DD2437 – Artificial Neural Networks and Deep Architectures (annda)

Lecture 9: **Deep learning fundamentals**

General philosophy and a review of deep architectures

Pawel Herman

Computational Science and Technology (CST)

KTH Royal Institute of Technology

AI ambition behind Deep Learning

The grand plan is to *”allow computers to model our world well enough to exhibit what we call intelligence”.*

(Bengio, 2006)

- The need for capturing high-level of abstraction
- Hope in learning algorithms that can help to exploit large quantities of available information (big data in the future) and generalise it to new contexts
- The assumption about the need for highly nonlinear (varying) mathematical functions (accounting for variations in the multivariate, often high-dimensional, domain of interest) to model complex behaviours

AI ambition behind Deep Learning

The grand plan is to *”allow computers to model our world well enough to exhibit what we call intelligence”.*

So, we need

- knowledge
- learning
    - complex functions,
    - from unlabeled data
    - with little human input
- generalisation
- understanding/identifying the underlying explanatory factors

AI ambition behind Deep Learning

The grand plan is to *"allow computers to model our world well enough to exhibit what we call intelligence".*

So, we need

- knowledge
- learning
  - complex functions,
  - from unlabeled data
  - with little human input
- generalisation
- understanding/identifying the underlying explanatory factors

Is there a hope in

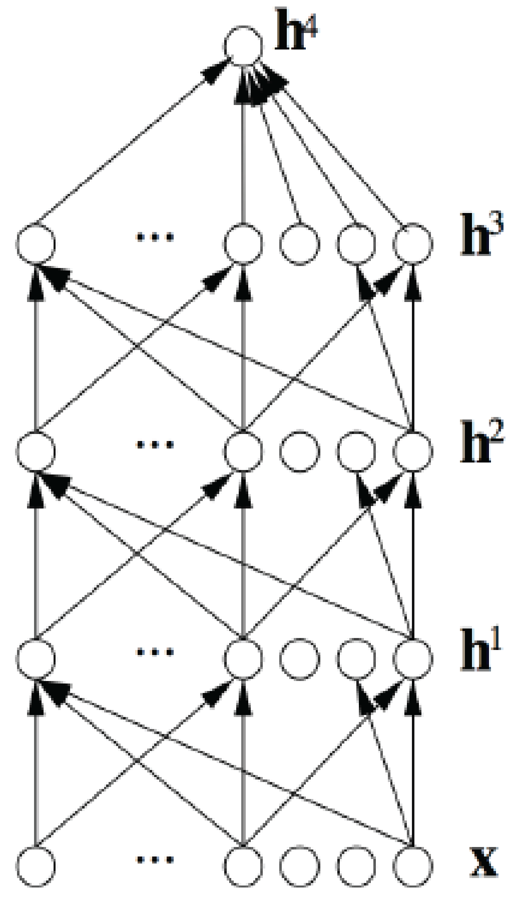
DEEP LEARNING in

DEEP ARCHITECTURES ?

What is depth in ML?

- Depth of architecture
  - the number of levels of composition of nonlinear operations in the function learnt
  - the length of the longest path from input to output in the graph

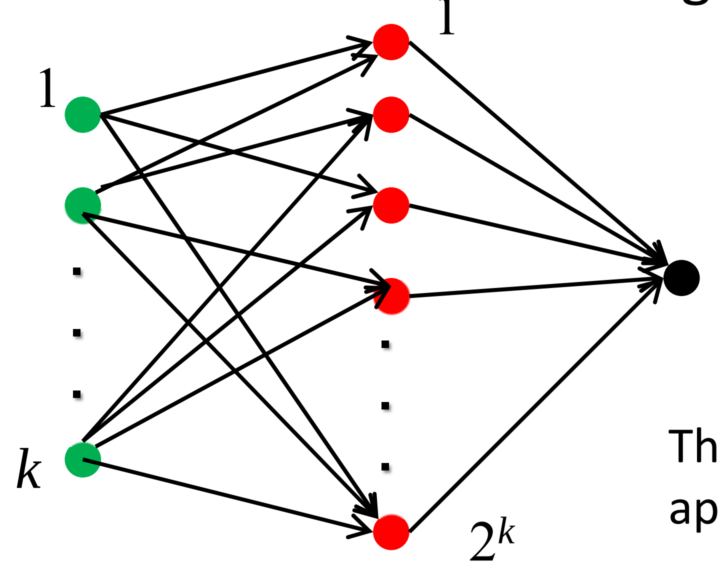
- Deep learning
  - using multiple layers of inf. processing stages in hierarchical architectures for pattern recognition and representation learning
  - originally, focus on (*incremental*) learning of feature hierarchies



Motivation for deep structures

Why go deep? Do we need deep structures?

- Expressive power and compactness of models (*expressibility* and *efficiency*)
  - enhances generalisation, especially with limited training examples
  - less degrees of freedom when handling complexity and nonlinearity – exponential gain

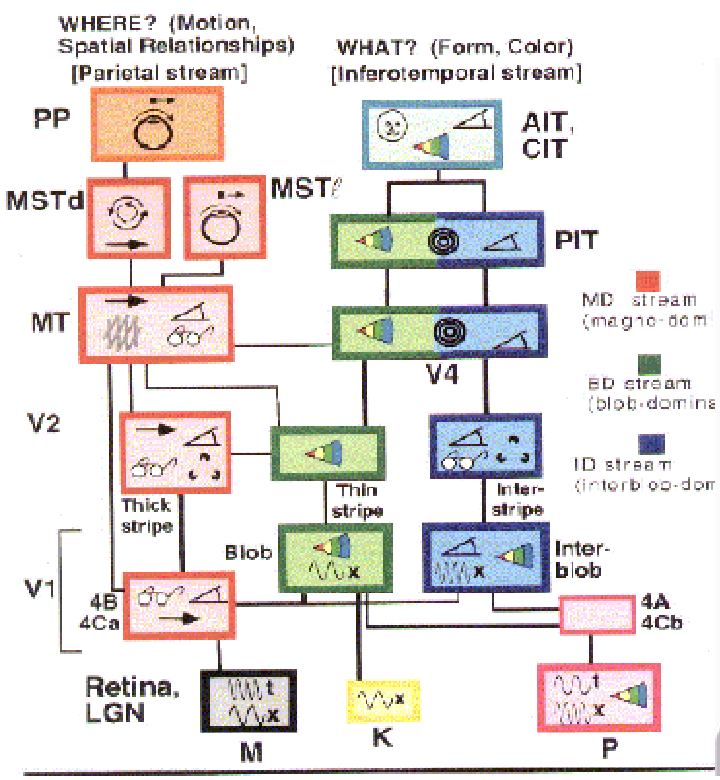


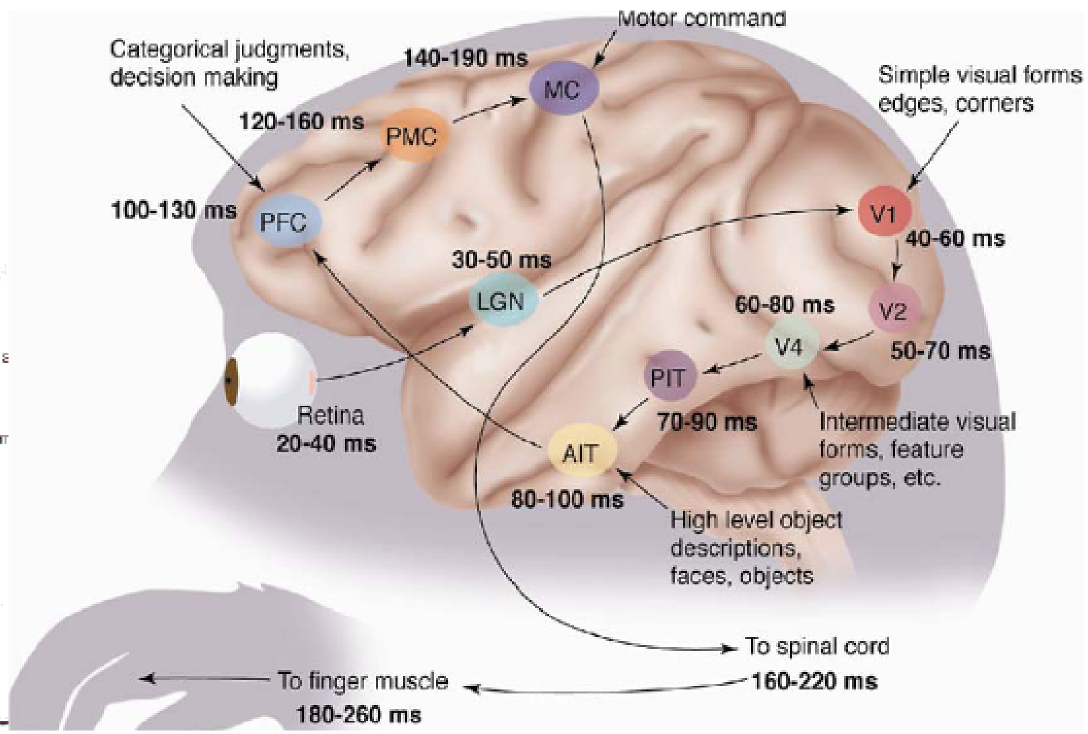
Shallow structure may need exponential size of hidden layer(s)

The universal approximation theorem and approximation costs.

