博士生课程：模式识别－－第五讲

人工神经网络

An Introduction to Artificial Neural Networks

向 世 明

Shiming Xiang

smxiang@nlpr.ia.ac.cn

Institute of Automation,
National Laboratory of Pattern Recognition (NLPR),
Chinese Academy of Sciences, China

Oct., 30, 2014

课前阅读

- 精读
  - G. Hinton. A Practical Guide to Training Restricted Boltzmann Machines, Tech Report, No. UTML TR 2010-003, Department of Computer Science, University of Toronto, Canada
  - Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. Gradient-Based Learning Applied to Document Recognition, Proceedings of IEEE, vol. 86, no. 11, pp. 2278-2324, 1998. (注：只要求读该文的第1至第17页即可)

课内互动

- 事先必须通读整个PPT，鼓励推导其中的所有公式
- 课堂随机点名，讲：BP算法及思想
- 对深度学习：
  - 对PPT内容，课堂随机提问
  - 课堂讲座深度学习的相关问题
  - 就特征学习和分类问题：课堂讨论如何应用和扩展

内容提要

- 介绍
  - 发展历史
  - 网络结构
- 基本模型
  - 单层感知器、多层感知器、RBF网络
- 扩展模型
  - Hopfield网络、RBM、DNN、CNN等

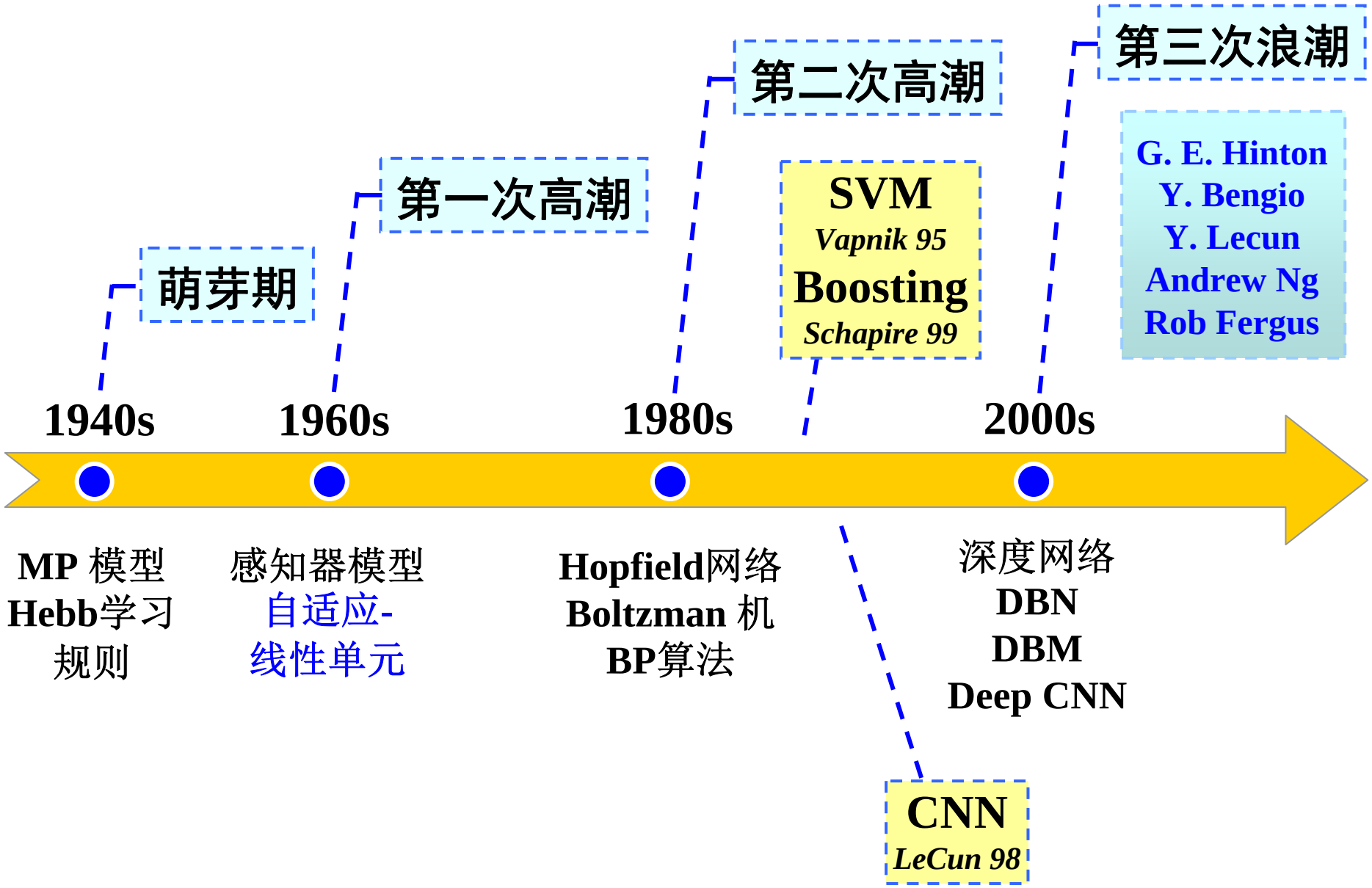
介绍

- 起源
  - Neural network inspired by **biological nervous systems**, such as our brain (生物神经系统)
- 发展历史
  - 1940年代
    - 心理学家McCulloch和数学家Pitts建立了阈值加权和模型 (1943)
    - 心理学家Hebb提出神经元之间突触联系是可变(可学习)的假说——Hebb学习律 (1949)

发展历史

- 1950年代、1960年代
  - 提出并完善了单级感知器 (Perceptron)
    - 代表性人物：Marvin Minsky，Frank Rosenblatt，Bernard Widrow
- 1980年代
  - J. Hopfield提出Hopfield网络 (1984)
  - Hinton、Sejnowsky、Rumelhart等人提出了著名的Boltzmann机 (1985)
  - Rumelhart等提出多层网络的学习算法—BP算法 (1986)
- 2000年代
  - Hinton et al. Deep Neural Networks (2007)

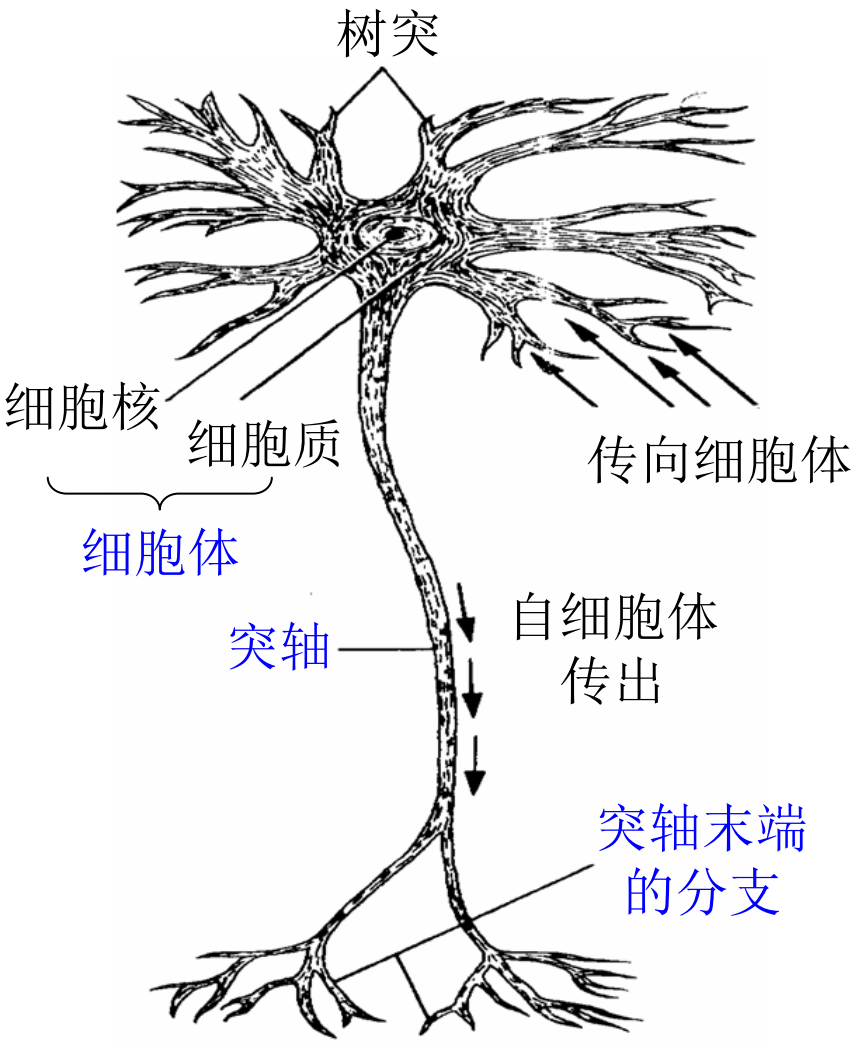
人工神经网络发展历程


第三次浪潮
G. E. Hinton
Y. Bengio
Y. Lecun
Andrew Ng
Rob Fergus
第二次高潮
SVM
Vapnik 95
Boosting
Schapire 99
第一次高潮
萌芽期
1940s
1960s
1980s
2000s
MP 模型
Hebb学习
规则
感知器模型
自适应-
线性单元
Hopfield网络
Boltzman 机
BP算法
深度网络
DBN
DBM
Deep CNN
CNN
LeCun 98

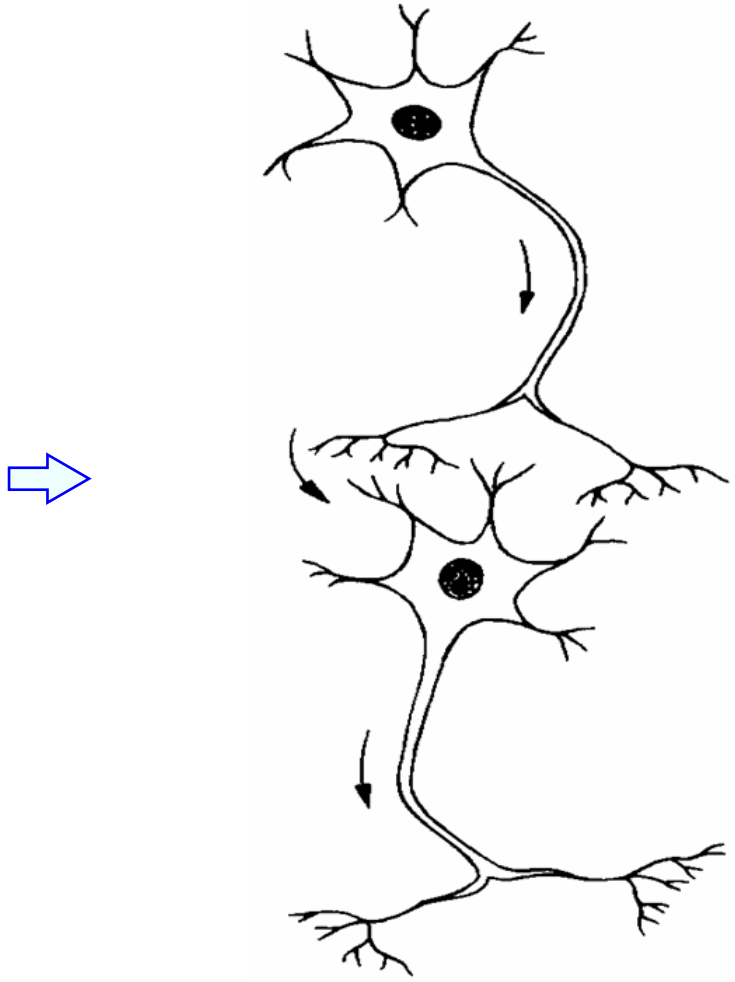
Research Circle

- Neural networks have evolved into a broader concept
  - Seems any intelligent algorithm can be cast into NNs
- Top journals for neural networks
  - IEEE Trans. on Neural Networks
  - Neural Computation
  - Neural Networks
- Other related journal or conferences
  - Journal of Machine Learning Research
  - Machine Learning
  - Neural Information Processing Systems (NIPS)
- 深度学习
  - Geoffrey Hinton, Ruslan Salakhutdinov
  - Yann LeCun, Yoshua Bengio

