支持向量机与深度学习

Support Vector Machine and Deep Learning

张三国(中国科学院大学)
sgzhang@ucas.ac.cn

January 9, 2015

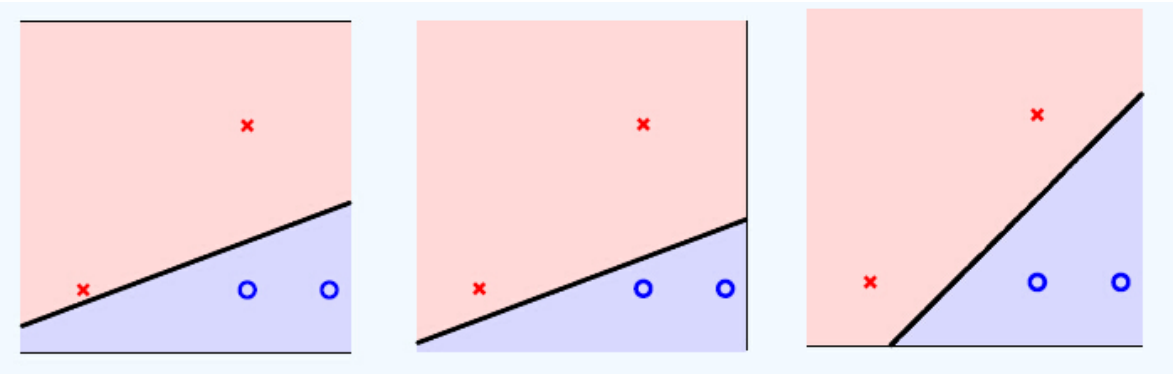
目录

源起

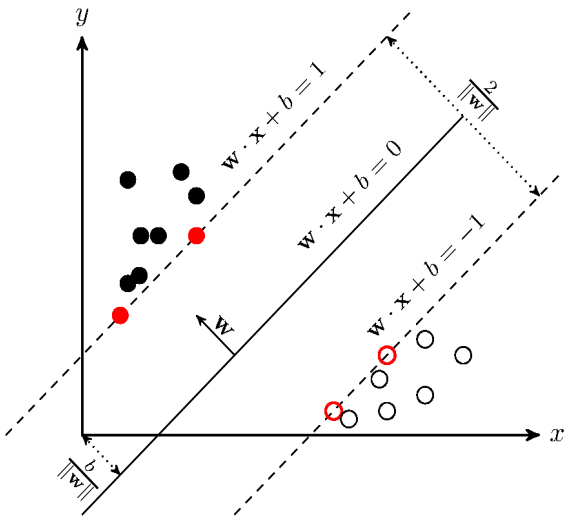
- 1963 年, *Vapnik* 在解决模式识别问题时提出了支持向量方法, 这种方法从训练集中选择一组特征子集, 使得对特征子集的划分等价于对整个数据集的划分, 这组特征子集就被称为支持向量(*Support Vector*, *SV*)。
- 1971 年, *Kimeldorf* 提出使用线性不等约束重新构造*SV*的核空间, 解决了一部分线性不可分问题。
- 1990 年, *Grace*, *Boser* 和 *Vapnik* 等人开始对支持向量机进行研究。
- 1995 年, *Cortes* 和 *Vapnik* 发在在《*Machine Leranming*》期刊上的 “*Support – vector networks*” 一文标志着 *SVM* 模型诞生。
- *SVM* 引发研究热潮, 截至目前, 在*Google Scholar*上有约2,210,000条关于*Support Vector Machine*结果。

线性分类器

- 在机器学习和统计领域，分类是指基于带属性及已知类别标签的样本，训练得到分类器，实现对新样本类别的预测。
- 考虑两分类问题，设 $\mathbf{x} \in \mathbb{R}^d$ 表示数据点，$\mathbf{w} \in \mathbb{R}^d$ 表示系数向量，$b \in \mathbb{R}$表示截距项，$y \in \{\pm 1\}$ 表示两个不同的类别。
- 线性分类器是指在 d 维空间中的超平面，其方程表示为$\mathbf{w}^T\mathbf{x} + b = 0$，恰能分隔开两类数据。
- 如下图所示，哪个超平面是最优越的？



最大间隔



设$\{(\mathbf{x}_i, y_i),\ i = 1, \cdots, n,\ \mathbf{x}_i \in \mathbb{R}^d,\ y_i \in \{\pm 1\}\}$，寻找最优的线性分类函数$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$使得两类样本点分居此超平面两侧。设$f(\mathbf{x}) < 0$的点对应的$y = -1$，而$f(\mathbf{x}) > 0$的点对应$y = 1$。训练数据满足$y_i \cdot f(\mathbf{x_i}) \geqslant 1,\ i = 1, \cdots, n$。新样本的类别预测为$sgn(f(\mathbf{x}))$。

- 泛函间隔：$y \cdot f(\mathbf{x})$　样本被正确分类当且仅当$y \cdot f(\mathbf{x}) > 0$
- 几何间隔：$\frac{2}{\|\mathbf{w}\|}$　超平面$f(\mathbf{x}) = \pm 1$之间的间隔

$$\begin{aligned} \max_{\mathbf{w},\ b} \quad & \frac{2}{\|\ \mathbf{w}\ \|} \\ s.t. \quad & y_i \cdot (\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1, \quad i = 1, \cdots, n. \end{aligned} \tag{1}$$

对偶问题

$$\begin{aligned} \min_{\mathbf{w},\ b} \quad & \frac{1}{2} \parallel \mathbf{w} \parallel^2 \\ s.t. \quad & y_i \cdot (\mathbf{w}^T \mathbf{x}_i + b) \geqslant 1, \quad i = 1, \cdots, n. \end{aligned} \tag{1*}$$

这是二次规划问题, 属凸优化问题。其*Lagrange*对偶函数为

$$\mathbb{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 - \sum_{i=1}^{n} \alpha_i (y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) - 1)$$

由对偶理论及 *KKT* 条件，得对偶问题(*Dual Problem*)为

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ s.t. \quad & \sum_{i=1}^{n} \alpha_i y_i = 0; \quad \alpha_i \geqslant 0, \quad i = 1, \cdots, n. \end{aligned} \tag{2}$$

KKT 条件及解的性质

对Lagrange函数$\mathbb{L}(\mathbf{w}, b, \boldsymbol{\alpha})$的处理过程中, 可得KKT条件:

$$\frac{\partial \mathbb{L}(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \Longrightarrow \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \quad \text{(表示定理)}$$

$$\frac{\partial \mathbb{L}(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \Longrightarrow \sum_{i=1}^{n} \alpha_i y_i = 0$$

互补松弛条件: $\alpha_i (y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) - 1) = 0, \quad i = 1, \cdots, n.$

- 支持向量: $\{i : \alpha_i > 0\}$ 的样本点,位于间隔(Margin)上
- 稀疏性: 仅依赖少部分支持向量的样本点
- 鲁棒性: 对非支持向量的样本点不敏感
- 泛化能力强: 极大化分类边界的间隔

数据中存在噪声

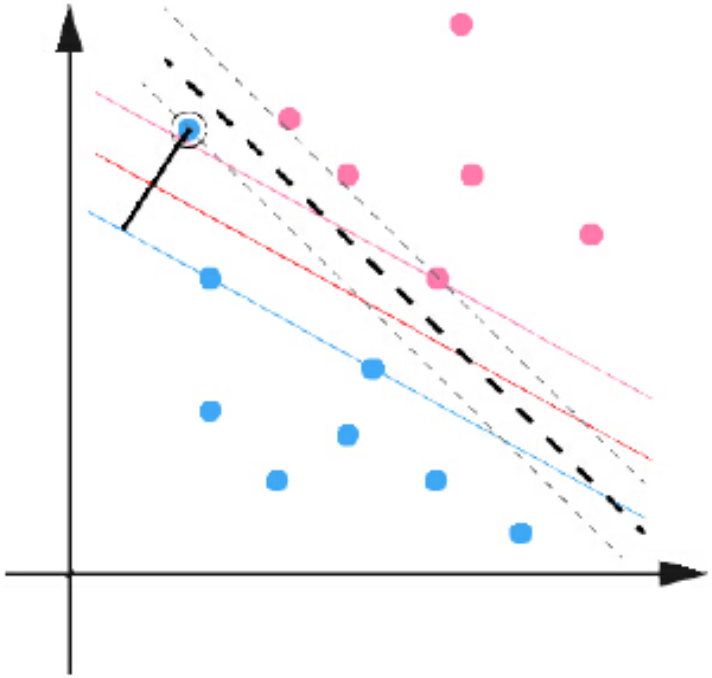


Figure: 数据中存在离群点的情况

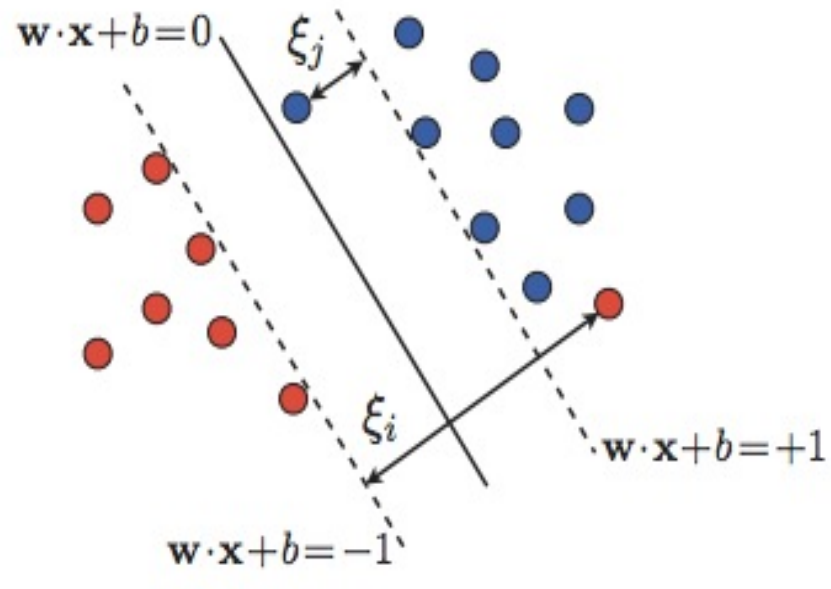


Figure: 引入松弛因子

松弛技巧

当数据中裹挟着噪音时，允许适度的松弛 $\xi_i \geqslant 0$，问题变为

$$\begin{aligned} \min_{\mathbf{w},\ b,\ \boldsymbol{\xi}} \quad & \frac{1}{2} \parallel \mathbf{w} \parallel_2^2 \ + \ C \sum_{i=1}^{n} \xi_i \\ s.t. \quad & y_i \cdot (\mathbf{w}^T \mathbf{x}_i + b) \geqslant 1 - \xi_i; \quad \xi_i \geqslant 0, \quad i = 1, \cdots, n. \end{aligned} \tag{3}$$