Why go deep? Do we need deep structures?

- Inspirations from hierarchical brain organisation





LeCun & Ranzato, 2013

Motivation for deep structures

Why go deep? Do we need deep structures?

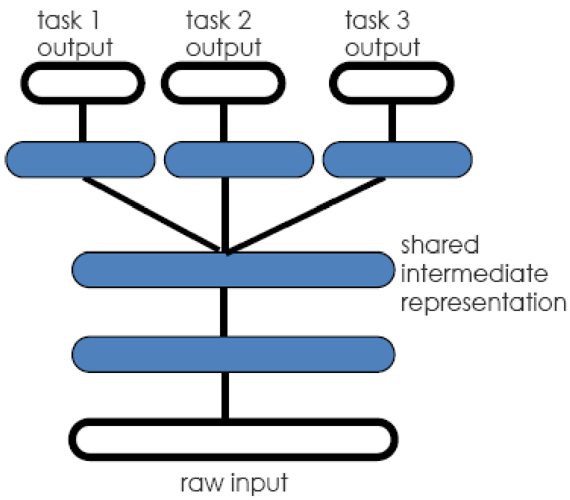
- Expressive power and compactness of models
  - enhances generalisation, especially with limited training examples
  - less degrees of freedom when handling complexity and nonlinearity
- Inspirations from hierarchical brain organisation
- Cognitive inspiration – multiple levels of abstraction

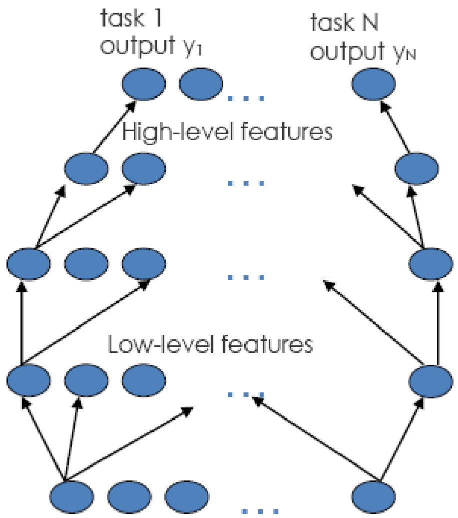
Motivation for deep structures

Why go deep? Do we need deep structures?

- Finally, ...

  multiple levels of representations facilitate transfer and multi-task learning (hierarchy of representations, non-local generalisation)





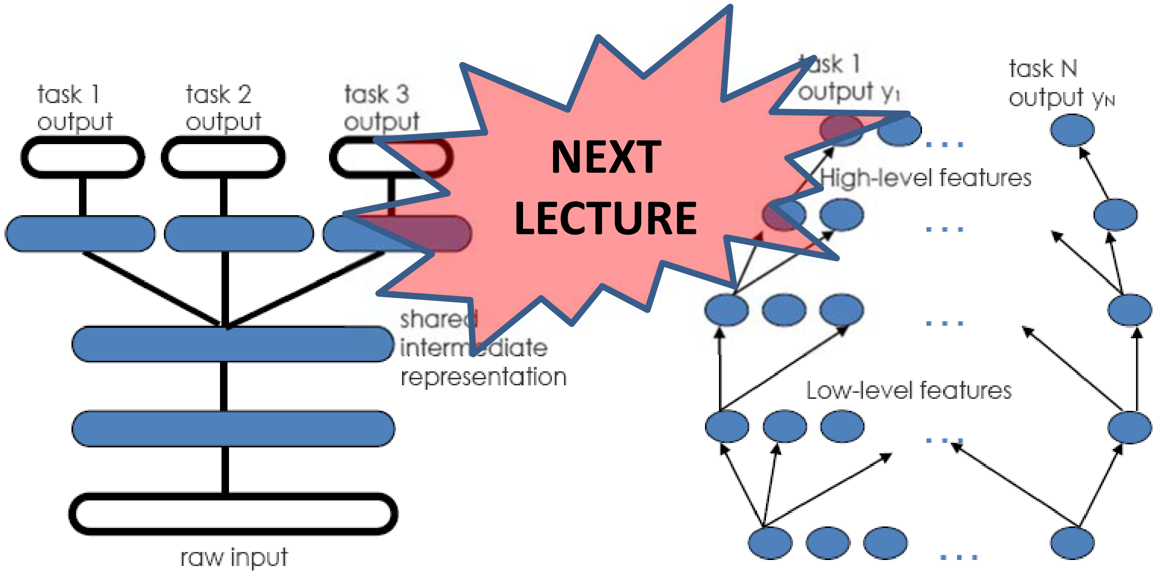
Lee, 2011

Motivation for deep structures

Why go deep? Do we need deep structures?

- Finally, ...

  multiple levels of representations facilitate transfer and multi-task learning (hierarchy of representations, non-local generalisation)

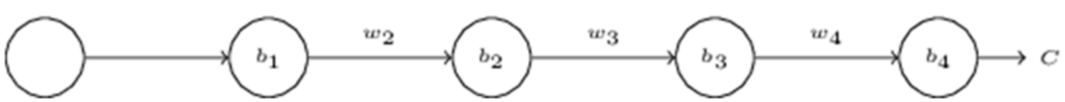


Lee, 2011

Trouble with classical multi-layer ANNs

- Hard to train
  - the problem of *vanishing gradients* (*diffusion* of gradients) in backpropagation algorithm

$$\frac{\partial C}{\partial b_1} = \sigma'(z_1) \times w_2 \times \sigma'(z_2) \times w_3 \times \sigma'(z_3) \times w_4 \times \sigma'(z_4) \times \frac{\partial C}{\partial a_4}$$

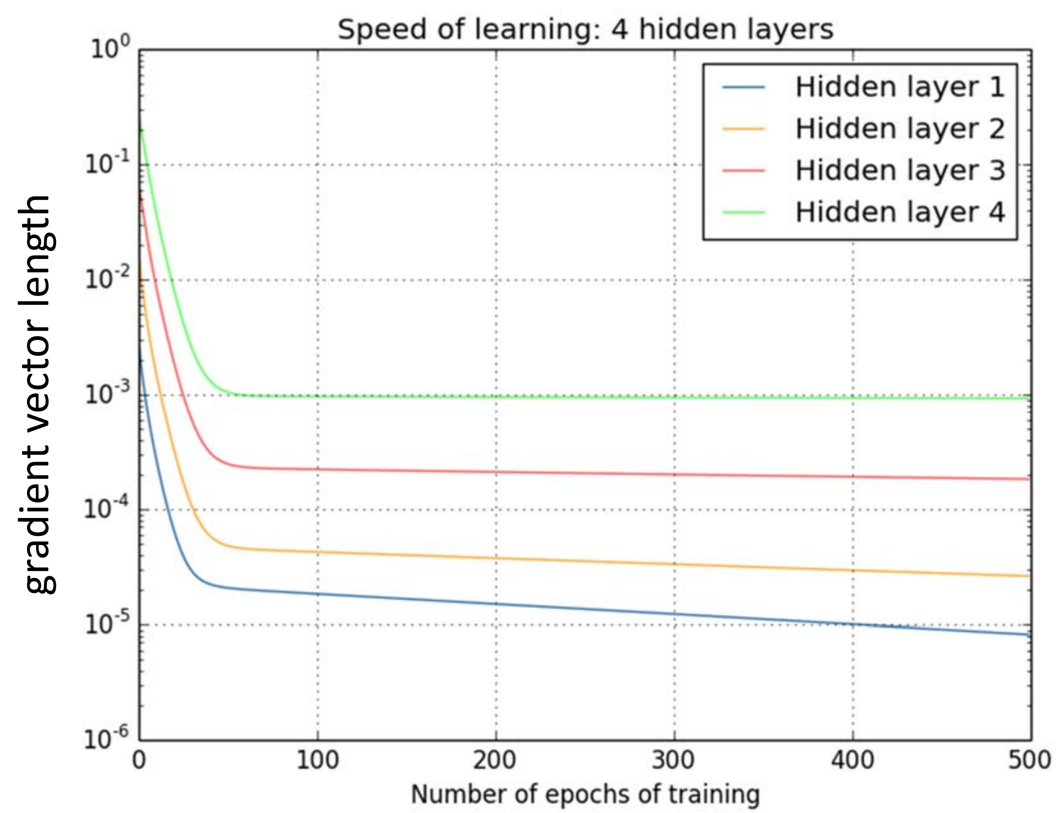


$$\frac{\partial C}{\partial b_1} = \sigma'(z_1)\overbrace{w_2\sigma'(z_2)}^{<\frac{1}{4}}\overbrace{w_3\sigma'(z_3)}^{<\frac{1}{4}}\underbrace{w_4\sigma'(z_4)\frac{\partial C}{\partial a_4}}_{\text{common terms}}$$

$$w_i \sim \mathbb{N}(0,1); \quad b_i \leq 1$$

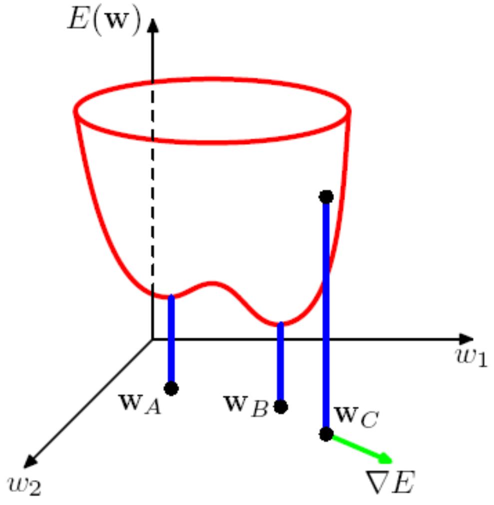
$$\frac{\partial C}{\partial b_3} = \sigma'(z_3)\overbrace{w_4\sigma'(z_4)\frac{\partial C}{\partial a_4}}$$

