Course ID 80250993 Dates: 9/16-12/23/2021 @ I-205
Tencent/VooV Meeting 521 4678 6257 (passwd 1205)
Broadcast and Reply: https://meeting.tencent.com/l/p11V4drTYoa4 (passwd 1205)

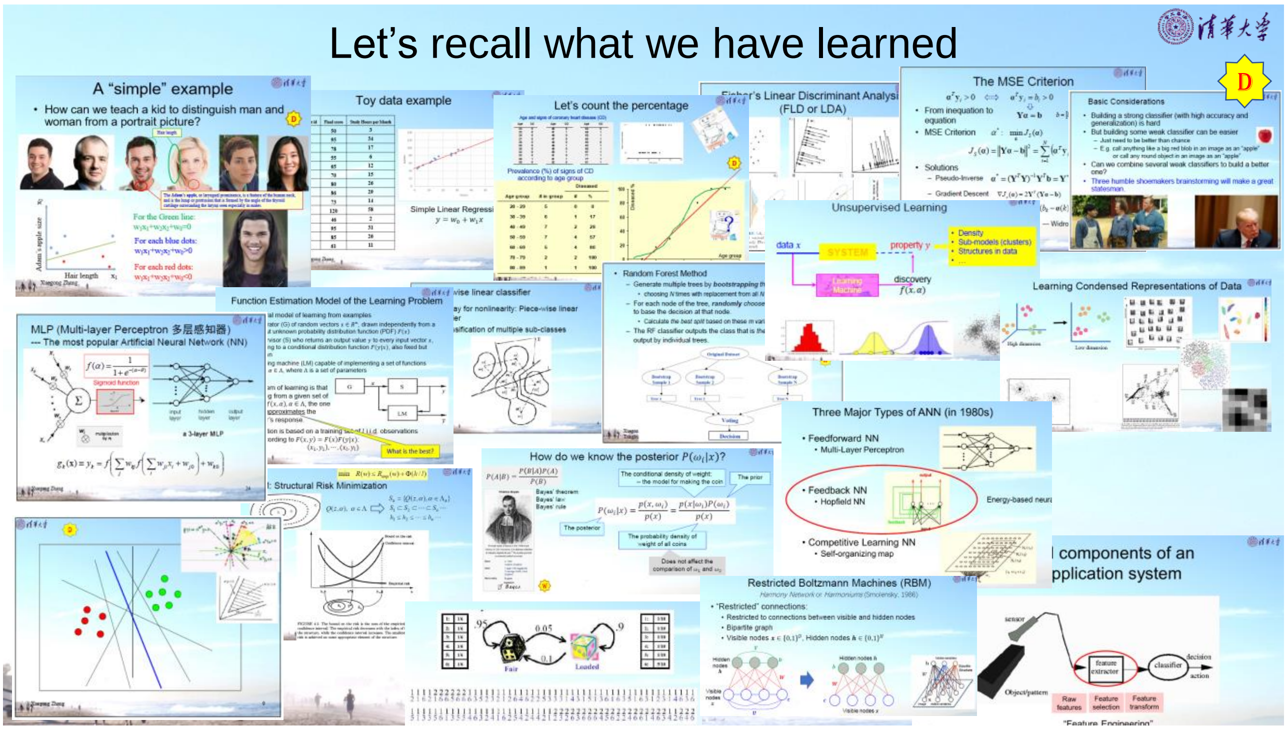
Part IV
Deep Learning and Beyond

Xuegong Zhang
Dec 2, 2021

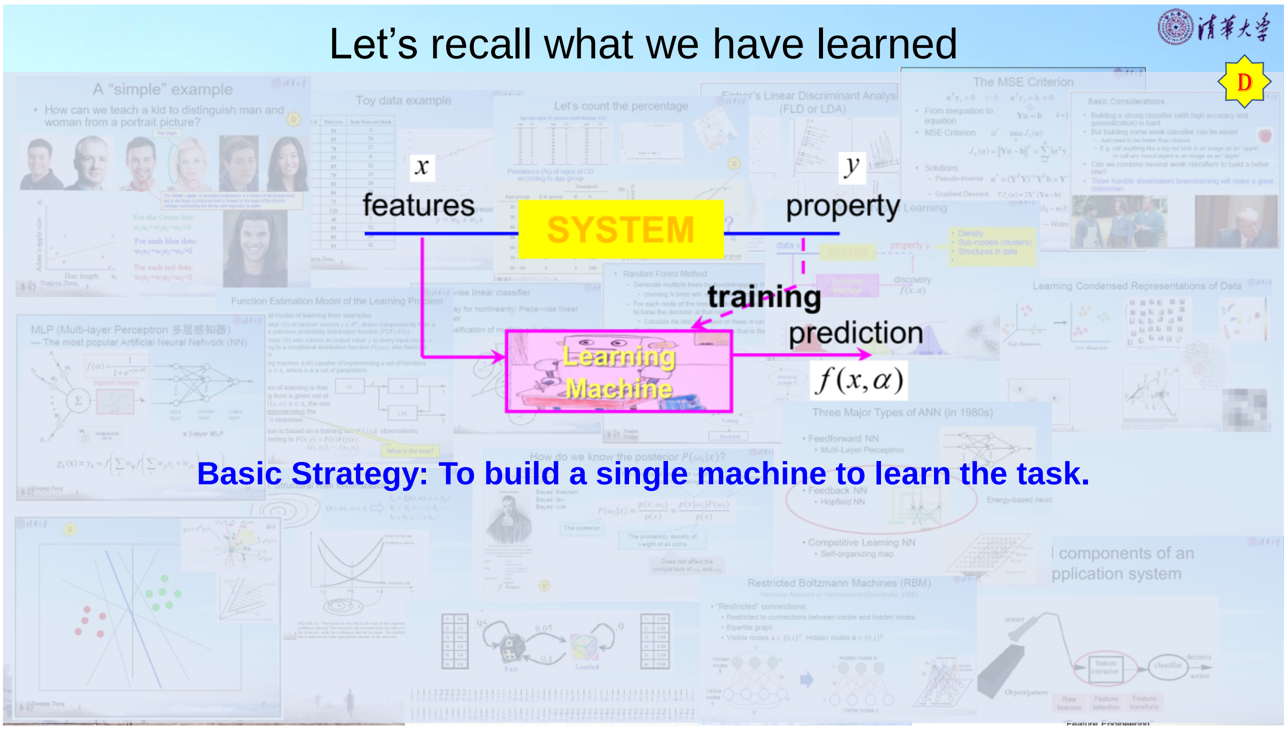
80250993 Machine Learning
@Tsinghua University