C 是平衡间隔最大和松弛因子的参数。其*Lagrange*对偶函数为:

$$\mathbb{L}(\mathbf{w}, b, \alpha, \beta) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 - \sum_{i=1}^{n} \alpha_i (y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) - 1 + \xi_i) - \sum_{i=1}^{n} \beta_i \xi_i$$

其对偶问题为:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ s.t. \quad & \sum_{i=1}^{n} \alpha_i y_i = 0; \quad 0 \leqslant \alpha_i \leqslant C, \quad i = 1, \cdots, n. \end{aligned} \tag{4}$$

KKT 条件及解的性质

对Lagrange函数$\mathbb{L}(\mathbf{w}, b, \boldsymbol{\alpha})$的处理过程同样可得KKT条件:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \ , \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \ , \ \alpha_i + \beta_i = C$$

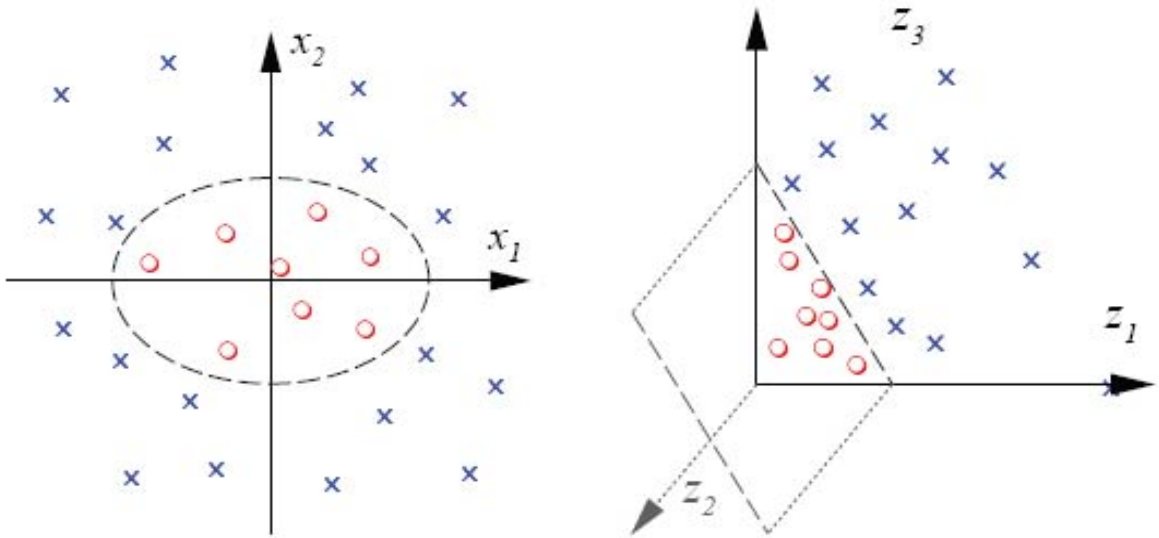
互补松弛条件:

$$\alpha_i (y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) - 1 + \xi_i) = 0 \ , \quad \beta_i \xi_i = 0, \qquad i = 1, \cdots, n.$$

- $\alpha_i = 0$ 时, $\beta_i = C$, $\xi_i = 0$, $y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) \geqslant 1$
  间隔外, 分类正确, 无松弛
- $0 < \alpha_i < C$ 时, $\beta_i = C - \alpha_i > 0$, $\xi_i = 0$, $y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) = 1$
  间隔上, 分类正确, 支持向量(SV), 无松弛
- $\alpha_i = C$ 时, $\beta_i = 0$, $\xi_i \geqslant 0$, $y_i \cdot (\mathbf{w}^T \mathbf{x_i} + b) = 1 - \xi_i \leqslant 1$
  间隔内, 分类可能会出错, 支持向量(SV), 可能有松弛

核技巧 (1)

在更为一般的场合，数据点呈现出非线性特征时，线性分类器面临更大的挑战。例如:



当线性不可分时，*SVM* 通过某预先选择的非线性映射 ϕ 将数据映射到某个高维特征空间，在这个空间中构造最优分类超平面。

核技巧 (2)

设 $\phi: \mathbb{X} \to \mathbb{F}$ 是从输入空间 $\mathbb{X}$ 到某个特征空间 $\mathbb{F}$ 的映射，建立非线性学习器分为两步：

1. 使用一个非线性映射 ϕ 将数据变换到一个特征空间 $\mathbb{F}$；
2. 在特征空间学习得到线性分类器 $f(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$ 。

其对偶问题形式为

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) \\ s.t. \quad & \sum_{i=1}^{n} \alpha_i y_i = 0; \quad 0 \leqslant \alpha_i \leqslant C, \quad i = 1, \cdots, n. \end{aligned} \tag{5}$$

根据对偶问题的表示定理可得: $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x_i})^T\phi(\mathbf{x}) + b$。

ϕ 函数的引入，使得原问题只依赖于内积 $\phi(u)^T\phi(v)$，即可解得最优解 $(\mathbf{w}, b)$。称 $\kappa(u, v) = \phi(u)^T\phi(v)$ 为核函数。

核技巧 (3)

在实际应用中，往往依赖先验领域知识才能选择有效的核函数。常用的核函数有

- 线性核函数: $\kappa(u,v) = u^T v$
- 多项式核函数: $\kappa(u,v) = (u^T v + \gamma)^d$
- 高斯核函数: $\kappa(u,v) = \exp\left\{-\frac{\|u-v\|^2}{2\sigma^2}\right\}$
- *Sigmoid*核函数: $\kappa(u,v) = \tanh(\gamma u^T v + c)$; $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

注1: $G = (\kappa(\mathbf{x}_i, \mathbf{x}_j))_{n\times n}$ 为 κ 的 *Gram* 矩阵, 则 G 为半正定矩阵。
注2: 可通过半正定矩阵B来构造核函数。

统一框架

两分类支持向量机的正则化(*Regularization*)框架

$$\min_{f\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n} L\left(y_i,\ f(\mathbf{x}_i)\right)+\lambda J(f) \qquad (*)$$

设$\mathscr{F}$为候选函数族, $L(y,\ f)$为损失函数, $J(f)$为正则项, λ为调整参数。第一部分为平均训练误差, 刻画模型精度; 第二部分为正则项, 对模型复杂度施加惩罚, 避免模型过拟合。参数λ用以平衡二者。
经典的带松弛因子的*SVM*优化问题 (3) 可表为

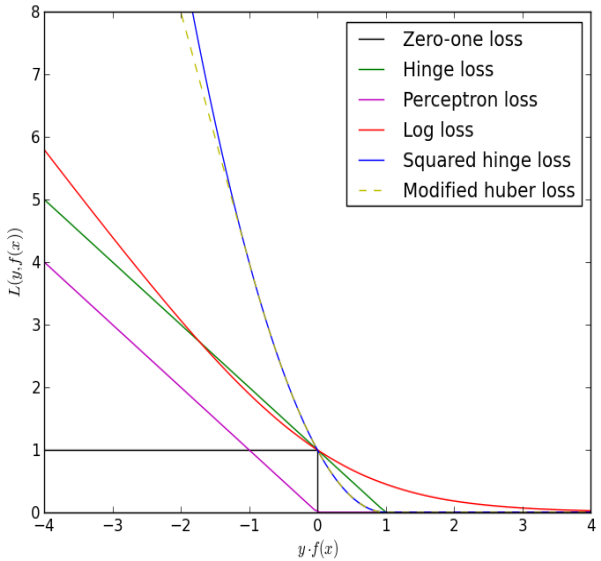
$$\min_{f\in\mathscr{F}} \frac{1}{2}\parallel w\parallel_2^2 + C\sum_{i=1}^{n}[1-y_i\cdot f(\mathbf{x}_i)]_+$$

$L(y,\ f)=[1-y\cdot f]_+ \triangleq \max(0,1-y\cdot f),\ J(f)=\parallel \mathbf{w}\parallel_2^2,\ C=\frac{1}{2n\lambda}$。

损失函数

常见的损失函数有

- $0-1$ *loss* : $L(y,\ f) = I\{y \neq f\}$
- *hinge loss* : $L(y,\ f) = [1 - y \cdot f]_+$
- *square hinge loss* : $L(y,\ f) = [1 - y \cdot f]_+^2$
- *log loss* : $L(y,\ f) = \log[1 + \exp(-y \cdot f)]$
- *exponential loss* : $L(y,\ f) = \exp(-y \cdot f)$



- *LUM loss*(*Liu et.al* 2012) : $a > 0,\ c \geqslant 0$

$$L(y,\ f) = \begin{cases} 1 - y \cdot f & y \cdot f < \frac{c}{1+c}, \\ \frac{1}{1+c}\left(\frac{a}{(1+c)y \cdot f - c + a}\right)^a & y \cdot f \geqslant \frac{c}{1+c}. \end{cases}$$