Nielsen, 2015



Trouble with classical multi-layer ANNs

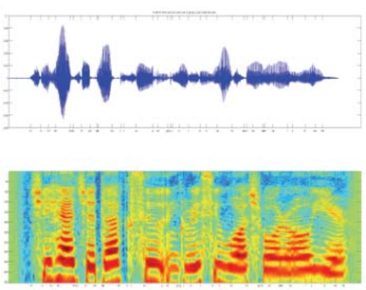
- Hard to train
  - the problem of *vanishing gradients* (*diffusion* of gradients) in backpropagation algorithm
  - it is really about unstable gradients
  - non-convex optimisation
    - local minima
    - susceptibility to overfitting

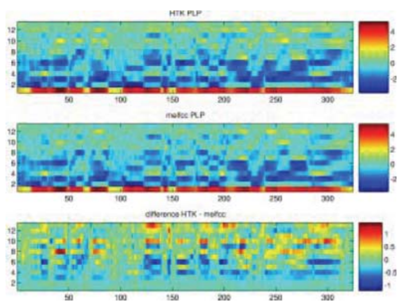


Learning high-level features – data representations

Traditional pattern recognition

- Human-designed representations (hand-engineered features)





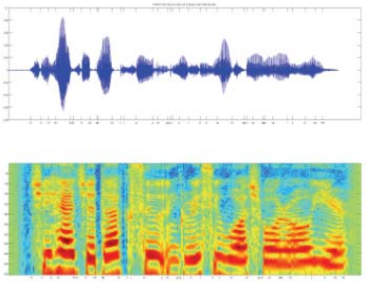
- Focus on optimisation to make best predictions
- Importance of data labels in supervised learning

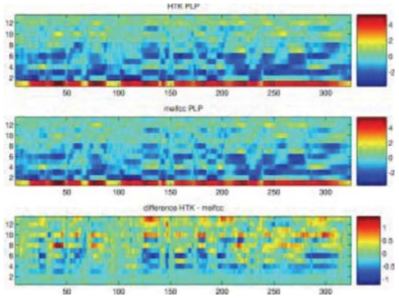
$$(\boldsymbol{x}, \color{red}{y})$$

Learning high-level features – data representations

Traditional pattern recognition

- Human-designed representations (hand-engineered features)





- Focus on optimisation to make best predictions
- Importance of data labels in supervised learning

$$(\boldsymbol{x}, y)$$

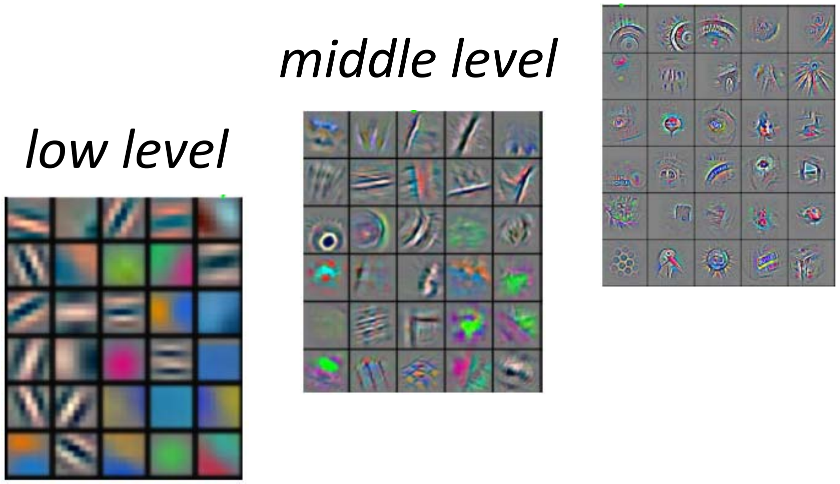
Deep learning approach

- Representation learning where good features are automat. learnt
- Potential to learn multiple levels of representation in DL algorithms

high level

middle level

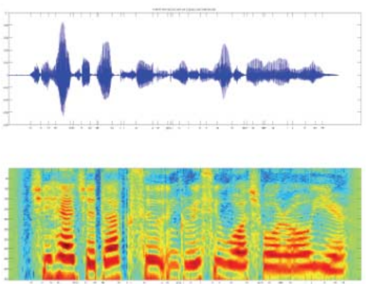
low level

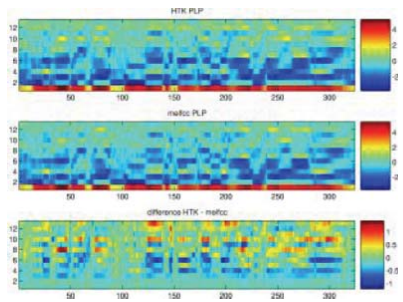


Learning high-level features – data representations

Traditional pattern recognition

- Human-designed representations (hand-engineered features)





- Focus on optimisation to make best predictions
- Importance of data labels in supervised learning

$$(\boldsymbol{x}, y)$$

Deep learning approach

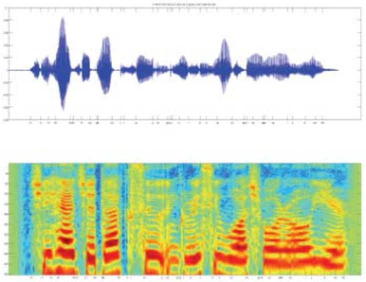
- Representation learning where good features are automat. learnt
- Potential to learn multiple levels of representation in DL algorithms

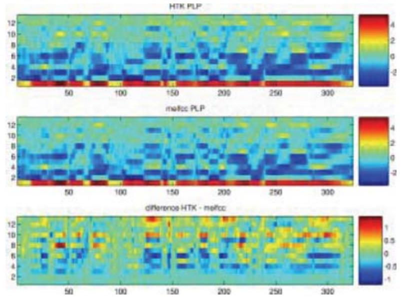
BUT: ***Extracting low-level features specific to the problem domain helps a lot!***

Learning high-level features – data representations

Traditional pattern recognition

- Human-designed representations (hand-engineered features)





- Focus on optimisation to make best predictions
- Importance of data labels in supervised learning

$$(\boldsymbol{x}, {\color{red}y})$$

Deep learning approach

- Representation learning where good features are automat. learnt
- Potential to learn multiple levels of representation in DL algorithms
- For example,

  character –> word –> word group, phrase –> clause –> sentence –> story

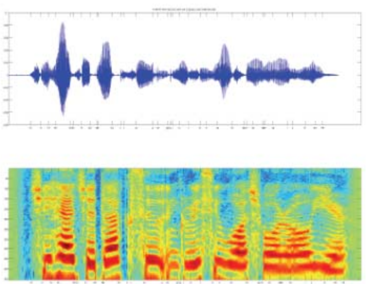
  pixel –> edge –> motif –> object

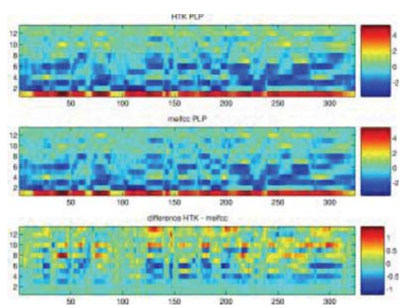
  sample –> spectral feature –> sound –> phoneme –> word

Learning high-level features – data representations

Traditional pattern recognition

- Human-designed representations (hand-engineered features)





- Focus on optimisation to make best predictions
- Importance of data labels in supervised learning

$$(\boldsymbol{x}, {\color{red}y})$$

Deep learning approach

- Representation learning where good features are automat. learnt
- Potential to learn multiple levels of representation in DL algorithms
- Good predictions are v. important but so is <u>data representation</u>
- Both unsupervised and supervised mode is heavily exploited – <u>unlabeled data are also useful</u>

Short historical note on deep architectures in ML (1)

- Perceptron - the first learning machine (Rosenblatt, ~1960)
- Deep learning in artificial neural networks
  - revival of interest with backpropagation in 1980s
  - ”better” backprop with advanced gradient descent
  - generalisation –complexity issues and bias/variance dilemma

Short historical note on deep architectures in ML (1)

- Perceptron - the first learning machine (Rosenblatt, ~1960)
- Deep learning in artificial neural networks
  - revival of interest with backpropagation in 1980s
  - ”better” backprop with advanced gradient descent
  - generalisation –complexity issues and bias/variance dilemma

  BUT still......