Chapter 17
Deep Neural Networks

Xuegong Zhang
Dec 2, 2021



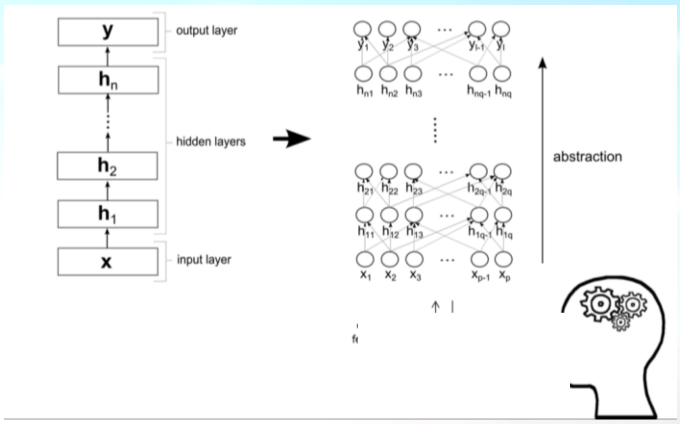
Let’s recall what we have learned
A “simple” example
How can we teach a kid to distinguish man and woman from a portrait picture?
Toy data example
Let’s count the percentage
(FLD or LDA)
The MSE Criterion
Basic Considerations
Unsupervised Learning
Learning Condensed Representations of Data
MLP (Multi-layer Perceptron 多层感知器)
--- The most popular Artificial Neural Network (NN)
Function Estimation Model of the Learning Problem
Random Forest Method
Three Major Types of ANN (in 1980s)
Feedforward NN
Feedback NN
Competitive Learning NN
Restricted Boltzmann Machines (RBM)
components of an
pplication system



Let’s recall what we have learned
x
features
SYSTEM
y
property
training
Learning Machine
prediction
$f(x,\alpha)$
Basic Strategy: To build a single machine to learn the task.

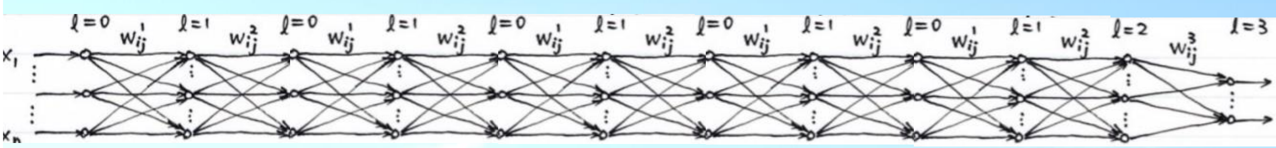
Why Deep Learning?

- The world is complicated
  - Human/animal brains do multi-level abstraction to perceive the world
  - Single “shallow” machines may not capture the underlying rules

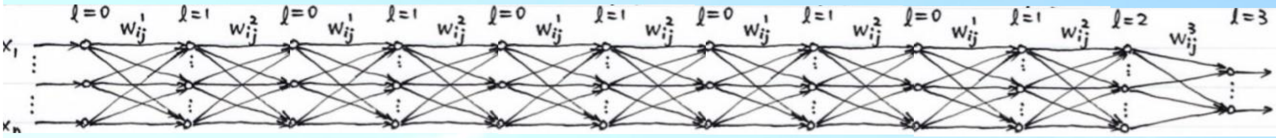


http://www.simonaullo.com/wordpress/wp-content/uploads/2013/07/deep-learning-1024x768.png

So, let’s go deeper!