适当构造 $0-1$ 损失函数的凸替代函数, 使问题“凸化”, 更易求解。

惩罚函数

常见的惩罚函数有

- *square penlty* : $J(f) = \| \mathbf{w} \|_2^2 = \sum_{i=1}^{d} w_i^2$
- $1-$ *norm penlaty* : $J(f) = \| \mathbf{w} \|_1 = \sum_{i=1}^{d} |w_i|$
- L_q *penalty* : $J(f) = \| \mathbf{w} \|_1 = \sum_{i=1}^{d} |w_i|^q, \quad q > 0$
- *elastic net penalty* : $J(f) = \lambda_1 \| \mathbf{w} \|_1 + \lambda_2 \| \mathbf{w} \|_2^2, \quad \lambda_1, \lambda_2 \geqslant 0$
- *SCAD penalty* :

$$SCAD_\lambda(\theta) = \begin{cases} \lambda|\theta| & |\theta| < \lambda, \\ -\frac{|\theta|^2 - 2a\lambda|\theta| + \lambda^2}{2(a-1)} & \lambda < |\theta| \leqslant \lambda, \\ \frac{(a+1)\lambda^2}{2} & |\theta| \geqslant \lambda. \end{cases}$$

$$J(f) = \sum_{i=1}^{d} SCAD_{\lambda_j}(w_j)$$

L_1 SVM

- 在问题(3)中, 用 L_1 惩罚代替原来的 L_2 惩罚
- L_1 惩罚倾向于获得稀疏解, 适用于高维问题
- L_1 SVM 的具体问题形式为

$$\begin{aligned} \min_{\mathbf{w},\ b,\ \boldsymbol{\xi}} \quad & \| \mathbf{w} \|_1 \ + \ C \sum_{i=1}^{n} \xi_i \\ s.t. \quad & y_i \cdot (\mathbf{w}^T \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1, \cdots, n; \\ & \xi_i \geqslant 0, \quad i = 1, \cdots, n. \end{aligned} \tag{6}$$

- 最优解 $\mathbf{w}$ 至多有 d 个非零。

鲁棒支持向量机

Robust Support Vector Machine, *Wu and Liu*, 2007

- *hinge loss* 是无界的; 对异常值敏感(例如: 错分的样本)
- 支持向量: $y_i \cdot (\mathbf{w}^T x_i + b) \leqslant 1$
- 错分样本: $y_i \cdot (\mathbf{w}^T x_i + b) \leqslant 0$
- 消除那些“坏”的 *SV* 点的影响
- 新的损失函数: 截断 *hinge loss*

$$l_s(y,\ f) = \begin{cases} 1-s & y \cdot f < s, \\ 1-y \cdot f & s \leqslant y \cdot f < 1, \\ 0 & y \cdot f \geqslant 1. \end{cases}$$

$s \leqslant 0$为截断参数。

自适应加权支持向量机

Adaptively Weighted Support Vector Machine, *Wu and Liu*, 2013

- 设训练样本点的权重依次为 $v_i \geqslant 0,\ i = 1,\ \cdots,\ n$
- 靠近(远离)分类边界的样本点权重大(小)
- 远离分类边界的样本点的权重小
- 适当选取权重可兼顾鲁棒性
- 问题形式为

$$\begin{aligned} \min_{\mathbf{w},\ b,\ \boldsymbol{\xi}} \quad & \frac{1}{2} \parallel \mathbf{w} \parallel_2^2 \ + \ C \sum_{i=1}^{n} v_i \ \xi_i \\ s.t. \quad & y_i \cdot (\mathbf{w}^T \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1, \cdots, n; \\ & \xi_i \geqslant 0, \quad i = 1, \cdots, n. \end{aligned} \tag{7}$$

代价敏感支持向量机

Cost – Sensitive Support Vector Machines

- 对两类类样本点的错分代价存在差异, 可作为*AWSVM*的特例
- 可应用于非平衡数据处理
- 问题形式为

$$\begin{aligned} \min_{\mathbf{w},\ b,\ \boldsymbol{\xi}} \quad & \frac{1}{2}\parallel \mathbf{w} \parallel_2^2 \ + \ C_+ \sum_{i:\ y_i=1}^{n} \xi_i + C_- \sum_{i:\ y_i=-1}^{n} \xi_i \\ s.t. \quad & y_i \cdot (\mathbf{w}^T \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1, \cdots, n; \\ & \xi_i \geqslant 0, \quad i = 1, \cdots, n. \end{aligned} \tag{8}$$

C_+, C_-分别为正负两类样本的错分代价。

k ($k \geqslant 3$) 分类问题

- 转化为多个二分类器(*Multiple Binary*)
  - 一对余 (*One Versus Rest*)
    将第 i 类视作正类, 其他类视作负类, 得到二分类器 f_i, 最终得到 k 个二分类器。新样本 $\mathbf{x}$ 经过全部分类器后，若只有一个 $+1$ 出现，则判断为相应类别; 若输出不止一个 $+1$, 或者没有一个输出为 $+1$，则比较 $f_i(\mathbf{x})$, $i = 1, \cdots, n$, 最大者对应类别作为新样本的类别。
  - 一对一 (*One Versus One*)
    选取分属类别 i, $j (i < j)$ 的样本训练得到二分类器 f_{ij}, 最终得到 $\frac{k(k-1)}{2}$ 个二分类器。新样本 $\mathbf{x}$ 经过全部分类器, 每个分 类器为其“投上一票”, 最后输出为得票最高的类别。
- 同时 k 分类器(*All together*)
  构造向量函数 $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_k(\mathbf{x}))$ 作为同时 k 分类器。对新样本 $\mathbf{x}$ 类别的预测为 $\hat{y} = \underset{j=1,\cdots,k}{\operatorname{argmax}} f_j(\mathbf{x})$。

正则框架

$$\min_{\mathbf{f}\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n} L\left(\mathbf{f}(\mathbf{x}_i),\ y_i\right) + \lambda J(\mathbf{f}) \qquad (**)$$

设 $\mathscr{F}$ 为候选函数族, L 为损失函数, $J(\mathbf{f})$ 为正则项, λ 为调整参数。为消除冗余和降维，通常对函数 $\mathbf{f}$ 加以约束$\sum_{j=1}^{k} f_j(\mathbf{x}) = 0$。
常用损失函数有

- *Naive hinge loss* : $[1 - f_y(\mathbf{x})]_+$
- *Vapnik*, 1998 : $\sum_{j\neq y}[1 - (f_y(\mathbf{x}) - f_j(\mathbf{x}))]_+$
- *Crammer et al.*, 2001 : $[1 - \min_j(f_y(\mathbf{x}) - f_j(\mathbf{x}))]_+$
- *Lee et al.*, 2004 : $\sum_{j\neq y}[1 + f_j(\mathbf{x})]_+$
- *Liu et al.*, 2011:

$$\gamma[(k-1) - f_y(x)]_+ + (1-\gamma)\sum_{j\neq y}[1 + f_j(\mathbf{x})]_+, \quad \gamma \in [0,1]$$

增强多分类支持向量机

Reinforced Multicategory Support Vector Machine, *Liu and Yuan*(2011)

- 问题形式为

$$\begin{aligned} \min_{\mathbf{f}\in\mathscr{F}} \quad & \frac{1}{n}\sum_{i=1}^{n} l\left(\mathbf{f}(\mathbf{x}_i),\ y_i\right)+\lambda J(\mathbf{f}) \\ s.t. \quad & \sum_{j=1}^{k} f_j(\mathbf{x})=0. \end{aligned} \tag{9}$$

- 损失: $\gamma[(k-1)-f_y(x)]_+ + (1-\gamma)\sum_{j\neq y}[1+f_j(\mathbf{x})]_+, \quad \gamma\in[0,1]$
- “双核驱动”
  - 第一部分强化 $f_j(x)=k-1,\ j=y$
  - 第二部分强化 $f_j(x)=-1,\ j\neq y$
- 两部分适当加权平衡, 全面提升

基于角度的多分类支持向量机

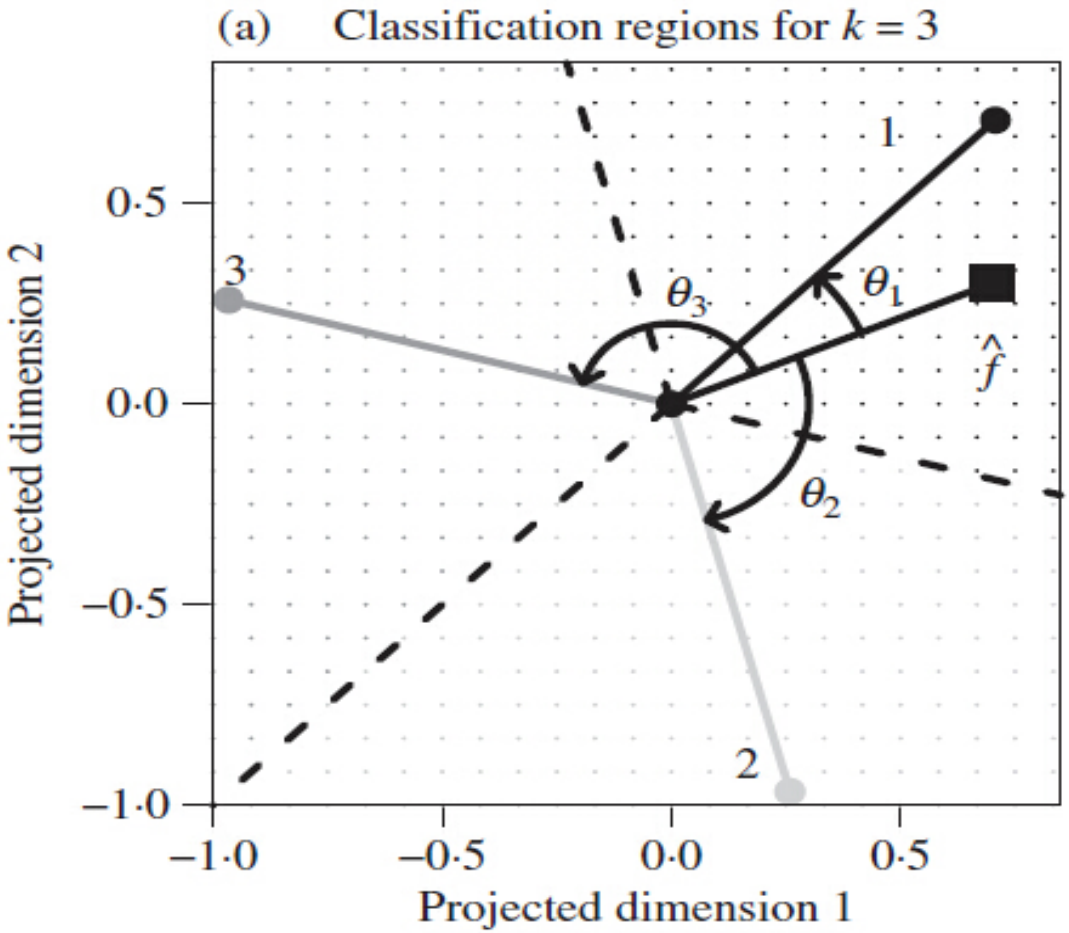
Angle – based Multicategory Support Vector Machine, *Zhang and Liu*(2014)