神经元结构

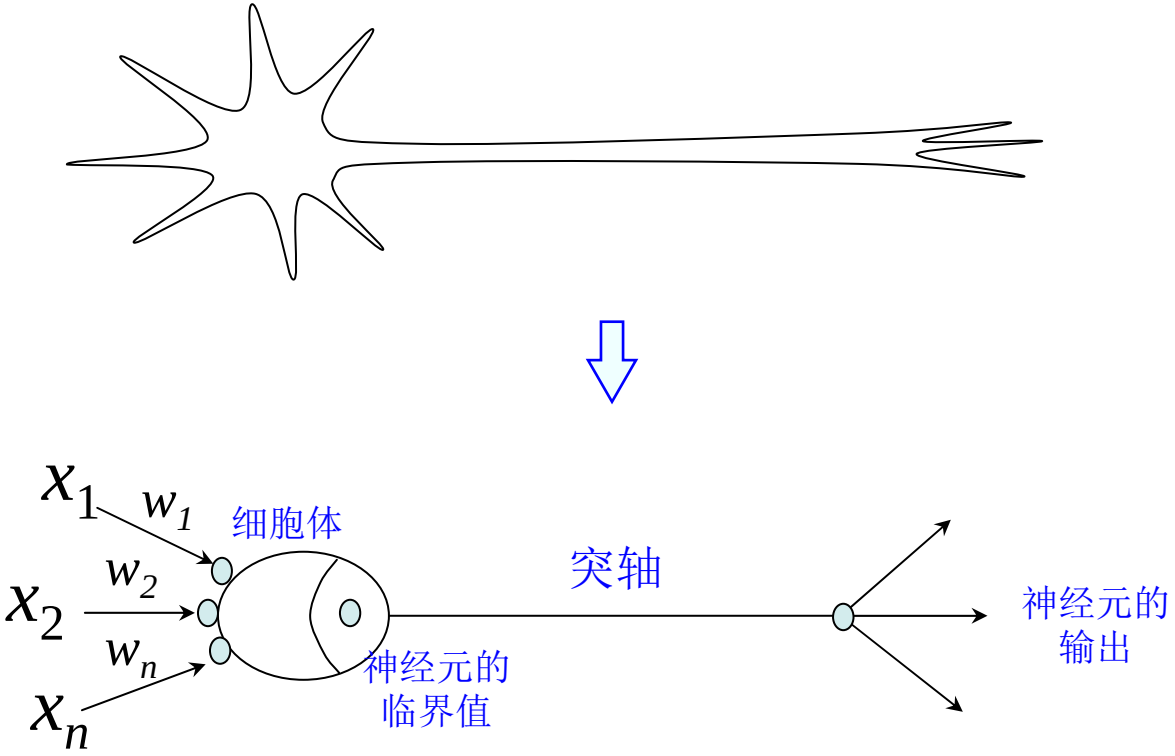


神经元之间通过突轴两两相连。
突轴记录了神经元间联系的强弱。
只有达到一定的兴奋程度，神经元才向外界传输信息。
每个神经元可抽象成一个激励函数（非线性处理）。



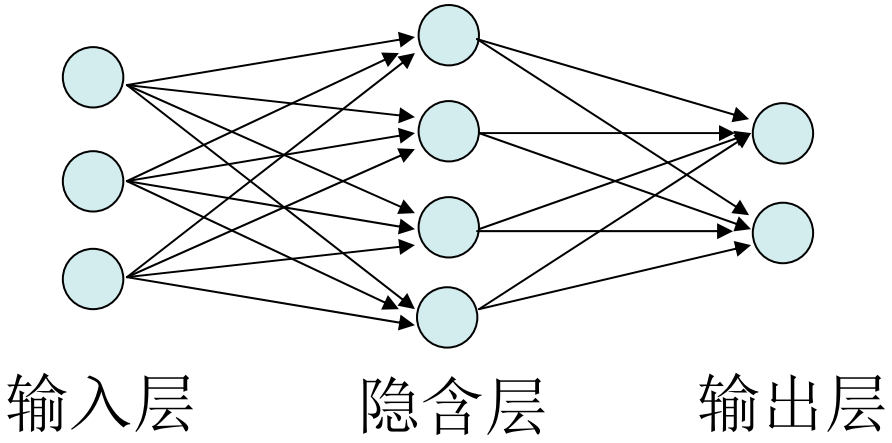
神经元模型

- 从生物学结构到数学模型



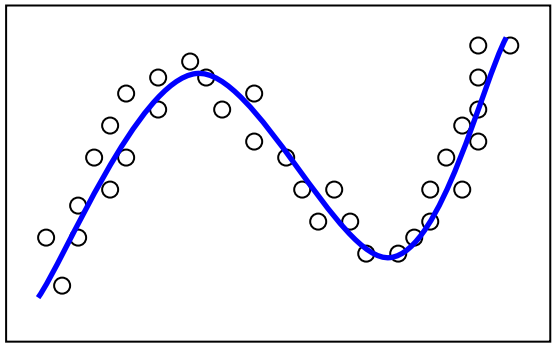
人工神经网络模型

- 描述
  - 人工神经网络由若干人工神经元结点互联而成
  - The objective of the neural network is to find the suitable connection weights so as to transform the inputs into meaningful outputs



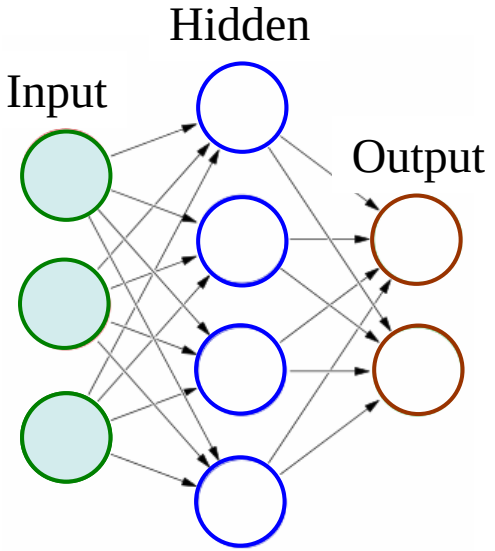
Applications

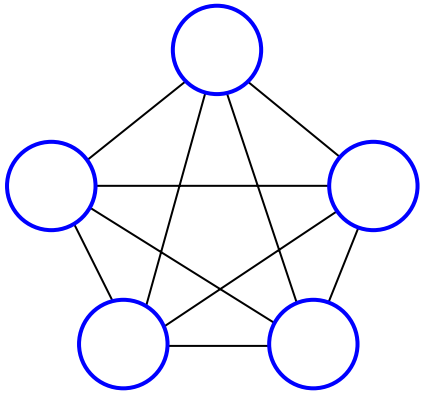
- Applications
  - Approximating function
  - Making prediction
  - Classification
  - Clustering
  - ……



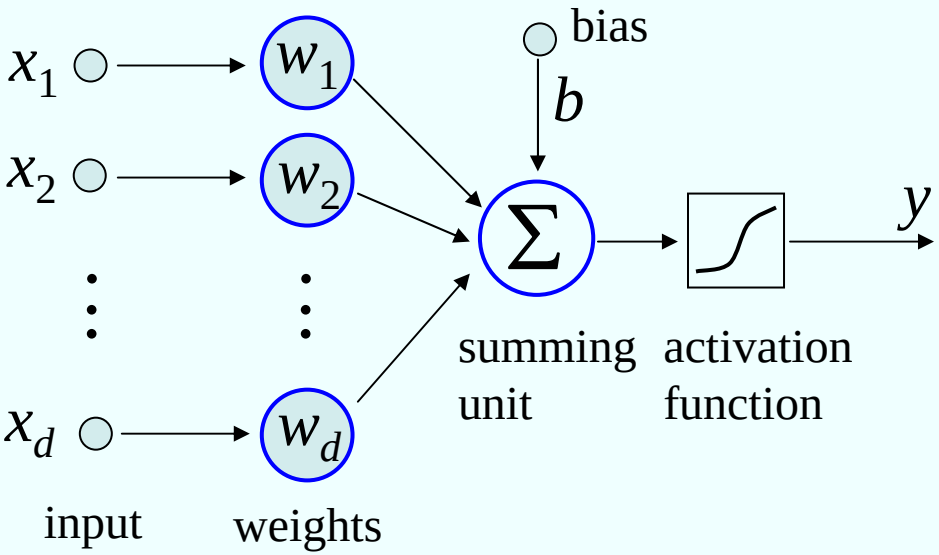
模型分类

- 联接类型
  - 前馈网络
    - 单层感知器
    - 多层感知器
    - 径向基函数网络
  - 反馈网络
    - Hopfield 网络
    - 联想存储网络
    - SOM
    - Boltzman 机





单层感知器

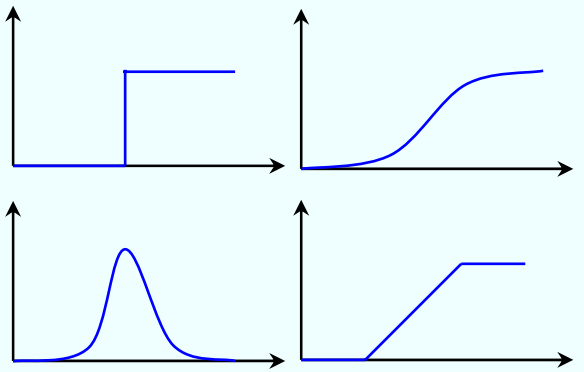


$$v = \sum_{i=1}^{d} w_i x_i + b$$

$$y = h(v)$$

Activation function:

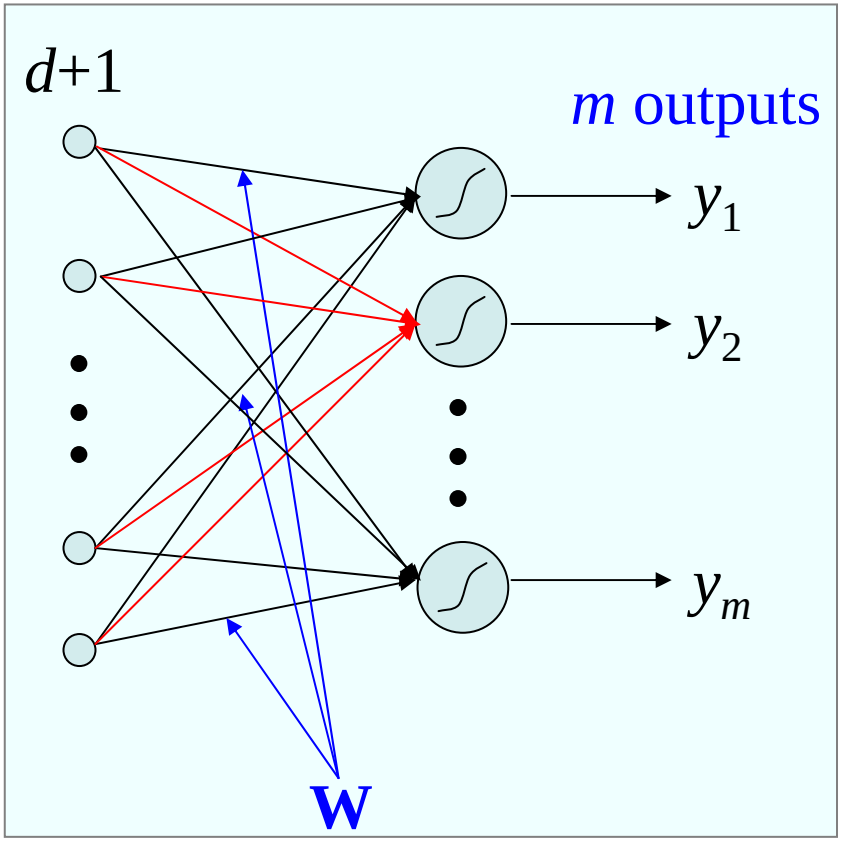
$$g(v) = \frac{1}{1+e^{-v}}$$



Informal problem definition:

Given a data set $\{(x_i, c_i) \mid i=1,2,\ldots,n\}$, where y_i denotes the prediction value (class label) for the i-th sample, the task is to find the best weight **W** to make the output of the NN for each sample fit c_i .

单层感知器



Typical setting for classification:

Training data: $\{(x_i, c_i) \mid i=1,2,\ldots,n\}$

Goal: Find the best weights $\mathbf{W}_{m\times(d+1)}$ to make the output of SLP approximate the true label of each sample

– d: dimensionality of each sample

– m: the number of classes

– True label of x_i is equivalently written as label vector:

$$\mathbf{t}_i = [0, \ldots, 0, t_{k,i}, 0, \ldots, 0]^T \in \mathrm{R}^m$$

where $t_{k,i}=1$, if $k = c_i$; 0, otherwise

$$\mathbf{W} = \arg\min_{\mathbf{w}} \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m}[y_k(\mathbf{x}_i) - t_{k,i}]^2$$

← Our task

单层感知器

- 训练
  - If the activation function is a linear function

$$y_k(\mathbf{x}) = v_k = \sum_{i=1}^{d} w_{k,i} x_i + w_{k,0} = \mathbf{w}_k^T \tilde{\mathbf{x}},\ k = 1, 2, \ldots, m$$

$$\mathbf{Y} = \mathbf{X}\mathbf{W}^T \quad (\mathbf{T} \approx \mathbf{X}\mathbf{W}^T)$$

$$\min_{\mathbf{W}} E = \min_{\mathbf{W}} \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{m} [y_k(\mathbf{x}_i) - t_{k,i}]^2 = \min_{\mathbf{W}} \frac{1}{2} \| \mathbf{X}\mathbf{W}^T - \mathbf{T} \|_F^2$$

$$\frac{\partial E}{\partial \mathbf{W}} = 0 \Rightarrow \mathbf{W}^T = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

完全等价于线性回归！

单层感知器

- 训练
  - If the activation function is a sigmoid function
    - $\mathbf{W}_{m\times(d+1)}$ can be updated by gradient descent method

$$E=\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m}[y_k(\mathbf{x}_i)-t_{k,i}]^2$$

$$\mathbf{w}_k(t+1)=\mathbf{w}_k(t)-\eta_t\frac{\partial E}{\partial \mathbf{w}_k}$$

齐次坐标

$$\frac{\partial E}{\partial \mathbf{w}_k}=\sum_{i=1}^{n}\sum_{k=1}^{m}[y_k(\mathbf{x}_i)-t_{k,i}]\cdot y_k(\mathbf{x}_i)\cdot[1-y_k(\mathbf{x}_i)]\cdot\mathbf{x}_i,\quad y_k(\mathbf{x})=g(v_k)=\frac{1}{1+e^{-\mathbf{w}_k^T\tilde{\mathbf{x}}}}$$

Chapter 4, M. Cheriet, N. Kharma, C.-L. Liu, C.Y. Suen, Character Recognition Systems: A Guide for Students and Practitioners, John Wiley & Sons

单层感知器

- Stochastic approximation （每次考虑一个训练样本）

$$E_i = \frac{1}{2}\sum_{k=1}^{m}[y_k(\mathbf{x}_i) - t_{k,i}]^2$$

Sequential updating

$$\mathbf{w}_k(t+1) = \mathbf{w}_k(t) - \eta_t \frac{\partial E_i}{\partial \mathbf{w}_k}$$

第i个样本

齐次坐标表示

$$\frac{\partial E_i}{\partial \mathbf{w}_k} = \sum_{k=1}^{m}[y_k(\mathbf{x}_i) - t_{k,i}] \cdot y_k(\mathbf{x}_i) \cdot [1 - y_k(\mathbf{x}_i)] \cdot \tilde{\mathbf{x}}_i$$

Lemma: The stochastic approximation converges to a local minimum of E with probability 1 under the following conditions:

$$\lim_{t\to\infty} \eta(t) = 0, \quad \sum_{t=1}^{\infty} \eta(t)^2 < \infty$$

单层感知器

- Any other loss function except square error?
  - Any physically meaningful loss function as long as it is smooth and continuous
  - Cross entropy loss function