Problems with “naïve” Deep Neural Networks

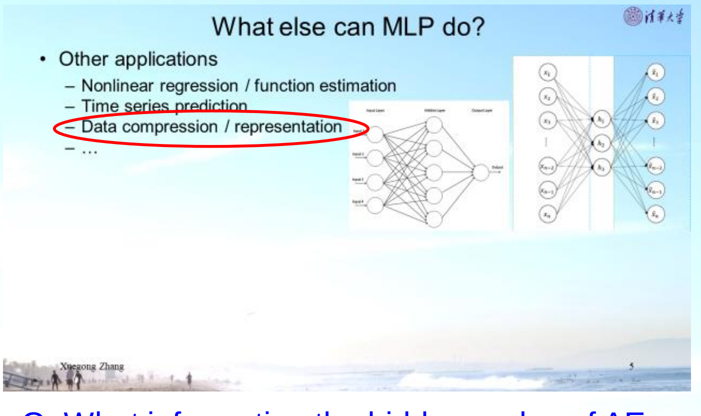


- Perform poorly on visual object recognition
  - Too many layers → too many weights to learn
    - Need many labeled data
    - Slow learning
    - Local optima
    - Gradient diffusion
    - …
  - Spatial/context/relation information not used

17.1
Deep AutoEncoder (DAE)

17.1.1
AutoEncoders

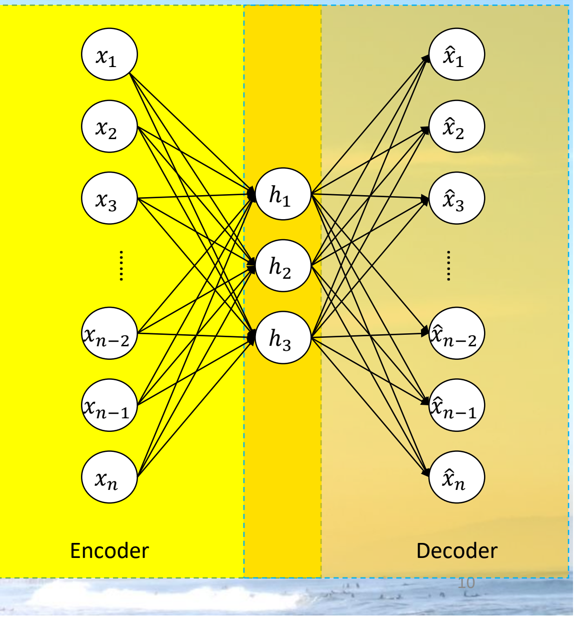
Autoencoders (AE)



Q: What information the hidden nodes of AE will capture if all activation functions are linear?

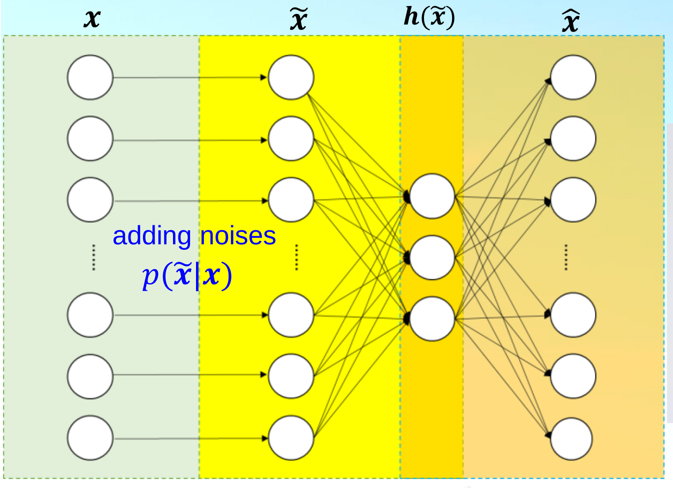
D

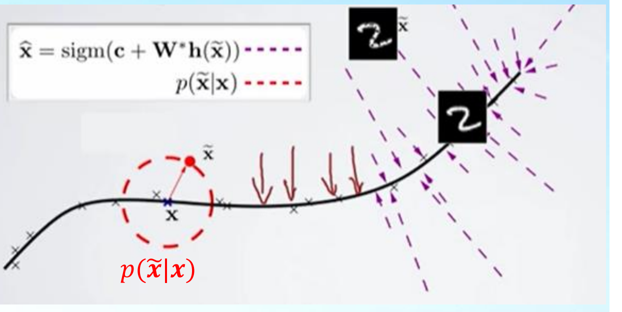
$$\hat{\boldsymbol{x}} = f(\boldsymbol{h}(\boldsymbol{x})) \triangleq \boldsymbol{x}$$



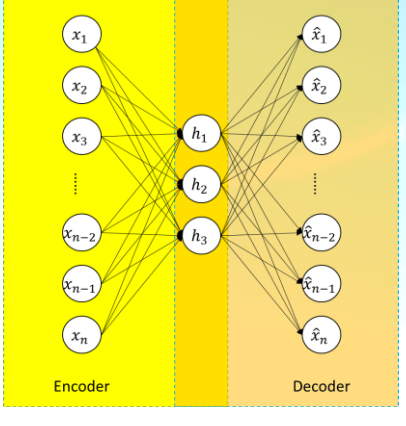
Denoising Autoencoders (DAE)

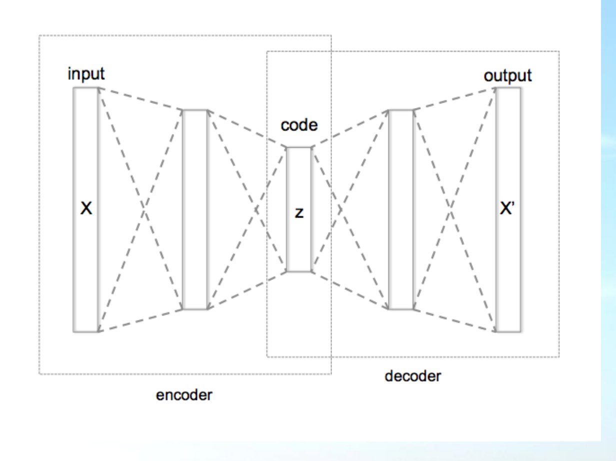
$$\hat{\boldsymbol{x}} = f(\boldsymbol{h}(\tilde{\boldsymbol{x}})) \triangleq \boldsymbol{x}$$