- 构造 $\mathbb{R}^{k-1}$ 上中心在原点的正 k 面体, k 个顶点到原点距离为1
- 将样本 $(\mathbf{x}_i,\ y_i)$ 按所属类别映射到顶点向量W_{y_i}, $i = 1,\ \cdots,\ n$
- 以内积 $< f_y(\mathbf{x}), W_y >$ 表征泛函间隔(*functional margin*)
- 约束 $\sum_{j=1}^{k} < f_j(\mathbf{x}), W_j >= 0$ 自然成立
- 问题形式为

$$\min_{\mathbf{f}\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n} l\left(< \mathbf{f}(\mathbf{x}_i),\ W_{y_i} >\right) + \lambda J(\mathbf{f}) \tag{10}$$

- 二分类支持向量机可视作其特例

基于角度的多分类问题图解



计算实现

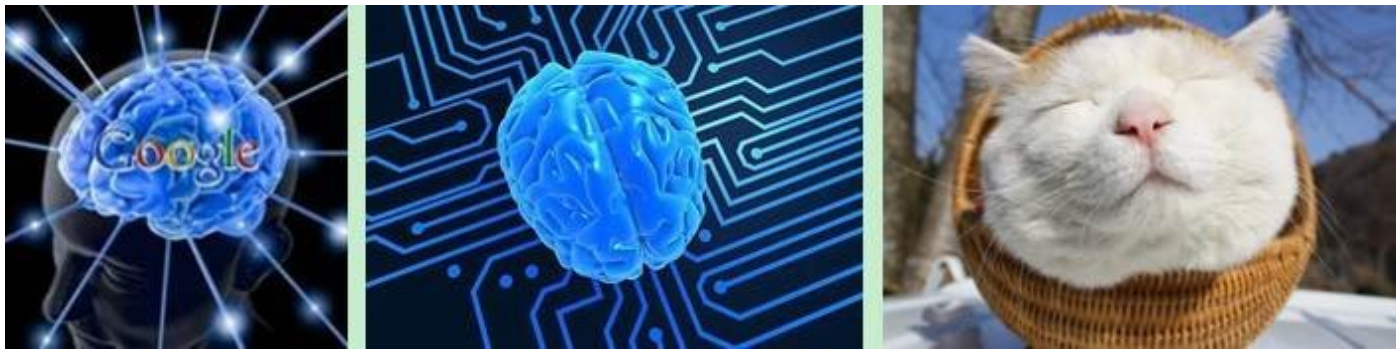
- LibSVM (C++):
- R(e1071 package):
- SVMLight(C)
- Matlab SVM toolbox
- Torch (C++)
- Weka (Java)

总结

- *SVM* 是一种常用的模式分类方法, 应用范围广
- 最大间隔的思想
- 处理非线性问题的核技巧
- 调整 *SVM* 正则框架改进方法
- *SVM* 可以应对大数据的挑战

Deep Learning

2012年6月，《纽约时报》披露了Google Brain项目，吸引了公众的广泛关注。这个项目由斯坦福大学教授Andrew Ng和世界顶尖专家Jeff Dean共同主导，用16000个CPU Core的并行计算平台训练一种称为“深度神经网络”(Deep Neural Networks)的机器学习模型,内部共有10亿个节点,在语音识别和图像识别等领域获得了巨大的成功。



2012年11月，微软在中国天津的一次活动上公开演示了一个全自动的同声传译系统。后面支撑的关键技术也是DNN。

2013年1月，百度年会上，CEO李彦宏宣布要成立百度研究院，其中第一个成立的就是“深度学习研究所”（IDL，Institue of Deep Learning）。

传统的Machine Learning

Machine Learning基本过程：

- 最后一个部分是机器学习的部分
- 中间三部分，概括起来就是特征表达(但实际中一般都是靠人工提取特征)
- Deep Learning做的就是不要人参与特征的选取过程

Neural Network的发展

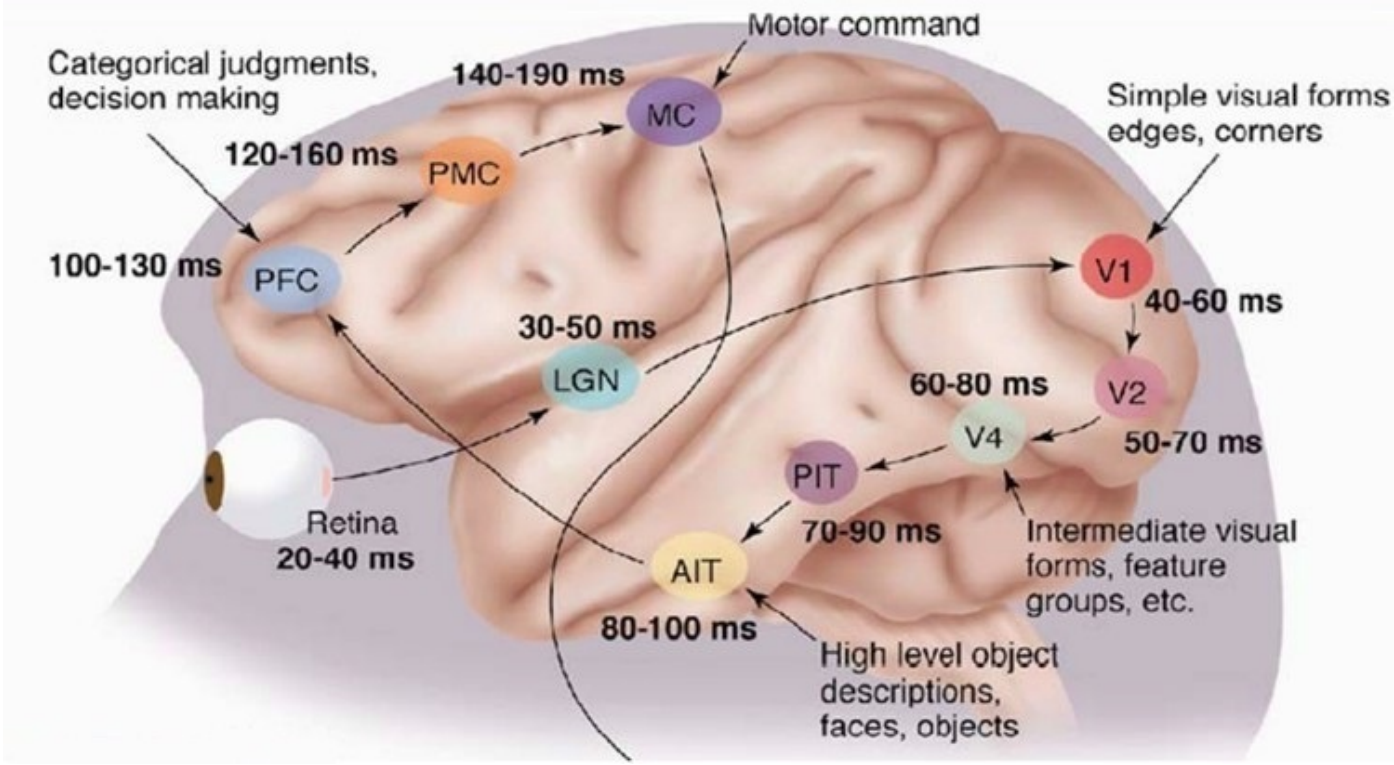
- **浅层学习浪潮**——20世纪80年代末期，用于人工神经网络的反向传播算法(Back Propagation算法)，掀起了基于统计模型的机器学习热潮。
  20世纪90年代，各种各样的浅层机器学习模型相继被提出，例如支撑向量机(SVM)、Boosting、 最大熵方法等。
- **深度学习浪潮**——2006年，加拿大多伦多大学教授Geoffrey Hinton和他的学生在《Sciences》上发表了一篇文章，开启了深度学习在学术界和工业界的浪潮。

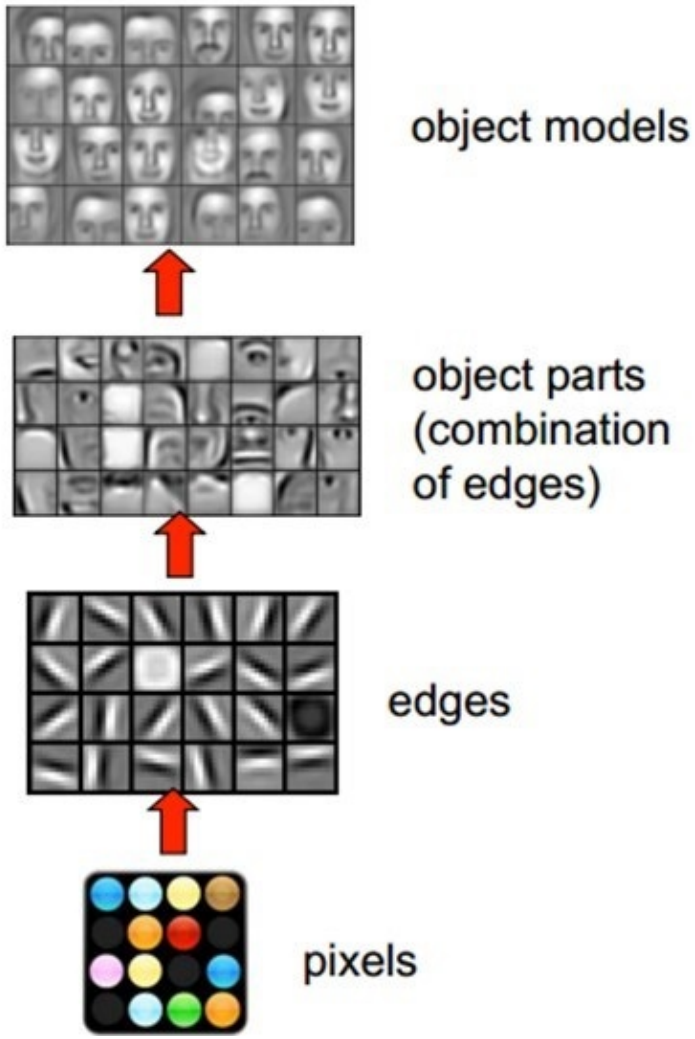
Deep Learning

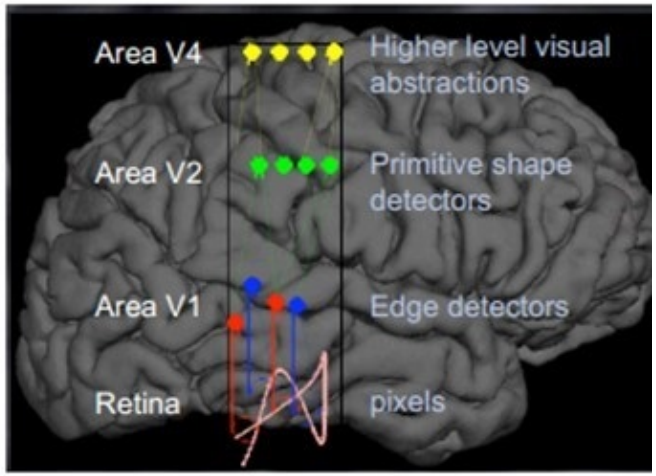
- 建立、模拟人脑进行分析学习的神经网络，它模仿人脑的机制来解释数据，例如图像，声音和文本。
- 浅层结构算法局限性在于有限样本和计算单元情况下对复杂函数的表示能力有限，针对复杂分类问题其泛化能力受到一定制约。
- 深度学习通过学习一种深层非线性网络结构，实现复杂函数逼近，表征输入数据分布式表示，并展现了从少数样本集中学习数据集本质特征的能力。