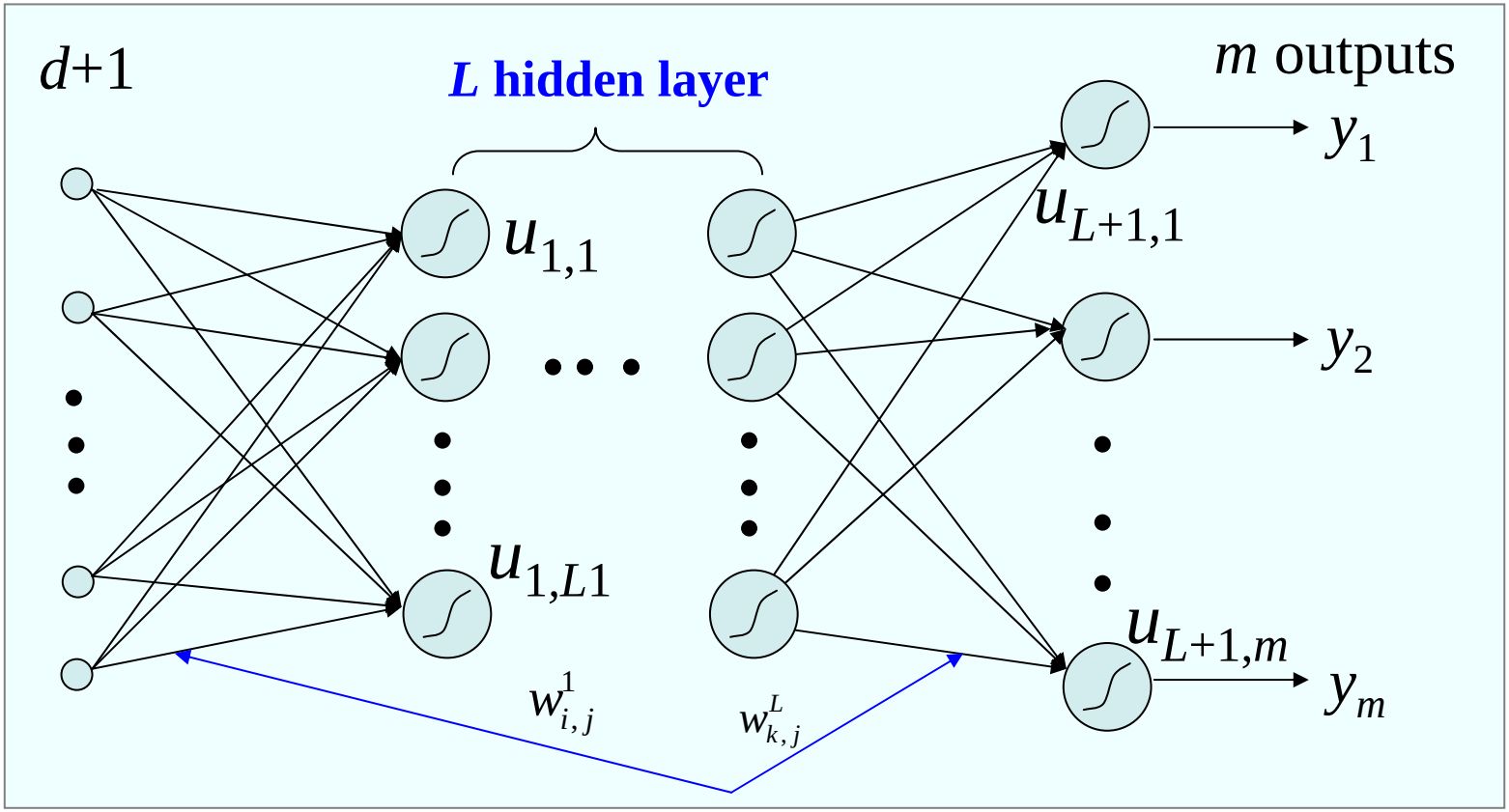
$$H(p,q) = -\sum_{x} p(x)\log(q(x))$$

$$E_h = \sum_{i=1}^{n}\sum_{k=1}^{m}\left(t_{k,i}\log[y_k(\mathbf{x}_i)] + (1-t_{k,i})\log[1-y_k(\mathbf{x}_i)]\right)^2$$

单层感知器-小结

- SLP has an analytic solution, if the activation function is linear
- Batch model
  - Update the weight only once in a cycle of feeding all the samples (一次性更新权值)
  - Disadvantages: slow convergence
- Sequential model
  - Update the weight when a sample arrives (逐样本更新权值)
  - Stochastic approximation, but fast and guaranteed to converge

多层感知器



Typical setting for classification:

Training data: $\{(x_i, c_i) \mid i=1,2,\ldots,n\}$

Goal: Find the best weights to make the output of MLP approximate the true label of each sample

Our goal:

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m}[y_k(\mathbf{x}_i) - t_{k,i}]^2$$

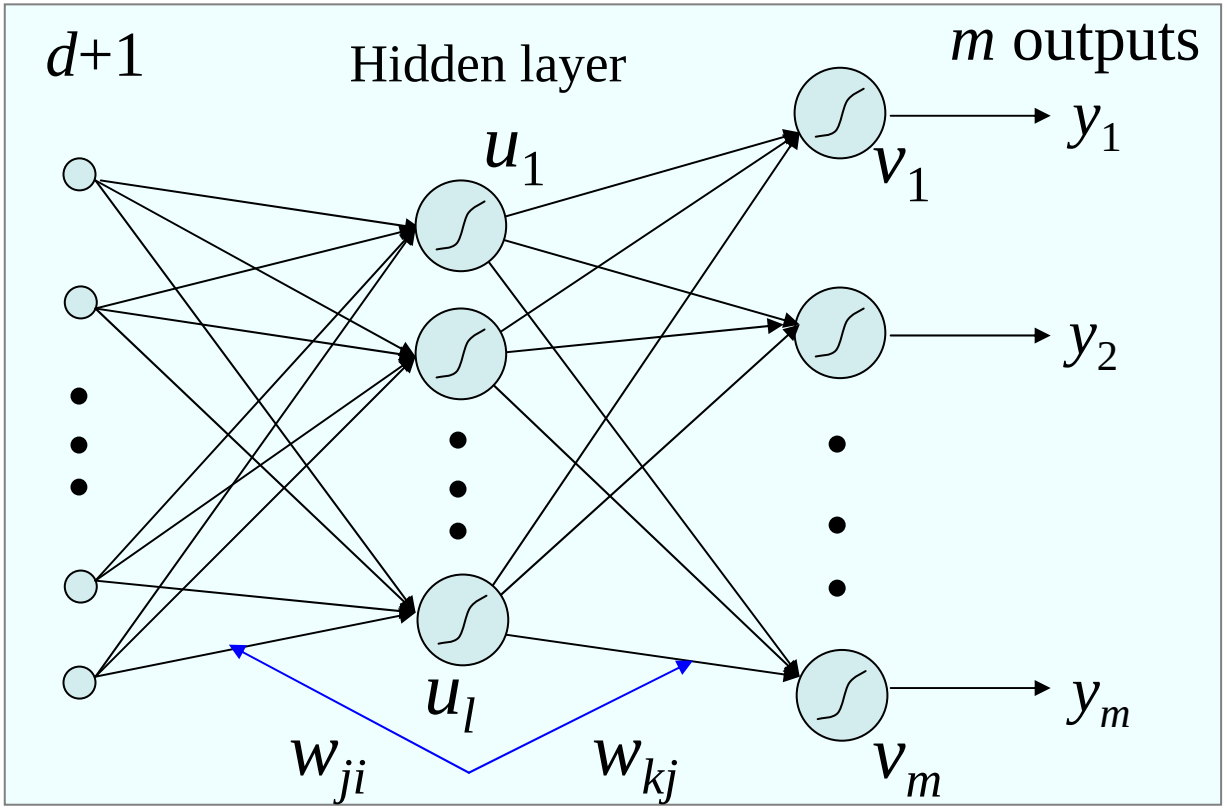
多层感知器

Problem formulation:

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m}\left(y_k(\mathbf{x}_i) - t_{k,i}\right)^2$$

逐层展开输出函数：

$$\begin{aligned} y_k(\mathbf{x}) &= g\left(u_{L,k}(\mathbf{x})\right) = g\left(\sum_{j=1}^{L_L} w_{k,j}^{L} h_{L,j} + w_{k,0}^{L}\right), \\ &= g\left(\sum_{j=1}^{L_L} w_{k,j}^{L} g\left(\sum_{i=1}^{L_{L-1}} w_{j,i}^{L-1} h_{L-1,i} + w_{k,0}^{L-1}\right) + w_{k,0}^{L}\right), \\ &= .... \end{aligned}$$



举例：

仅含有一个隐含层

能量：

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m}\left(y_k(\mathbf{x}_i) - t_{k,i}\right)^2$$

$$y_k(\mathbf{x}) = g\left(u_k(\mathbf{x})\right) = g\left(\sum_{j=1}^{l} w_{kj} h_j + w_{k0}\right)$$

$$= g\left(\sum_{j=1}^{l} w_{kj}\, g\left(\sum_{i=1}^{d} w_{ji} x_i + w_{j0}\right) + w_{k0}\right)$$

$$= g\left(\sum_{j=1}^{l} w_{kj}\, g(u_j(\mathbf{x})) + w_{k0}\right)$$

$$v_k(\mathbf{x}) = \sum_{j=1}^{l} w_{kj} h_j + w_{k0}$$

$$u_j(\mathbf{x}) = \sum_{i=1}^{d} w_{ji} x_i + w_{j0}$$

多层感知器

- Stochastic gradient descent optimization

第i个样本

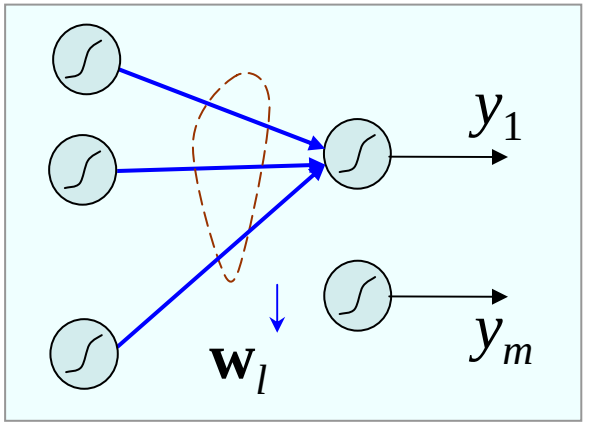
$$E_i = \frac{1}{2}\sum_{k=1}^{m}\left(y_k(\mathbf{x}_i) - t_{k,i}\right)^2 \longrightarrow \mathbf{w}_l(t+1) = \mathbf{w}_l(t) - \eta_t \frac{\partial E_i}{\partial \mathbf{w}_l},\ l=1,2,\ldots m$$

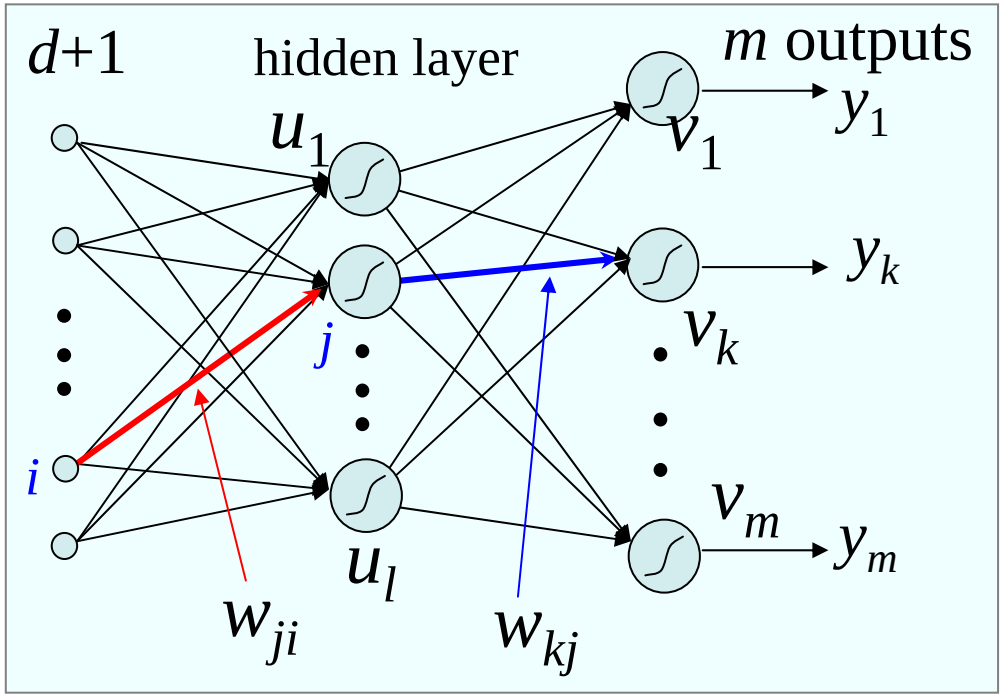
$$\frac{\partial E_i}{\partial \mathbf{w}_l} = \sum_{k=1}^{m}\left(y_k(\mathbf{x}_i) - t_{k,i}\right)\frac{\partial y_k(\mathbf{x}_i)}{\partial \mathbf{w}_l}$$

$$= \sum_{k=1}^{m}\left(y_k(\mathbf{x}_i) - t_{k,i}\right)\frac{\partial y_k(\mathbf{x}_i)}{\partial \mathbf{w}_l}$$

$$= \sum_{k=1}^{m}\left(y_k(\mathbf{x}_i) - t_{k,i}\right)\cdot y_k(\mathbf{x}_i)\cdot\left(1 - y_k(\mathbf{x}_i)\right)\frac{\partial v_k(\mathbf{x}_i)}{\partial \mathbf{w}_l}$$

$$= \sum_{k=1}^{m}\delta_k(\mathbf{x}_i)\frac{\partial v_k(\mathbf{x}_i)}{\partial \mathbf{w}_l}$$





$$v_k(\mathbf{x}) = \sum_{j=1}^{m} w_{kj} h_j + w_{k0}$$

$$h_j(\mathbf{x}) = g(u_j(\mathbf{x}))$$

$$u_j(\mathbf{x}) = \sum_{i=1}^{l} w_{ji} x_i + w_{j0}$$

$$\frac{\partial v_k(\mathbf{x})}{\partial w_{kj}} = h_j,\ j = 1,\ldots,m. \quad \longleftarrow \text{(隐层—输出层)}$$

$$\frac{\partial v_k(\mathbf{x})}{\partial w_{ji}} = w_{kj}\frac{\partial h_j}{\partial w_{ji}} = w_{kj} h_j(\mathbf{x})\left(1 - h_j(\mathbf{x})\right) x_i, \quad j = 1,2,\ldots,m;\ i = 0,1,\ldots,d$$

(输入层 ↔ 隐层)