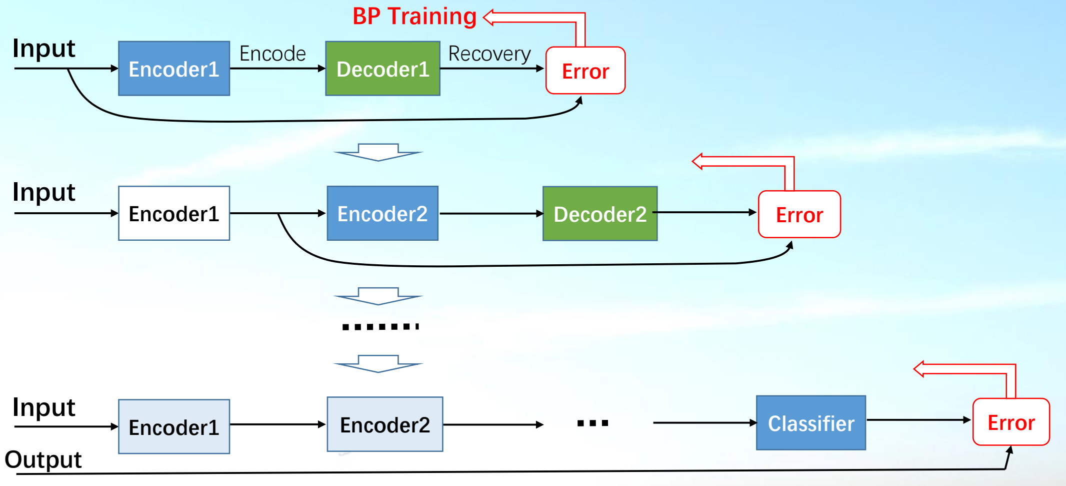


Autoencoder ➔ Deep Autoencoder





Cascade Auto-Encoder: Unsupervised pre-training for classification



17.1.2
Deep AutoEncoders (DAE)

Restricted Boltzmann Machines (RBM)

COGNITIVE SCIENCE 9, 147–169 (1985)

A Learning Algorithm for Boltzmann Machines*

DAVID H. ACKLEY
GEOFFREY E. HINTON
Computer Science Department
Carnegie-Mellon University
TERRENCE J. SEJNOWSKI
Biophysics Department
The Johns Hopkins University

The computational power of massively parallel networks of simple processing elements resides in the communication bandwidth provided by the hardware connections between elements. These connections can allow a significant

Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such "autoencoder" networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer "encoder" network

28 JULY 2006 VOL 313 SCIENCE www.sciencemag.org

Deep Autoencoder

Hinton & Salakhutdinov, *Science*, 2006

Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such "autoencoder" networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than princi

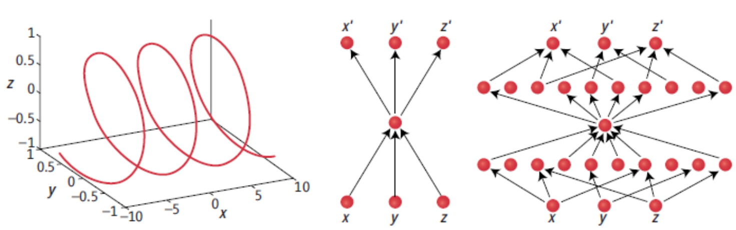
28 JULY 2006 VOL 313

New Life for Neural Networks

With the help of neural networks, data sets with many dimensions can be analyzed to find lower dimensional structures within them.

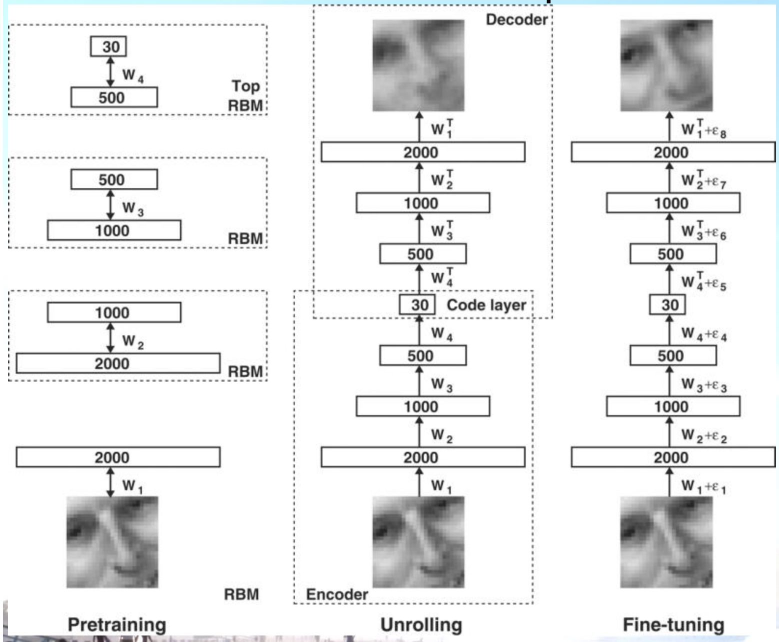
Garrison W. Cottrell

As many researchers have found, the data they have to deal with are often high-dimensional—that is, expressed by many variables—but may contain a great deal of latent structure. Discovering that structure, however, is nontrivial. To illustrate the point, consider a case in the relatively low dimension of three. Suppose you are handed a large number of three-dimensional points in random order (where each point is denoted by its coordinates along the x, y, and z axes): $\{(-7.4000, -0.8987, 0.4385), (3.6000, -0.4425, -0.8968), (-5.0000, 0.9589, 0.2837), \ldots\}$. Is there a more compact, lower dimensional description of these data? In this case, the