人脑视觉机理

- 1981 年的诺贝尔医学奖，颁发给了David Hubel和Torsten Wiesel以及 Roger Sperry。前两位的主要贡献是发现了视觉系统的信息处理：可视皮层是分级的。
- David Hubel和Torsten Wiesel 发现了一种被称为“方向选择性细胞(Orientation Selective Cell)” 的神经元细胞。当瞳孔发现了眼前的物体的边缘，而且这个边缘指向某个方向时，这种神经元细胞就会活跃。
- 神经-中枢-大脑的工作过程，或许是一个不断迭代、不断抽象的过程。


Motor command
Categorical judgments, decision making
140-190 ms
MC
Simple visual forms edges, corners
120-160 ms
PMC
100-130 ms
PFC
V1
40-60 ms
30-50 ms
LGN
60-80 ms
V2
V4
50-70 ms
PIT
Retina
20-40 ms
70-90 ms
Intermediate visual forms, feature groups, etc.
AIT
80-100 ms
High level object descriptions, faces, objects


object models
object parts
(combination
of edges)
edges
pixels


Area V4
Higher level visual
abstractions
Area V2
Primitive shape
detectors
Area V1
Edge detectors
Retina
pixels

- 从原始信号摄入开始（瞳孔摄入像素Pixels）
- 接着做初步处理（大脑皮层某些细胞发现边缘和方向）
- 然后抽象（大脑判定，眼前的物体的形状）
- 然后进一步抽象（大脑进一步判定该物体是谁的脸）
- 关键在于抽象和迭代。从原始信号开始，做低级抽象，逐渐向高级抽象迭代。
- 从低层到高层的特征表示越来越抽象，越来越能表现语义或者意图。抽象层面越高，存在的可能猜测就越少。

Deep Learning基本原理

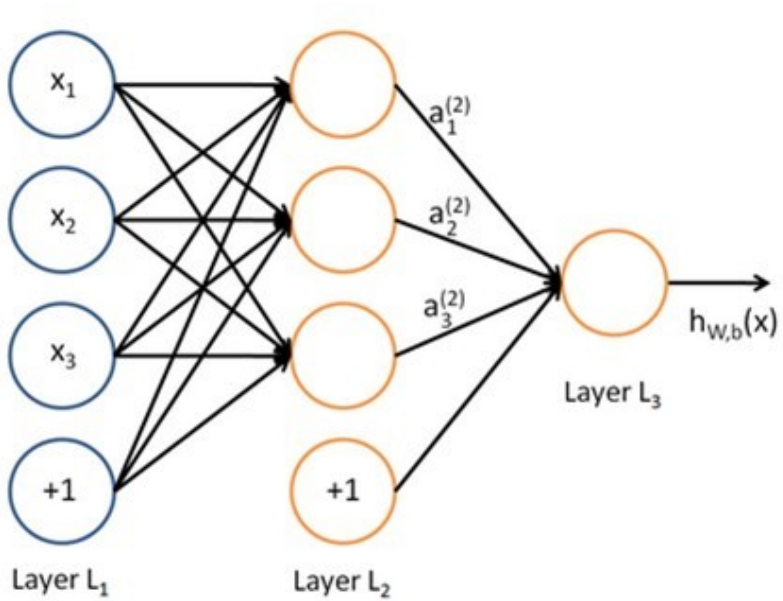
- 组合低层特征形成更加抽象的高层表示属性，以发现数据的分布式特征表示
- 堆叠多个层，实现对输入信息进行分级表达了。
- “深度模型”是手段，“特征学习”是目的。

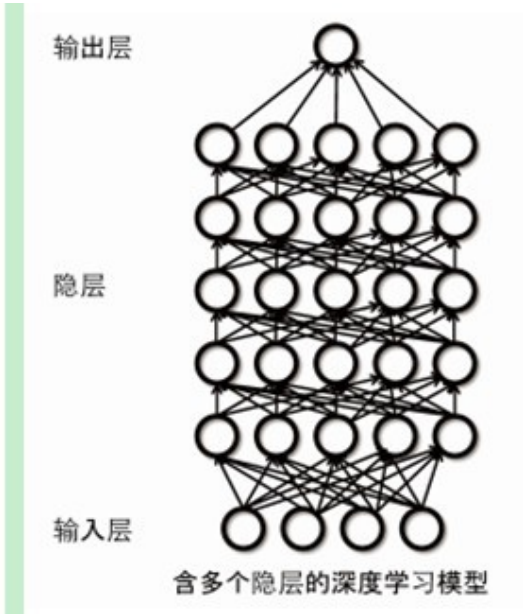
Deep Learning训练过程

- 自下上升非监督学习（feature learning过程）：这一步可以看作是一个无监督训练过程，区别于传统神经网络初值随机初始化
- 自顶向下的监督学习（fine-tune）：
  这一步是在前面学习获得各层参数进的基础上，在最顶的编码层添加一个分类器，而后通过带标签数据的监督学习，利用梯度下降法去微调整个网络参数。

Deep Learning与BP神经网络区别

- 二者的相同在于Deep Learning采用了神经网络相似的分层结构，系统由包括输入层、隐层（多层）、输出层组成的多层网络，只有相邻层节点之间有连接，同一层以及跨层节点之间相互无连接；这种分层结构接近人类大脑的结构。
- 不同在于BP神经网络采用采用迭代的算法来训练整个网络，随机设定初值，计算当前网络的输出，然后根据当前输出和label之间的差去改变前面各层的参数，直到收敛（整体是一个梯度下降法）。而Deep Learning 整体上是一个layer-wise（分层计算）的训练机制。


x_1
x_2
x_3
+1
$a_1^{(2)}$
$a_2^{(2)}$
$a_3^{(2)}$
+1
$h_{W,b}(x)$
Layer L_1
Layer L_2
Layer L_3


输出层
隐层
输入层
含多个隐层的深度学习模型

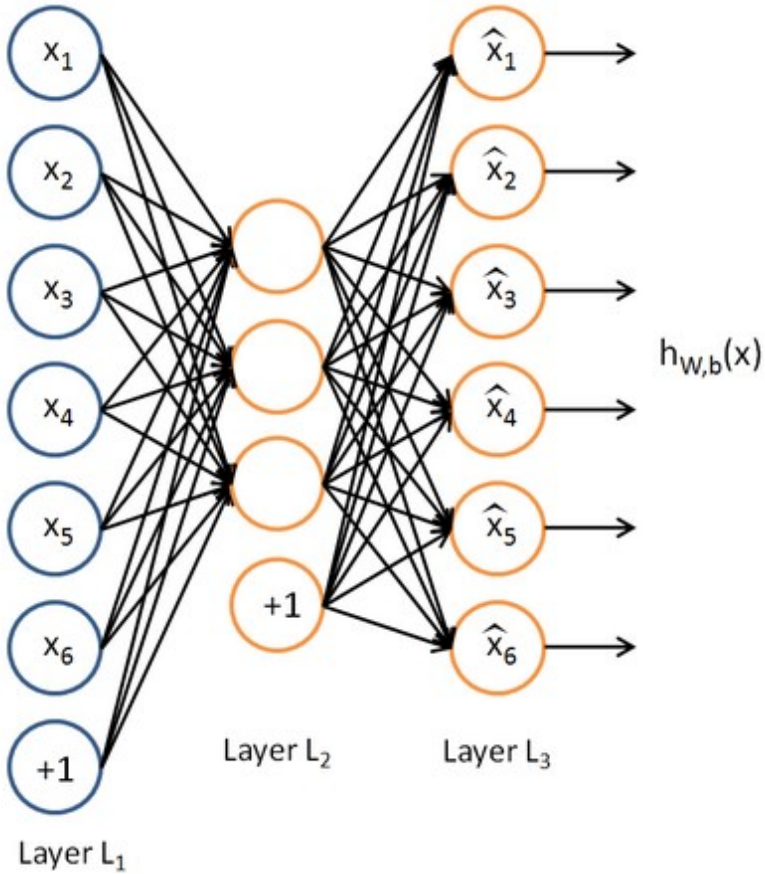
BP算法存在的问题

- 梯度弥散问题（Gradient Diffusion）：层数较多时，传到前面层的误差校正信号越来越小；
- 局部极值问题:收敛到局部最小值：尤其是从远离最优区域开始的时候（随机值初始化会导致这种情况的发生）；
- 一般只能用有标签的数据来训练，但大部分的数据是没标签的；

Deep Learning的优势

- 多隐层的人工神经网络具有优异的特征学习能力，学习得到的特征对数据有质的刻画，对输入信息进行分级表达，有利于可视化或分类；
- 深度神经网络在训练上的难度，可以通过“逐层初始化”（layer-wise pre-training）来有效克服，逐层初始化是通过无监督学习实现的。