  - lack of ability to learn from the unlabeled data (most data is unlabeled)
  - slow learning, problems with convergence, sensitivity to local minima

Short historical note on deep architectures in ML (2)

- Perceptron - the first learning machine (Rosenblatt, ~1960)
- Deep learning in artificial neural networks
  - revival of interest with backpropagation in 1980s
  - ”better” backprop with advanced gradient descent
  - generalisation –complexity issues and bias/variance dilemma
- Shallow architectures with Support Vector Machines (SVMs)
  - effective in addressing simple and well-constrained problems
  - kernels arbitrarily define features (not hand-crafted but still ”fixed”)
  - limited modelling and representational power

Short historical note on deep architectures in ML (2)

- Perceptron - the first learning machine (Rosenblatt, ~1960)
- Deep learning in artificial neural networks
  - revival of interest with backpropagation in 1980s
  - ”better” backprop with advanced gradient descent
  - generalisation –complexity issues and bias/variance dilemma
- Shallow architectures with Support Vector Machines (SVMs)
  - effective in addressing simple and well-constrained problems
  - kernels arbitrarily define features (not hand-crafted but still ”fixed”)
  - limited modelling and representational power

Prior knowledge is arbitrary, not learnt

Short historical note on deep architectures in ML (3)

- Exploration of potential benefits of unsupervised or semi-supervised techniques in the context of supervised learning

Short historical note on deep architectures in ML (3)

- Exploration of potential benefits of unsupervised or semi-supervised techniques in the context of supervised learning
- Identification of *Fundamental Deep Learning Problem* in 1991
  - vanishing or exploding gradients – unstable learning
  - progresive ideas with deep hierarchies of recurrent networks, auto-encoders and first deep belief networks

Short historical note on deep architectures in ML (3)

- Exploration of potential benefits of unsupervised or semi-supervised techniques in the context of supervised learning
- Identification of *Fundamental Deep Learning Problem* in 1991
  - vanishing or exploding gradients – unstable learning
  - progresive ideas with deep hierarchies of recurrent networks, auto-encoders and first deep belief networks
- Major breakthrough in 2006
  - the idea to pre-train deep architectures with layer-wise unsupervised learning (groups led by G.E. Hinton, Y. Bengio and Y. LeCun)
  - more efficident parameter estimation methods

Short historical note on deep architectures in ML (3)

- Exploration of potential benefits of unsupervised or semi-supervised techniques in the context of supervised learning
- Identification of *Fundamental Deep Learning Problem* in 1991
  - vanishing or exploding gradients – unstable learning
  - progresive ideas with deep hierarchies of recurrent networks, auto-

[1] Hinton, G. et al. (2006) A fast learning algorithm for deep belief nets. *Neural Computation* 18:1527-1554,

[2] Bengio, Y. et al. (2006) Greedy Layer-Wise Training of Deep Networks, in J. Platt et al. (Eds), Advances in Neural Information Processing Systems 19 (NIPS 2006), pp. 153-160.

[3] Ranzato, M. et al. & Yann LeCun, Y. (2006) Efficient Learning of Sparse Representations with an Energy-Based Model, in J. Platt et al. (Eds), *NIPS*.

Short historical note on deep architectures in ML (3)

- Exploration of potential benefits of unsupervised or semi-supervised techniques in the context of supervised learning
- Identification of *Fundamental Deep Learning Problem* in 1991
  - vanishing or exploding gradients – unstable learning
  - progresive ideas with deep hierarchies of recurrent networks, auto-
-

Shared principles in these papers:

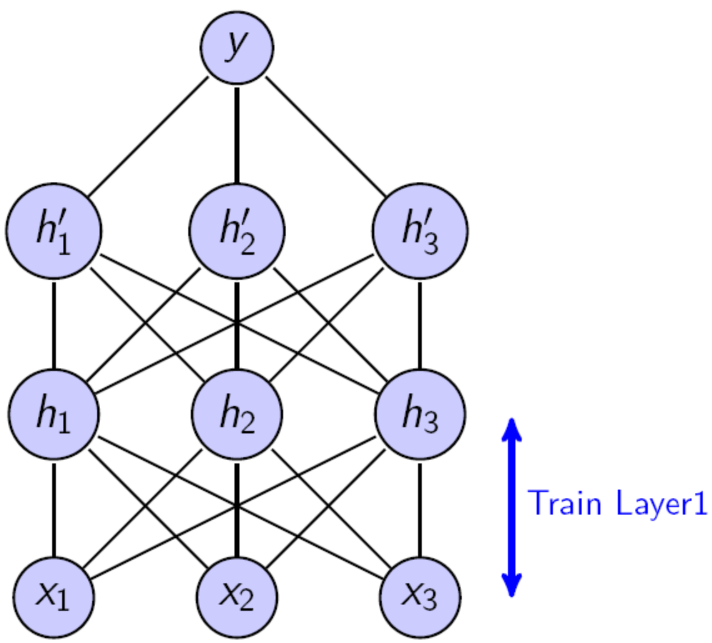
- Unsupervised learning of representations is used to (pre-)train each layer.
- Unsupervised training of one layer at a time, on top of the previously trained ones. The representation learned at each level is the input for the next layer.
- Use supervised training to fine-tune all the layers (in addition to one or more additional layers that are dedicated to producing predictions).

Successful applications as a driver for development

- Convolutional nets (CNNs) in *computer vision*
- Deep learning based *speech recognition* systems developed by Google and Microsoft
- Deep learning is becoming a hot topic in *natural language processing* (NLP)
- Advances in machine translation (RNNs, LSTM)
- Growing importance in reinforcement learning (deep RL)
- Scope of applications massively grows

General theme of the early deep learning protocol – deep belief networks, stacked autoencoders

- Greedy layer-wise unsupervised pre-training + supervised tuning (the legacy of Hinton, Bengio and LeCun)

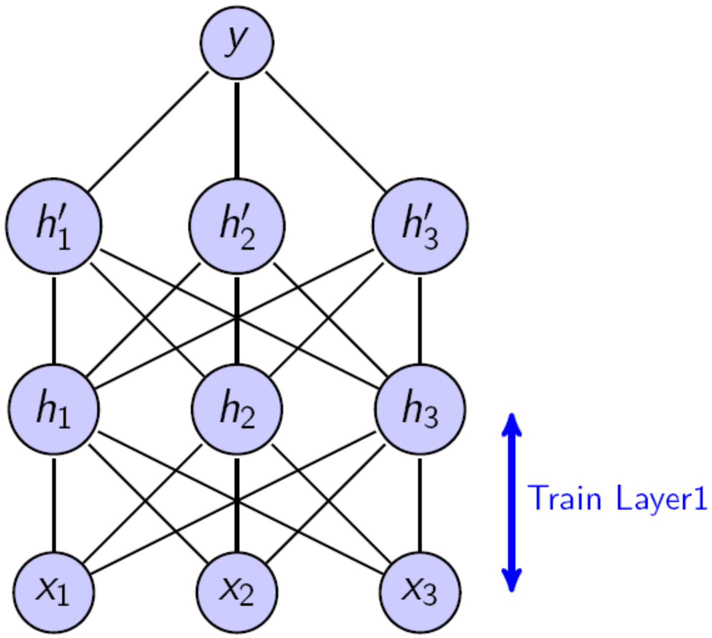


Single layer at a time

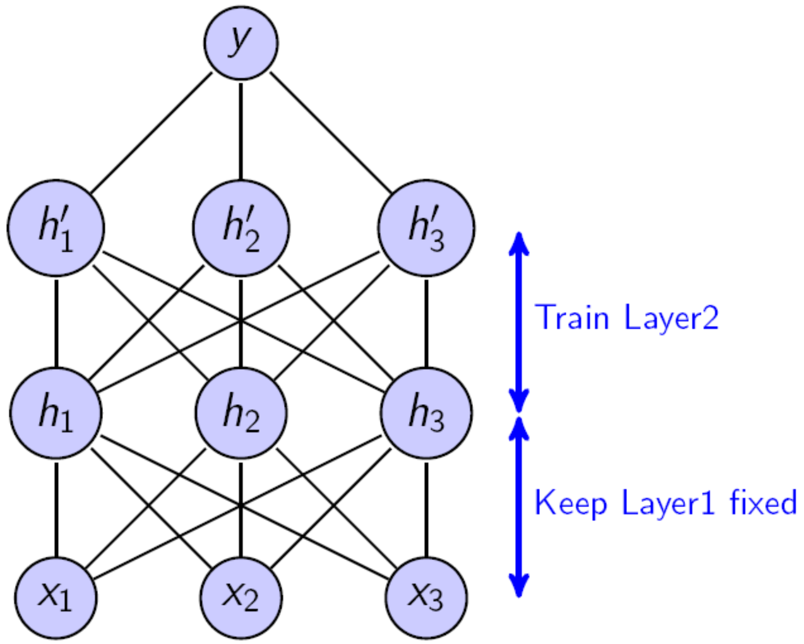
Hinton et al., 2006
Duh, 2013

General theme of the early deep learning protocol – deep belief networks, stacked autoencoders

