

Use case 2: Novelty detection

Idea
- Because the AE learns to **encode** / capture **variations in the training data**
- …it is by design **bad in encoding** previously **unseen variation**

Application: Predictive maintenance
- Vibration signal → feature extraction via spectrogram → autoencoder
- Monitor reconstruction error as a «novelty signal»

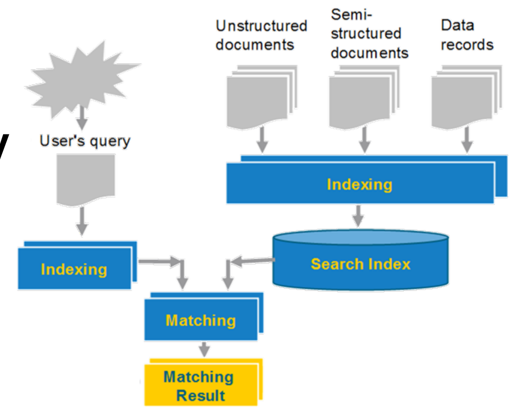


Stadelmann, Tolkachev, Sick, Stampfli, and Dürr, *“Beyond ImageNet - Deep Learning in Industrial Practice”*, in: Braschler, Stadelmann, and Stockinger (Eds), “Applied Data Science - Lessons Learned for the Data-Driven Business”, Springer, 2019.

Use case 3: Information retrieval via semantic hashing

Efficient IR by dimensionality reduction

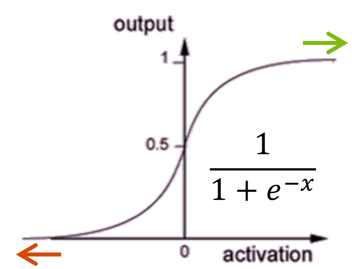
- Given a set of documents (e.g., texts & queries)...
- Train an AE to produce a code that is low dimensional and **binary**
- Create a **hash table** from binary code to document



➔ retrieve all docs that have the same binary code as the query
➔ enlarge to **similar results**: **flip bits** from the query's encoding

Implementation

- Use **sigmoid**al units in the final encoding layer
- Train to **saturate** (nearly 1 or 0) the units for all training data, e.g., by:
  - Add noise just before the sigmoid, increase its magnitude over time
    ➔ network will learn to increase data magnitude to preserve SNR
    ➔ saturation will occur



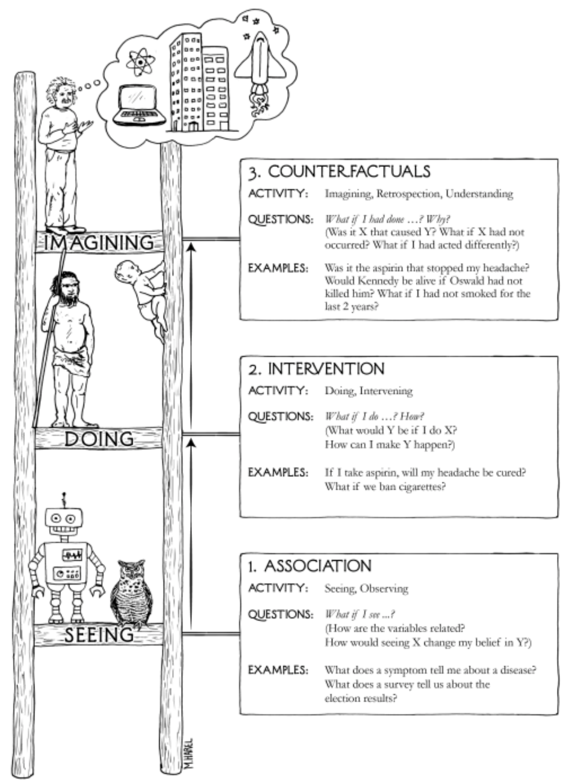
Salakhutdinov and Hinton, *"Semantic Hashing"*, International Journal of Approximation Reasoning, Elsevier, 2009.

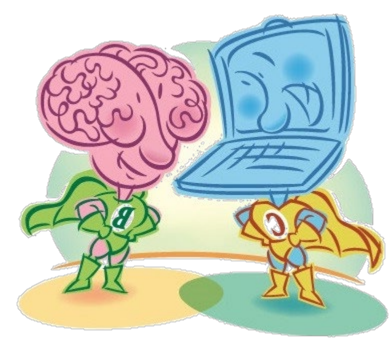
Where’s the intelligence?
Man vs. machine

- **Learning from the data itself** (rather than from human-provided labels) is also the main learning signal in biological learning

- In human learning, (self-)supervision does not seem to come merely from autoencoding, but from playing “**prediction games**”: thinking through the possible outcomes of events

Judea Pearl, *“The book of why: the new science of cause and effect”*, Penguin, 2019.