多层感知器

Updating the weights:

$$\Delta w_{kj}(t) = -\eta_t \delta_k(\mathbf{x}) h_j, \quad k = 1, 2, ..., m; \quad j = 1, 2, ..., l$$

$$\Delta w_{ji}(t) = -\eta_t \left[\sum_{k=1}^{m} \delta_k(\mathbf{x}) w_{kj} h_j(\mathbf{x}) \left(1 - h_j(\mathbf{x})\right) \right] \cdot x_i$$

$$= -\eta_t \delta_j(\mathbf{x}) x_i,$$

$$j = 1, 2, ..., l, \quad i = 0, 1, 2, ..., d$$

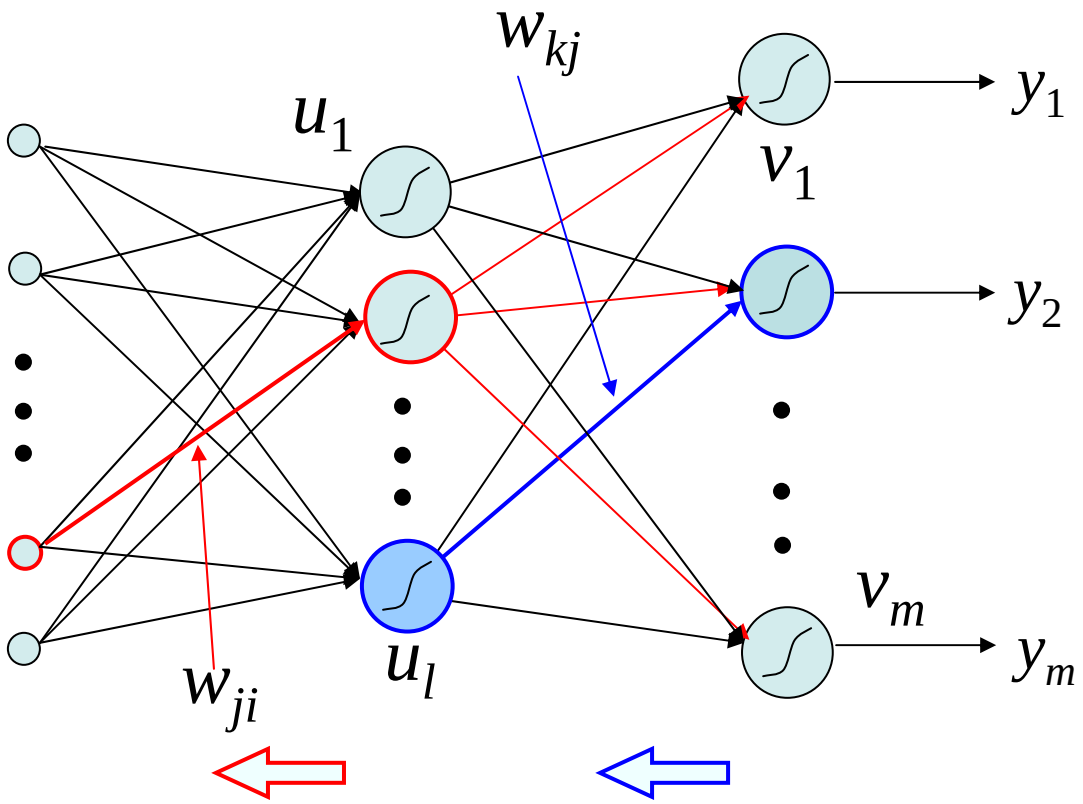
$\delta_j(\mathbf{x})$

$$\Delta w_{kj}(t) = -\eta_t \delta_k(\mathbf{x}) h_j \quad k = 1,2,\ldots,m; \quad j = 1,2,\ldots,l$$

$$\Delta w_{ji}(t) = -\eta_t \sum_{k=1}^{m} \delta_k(\mathbf{x}) w_{kj} h_j(\mathbf{x})\left(1 - h_j(\mathbf{x})\right) \cdot x_i$$

$$= -\eta_t \delta_j(\mathbf{x}) x_i, \qquad j = 1,2,\ldots,l, \quad i = 0,1,2,\ldots,d$$

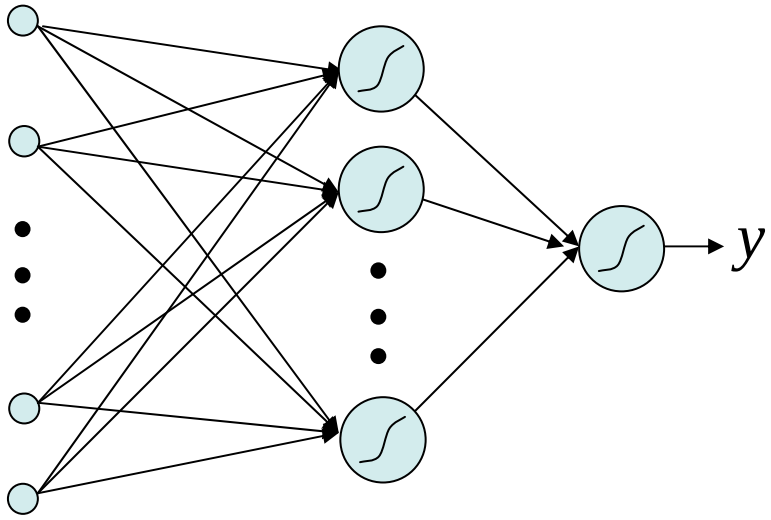
权值更新示意图



Back Propagation (收集信息,反向传播)

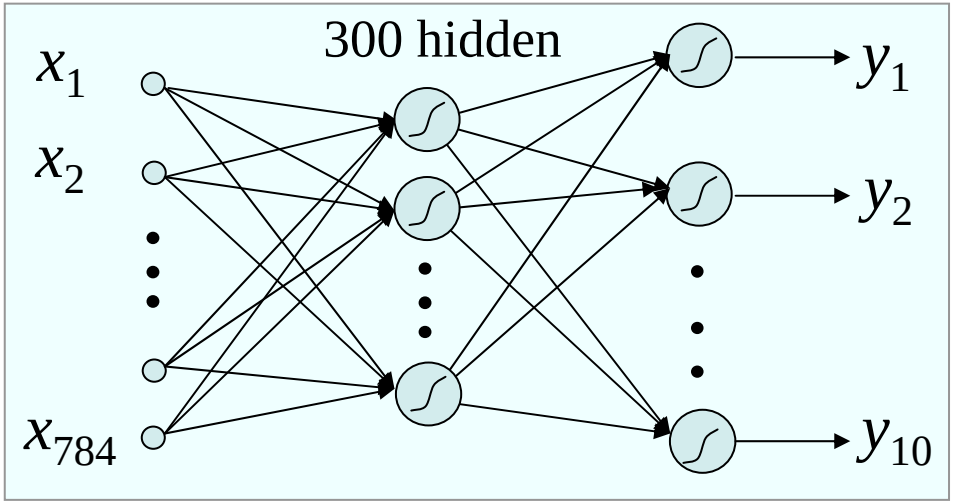
多层感知器

- For function application
  - Training the weights with one single output unit



多层感知器

- An example
  - Target: Recognize 10 digits
  - Input data, MNIST (each sample of 28×28=784d)
    - 60k training samples
    - 10k test samples



Step1: Initialization --Randomly sort the training set
--Randomly initialize the weights w_{ji} and w_{kj}
Step2: Do - input each training sample sequentially to the input nodes
-Update the weights
Until converged

Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 86(11):2278-2324, 1998

Test for a new sample $\mathbf{z}$:

$$c(\mathbf{z}) = \arg\min_k y_k(\mathbf{z})$$

多层感知器

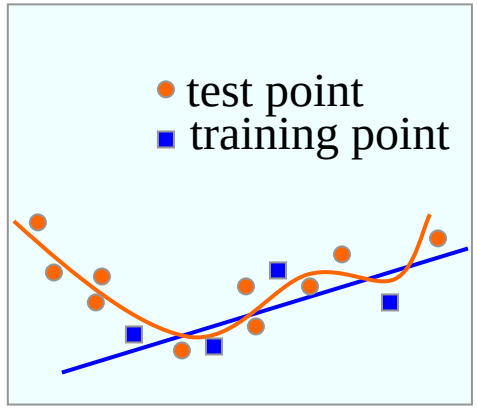
- Some practical issues
  - Preprocessing is important
    - Normalize each dimension of data to [-1, 1]
    - Normalize each dimension of data by: $x = (x-m)/\sigma$
  - Adapting the learning rate
    - $\eta_t = 1/t$

MLP: Discussion

- Good expression ability
  - A three-layer network can map any continuous function exactly from *d* input variables to an output variable *
  - A benchmark classifier in many pattern recognition tasks (常用的比较方法)
  - Simple to implement

*: **C. Bishop. Neural Networks for Pattern Recognition. Oxford University Press, 1995.**

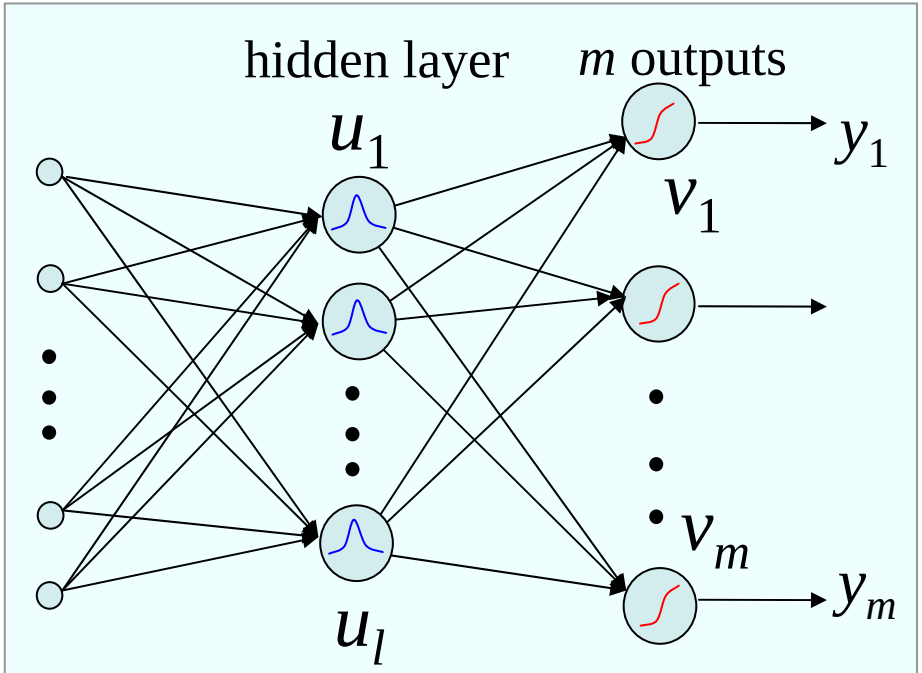
MLP: Discussion



- 缺点
  - Slow convergence
    - Using momentums： $\Delta \mathbf{w}_i(t+1) = \gamma \mathbf{w}_i(t) - \eta_t \frac{\partial E_i}{\partial \mathbf{w}_i}$
      - »
    - Using Newton's method (second order method)
  - Over-fitting
    - Regularization on the smoothness of weight *W*
    - Fewer but sufficient number of hidden nodes
  - Local minimum
    - Try different initial points for NN
    - Perturbation

RBF Network

- **A three-layer NN** with each node performing a localized non-linear function (usually RBF/Gaussian) in the hidden layer



$$u_j(\mathbf{x}) = \exp\left(\frac{-\|\mathbf{x}-\mathbf{m}_j\|^2}{2\sigma_j^2}\right), j = 1:l$$

$$v_k(\mathbf{x}) = \sum_{j=1}^{l} w_{kj} u_j(\mathbf{x}) + w_{k0} = \sum_{j=0}^{l} w_{kj} u_j(\mathbf{x})$$

$$y_k(\mathbf{x}) = g(v_k(\mathbf{x})) = \frac{1}{1+e^{-v_k(\mathbf{x})}}$$

Still optimize **W** by gradient descent when σ_j and m_j are known

$$w_{kj}(t+1) = w_{kj}(t) - \eta(t)\frac{\partial E_i}{\partial w_{kj}}$$

$$\frac{\partial E_i}{\partial w_{kj}} = \left(y_k(\mathbf{x}_i) - t_{k,i}\right) y_k(\mathbf{x}_i)\left(1 - y_k(\mathbf{x}_i)\right) u_j(\mathbf{x}_i) = \delta_k(\mathbf{x}_i) u_j(\mathbf{x}_i)$$

RBF Network

- When σ_j and m_j are known
  - Update them by gradient descent

$$m_j(t+1) = m_j(t) - \eta(t)\frac{\partial E_i}{\partial m_j}$$

$$\frac{\partial u_j(\mathbf{x}_i)}{\partial m_j} = -\frac{u_j(\mathbf{x}_i)}{2\tau_j}(\mathbf{x}_i - \mathbf{m}_j)$$

$$\frac{\partial E_i}{\partial m_j} = \sum_{k=1}^{m} w_{kj}\delta_k(\mathbf{x}_i)\frac{\partial u_j(\mathbf{x}_i)}{\partial m_j}$$

$$\sigma_j(t+1) = \sigma_j(t) - \eta(t)\frac{\partial E_i}{\partial \sigma_j}$$

$$\frac{\partial E_i}{\partial \tau_j} = \sum_{k=1}^{m} w_{kj}\delta_k(\mathbf{x}_i)\frac{\partial u_j(\mathbf{x}_i)}{\partial \tau_j}, \quad (\tau_j = \sigma_j^2)$$

$$\frac{\partial u_j(\mathbf{x}_i)}{\partial \tau_j} = -\frac{u_j(\mathbf{x}_i)}{2\tau_j^2}\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^2$$

RBF Network

- Very good model for pattern classification or function approximation
  - Like MLP, RBF can form an arbitrarily close approximation to any continuous nonlinear mapping
  - Usually converges faster than MLP
  - Can perform better than MLP in some cases

D. Hush and B. Horne, Progress in Supervised Neural Networks, IEEE Signal Processing Magazine, 8-39, 1993

RBF Network

- Another viewpoint
  - RBF is learning the best basis and the best weight coefficients
  - Assume the number of hidden node is equal to the sample size and $\mathbf{m}_j$ equal each training sample $\boldsymbol{x}_j$, then it is very similar to parzen window
  - If the weight $\mathbf{W}$ is further regularized by $\|\mathbf{W}\|^2$ → Least Square Support Vector Machine (with RBF kernel)

Neural networks are something related to SVM but without a regularization term, making it easily over-fitting

深度学习

深度学习浪潮！

Deep Learning Since 2006

materials are identical for all configurations. The blue bars in Fig. 1 summarize the measured SHG signals. For excitation of the *LC* resonance in Fig. 1A (horizontal incident polarization), we find an SHG signal that is 500 times above the noise level. As expected for SHG, this signal closely scales with the square of the incident power (Fig. 2A). The polarization of the SHG emission is nearly vertical (Fig. 2B). The small angle with respect to the vertical is due to deviations from perfect mirror symmetry of the SRRs (see electron micrographs in Fig. 1). Small detuning of the *LC* resonance toward smaller wavelength (i.e., to 1.3-μm wavelength) reduces the SHG signal strength from 100% to 20%. For excitation of the Mie resonance with vertical incident polarization in Fig. 1D, we find a small signal just above the noise level. For excitation of the Mie resonance with horizontal incident polarization in Fig. 1C, a small but significant SHG emission is found, which is again po-

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

Neural networks are coming back!