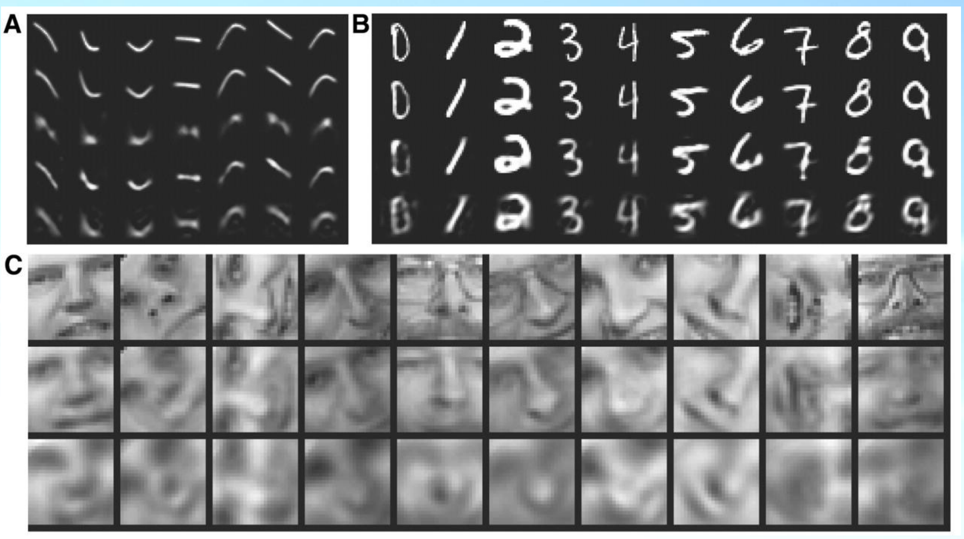
Searching for structure. (Left) Three-dimensional data that are inherently one-dimensional. **(Middle)** A simple "autoencoder" network that is designed to compress three dimensions to one, through the narrow hidden layer of one unit. The inputs are labeled x, y, z, with outputs x', y', and z'. **(Right)** A more complex autoencoder network that can represent highly nonlinear mappings from three dimensions to one, and from one dimension back out to three dimensions.

Deep Autoencoder



Hinton & Salakhutdinov, *Science*, 2006

- Encoding:
  - a stack of RBMs with cascade pre-training
- Decoding:
  - “unrolling” the encoder RBMs
- Fine-tuning:
  - BP training for the whole encoder-decoder network



Original

Reconstruction by AE

Reconstruction by PCA

Original

Reconstruction by AE

Reconstruction by PCA

Fig. 2. (**A**) Top to bottom: Random samples of curves from the test data set; reconstructions produced by the six-dimensional deep autoencoder; reconstructions by “logistic PCA” using six components; reconstructions by logistic PCA and standard PCA using 18 components. The average squared error per image for the last four rows is 1.44, 7.64, 2.45, 5.90. (**B**) Top to bottom: A random test image from each class; reconstructions by the 30-dimensional autoencoder; reconstructions by 30-dimensional logistic PCA and standard PCA. The average squared errors for the last three rows are 3.00, 8.01, and 13.87. (**C**) Top to bottom: Random samples from the test data set; reconstructions by the 30-dimensional autoencoder; reconstructions by 30-dimensional PCA. The average squared errors are 126 and 135.

Hinton & Salakhutdinov, *Science*, 2006

Deep Autoencoder

Hinton & Salakhutdinov, *Science*, 2006

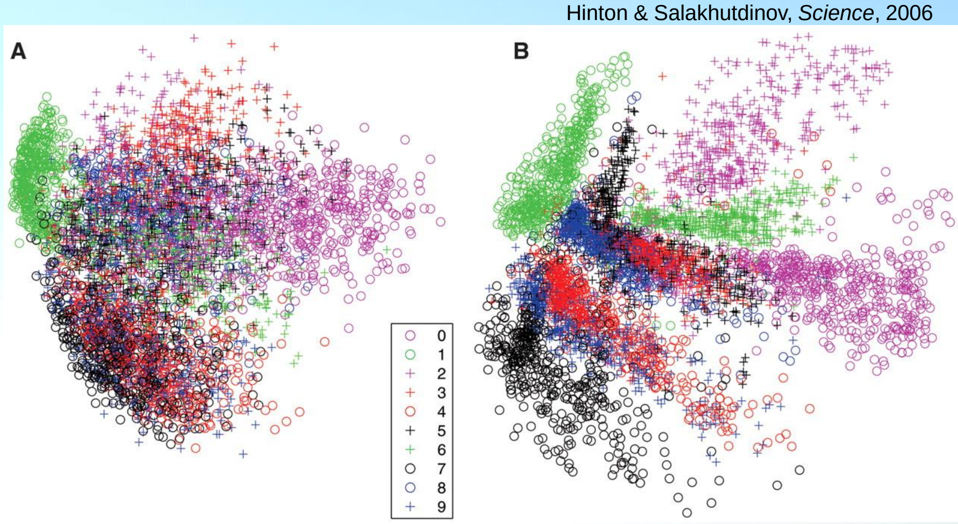
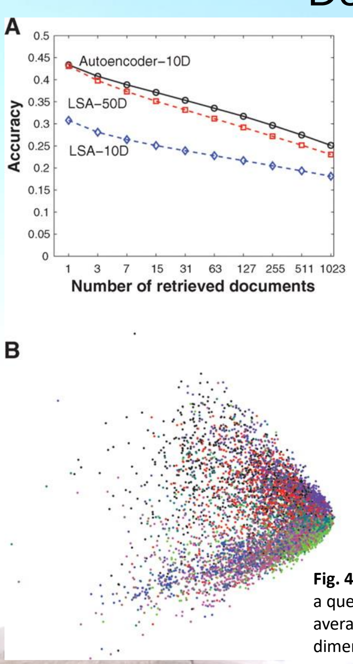