- Greedy layer-wise unsupervised pre-training + supervised tuning (the legacy of Hinton, Bengio and LeCun)



Single layer at a time

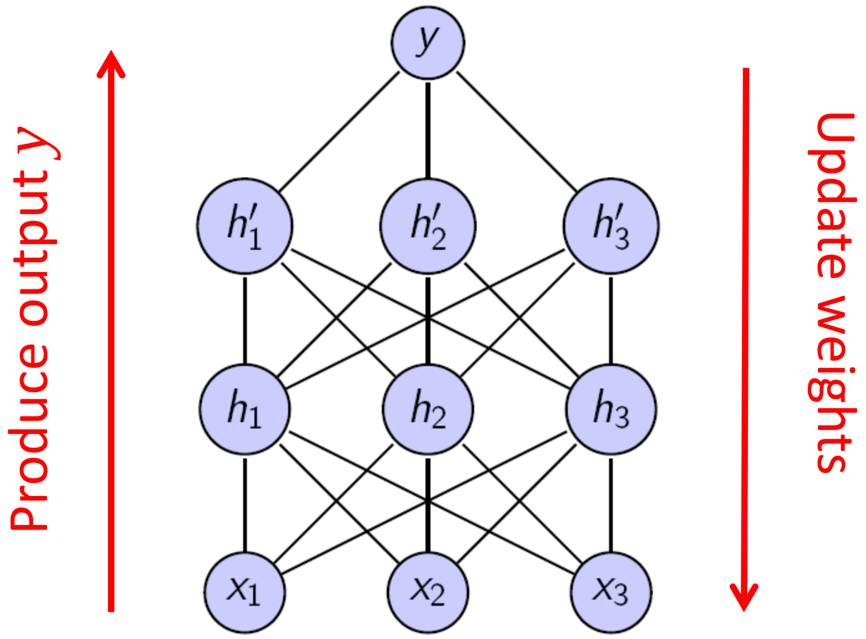


Train another layer while keeping the lower layer fixed

Hinton et al., 2006
Duh, 2013

General theme of the early deep learning protocol – deep belief networks, stacked autoencoders

- Greedy layer-wise unsupervised pre-training + supervised tuning (the legacy of Hinton, Bengio and LeCun)



Gradient-based fine tuning

1. Add a classifier layer and retrain globally the entire structure.
2. Train only a supervised classifier on top and keep other layers fixed.

Hinton et al., 2006
Duh, 2013
LeCun & Ranzato, 2013

Hypothetical role of unsupervised pre-training

- Regularisation hypothesis (Erhan et al., 2010)
  - Pre-training minimises variance
  - It also helps to control complexity for architectures with large sizes of hidden layers
  - Acts like an implicit penalisation term – regularisation

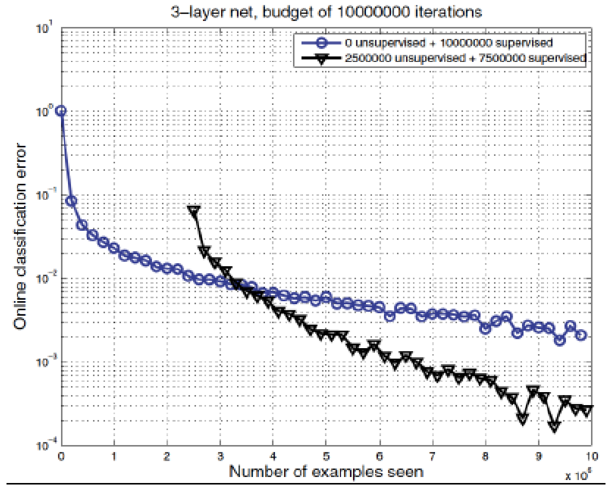
Hypothetical role of unsupervised pre-training

- Regularisation hypothesis (Erhan et al., 2010)
  - Pre-training minimises variance
  - It also helps to control complexity for architectures with large sizes of hidden layers
  - Acts like an implicit penalisation term – regularisation
- Optimisation hypothesis (Bengio et al., 2007)


Hypothetical role of unsupervised pre-training

- Regularisation hypothesis (Erhan et al., 2010)
  - Pre-training minimises variance
  - It also helps to control complexity for architectures with large sizes of hidden layers
  - Acts like an implicit penalisation term – regularisation

- Optimisation hypothesis (Bengio et al., 2007)
  - pre-training finds a better initial condition for further gradient-based optimisation
  - good initial conditions are very important
  - it facilitates training of the entire architecture (lower and higher layers benefit from tuning)



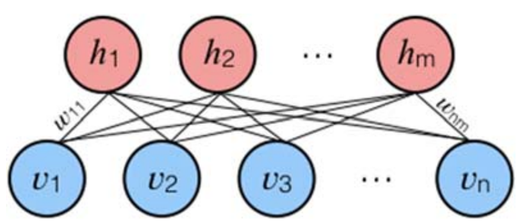
The fate of "pretraining" concept

- Pretraining actually sparked off developments in deep learning *("revived DNNs from obscurity", McKay*)
- The original ideas: *learning input distribution is useful and initialization is important*
- Now, unsupervised pretraining has mostly been abandoned due to more advanced regularization techniques and ReLU units
- However, pretraining concept has inspired much of the modern research in transfer learning

Most common network architecture and learning types

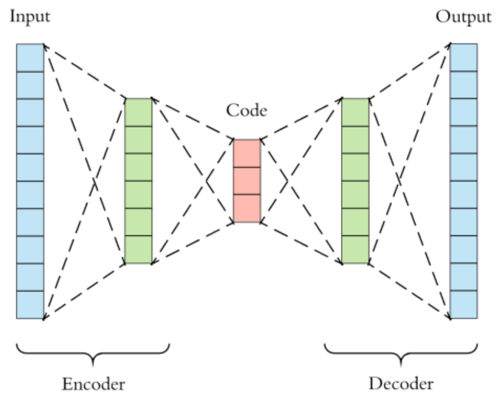
Restricted Boltzmann machine (RBM) layer

(contrastive divergence for pre-training)

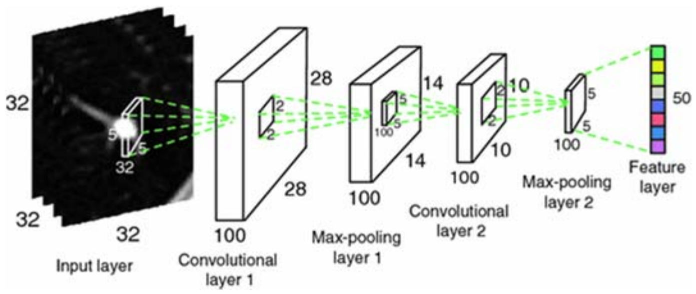


Auto-encoder (AE) layer

(gradient descent based algorithms for pre-training)



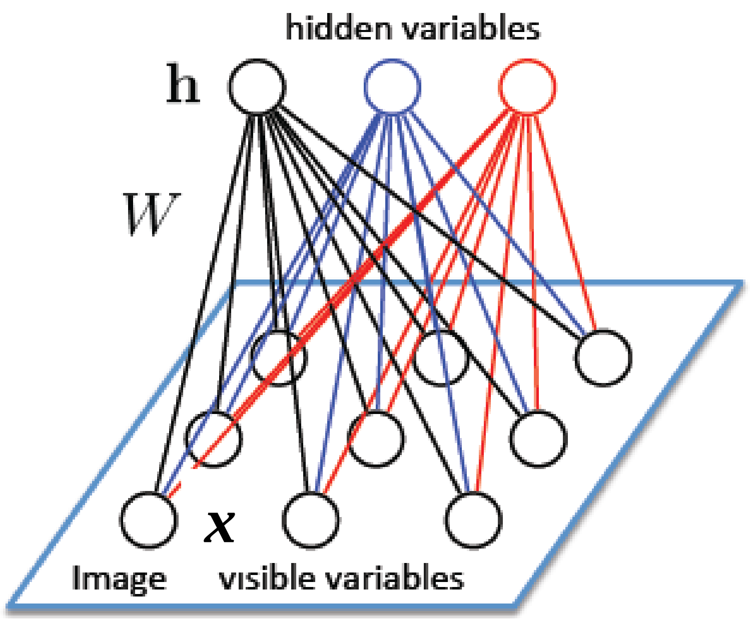
Convolutional neural networks (CNNs)



Greedy layer-wise unsupervised pre-training, which is increasingly omitted once **ReLU** units are employed

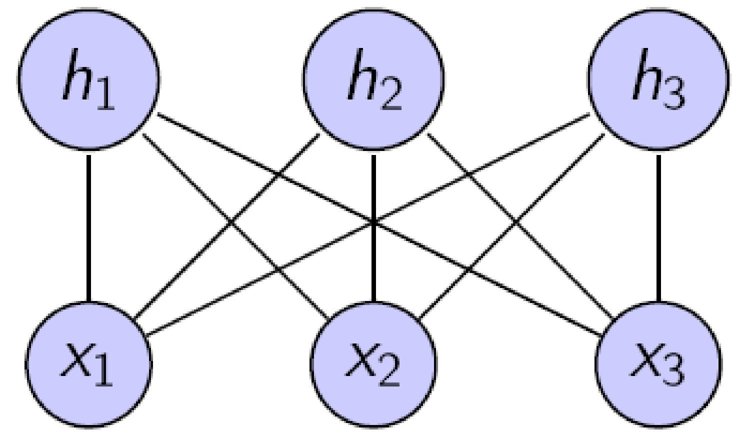
Network can be initialised without any pre-training, though transfer learning is exploited