Review

- UL is a deemed the **greatest innovation area** in ML by many experts
- UL is **more than clustering**; in particularly, feature learning via deep models
- UL to facilitate some SL task may benefit from **output distribution matching**

- **AEs learn** the **structure** of the data **by balancing** approximate **reconstruction with** some regularization **penalty**
  ➔ they thus learn to capture lower-dimensional **manifolds**
  ➔ … and important aspects of the underlying data-generating **distribution**

- **AEs** are thus important approaches to build **generative model**s

APPENDIX

P04.1: Predictive maintenance with AEs

Work through the lab description of P04.1 and build different sorts auf autoencoders in order to decide on the faultiness of bearings in rotating machinery.

You are provided with extracted features from the NASA dataset.

AE types you will implement using `keras`:

- Compressing AE
- Sparse AE
- Denoising AE

Is unsupervised learning broken?
Contd. (→ see appendix for an example)

Methods
- Autoencoder-like models (→ see later)
- Variational Autoencoders (VAE, *«A Tutorial on Variational Autoencoders»*, Doersch 2016)
- Generative Adversarial Nets (GAN, *«Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks»*, Radford, Metz and Chintala 2016) → see V13c

Success factors
- **Label space has to be large** (e.g., regression; letter distribution in a language), otherwise ODM cost optimization is trivial
- Hidden **structure of the two spaces has to be** sufficiently **similar**
- ODM cost is likely to improve generalization by eliminating unsuitable F from consideration; but if the function class (e.g., a DNN) has high capacity, ODM cost optimization cannot recover the true F

→ ODM cost is a new concept; better generative models for optimization are expected to make it universally applicable / useful in the future

ODM cost example

From the *"Towards principled unsupervised learning"* paper

Data

- Sequences of digits [0-9] based on the characters from *"On the origin of species"*
- Each digit represented by a random MNIST example of this class
- Only 4 labeled digit images (otherwise only knowledge of character/digit distribution in *"Otoos"*)

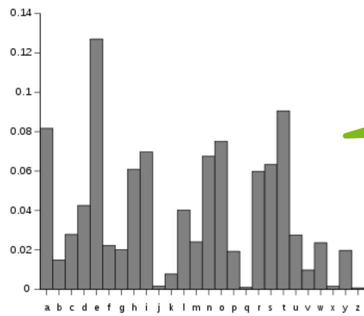
Goal

- Train GAN model to map digit images to 10-dim outputs (probability over [0-9])

Result

- 4.7% test set error on MNIST
- GAN training very sensitive to hyper parameters like learning rate

MNIST images



Distribution of letters in the English language

Yann LeCun on unsupervised learning & AI

See http://www.cnbc.com/2017/04/17/meet-the-man-who-makes-facebooks-machines-think.html

Zurich University of Applied Sciences

*"The **essence of intelligence**, to some extent, **is the ability to predict**,"* LeCun explained. *"If you can predict what's going to happen as a consequence of your actions then you can plan. You can plan a sequence of actions that will reach a particular goal."*

Helping AI understand and embrace uncertainty is part of an AI discipline called "unsupervised learning," currently the field's cutting edge. **When AI has observed enough to know how the world works and predict what's going to happen next, it can start thinking a bit more like humans, gaining a kind of common sense**, which, LeCun believes, is key to making machines more intelligent.

Still, sometimes LeCun can't restrain his enthusiasm. He's particularly excited about adversarial training, a relatively new form of AI research that could help solve the prediction and uncertainty challenges facing the field today. Adversarial training pits two AI systems against each other in an attempt to get them to teach themselves about the real world.