Sparse Auto-Encoder



- Auto-Encoder是一种无监督学习算法，它使用了BP传播算法，让目标值等于输入值 $h_{W,b}(x) \approx x$。
- 在隐藏层加Sparsity的限制来发现输入数据中的结构。
- 引入Sparsity参数$\rho = 0.05$，使得

$$\hat{p}_j = \frac{1}{N}\sum_{n=1}^{N}[a_j(x^n)] \approx \rho$$

其中$\hat{p}_j$隐含层节点的平均激活度，a_j隐含第j个节点的激活度

- 具体实现是在原Auto-Encoder BP算法中增加惩罚项

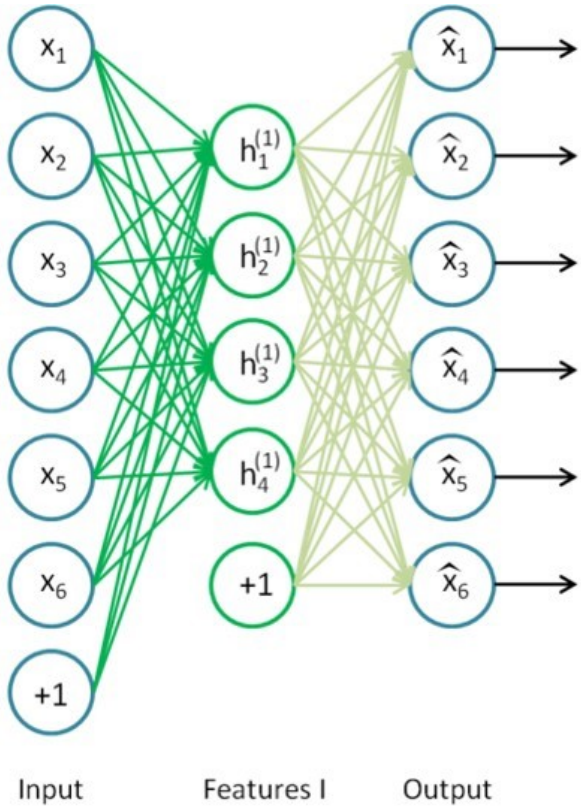
$$E_{sparse}(W,b) = E(W,b) + \beta \cdot \sum_{j=1}^{J} KL(\rho \| \hat{\rho}_j)$$

其中

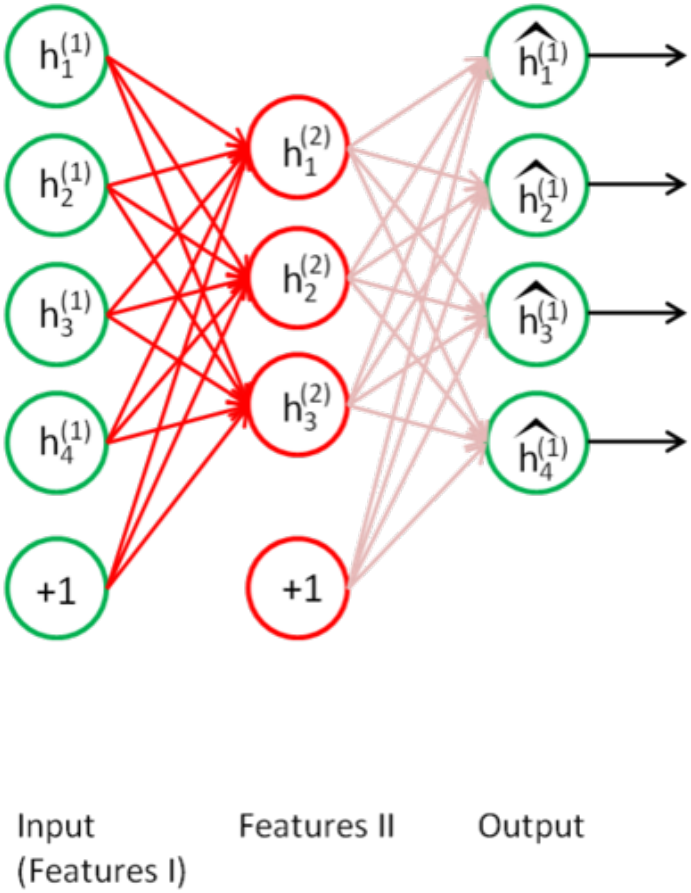
$$\sum_{j=1}^{J} KL(\rho \| \hat{\rho}_j) = \sum_{j=1}^{J} \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j}$$

Stacked Auto-Encoder

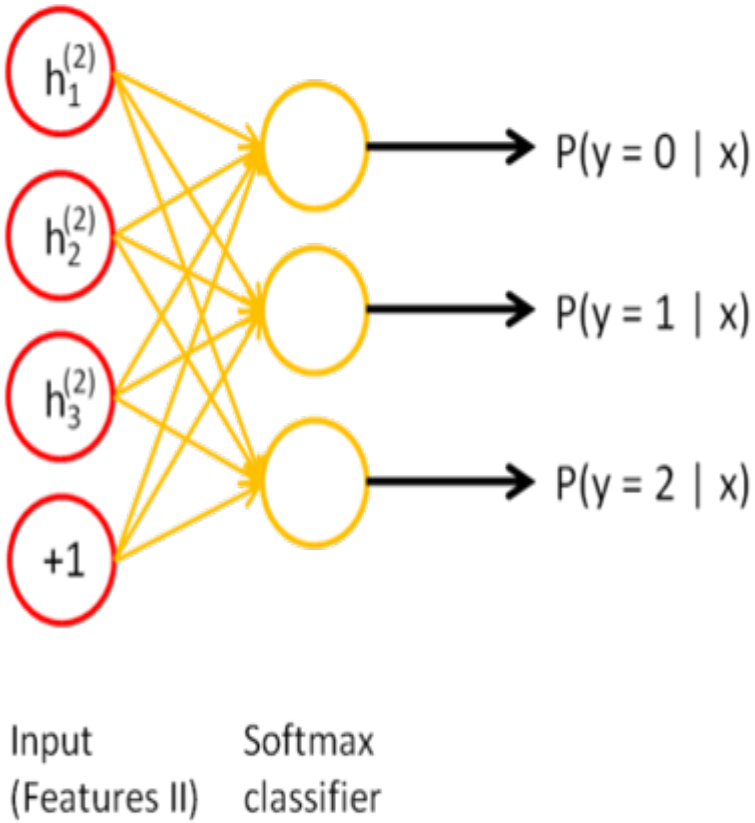
- 用原始输入数据作为输入，利用sparse autoencoder方法训练出第一个隐含层结构的网络参数，得到原始输入的一阶特征表示$h^{(1)}(k)$



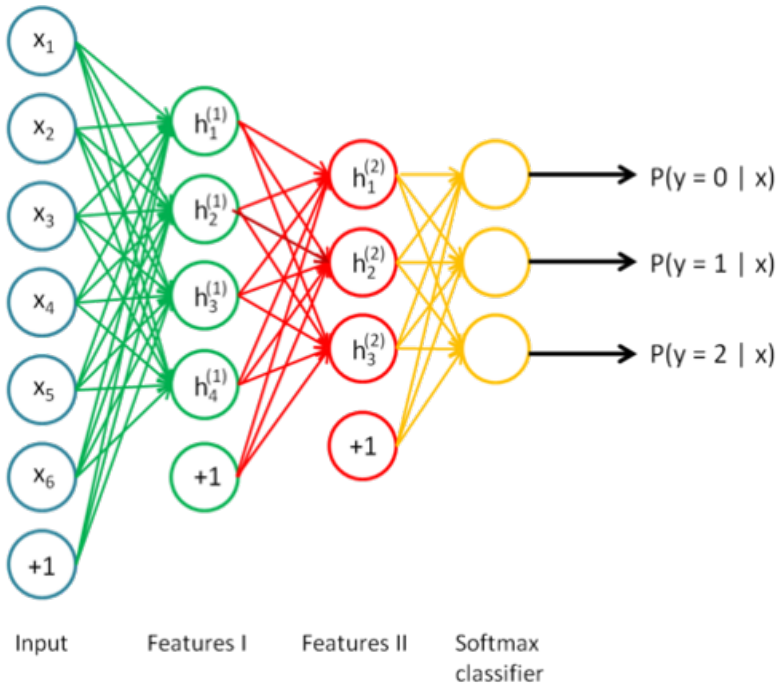
- 把上一层的输出作为下一层输入，再利用sparse autoencoder方法训练出第二个隐含层网络的参数，学习二阶特征$h^{(2)}(k)$



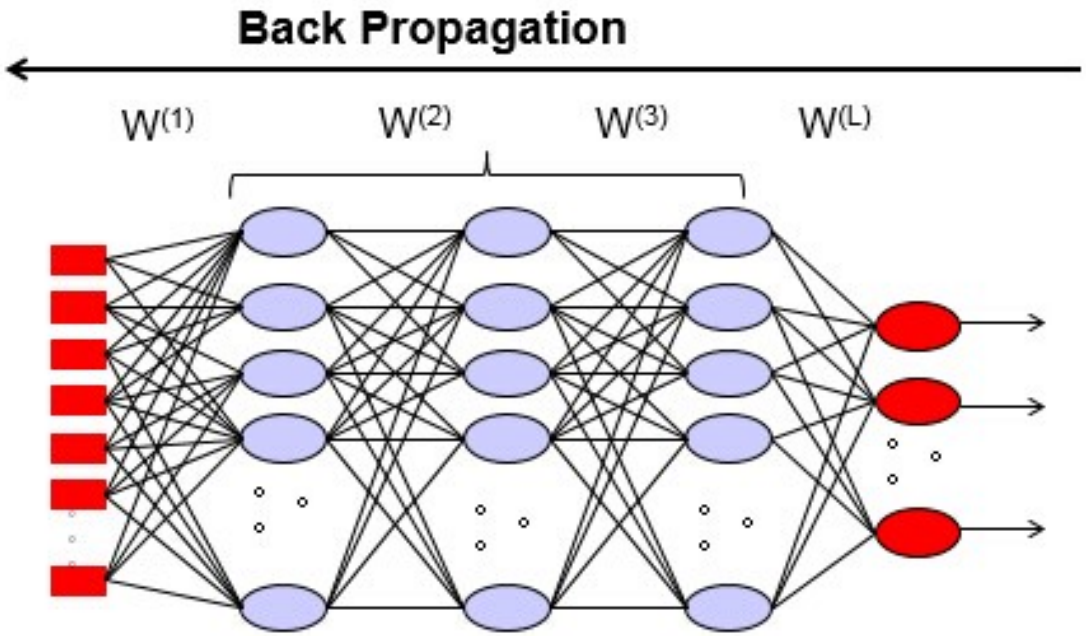
- 将二节特征作为多分类器softmax的输入，利用原始数据的标签来训练出softmax分类器的网络参数，构建一个包含两个隐藏层和一个最终softmax分类器层的Stacked Auto-Encoder网络