Restricted Boltzmann machine (RBM)



In traditional RBM, x_i and h_j are binary variables

Simple energy-based model

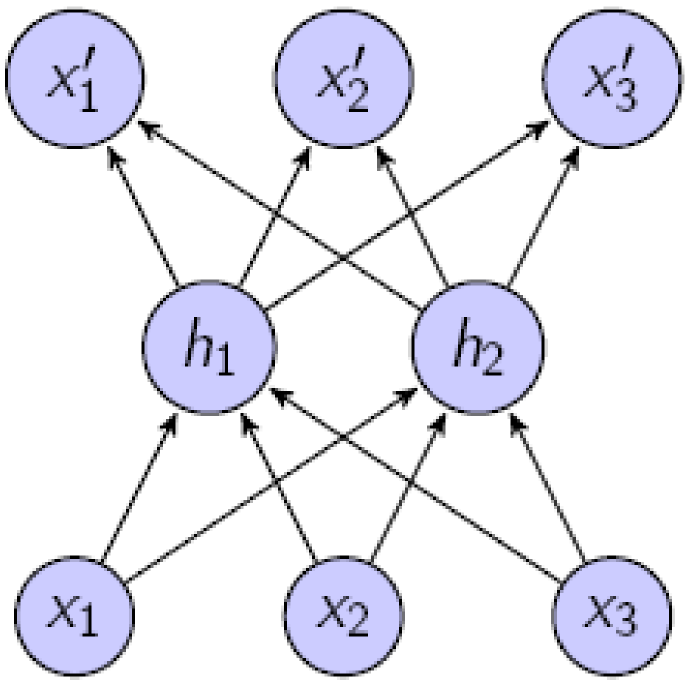


$$p(x,h) \sim e^{-E_\theta(x,h)}$$

$$E_\theta(x,h) = -x'Wh - b'x - d'h$$

The idea is to optimise log-likelihood with the use of approximative Gibbs sampling – Constrastive Divergence algorithm

Autoencoders



Decoder: $x' = \sigma(W'h + d)$

Encoder: $h = \sigma(Wx + b)$

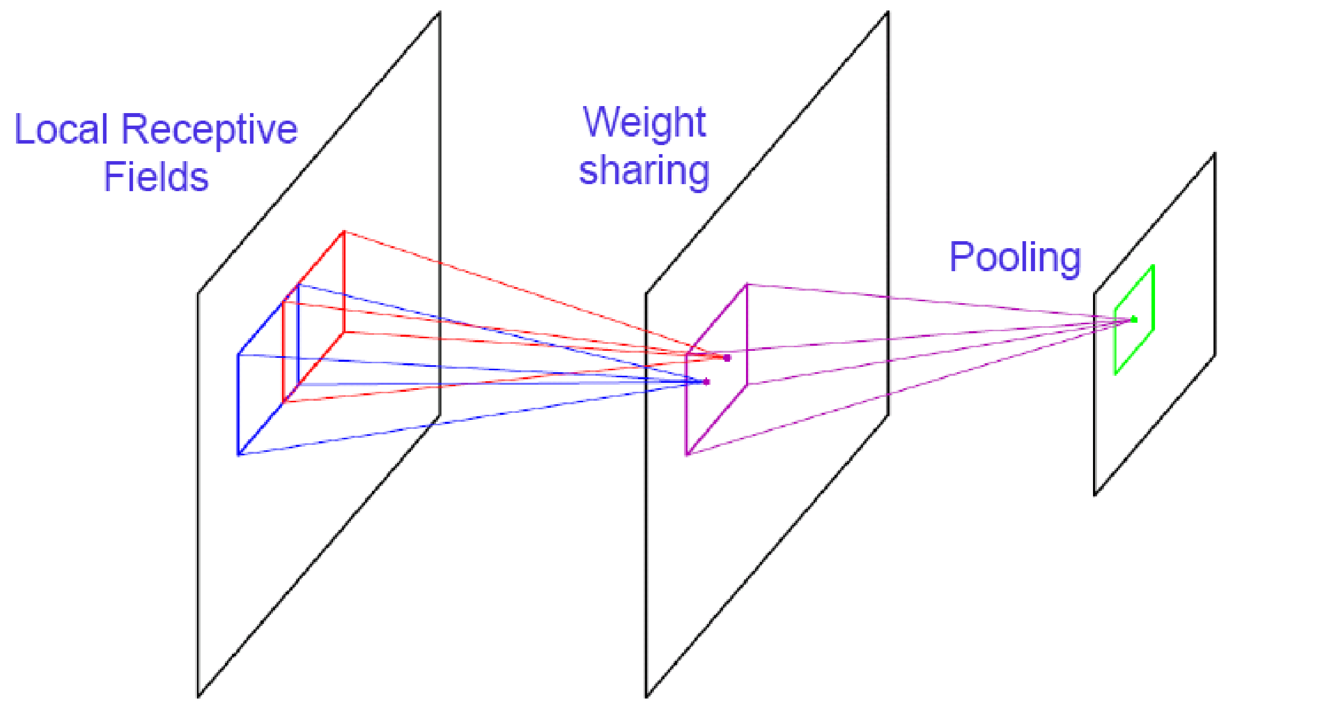
Encourage h to give small reconstruction error:

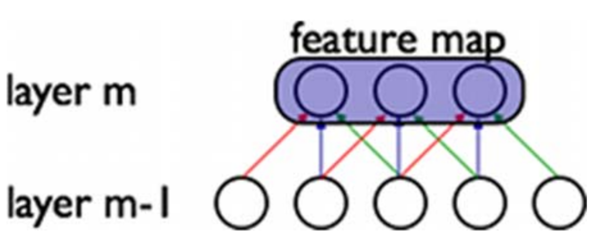
- e.g. $Loss = \sum_m ||x^{(m)} - DECODER(ENCODER(x^{(m)}))||^2$
- Reconstruction: $x' = \sigma(W'\sigma(Wx + b) + d)$

(REF)

Convolutional neural networks (CNNs)

Input image convolutional layer pooling

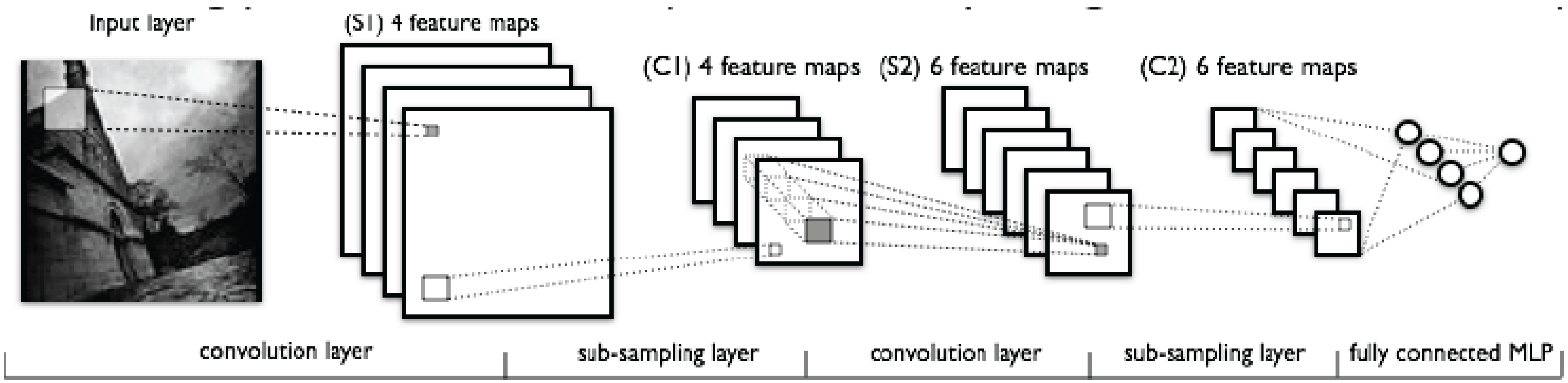


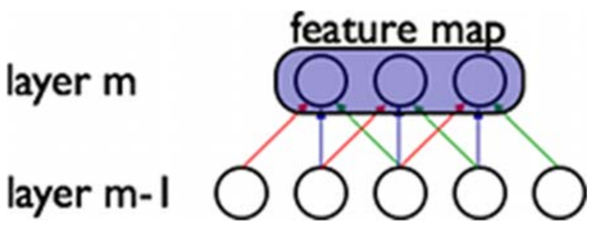


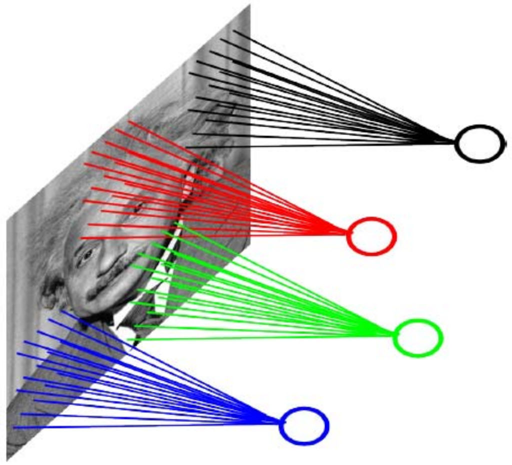
LeCun et al., 1989

Convolutional neural networks (CNNs)