Fig. 3. (**A**) The two-dimensional codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images. (**B**) The two-dimensional codes found by a 784-1000-500-250-2 autoencoder.

Deep Autoencoder



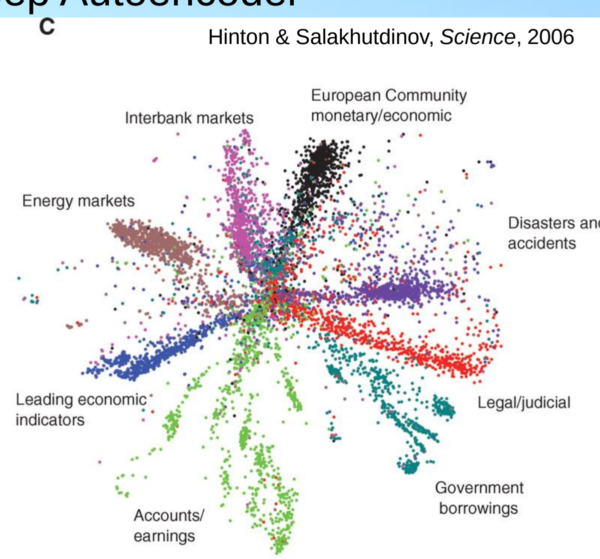
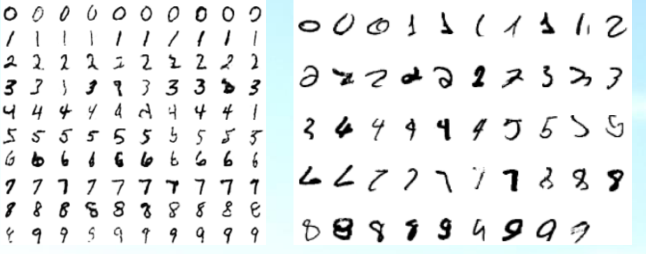


Fig. 4. (**A**) The fraction of retrieved documents in the same class as the query when a query document from the test set is used to retrieve other test set documents, averaged over all 402,207 possible queries. (**B**) The codes produced by two-dimensional LSA. (**C**) The codes produced by a 2000-500-250-125-2 autoencoder.

Application on images

- On MINST



- Generative model based on RBM's 1.25%
- Support Vector Machine (Decoste et. al.) 1.4%
- Backprop with 1000 hiddens (Platt) ~1.6%
- Backprop with 500 -->300 hiddens ~1.6%
- K-Nearest Neighbor ~ 3.3%

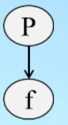
Geoffrey Hinton, UCL Turorial on Deep Bleief Nets,2009

10-second break

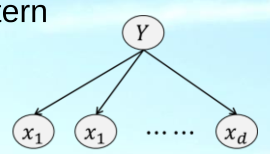
17.2
Deep Belief Networks

Recall: What are "Patterns"?

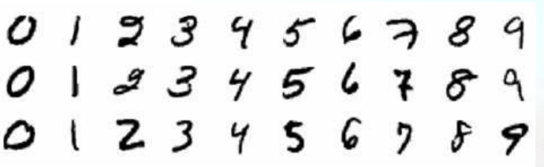
- Simple patterns
  - Single feature: distribution → pattern

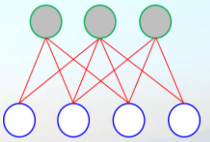


  - Multiple independent features: distributions → pattern



- Complex patterns
  - Multiple related features:
    relations among features → pattern

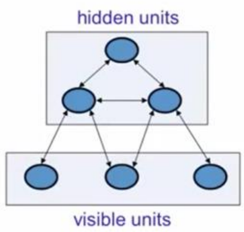


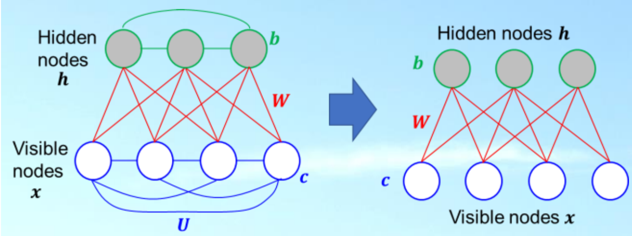


What are “Patterns”?

A different computational role for Hopfield nets

- Instead of using the net to store memories, use it to construct interpretations of sensory input.
  - The input is represented by the visible units.
  - The interpretation is represented by the states of the hidden units.
  - The badness of the interpretation is represented by the energy.