Adversarial training, LeCun said, *"is **the best, coolest idea in machine learning** in the last 10 or 20 years."*

Related UL concepts not covered in this lecture

Unsupervised pre-training

- Build a deep model for a supervised task by
- …consecutively stacking layers on top of each other that
- …have been trained in an unsupervised fashion (e.g., autoencoder)

→ See e.g. [Hinton et al., 2006]: *«A fast learning algorithm for deep belief nets»*

Transfer learning

- Learn a model in one domain (e.g., for classifying ImageNet photographs)
- Use this pre-trained model (at least the feature-extraction part) as the initialization for
- …learning in another domain (e.g., classifying paintings) or
- …training for another task (e.g., exchange the output layer of a deep net to draw paintings)

→ See e.g. http://cs231n.github.io/transfer-learning/

Semi-supervised learning

- Learn both from labeled training examples and
- …the general distribution of unlabeled data

→ See e.g. [Simmler et al., 2021]: *«A Survey of Un-, Weakly-, and Semi-Supervised Learning Methods for Noisy, Missing and Partial Labels in Industrial Vision Applications»*

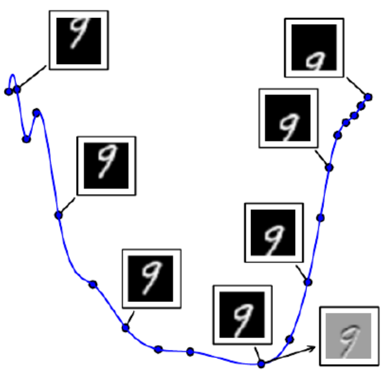
Manifold learning

Definition

- Manifolds are characterized by their tangent planes: The **axes of variation** at each training point in the lower-dimensional subspace

Why AEs learn manifolds

- AEs balance two opposing forces: learning approximate reconstruction while satisfying the regularizer's constraints
- Thus AEs learn h to **represent only those variations** in the training data that are **needed for reconstructing** → they learn to capture the manifold
- I.e., the learned mapping to h will be insensitive to variations that do not lie on the manifold

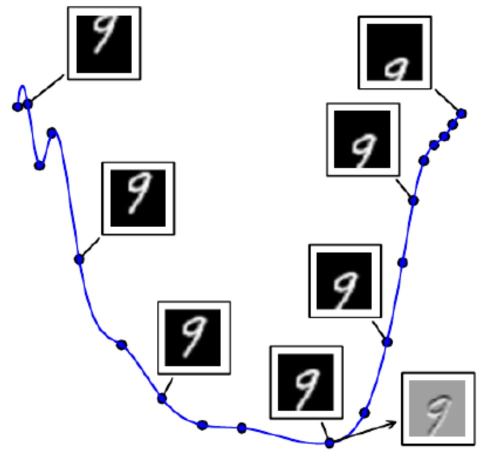


Example for a tangent hyperplane: The picture shows a one-dimensional manifold in 784-dimensional space created by translating MNIST images vertically. The 1D operation of vertical translation lies on a complicated curved path in image space (shown here in 2D via PCA).

The black arrow indicates an example tangent line at one point, with an image showing how this tangent direction appears in image space. Gray pixels indicate pixels that do not change as we move along the tangent line, white pixels indicate pixels that brighten, and black pixels indicate pixels that darken
→ i.e., the "9" moves down as we move to the right on the manifold

Manifold learning (contd.)
A plea for depth

- Traditional approaches rely on local interpolation between nearest neighbors
- Many manifolds are not “smooth” ➔ need for a very large number of training examples and still unable to generalize to unseen variations
- Manifolds involved in AI problems can have very complicated structure
- Example: Observing a single coordinate in the MNIST image translation picture



➔ the patterns of brightness in the training data drive the complexity of the manifold (even if the image transformations are simple)
➔ distributed deep representations are able to capture this

Deep or shallow?

Neural network guarantees (a.k.a. universal approximation theorem) are not sufficient

- 1 hidden layer in a feed forward NN is enough to **approximate any function** (within a broad class) to an arbitrary degree, **given *enough* hidden units**

Drawbacks of shallow AEs

- Shallow mapping from/to code layer h → **no arbitrary constraints** learnable (e.g. sparsity)

Advantages of deep AEs

- can exponentially **reduce** the **computational cost**
- can exponentially **decrease** the **amount of training data**

Best practice of the early deep learning days

- greedily pre-train the deep architecture by training a stack of shallow AEs

Exercise (at home): Situating autoencoders

How deep should an autoencoder be?

➔ Extract the arguments from the reasoning in [Bengio, 2009]: “Learning Deep Architectures for AI”, chapter 2.

➔ Use them to argue for AE’s advantages in manifold learning (→ slide 15 and appendix)

➔ …and representation learning in general (→ slide 16)