- 计算含此Stacked Auto-Encoder的整个网络的损失函数，以及整个网络对每个参数的偏导函数值



- 调用标准的BP算法对网络权值进行Fine turning



应用举例（Benchmarks）

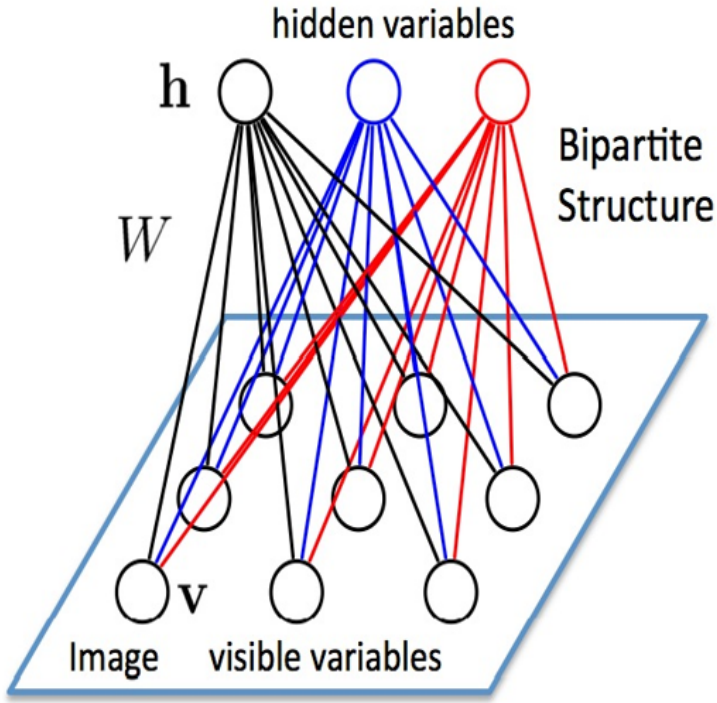
basic: subset of MNIST digits.	(10 000 training samples)
rot: applied random rotation (angle between 0 and 2π radians)	
bg-rand: background made of random pixels (value in $0 \ldots 255$)	
bg-img: background is random patch from one of 20 images	
rot-bg-img: combination of rotation and background image	
rect: discriminate between tall and wide rectangles.	
rect-img: same but rectangles are random image patches	
convex: discriminate between convex and non-convex shapes.	

测试误差比较

Problem	SVM_{rbf}	DBN-1	DBN-3	SAA-3	SdA-3 (ν)	$SVM_{rbf}(\nu)$
basic	3.03±0.15	3.94±0.17	3.11±0.15	3.46±0.16	2.80±0.14 (10%)	3.07 (10%)
rot	11.11±0.28	14.69±0.31	10.30±0.27	10.30±0.27	10.29±0.27 (10%)	11.62 (10%)
bg-rand	14.58±0.31	9.80±0.26	6.73±0.22	11.28±0.28	10.38±0.27 (40%)	15.63 (25%)
bg-img	22.61±0.37	16.15±0.32	16.31±0.32	23.00±0.37	16.68±0.33 (25%)	23.15 (25%)
rot-bg-img	55.18±0.44	52.21±0.44	47.39±0.44	51.93±0.44	44.49±0.44 (25%)	54.16 (10%)
rect	2.15±0.13	4.71±0.19	2.60±0.14	2.41±0.13	1.99±0.12 (10%)	2.45 (25%)
rect-img	24.04±0.37	23.69±0.37	22.50±0.37	24.05±0.37	21.59±0.36 (25%)	23.00 (10%)
convex	19.13±0.34	19.92±0.35	18.63±0.34	18.41±0.34	19.06±0.34 (10%)	24.20 (10%)

Restricted Boltzmann Machine, RBM

- 受限玻尔兹曼机(简称RBM)是由Hinton和Sejnowski于1986年提出的一种生成式随机神经网络(generative stochastic neural network)
- RBM 由一些visible units(即数据样本)和一些hidden units构成，visible units和hidden units都是二元变量，即其状态取$\{0,1\}$
- 整个网络是一个二部图，只有visible units和hidden units之间才会存在边，visible units 之间以及hidden units之间都不会有边连接



上图所示的RBM含有9个visible units (构成一个向量v)和3个hidden units (构成一个向量h)，W是一个9×3的矩阵，表示visible units和hidden units之间的边的权重。

RBM的学习目标-极大化似然(Maximizing likelihood)

RBM是一种基于能量(Energy-based)的模型，其可见变量v和隐藏变量h的联合配置(joint configuration)的能量为：

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

其中θ是RBM的参数$\{W,a,b\}$, W为visible units和hidden units之间的边的权重，b和a分别为visible units 和hidden units的偏置(bias)。

有了v和h的联合配置的能量之后，可以得到v和h的联合似然:

$$P_\theta(v,h) = \frac{1}{Z(\theta)}\exp(-E(v,h;\theta)) = \frac{1}{Z(\theta)}\prod_{ij} e^{W_{ij}v_i h_j}\prod_i e^{b_i v_i}\prod_j e^{a_j h_j}$$

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

其中$Z(\theta)$是归一化因子，也称为配分函数(partition function)。

矩阵表达:

$$P_\theta(V,H) = \frac{1}{Z(\theta)}\exp(V^T WH + a^T H + b^T V)$$

观测数据的似然函数$P(v)$，$P(v)$可由联合似然$P(v,h)$对h求边缘分布得到:

$$P_\theta(v) = \frac{1}{Z(\theta)}\sum_h \exp[v^T Wh + a^T h + b^T v]$$

极大化$P(v)$，也等同于极大化对数观测似然$L(\theta) = \log(P(v))$，来得到RBM的参数。

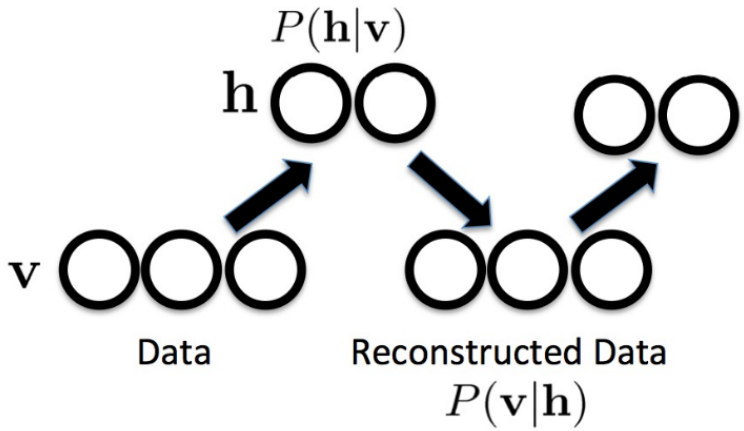
$$L(\theta) = \frac{1}{N}\sum_{n=1}^{N} \log P_\theta(v^{(n)})$$

通过随机梯度下降(stochastic gradient descent)来极大化$L(\theta)$，首先需要求$L(\theta)$对W的导数:

$$\begin{aligned}\frac{\partial L(\theta)}{\partial W_{ij}} &= \frac{1}{N}\sum_{n=1}^{N}\frac{\partial}{\partial W_{ij}}\log\left(\sum_h \exp[v^{(n)T}Wh + a^T h + b^T v^{(n)}]\right) \\ &\quad -\frac{\partial}{\partial W_{ij}}\log Z(\theta) \\ \text{化简得} &= E_{P_{data}}[v_i h_j] - E_{P_\theta}[v_i h_j] = E_{P_{data}}[v_i h_j] - \sum_{v,h} v_i h_j P_\theta(v,h)\end{aligned}$$

上式前一项比较好计算，只需要求$v_i h_j$在全部数据集上的平均值即可。而后者涉及到v，h的全部$2^{|v|+|h|}$种组合，计算量非常大，为此Hinton提出了Contrastive Divergence的学习方法(本质上是一种MCMC方法)。

RBM的学习方法-CD(Contrastive Divergence，对比散列)



首先根据数据v来得到h的状态，然后通过h来重构(Reconstruct)可见向量v^1，然后再根据v^1来生成新的隐藏向量h^1(Gibbs抽样)。因为RBM 的特殊结构(层内无连接，层间有连接)， 所以在给定v时，各个隐藏单元h_j的激活状态之间是相互独立的，反之，在给定h 时，各个可见单元的激活状态v_i也是相互独立的，可得到：

抽样分布:

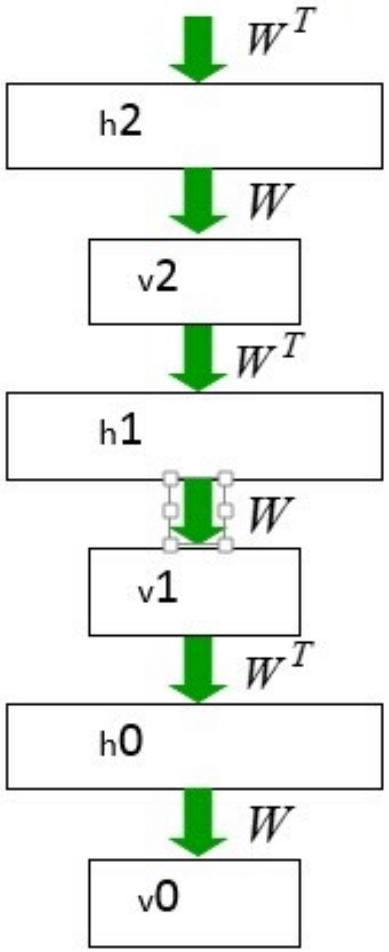
$$P(h|v) = \prod_j P(h_j v) P(h_j = 1|v) = \frac{1}{1 + \exp(-\sum_i W_{ij} v_i - a_j)}$$