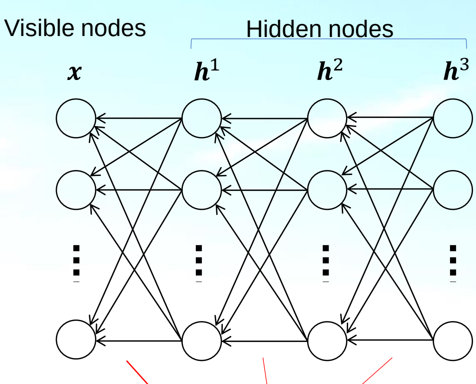
- The world is complicated
  - Human/animal brains do multi-level abstraction to perceive the world
  - Single “shallow” machines may not capture the underlying rules
- **Can we use multiple layers of hidden units to capture complex objects?**

D

Sigmoid Belief Network

(Raford M. Neal, 1992)

- Neural Network + Bayesian Network



Conditional Probabilities
(not NN connections)

$$P(x_i = 1|\boldsymbol{h}^1) = \mathrm{sigm}\left(b_i^0 + \sum_j W_{i,j}^1 h_j^1\right)$$

$$P\left(h_i^k = 1\middle|\boldsymbol{h}^{k+1}\right) = \mathrm{sigm}\left(b_i^k + \sum_j W_{i,j}^{k+1} h_j^{k+1}\right),$$
$$k = 1, \cdots, l-1$$

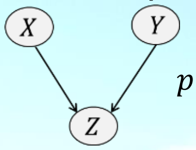
$$P\left(\boldsymbol{x}, \boldsymbol{h}^1, \cdots, \boldsymbol{h}^l\right) = P\left(\boldsymbol{h}^l\right)\left(\prod_{k=1}^{l-1} P\left(\boldsymbol{h}^k\middle|\boldsymbol{h}^{k+1}\right)\right) P\left(\boldsymbol{x}\middle|\boldsymbol{h}^1\right)$$

Powerful for modeling complicated data, but **intractable to compute the joint probability of the last layer.**

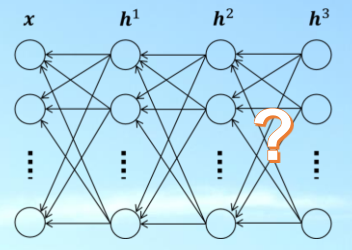
D

"Explaining Away" in sigmoid belief networks

- Conditional Dependency



$$p(x,y,z) = p(x)p(y)p(z|x,y)$$



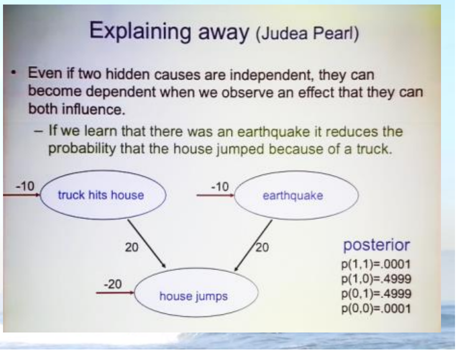
- When Z is unknown, X and Y are independent.

$$p(x,y) = \sum_{z' \in Z} p(x,y,z') = \sum_{z' \in Z} p(x)p(y)p(z'|x,y) = p(x)p(y)\sum_{z' \in Z} p(z'|x,y) = p(x)p(y)$$

- When Z is known, X and Y become dependent.

$$p(x,y|z) = p(x)p(y)p(z|x,y)/p\,(z)$$

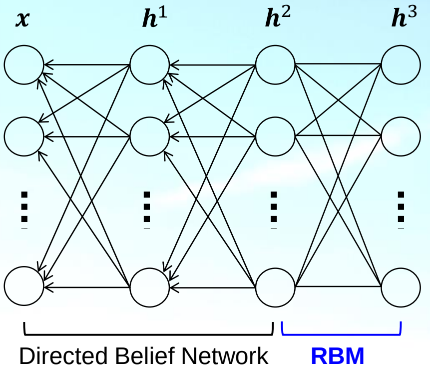
➔ Intractable to compute the joint probability of the last layer.



from Hinton's slide

Deep Belief Network (DBN)

(G.E. Hinton et al, 2006)



$$P(x_i = 1|\boldsymbol{h}^1) = \mathrm{sigm}\left(b_i^0 + \sum_j W_{i,j}^1 h_j^1\right)$$

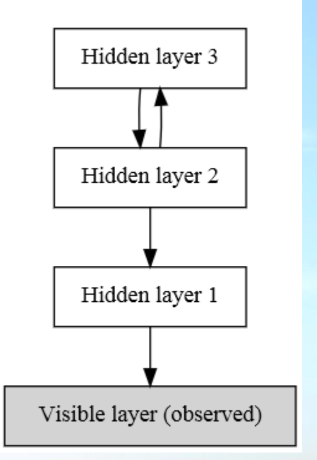
$$P\left(h_i^k = 1\middle|\boldsymbol{h}^{k+1}\right) = \mathrm{sigm}\left(b_i^k + \sum_j W_{i,j}^{k+1} h_j^{k+1}\right),$$
$$k = 1, \cdots, l-2$$

$$P(\boldsymbol{h}^{l-1}, \boldsymbol{h}^l) \propto \exp\left(\boldsymbol{c}^T \boldsymbol{h}^{l-1} + \boldsymbol{b}^T \boldsymbol{h}^l + \boldsymbol{h}^{l^T} \boldsymbol{W} \boldsymbol{h}^{l-1}\right)$$

$$P\left(\boldsymbol{x}, \boldsymbol{h}^1, \cdots, \boldsymbol{h}^l\right) = P(\boldsymbol{h}^{l-1}, \boldsymbol{h}^l)\left(\prod_{k=1}^{l-1} P\left(\boldsymbol{h}^k\middle|\boldsymbol{h}^{k+1}\right)\right)P(\boldsymbol{x}|\boldsymbol{h}^1)$$