深度学习浪潮！

- **Answer from G. Hinton， 2012.10**

72%, 2010

74%, 2011

85%, 2012

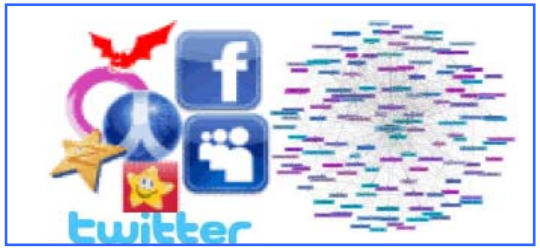
深度学习浪潮！

- 时代背景-数据爆炸

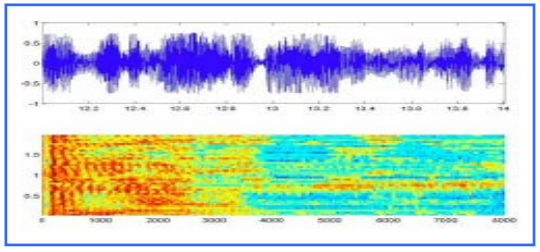
图像数据 

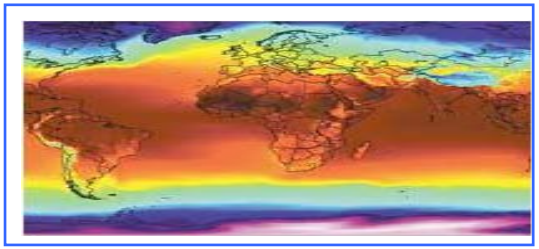
产品推荐

文本数据 



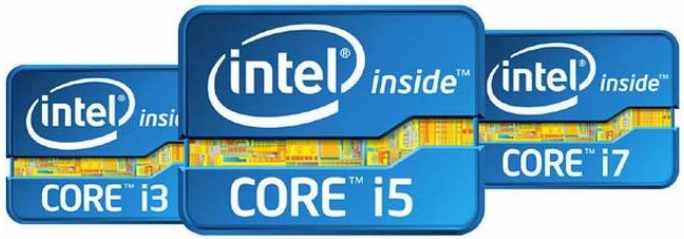
社交网络

语音数据 


科学计算

深度学习浪潮！

- 时代背景-计算性能提升



深度学习浪潮！

- **Academicals**
  - –Machine Learning @ UofT
    - Geoffrey E. Hinton, Rich Zemel, Ruslan Salakhutdinov, Brendan Frey, Radford Neal
  - Université de Montréal: LISA Lab
    - Yoshua Bengio, Pascal Vincent, Aaron Courville, Roland Memisevic
  - New York University: Yann Lecun’s andRob Fergus’ group
  - Stanford University: Andrew Ng’s group
  - UBC –Nando de Freitas’s group
  - UC Berkeley –Bruno Olshausen’s group
  - University of Washington –Pedro Domingos’ group
  - University of Michigan –Honglak Lee’s group

深度学习浪潮！

- Industrial
  - Google: Jeff Dean, Samy Bengio, Jason Weston, etc.
  - MSR: Li Denget al.
  - Facebook: Yann Lecunet al.
  - IBM Research –Brian Kingsburyet al.
  - Baidu: Kai Yu et al.
  - Huawei: Xiaogang Wang et al.
  - Alibaba:
  - Tencent:

深度学习浪潮！

- 深度学习

“Deep learning is a set of algorithms in machine learning that attempt to learn in multiple levels, corresponding to different levels of abstraction. It typically uses artificial neural networks. The levels in these learned statistical models correspond to distinct levels of concepts, where higher-level concepts are defined from lower-level ones, and the same lower-level concepts can help to define many higher-level concepts.” (Oct. 2013.)

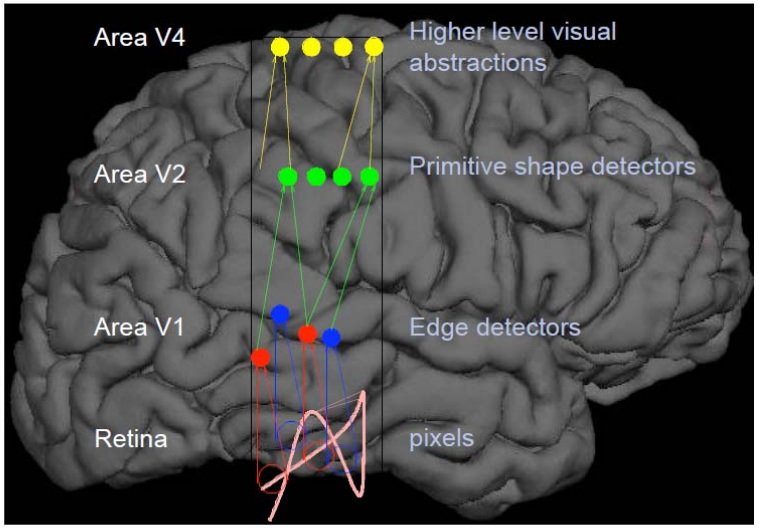
“Deep learning is a set of algorithms in machine learning that attempt to model high-level abstractions in data by using model architectures composed of multiple non-linear transformations.” (Aug. 2014)

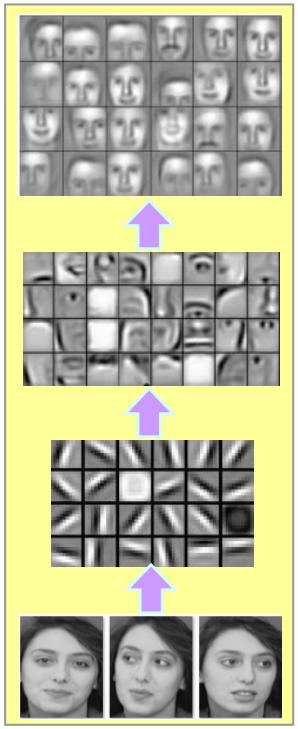
深度学习浪潮！

- Why

Hint from human learning

Deep Architecture in the Brain





Low-level sensing → Preprocessing → Feature extraction → Feature selection → Inference: prediction, recognition

Yoshua Bengio:
Learning Deep architectures for AI, Foundations and Trends in Machine Learning, 2009

深度学习浪潮！

- Why
  - 浅层神经网络可以近似任意函数，为何多层？
    - 深层网络结构中，高层可以综合应用低层信息
    - 低层关注“局部”，高层关注“全局”、更具有语义化信息
    - 为自适应地学习非线性处理过程提供了一种可能的简洁、普适的结构模型
    - 参数学习与分类器的训练可以一起学习

深度学习

—从学术研究的角度开始

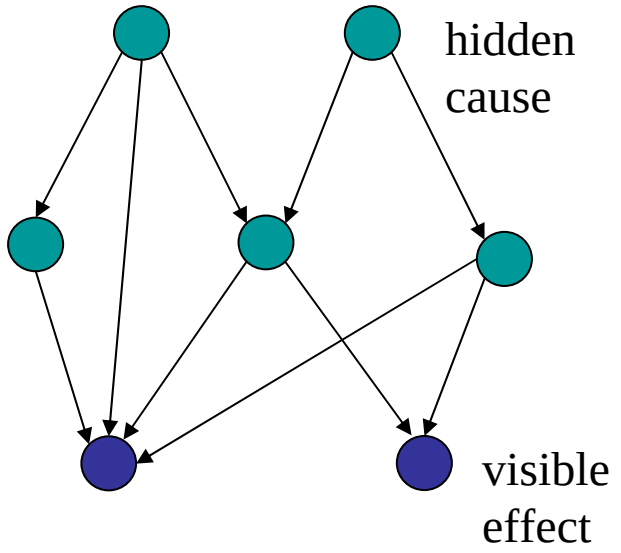
- 前向神经网络
  - 从学习的角度，是一个强有力的学习系统。系统结构简单、易于编程
  - 从计算的角度，是一个静态非线性映射。由简单的非线性处理单元的复合映射来获得复杂系统的非线性处理能力
  - 从系统的角度：它并不是一强有力系统，缺乏丰富的动力学行为

深度学习

- 发展历程
  - Hopfield network
  - Boltzman machine
  - Restricted Boltzman machine
  - CNN
  - DBN
  - DBM

Belief Network

- A belief net is a directed acyclic graph composed of stochastic variables.
- We get to observe some of the variables and we would like to solve two problems:
- The inference problem: Infer the states of the unobserved variables.
- The learning problem: Adjust the interactions between variables to make the network more likely to generate the observed data.

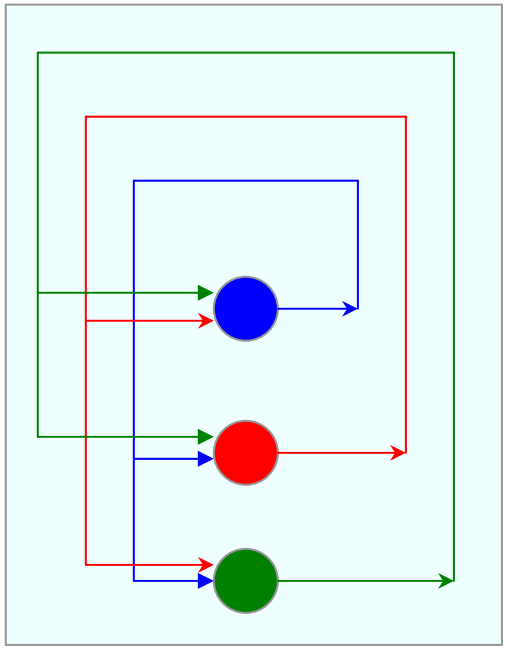


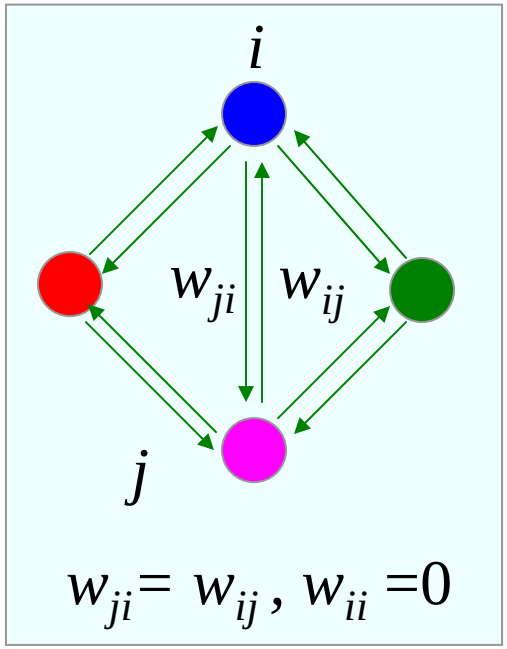
推理网络：层次结构不明显

Hard to training:
--需要估计后验分布 (依赖先验和似然)
--存在需要处理“得释”问题

Hopfield network

- 结构
  - 单层全互连、对称权值的反馈网络
  - 状态：-1(0)，+1





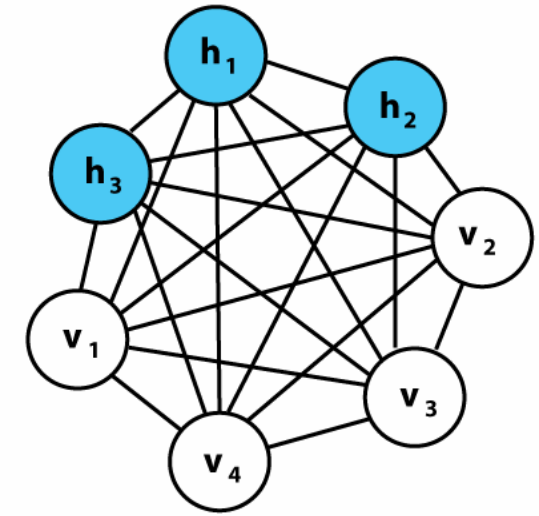
Hopfield网络按动力学方式运行，其工作过程为状态的演化过程，即从初始状态按能量减小的方向进行演化，直到达到稳定状态。稳定状态即为网络的输出

常见的两种形式

网络演化特点

Boltzman 机

- 结构
  - 结构类似于Hopfield 网络，但它是具有隐单元的反馈互联网络



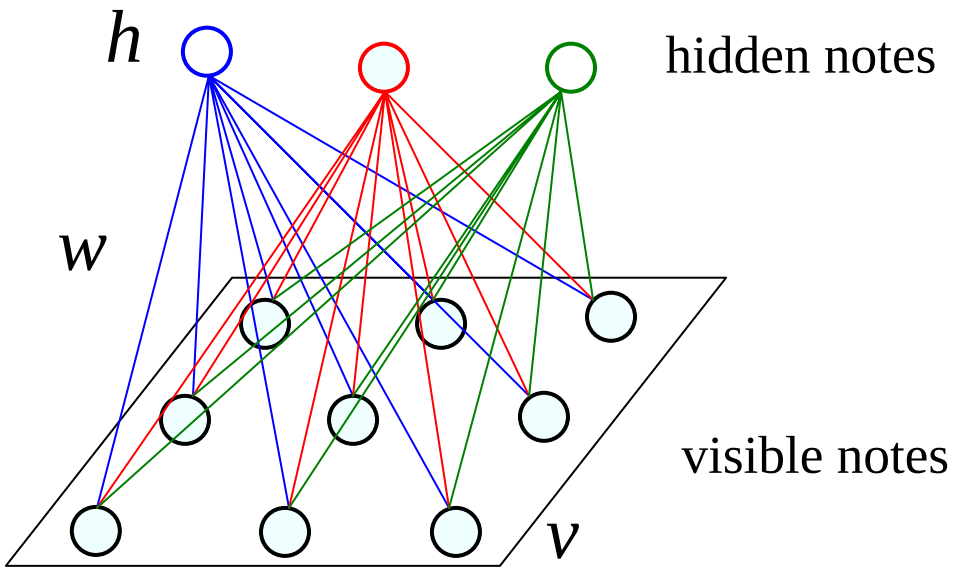
$w_{ji} = w_{ij}$, $w_{ii} = 0$

1. Hopfield网络的神经元的结构功能及其在网络中的地位是一样的。但BM中一部分神经元与外部相连,可以起到网络的输入、输出功能,或者严格地说可以受到外部条件的约束。另一部分神经元则不与外部相连，因而属于隐单元
2. 神经元的状态为0或1的概率取决于相应的输入。

网络结构复杂、训练代价大、局部极小

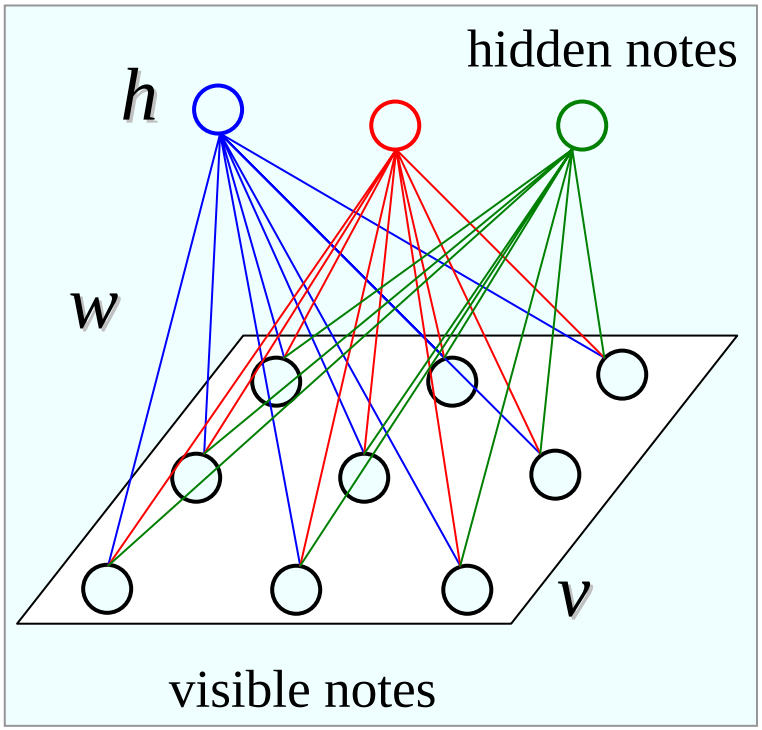
RBM

- 结构
  - In an RBM, the hidden units are conditionally independent given the visible states



RBM

- 结构 (Bipartite Structure)



Stochastic binary visible variables $\mathbf{v} \in \{0,1\}^d$ are connected to stochastic binary hidden variables $\mathbf{h} \in \{0,1\}^m$.