Input image   convolution   pooling







LeCun et al., 1989

Generative vs discriminative approach

1. Generative deep architectures
   - describe statistical distributions of data and associated classes, P(X,Y)
   - characterise higher-order correlational structure of data for pattern analysis (suitable for holistic training of complex systems)
   - energy-based models including auto-encoders

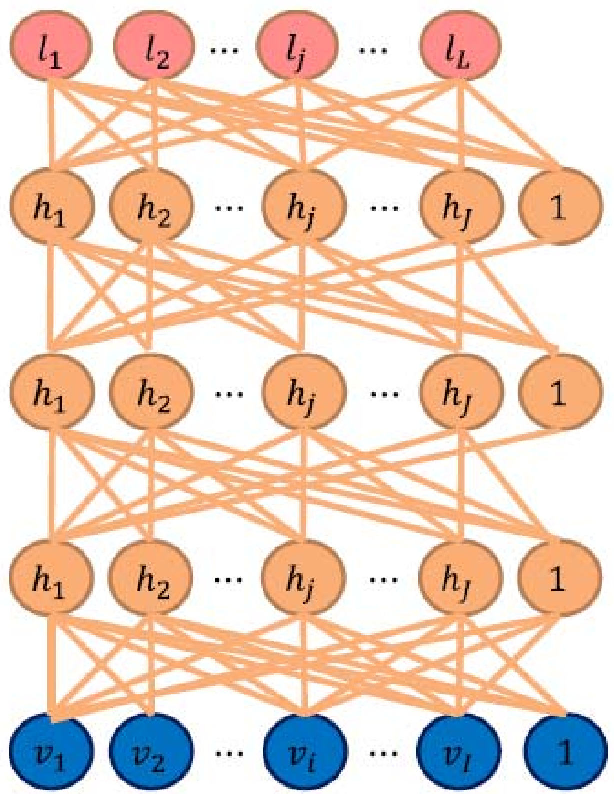
Generative vs discriminative approach

1. Generative deep architectures
   - describe statistical distributions of data and associated classes, P(X,Y)
   - characterise higher-order correlational structure of data for pattern analysis (suitable for holistic training of complex systems)
   - energy-based models including auto-encoders
2. Discriminative deep architectures
   - provide discriminative power for pattern classification by characterising the posterior distribution P(Y|X)
   - HMM, CNN, DBN-DNN

Generative vs discriminative approach

3. Hybrid deep architectures

- the goal is discrimination but is helped by the outcomes of generative modelling in deep architectures
- at the heart of early ideas for deep learning proposed by Hinton, Bengio and LeCun – unsupervised learning + supervised tuning
- deep belief networks (DBNs) are considered as a precursor component of hybrid deep architectures.

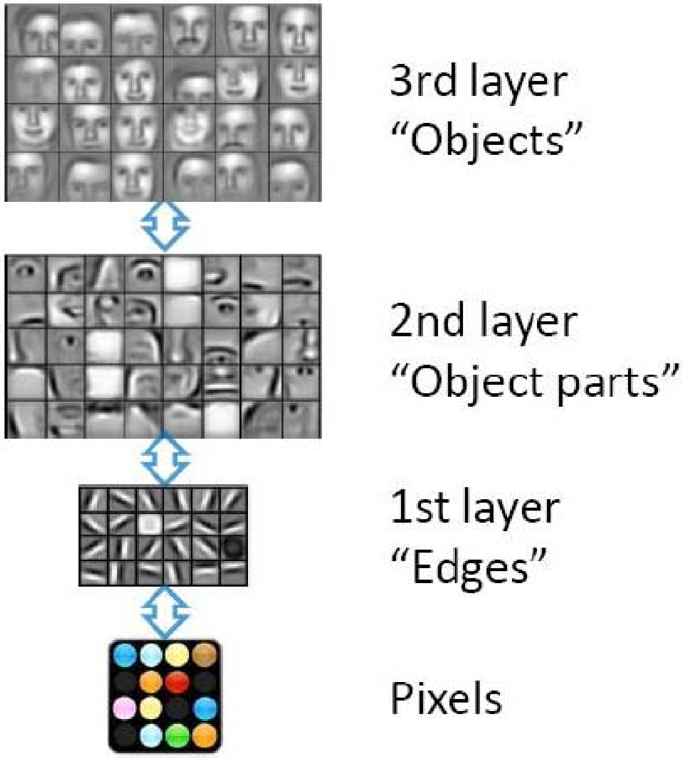


Deng, 2013

Why should we be bothered with deep learning?

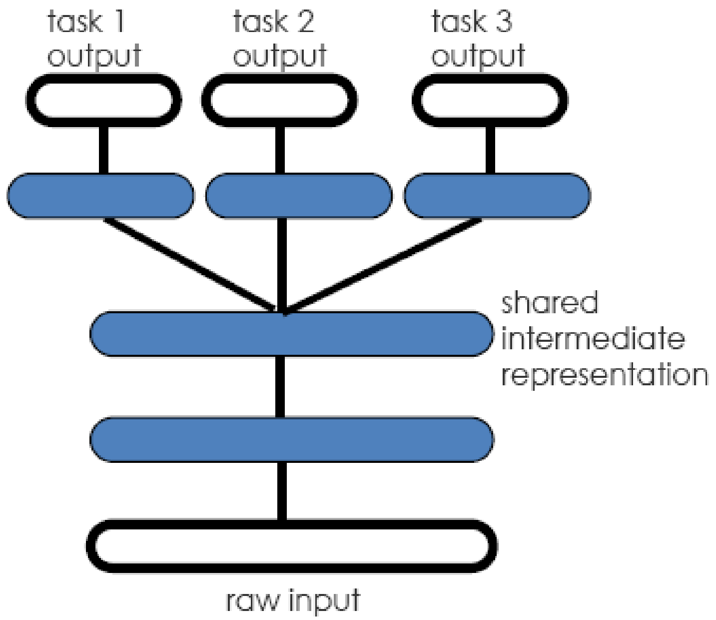
- Learning (distributed) representations
  - learning features as part of DL algorithms
  - multiple levels of abstraction and complexity (hierarchy)

NEXT LECTURE



Why should we be bothered with deep learning?

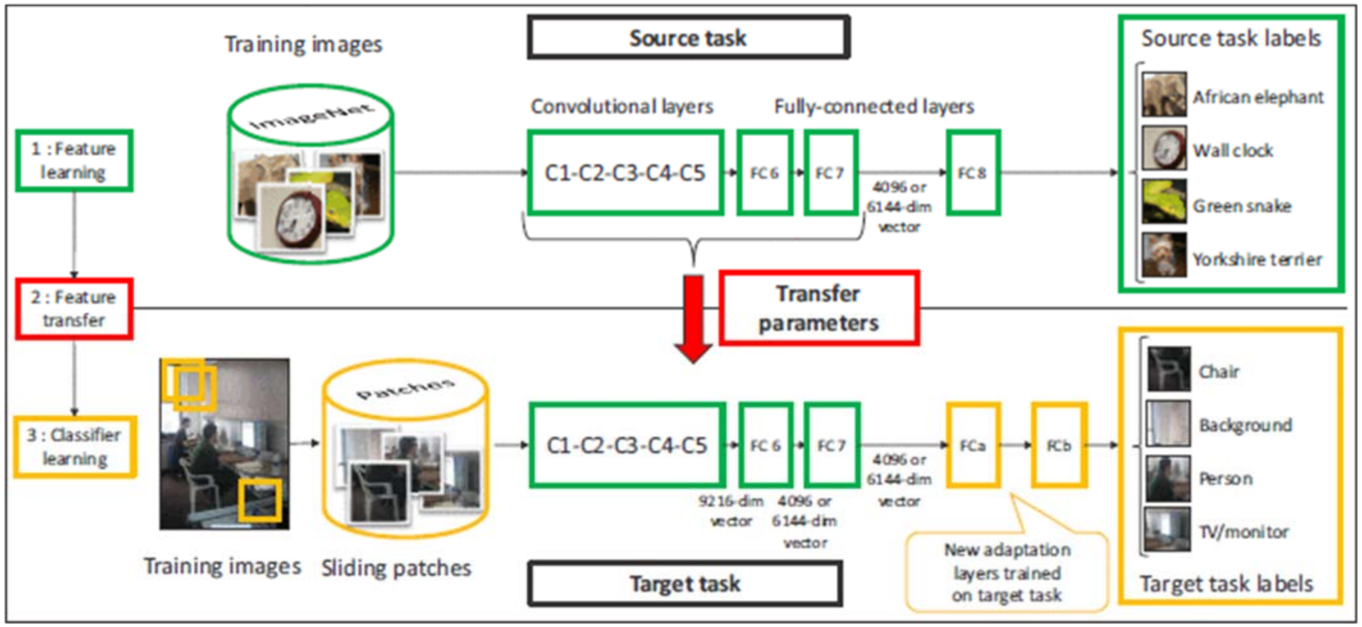
- Learning (distributed) representations
  - learning features as part of DL algorithms
  - multiple levels of abstraction and complexity (hierarchy)
  - **multi-task** or transfer learning



Bengio and Delalleau, 2013

Why should we be bothered with deep learning?