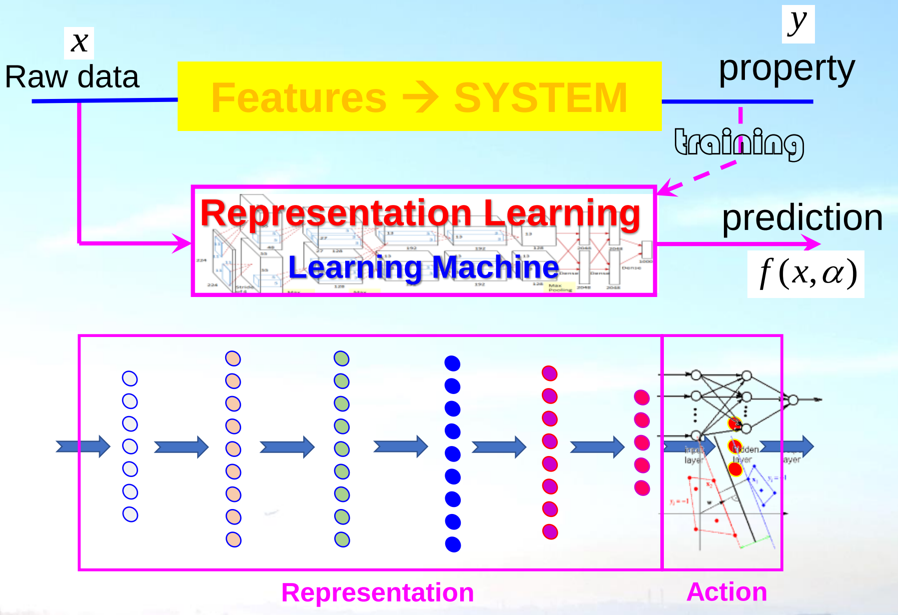
Deep Belief Network (DBN)

- Let X be a matrix of inputs, regarded as a set of feature vectors.
  1. Train a RBM on X to obtain its weight matrix W. Use this as the weight matrix between the lower two layers of the network.
  2. Transform X by the RBM to produce new data X', either by sampling or by computing the mean activation of the hidden units.
  3. Repeat this procedure with $X \leftarrow X'$ for the next pair of layers, until the top two layers of the network are reached.



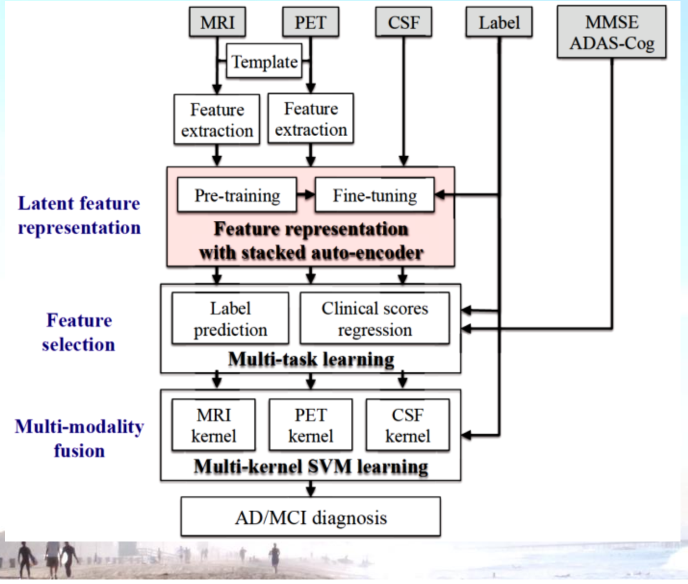
What can DBN be used for?

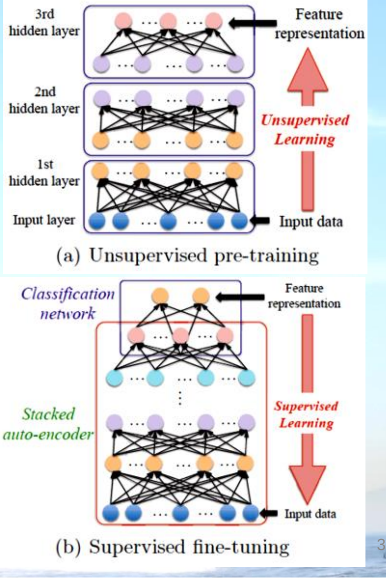
Decision based on Learned Representations



Example:
DL-based feature representation for AD/MCI classification

Suk & Shen, *MICCAI 2013*, LNCS 8150: 583-590





Discussion

- Deep neural network is a general name for many methods. There is no particular method named “deep neural network”.
- Major families of deep learning methods
  - Probabilistic models / Generative models
    - DAE, DBN, VAE, …
    - Unsupervised + supervised
  - Deterministic models
    - CNN, RNN, LSTM, …

D

Xuegong Zhang 33

Homework

- No homework for this chapter.

- It’s really time to start working on your course project now, if you haven’t started it yet.

- **Final exam: Dec.31 Friday, 9:00-11:00am, @ I-205**
- **Open-book exam**

D

Xuegong Zhang 34


See you next week
for
Convolutional Neural Networks
Xuegong Zhang
35