The energy of the joint configuration:

$$E(\mathbf{v},\mathbf{h};\theta) = -\sum_{ij} w_{ij} v_i h_j - \sum_i b_i v_i - \sum_j a_j h_j$$

θ={w,a,b}—model parameters

Probability of the joint configuration is given by the Boltzman distribution:

$$p_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{z(\theta)} \exp(-E(\mathbf{v},\mathbf{h};\theta)) = \frac{1}{z(\theta)} \prod_{ij} e^{w_{ij} v_i h_j} \prod_i e^{b_i v_i} \prod_j e^{a_j h_j}$$

$$z(\theta) = \sum_{\mathbf{v},\mathbf{h}} \exp(-E(\mathbf{v},\mathbf{h};\theta))$$

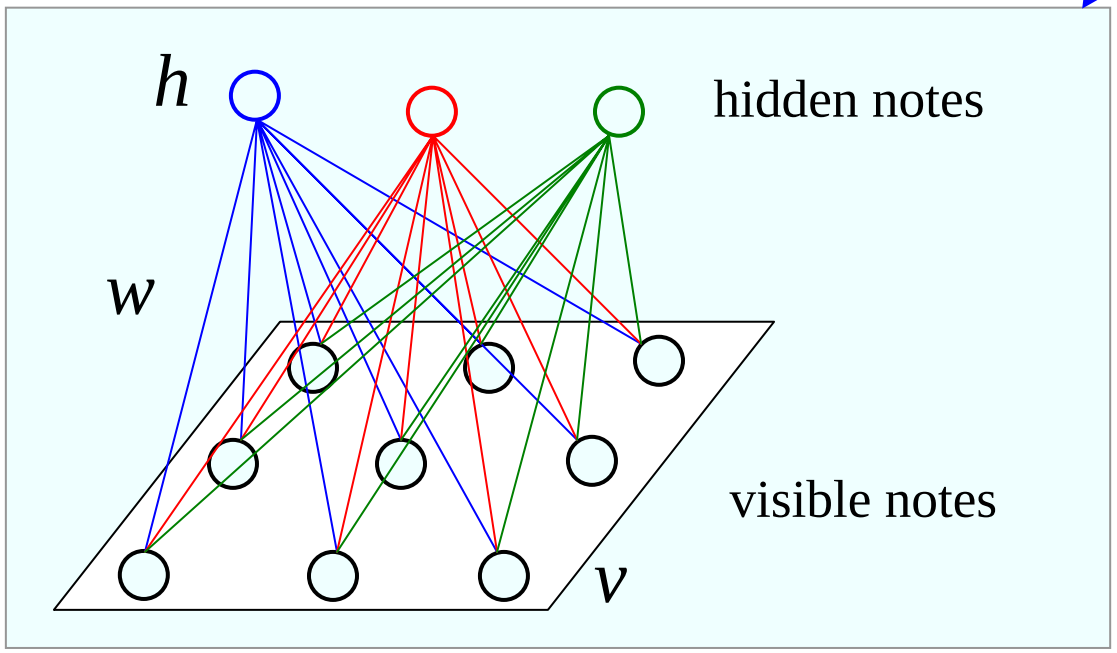
RBM

- The goal

$$\log(p_\theta(\mathbf{v})) = \log\left(\prod_i \exp(b_i v_i) \prod_j \left(1 + \exp(a_j + \sum_i w_{ij} v_i)\right)\right) - \log z(\theta)$$

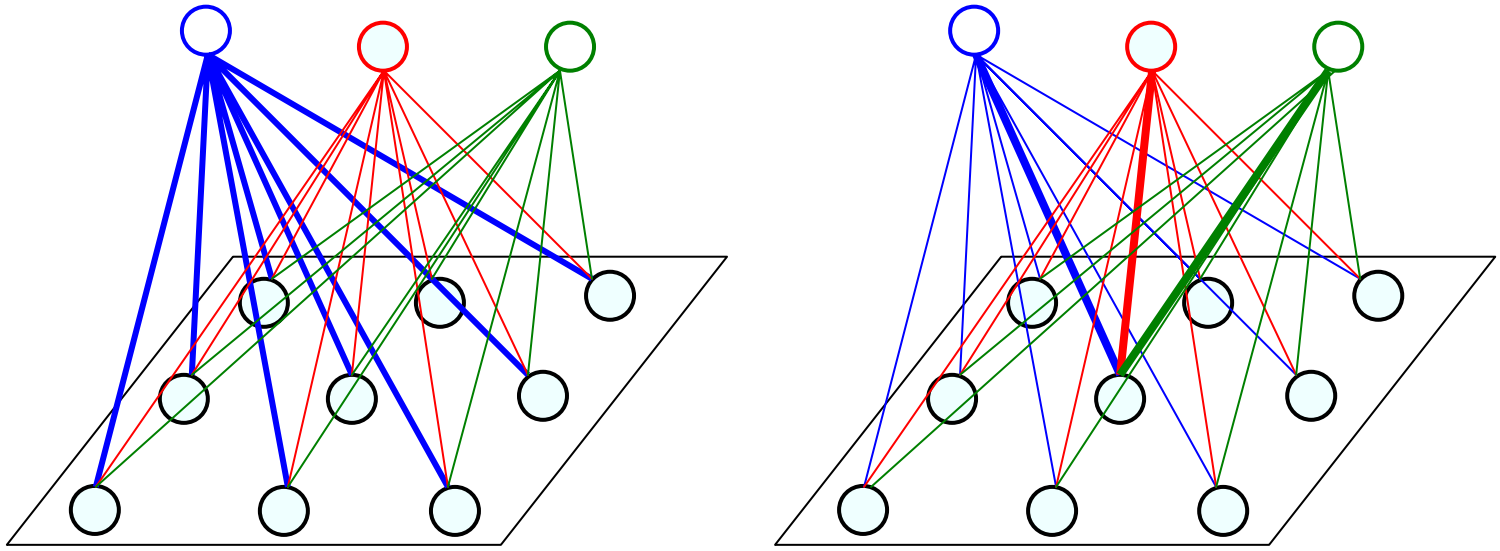
给定N个样本： $\max \sum_{i=1}^{N} \log p(\mathbf{v}_i)$

借助该模型



RBM

- 从马尔可夫随机场(MRF)的角度
  - 无向（双向）概率图模型
  - 所有可能的极大clique(团)的条件分布（或能量）已知时，可描述该MRF



$p(h_j \mid \mathbf{v})$ $p(v_i \mid \mathbf{h})$

RBM

- 最大似然(ML)
  - **v** 为观测变量，**h** 为隐变量，其能量函数为：$E(\mathbf{v}, \mathbf{h}; \theta)$
  - 概率形式：$p(\mathbf{v},\mathbf{h})$, $p(\mathbf{v})$, $p(\mathbf{h})$, $p(\mathbf{v}|\mathbf{h})$, $p(\mathbf{h}|\mathbf{v})$:

$$p(\mathbf{v},\mathbf{h}) = \frac{\exp^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}}$$

$$p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}} \qquad p(\mathbf{h}) = \frac{\sum_{\mathbf{v}} \exp^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}}$$

$$p(\mathbf{v}\,|\,\mathbf{h}) = \frac{\exp^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v}} \exp^{-E(\mathbf{v},\mathbf{h})}} \qquad p(\mathbf{h}\,|\,\mathbf{v}) = \frac{\exp^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}}$$

RBM

采用别名更清楚！

$z(\theta)$

- ML

$$\log p(\mathbf{v}) = \log \sum_{\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})} - \log \sum_{\mathbf{v}',\mathbf{h}'} \exp^{-E(\mathbf{v}',\mathbf{h}')}$$

$$\frac{\partial \log p(\mathbf{v})}{\partial \theta} = \frac{\sum_{\mathbf{h}} \left(\exp^{-E(\mathbf{v},\mathbf{h})} \frac{-\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} \right)}{\sum_{\mathbf{h}} \exp^{-E(\mathbf{v},\mathbf{h})}} - \frac{\sum_{\mathbf{v}',\mathbf{h}'} \left(\exp^{-E(\mathbf{v}',\mathbf{h}')} \frac{-\partial E(\mathbf{v}',\mathbf{h}')}{\partial \theta} \right)}{\sum_{\mathbf{v}',\mathbf{h}'} \exp^{-E(\mathbf{v}',\mathbf{h}')}}$$

$$= \sum_{\mathbf{h}} \left(p(\mathbf{h} \mid \mathbf{v}) \frac{-\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} \right) - \sum_{\mathbf{v}',\mathbf{h}'} \left(p(\mathbf{v}',\mathbf{h}') \frac{-\partial E(\mathbf{v}',\mathbf{h}')}{\partial \theta} \right)$$

$$= \mathrm{E}_{p(\mathbf{h}|\mathbf{v})} \left[\frac{-\partial E(\mathbf{v},\mathbf{h})}{\partial \theta} \right] - \mathrm{E}_{p(\mathbf{v}',\mathbf{h}')} \left[\frac{-\partial E(\mathbf{v}',\mathbf{h}')}{\partial \theta} \right]$$

positive　　negative

RBM

- CD 算法思想

$$\frac{\partial \log p(\mathbf{v})}{\partial \theta} = \mathrm{E}_{p(\mathbf{h}|\mathbf{v})}\left[\frac{-\partial E(\mathbf{v},\mathbf{h})}{\partial \theta}\right] - \mathrm{E}_{p(\mathbf{v}',\mathbf{h}')}\left[\frac{-\partial E(\mathbf{v}',\mathbf{h}')}{\partial \theta}\right]$$

观测分布

真实分布

Contrastive Divergence

RBM

- 具体参数**W, a, b**
  - RBM 的能量模型为：
  - 概率形式：

$$E(\mathbf{v},\mathbf{h}) \triangleq -\mathbf{v}^T\mathbf{W}\mathbf{h} - \mathbf{b}^T\mathbf{v} - \mathbf{a}^T\mathbf{h}$$

$$p(\mathbf{v},\mathbf{h}) = \frac{1}{Z} e^{\mathbf{v}^T\mathbf{W}\mathbf{h}} e^{\mathbf{b}^T\mathbf{v}} e^{\mathbf{a}^T\mathbf{h}}$$

$$p(\mathbf{h}\mid\mathbf{v}) = \prod_j p(h_j\mid\mathbf{v}) = \prod_j \frac{e^{\left(a_j + \sum_i w_{ij} v_i\right) h_j}}{1 + e^{a_j + \sum_i w_{ij} v_i}}$$

$$p(\mathbf{v}\mid\mathbf{h}) = \prod_i p(v_i\mid\mathbf{h}) = \prod_i \frac{e^{\left(b_i + \sum_j w_{ij} h_j\right) v_i}}{1 + e^{b_i + \sum_j w_{ij} h_j}}$$

$$\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial w_{ij}} = -v_i h_j, \quad \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial b_i} = -v_i, \quad \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial a_j} = -h_j$$

RBM

- 具体形式

$$\frac{\partial \log p(\mathbf{v})}{\partial \theta} = \mathrm{E}_{p(\mathbf{h}|\mathbf{v})}\left[\frac{-\partial E(\mathbf{v},\mathbf{h})}{\partial \theta}\right] - \mathrm{E}_{p(\mathbf{v}',\mathbf{h}')}\left[\frac{-\partial E(\mathbf{v}',\mathbf{h}')}{\partial \theta}\right]$$

$$\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \sum_{h_j}\left[p(h_j \mid \mathbf{v})(v_i h_j)\right] - \sum_{\mathbf{v}'}\left(p(\mathbf{v}')\sum_{h'_j}\left(p(h'_j \mid \mathbf{v}')v'_i h'_j\right)\right)$$

$$\frac{\partial \log p(\mathbf{v})}{\partial b_i} = \sum_{h_j}\left[p(h_j \mid \mathbf{v})h_j\right] - \sum_{\mathbf{v}'}\left(p(\mathbf{v}')\sum_{h'_j}\left(p(h'_j \mid \mathbf{v}')h'_j\right)\right)$$

$$\frac{\partial \log p(\mathbf{v})}{\partial a_j} = \sum_{h_j}\left[p(h_j \mid \mathbf{v})v_i\right] - \sum_{\mathbf{v}'}\left(p(\mathbf{v}')\sum_{h'_j}\left(p(h'_j \mid \mathbf{v}')v'_i\right)\right)$$

RBM

- 对于$h_j \in \{0;1\}, v_i \in \{0;1\}$, 可进一步化简为：

$$\frac{\partial \log(\mathbf{v})}{\partial w_{ij}} = p(h_j = 1 | \mathbf{v}) v_i - \sum_{\mathbf{v}'} [p(\mathbf{v}') p(h'_j = 1 | \mathbf{v}') v'_i]$$

$$\frac{\partial \log p(\mathbf{v})}{\partial b_i} = v_i - \sum_{\mathbf{v}'} [p(\mathbf{v}') v'_i]$$

$$\frac{\partial \log p(\mathbf{v})}{\partial a_j} = p(h_j = 1 | \mathbf{v}') - \sum_{\mathbf{v}'} [p(\mathbf{v}') p(h'_j = 1 | \mathbf{v}')]$$

- 第一项很好计算，关键是第二项的计算，例如：

$$\frac{\partial l \log p(\mathbf{v})}{\partial w_{ij}} = p(h_j = 1 | \mathbf{v}) v_i - \sum_{\mathbf{v}'} [p(\mathbf{v}') p(h'_j = 1 | \mathbf{v}') v'_i]$$

RBM

- 通过采样来计算第二项：

$$E[f(x)] = \sum_x p(x) f(x) \approx \frac{1}{L} \sum_{x \sim p(x)} f(x)$$

- CD－K算法

$$\mathbf{v}^{(0)} \xrightarrow{\mathbf{h} \sim p(\mathbf{h}|\mathbf{v})} \mathbf{h}^{(0)} \xrightarrow{\mathbf{v} \sim p(\mathbf{v}|\mathbf{h})} \mathbf{v}^{(1)} \rightarrow \mathbf{h}^{(1)} \rightarrow \cdots \rightarrow \mathbf{v}^{(k)}$$