- Learning (distributed) representations
  - learning features as part of DL algorithms
  - multiple levels of abstraction and complexity (hierarchy)
  - multi-task or **transfer learning**

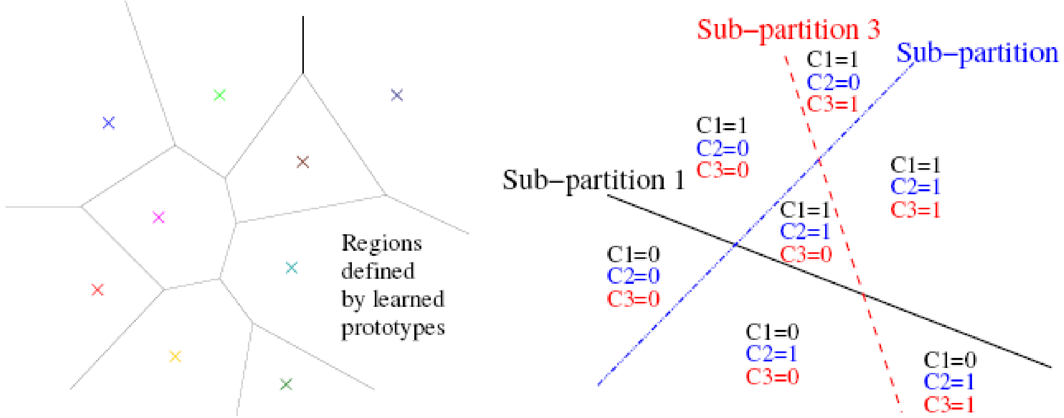


Oquab et al., 2014

Why should we be bothered with deep learning?

- Learning (distributed) representations
  - learning features as part of DL algorithms
  - multiple levels of abstraction and complexity (hierarchy)
  - multi-task or transfer learning
  - facilitates non-local generalisation (multi-clustering)

LOCAL CLUSTERING | **MULTI-CLUSTERING**



Bengio and Delalleau, 2013

Why should we be bothered with deep learning?

- Learning (distributed) representations
  - learning features as part of DL algorithms
  - multiple levels of abstraction and complexity (hierarchy)
  - multi-task or transfer learning
  - facilitates non-local generalisation (multi-clustering)
  - sparse coding

Why should we be bothered with deep learning?

- Effective use of widely available unlabeled data
  - Unsupervised pre-training or transfer learning (pre-trained networks)
  - Semi-supervised learning schemes

Why should we be bothered with deep learning?

- Effective use of widely available unlabeled data
  - Unsupervised pre-training or transfer learning (pre-trained networks)
  - Semi-supervised learning schemes
- Good performance and efficient solution (expressibility with relatively compact models)
  - better generalisation (lower error on unseen data and lower variance)
  - facilitated optimisation, distinct local minima → <u>still debatable</u>
  - capacity/complexity control

Why does deep learning seem to work?

- the notion of *”cheap learning”*
  - exponentially fewer parameters than ”generic” degrees of freedom (”swindle”)
  - we take advantage of the special nature of problems at hand:

    the laws of physics select a particular class of functions that are sufficiently “mathematically simple” to allow “cheap learning” to work

    benefitting from *smoothness*, *symmetry*, *invariance*, *locality* (local interactions boosting sparseness)

Henry W. Lin and Max Tegmark, Why does deep and cheap learning work so well?, arXiv:1608.08225

Why does deep learning seem to work?

- the notion of *”cheap learning”*
  - exponentially fewer parameters than ”generic” degrees of freedom (”swindle”)
  - we take advantage of the special nature of problems at hand:

    the laws of physics select a particular class of functions that are sufficiently “mathematically simple” to allow “cheap learning” to work

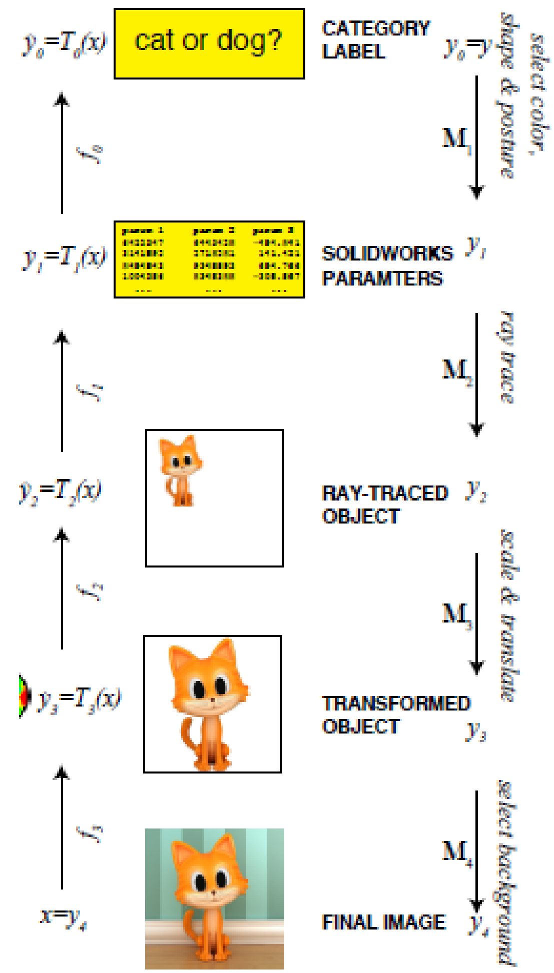
    benefitting from *smoothness*, *symmetry*, *invariance*, *locality* (local interactions boosting sparseness)

- *”no-flattening” theorems*
  - “flattening polynomials is exponentially expensive, with 2n neurons required to multiply n numbers using a single hidden layer, a task that a deep network can perform using only ∼ 4n neurons”

Henry W. Lin and Max Tegmark, Why does deep and cheap learning work so well?, arXiv:1608.08225

Why does deep learning seem to work?

- ***hierarchical*** structure of the physical world
  - hierarchy of the objects and hierarchy of generative processes to untangle
  - decomposition of the generative process into a hierarchy of simpler steps helps reduce the number of parameters ("swindle" paradox)



Henry W. Lin and Max Tegmark, Why does deep and cheap learning work so well?, arXiv:1608.08225

Key challenges ahead

I. Theoretical challenges

- insufficiently tight generalisation bounds (VC dimension)
- difficulty in theoretical handling of complexity of learning in deep architectures (”hard to prove anything”)
- is it just another (very efficient) parameterisation of solutions?

II. Visualisation, interpretation, *explanation*

- **explainable** deep networks (factors underlying inference outcomes)
- strong initiatives towards visualising and interpreting data representations (particularly in the realm of CNNs)
- how can the process of learning be monitored and controlled?

Key challenges ahead

III. Functionality

- multi-task learning, transfer learning
- multi-modal information processing
- local, incremental learning, self-organisation
- not addressing yet challenges that brain-like computing has ambition for

BUT: Is it really the direction for machine intelligence in the spirit of general AI?

Key challenges ahead

III. Functionality

- multi-task learning, transfer learning
- multi-modal information processing
- local, incremental learning, self-organisation
- not addressing yet challenges that brain-like computing has ambition for

IV. Computational challenges

- need for lowering computational costs (”equivalent” networks, performance cost etc.)
- need for better use of data and existing networks (pre-trained)
- dedicated hardware platforms

Recapitulation – key summary points

- What is the motivation for deep network architectures?
  - expressive power (*expressibility*) and compactness (*efficiency*)
  - hierarchical brain (cortex) organisation
  - multiple levels of abstraction
  - multiple levels of representations suitable for multi-task learning
- *Learning data representations* in deep learning approach vs *hand-engineering features* in traditional pattern recognition
- Learning protocol for DBNs, stacked autoencoders:
  - PHASE I: greedy layer-wise unsupervised pre-training (autoencoders or RBMs)
  - PHASE II: supervised tuning with gradient descent-like optimisation (the last layers or the entire network)

Recapitulation – key summary points

- Hypotheses about the role of unsupervised pre-training: *regularisation* vs *optimisation* hypotheses
- However, currently there is a trend to avoid pre-training and employ **ReLU units** (less risk for overfitting and local minima)
- What does DL have to offer?
  - learning data representations at multiple levels
  - hierarchy of distributed features (multi-task and transfer learning, non-local generalisation, mitigating the effect and consequences of curse of dimensionality)
  - good performance (large-scale problems) with relatively compact models --> the driving force behind R&D
  - semi-supervised learning opportunities

Recapitulation – key summary points

- Why does DL works so well?
  - “cheap learning”
  - ”no-flattening” theorems
  - hierarchical structure of the physical work
- Still plenty of challenges ahead!