- 再次回顾ML算法－给定N个样本，任务：

$$\max \sum_{i=1}^{N} \log p(\mathbf{v}_i)$$

RBM

- 算法流程(CD-1)：
  - 输入样本为{$\mathbf{v}_1,\mathbf{v}_2,\ldots,\mathbf{v}_n$}，设观测变量**v,** 隐变量**h**
  - 将对各参数的偏导数初始化为 $\Delta w_{ij}=0,\ \Delta a_j=0,\ \Delta b_i=0$;
  - For k =1,…, N: （样本数）
    - $\mathbf{v}^{(0)} \leftarrow \{\mathbf{v}_n\}$
  - For j = 1,...,m, do sample: $h_j^{(0)} \sim p(h_j \mid \mathbf{v}^{(0)})$ （隐结点数）
  - For i=1,...,n, do sample: $v_j^{(1)} \sim p(v_i \mid \mathbf{h}^{(0)})$ （维数）
  - 计算梯度，最后平均（在选样的过程中将别名统一回来）

$$\Delta w_{ij} \leftarrow \Delta w_{ij} + p(h_j=1\mid \mathbf{v}^{(0)})v_i^{(0)} - p(h_j=1\mid \mathbf{v}^{(1)})v_i^{(1)}$$

$$\Delta a_j \leftarrow \Delta a_j + p(h_j=1\mid \mathbf{v}^{(0)}) - p(h_j=1\mid \mathbf{v}^{(1)})$$

$$\Delta b_i \leftarrow \Delta b_i + v_i^{(0)} - v_i^{(1)}$$

学习率：更新多少？

RBM

- 解释

$$\frac{\partial \sum_{n=1}^{N} \log p(\mathbf{v})}{\partial w_{ij}}$$

$$= \sum_{n=1}^{N} p(h_j = 1 \mid \mathbf{v}_n) v_i^{(n)} - \sum_{n=1}^{N} \mathrm{E}_{\mathbf{v}'} \left(p(h'_j = 1 \mid \mathbf{v}') v'_i \right)$$

$$= \sum_{n=1}^{N} p(h_j = 1 \mid \mathbf{v}_n) v_i^{(n)} - \sum_{n=1}^{N} \frac{1}{N} \sum_{\mathbf{v}' \sim p(\mathbf{v}')} \left(p(h'_j = 1 \mid \mathbf{v}') v'_i \right)$$

$$= \sum_{n=1}^{N} p(h_j = 1 \mid \mathbf{v}_n) v_i^{(n)} - \sum_{\mathbf{v}' \sim p(\mathbf{v}')} \left(p(h'_j = 1 \mid \mathbf{v}') v'_i \right)$$

For k, i, j

$$\Delta w_{ij} \leftarrow \Delta w_{ij} + p(h_j = 1 \mid \mathbf{v}^{(0)}) v_i^{(0)} - p(h_j = 1 \mid \mathbf{v}^{(1)}) v_i^{(1)}$$

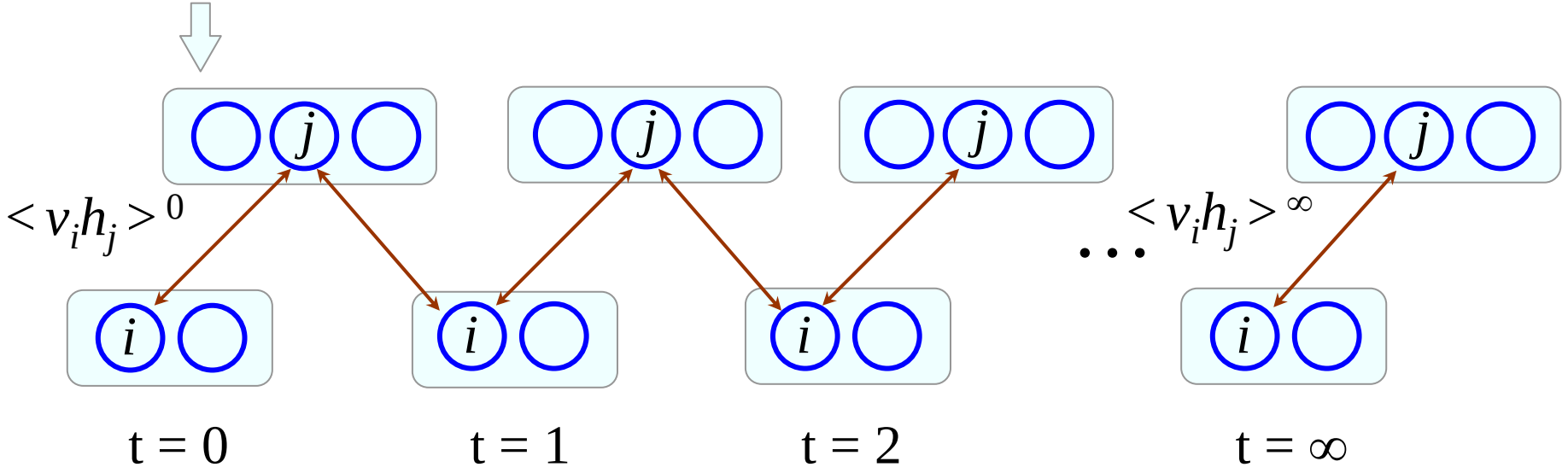
end

- 图解

$$\frac{\partial \log p(\mathbf{v})}{\partial \theta} = \mathrm{E}_{p(\mathbf{h}|\mathbf{v})}\left[\frac{-\partial E(\mathbf{v},\mathbf{h})}{\partial \theta}\right] - \mathrm{E}_{p(\mathbf{v}',\mathbf{h}')}\left[\frac{-\partial E(\mathbf{v}',\mathbf{h}')}{\partial \theta}\right]$$

$$\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \sum_{h_j}\left[p(h_j \mid \mathbf{v})(v_i h_j)\right] - \sum_{\mathbf{v}'}\left[p(\mathbf{v}')\sum_{h_j'}\left[p(h_j' \mid \mathbf{v}')v_i' h_j'\right]\right]$$

$$\hat{=} < v_i h_j >^0 - < v_i h_j >^\infty$$



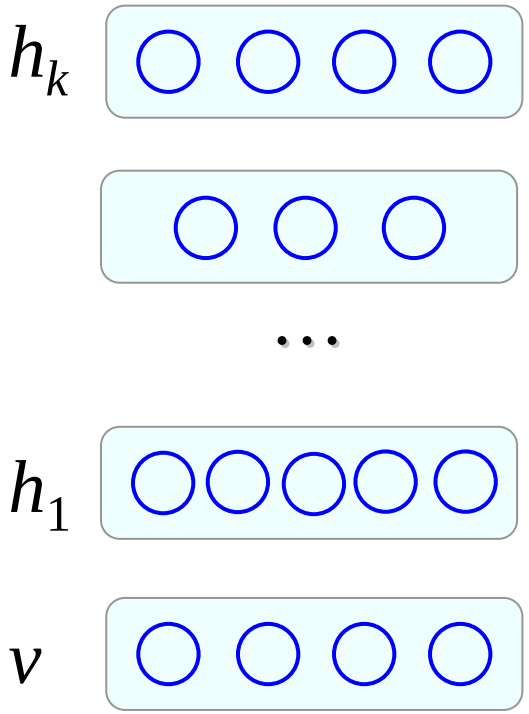
RBM

- 训练技巧（结构已定）
  - 将数据分成Batch, 在每个batch 内并行计算
  - 将CD-∞算法折衷成CD-1算法
  - 监控学习过程
  - 防止overfitting
  - 监控学习率
  - 增加动力机制（选样）
  - 增加稀疏机制（联接）

G. Hinton. A Practical Guide to Training Restricted Boltzmann Machines, Tech Report, No. UTML TR 2010-003, Department of Computer Science, University of Toronto, Canada

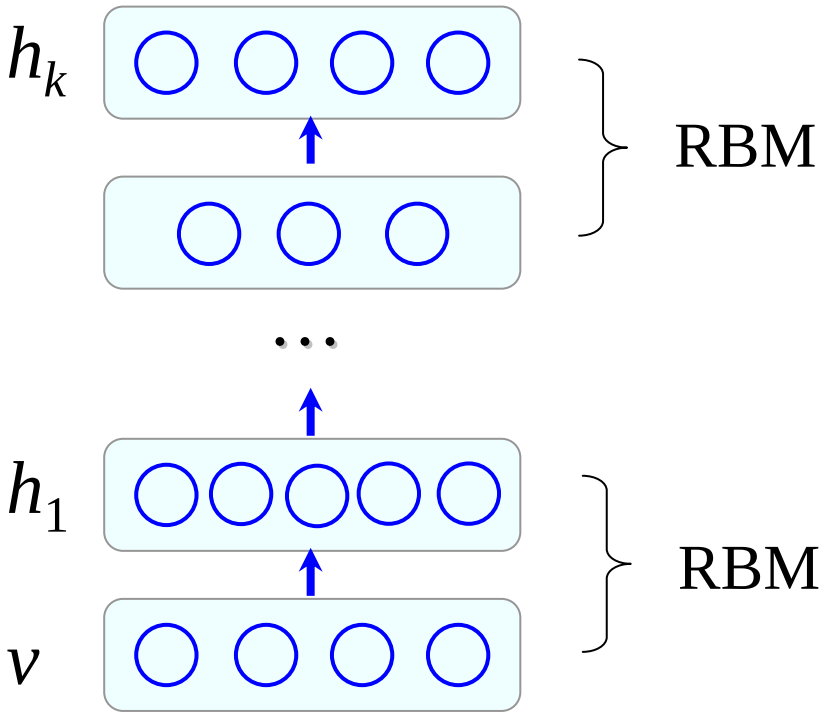
DBN

- 结构
  - 含有多个隐含层

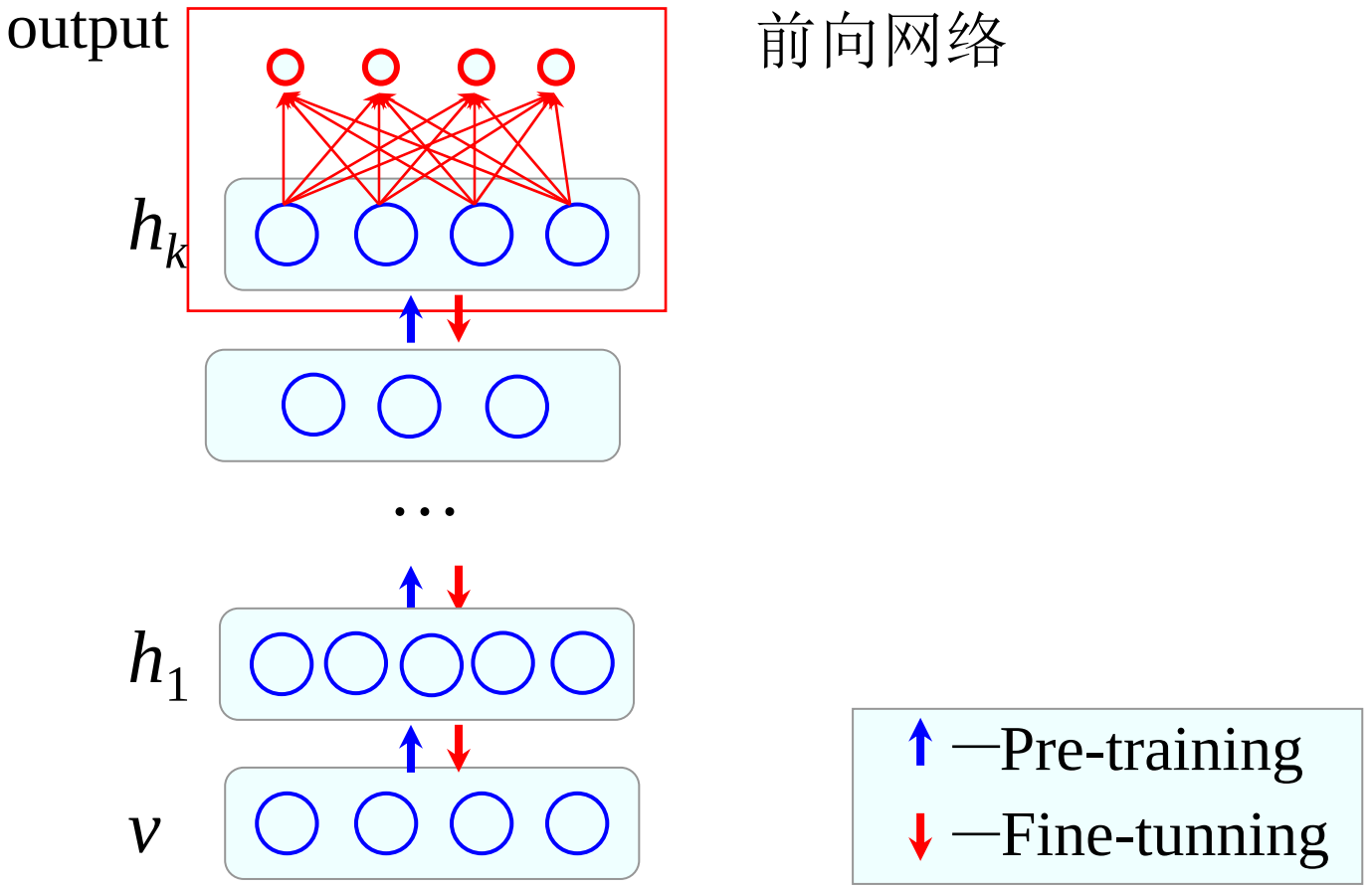


DBN

- Training for feature learning

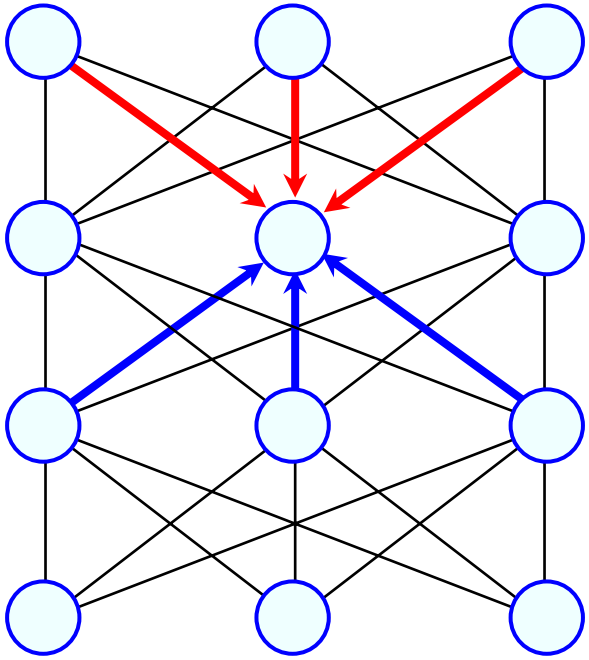


- Training for classification



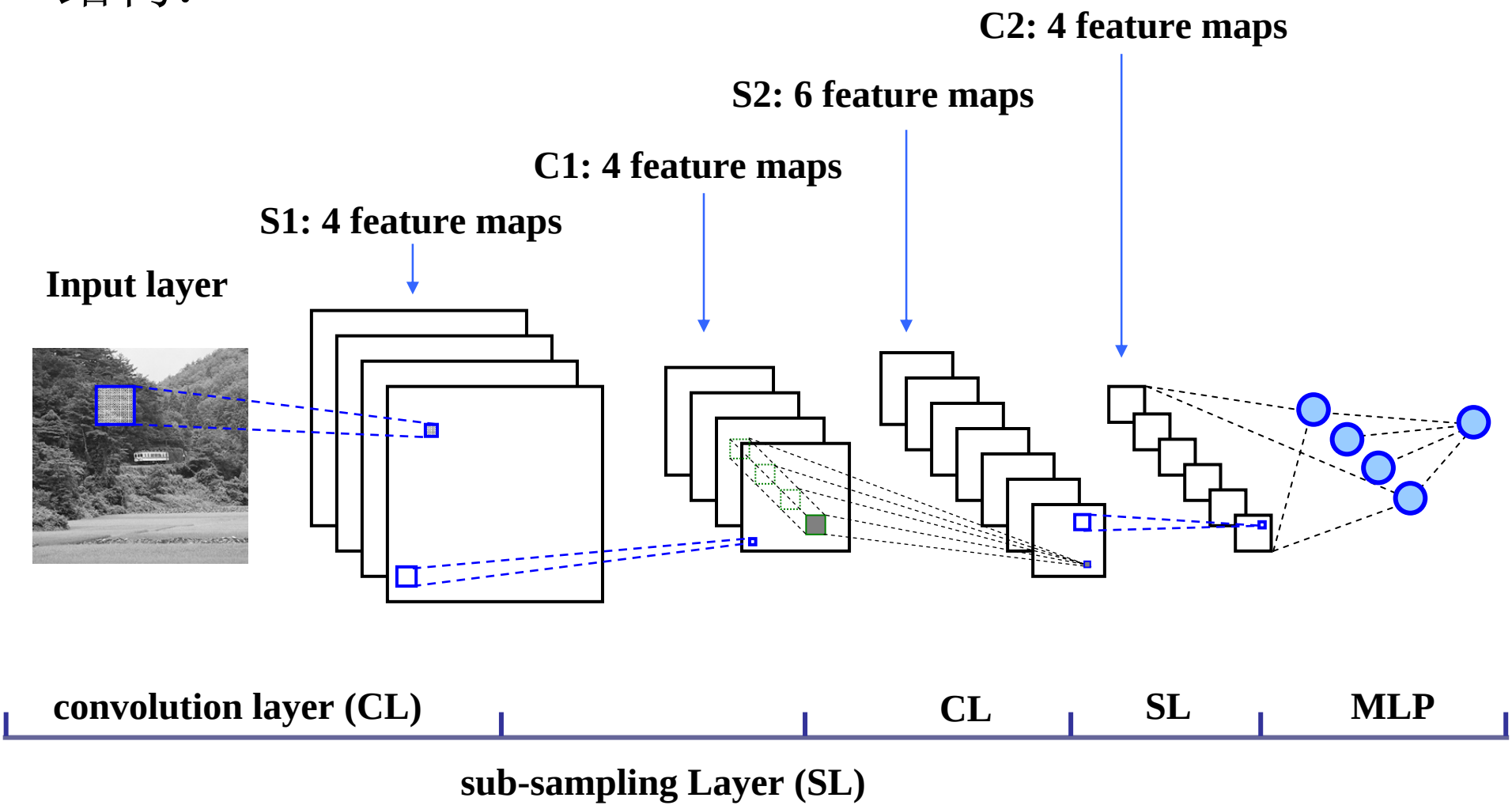
DBM

- 网络结构
  - 与DBN一样，但在训练时采用双方向（上下两层）
  - 在训练单层时需同时考虑两个或者多个隐含层（整个网络相当于一个MRF）
  - 能量模型与RBM不一样



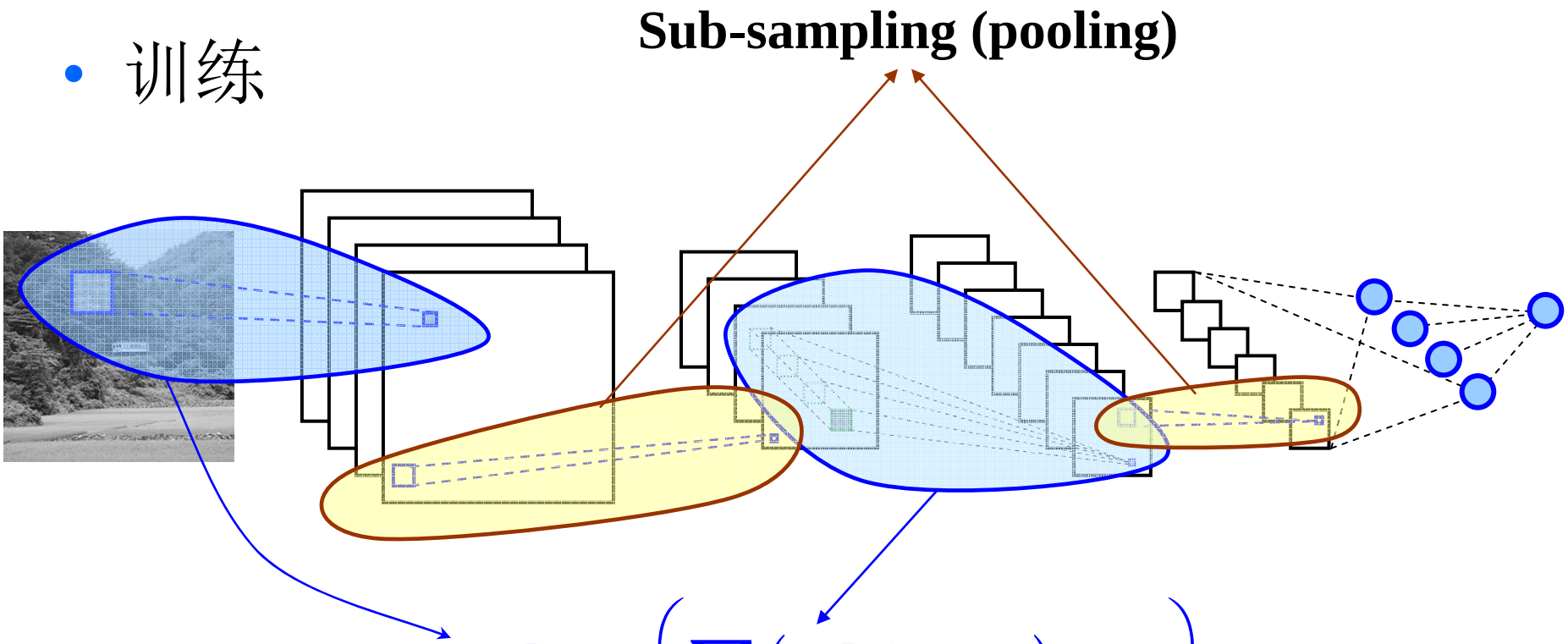
CNN (Convolutional Neural Networks)

结构：



CNN

- 训练



$$f_j^L = h\left(\sum_i\left(f_i^{L-1} * K_{ij}\right) + b_j\right)$$

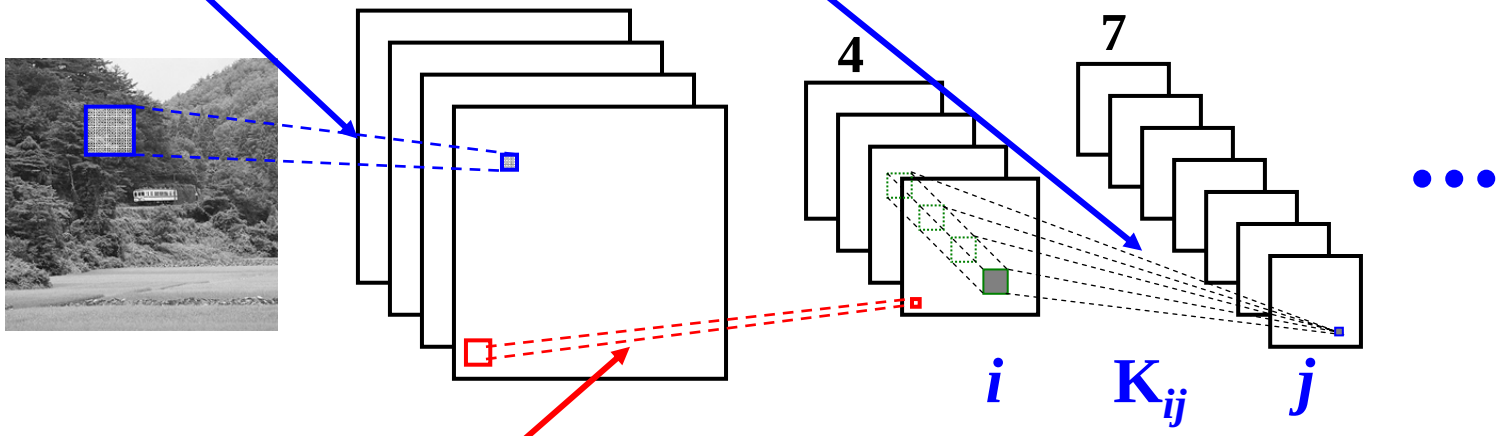
***L*: 当前层，K_{ij}: 待学习第 *j* 层filter，b_j: 待学习偏移量, *h*: 激励函数**

(*L*=0 为原始输入图像)

两个操作： Convolution and Pooling

4个卷积核，均作用于输入图像

28个卷积核，K_{ij} 作用于上一层第 *i* 个图像，结果累加至下一层第 *j* 个图像

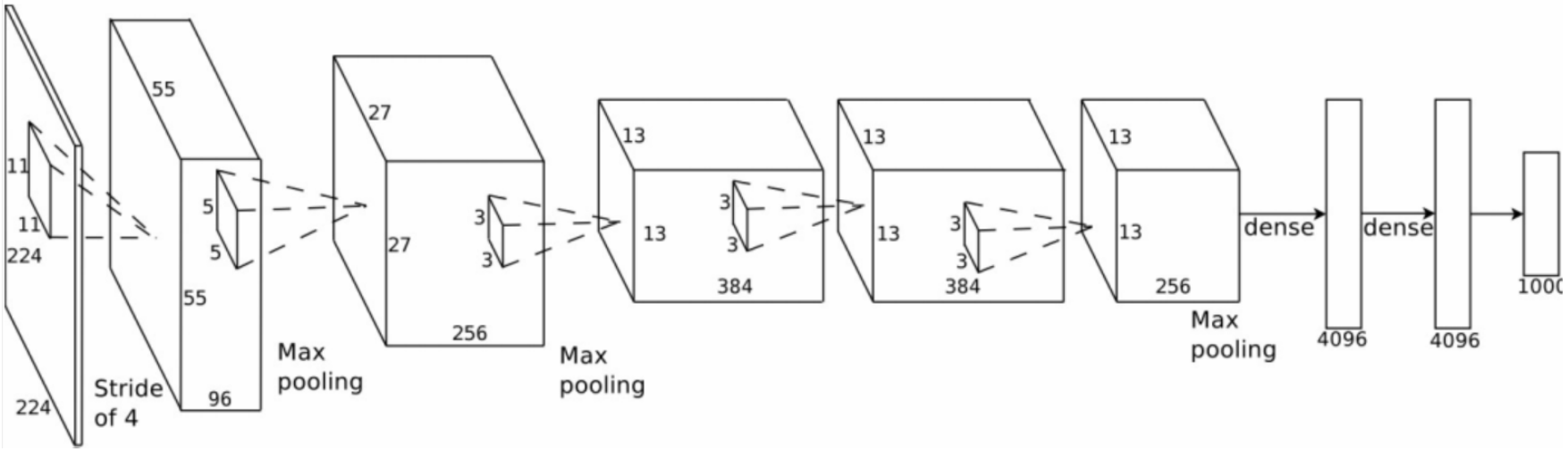


Max pooling: 用于同一卷积结果的(2x2)局部

$$f_j^L = h\left(\sum_i \left(f_i^{L-1} * K_{ij}\right) + b_j\right)$$

激励！ 收集！

CNN



- Two operations： Convolution, Pooling
- Application: Representation for almost all tasks (classification, segmentation, even real-time recognition or tracking on cell-phone)

CNN

- 课堂问题
  - 请从普通前向神经网络的角度来解释CNN
  - 请从RBM的角度对CNN进行扩展，并描述相应的训练方法

其他类型

- Autoencoder
  - G. Hinton and R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. Science, 2006

深度学习

- 课堂讨论
  - 深度学习
    - 关于训练...
    - 关于网络结构...
    - 关于算法理论...（网络应该达到什么样的性能）
    - 关于参数学习的要求...
  - 应用
    - 就特征学习和分类问题：课堂讨论如何应用和扩展

References

- C.M. Bishop, Chapter 6, Pattern Recognition and Machine Learning, Springer, 2006
- C. Bishop. Neural Networks for Pattern Recognition. Oxford University Press, 1995.

- Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. Gradient-Based Learning Applied to Document Recognition, Proceedings of IEEE, Vol. 86, No. 11, pp. 2278-2324, 1998.
- Y. LeCun, Y. Bengio: Convolutional Networks for Images, Speech, and Time-Series, In Arbib, M.A. The hand book of Brain Theory and Neural Networks, MIT Press, 1995, 255-258
- G. Hinton. A Practical Guide to Training Restricted Boltzmann Machines, Tech Report, No. UTML TR 2010-003, Department of Computer Science, University of Toronto, Canada
- G. Hinton and R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. Science, 2006

References

- R. Salakhutdinov and G. Hinton. Deep Boltzmann Machines. ICAIS, 2009
- Yoshua Bengio: Learning Deep architectures for AI, Foundations and Trends in Machine Learning, 2(1), 2009
- http://deeplearning.net/tutorial/
- G. Hinton, S. Osindero and Y. W. Teh. A fast learning algorithm for deep belief nets. Neural Computation, 2006
- R. Salakhutdinov and G. Hinton. A Better Way to Pretrain Deep Boltzmann Machines. NIPS, 2013
- Kevin P. Murphy. Machine Learning: A Probabilistic Perspective. Chapter 28 : Deep Learning. The MIT Press, 2012
- Bengioy 的书：
  http://www.iro.umontreal.ca/~bengioy/DLbook/

致 谢

- Courtesy for some slides modified from
  - Kaizhu Huang
  - Ying Wang
  - Kai Yu
  - Geoffrey Hinton
  - Zhongzhi Shi
  - Junliang Xing

Thank All of You!