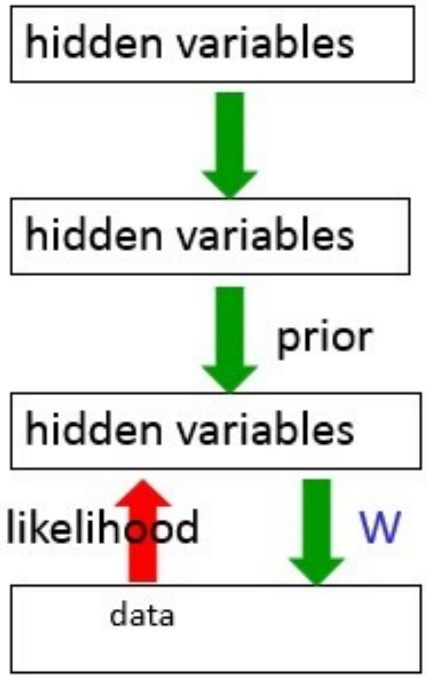
$$P(v|h) = \prod_i P(v_i h) P(v_i = 1|h) = \frac{1}{1 + \exp(-\sum_j W_{ij} h_j - b_i)}$$

重构的可见向量v^1和隐藏向量h^1就是对$P(v,h)$的一次抽样，多次抽样得到的样本集合可以看做是对$P(v,h)$的一种近似，使得均值计算变得可行。

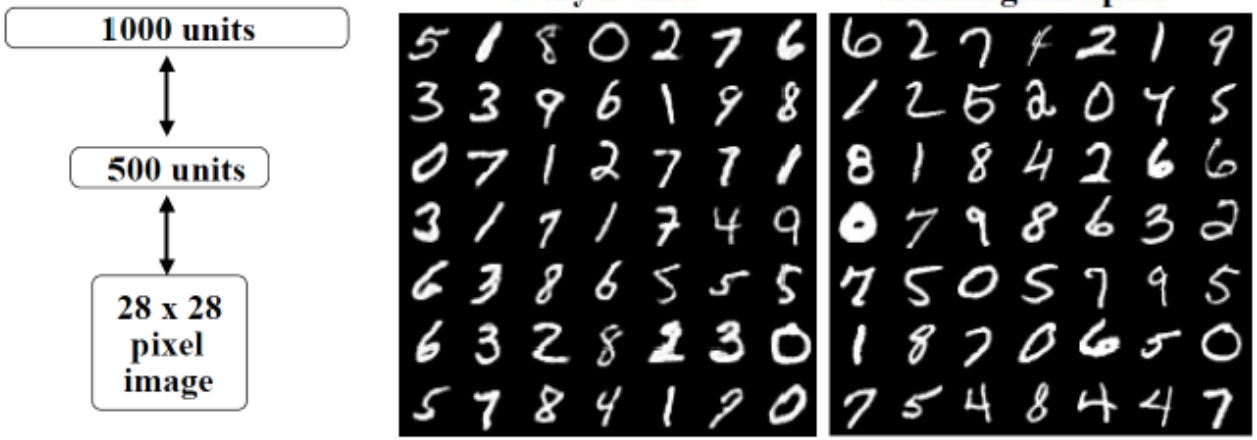
RBM的权重的学习算法

- 逐层迭代推断
- Fine Turning





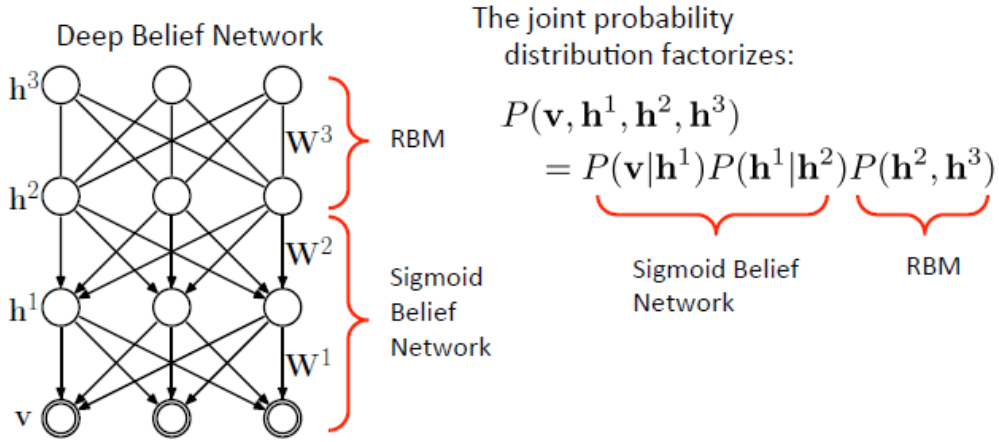
应用举例(MNIST:2-layer BM)



60,000 training and 10,000 testing examples
0.9 million parameters
Gibbs sampler for 100,000 steps

After discriminative fine-tuning: 0.95% error rate
Compare with DBN 1.2%, SVM 1.4%

深信度网络(Deep Belief Networks)



Deep Belief Networks是在靠近可视层的部分使用贝叶斯信念网络（即有向图模型），而在最远离可视层的部分使用Restricted Boltzmann Machine的模型。

$$P(h^2, h^3) = \frac{1}{Z(W^3)} \exp\left[h^{2T} W^3 h^3\right]$$

$$P(h^1|h^2) = \prod_j P(h_j^1|h^2) \qquad P(h_j^1 = 1|h^2) = \frac{1}{1 + \exp\left(-\sum_k W_{jk}^2 h_k^2\right)}$$

$$P(v|h^1) = \prod_i P(v_i|h^1) \qquad P(v_i = 1|h^1) = \frac{1}{1 + \exp\left(-\sum_j W_{ij}^1 h_j^1\right)}$$

DBN训练过程

1.自下而上的预训练过程

1. 可视层和第一个隐藏层之间进行吉布斯采样，直到在第一个隐藏层得到可视层的一个表达；
2. 固定可视层与第一个隐藏层之间的参数，即$\theta = \{W, a, b\}$；
3. 将第一个隐藏层作为可视层与第二个隐藏层进行无监督学习，以此类推直到最后一个隐藏层，得到了之前各层之间的参数表。

2.自上而下的微调过程

1. 初始化分类层（最后一个隐藏层作为分类层的输入层，分类结果作为分类层输出）的参数表；
2. 将数据从可视层经过每一个隐藏层的参数传递到最后一个隐藏层，在最后一个隐藏层与分类层参数表计算，解冻之前各层参数表，通过梯度下降算法对所有参数进行微调（所有参数是包括所有隐藏层之间的参数）。

应用举例（语音识别）

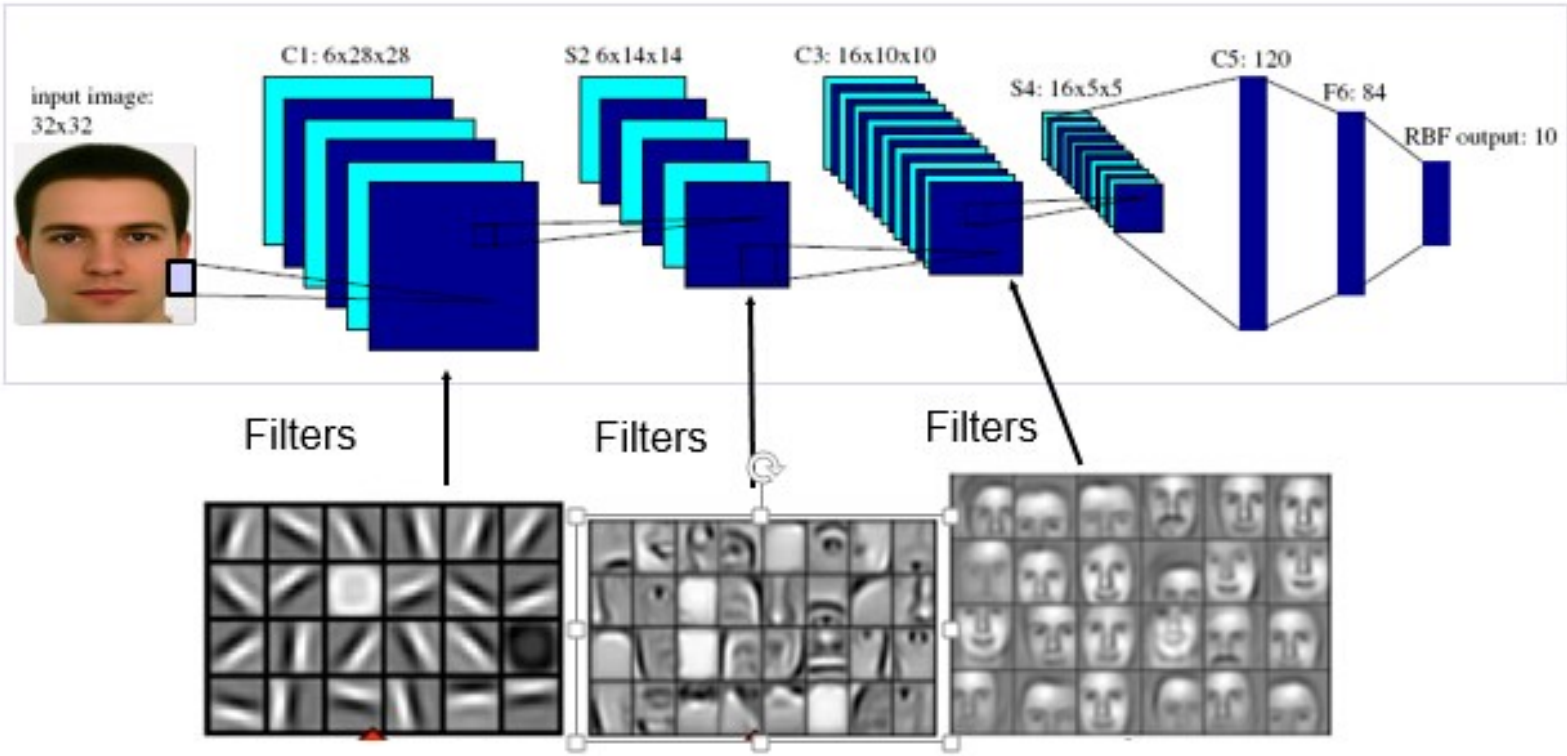
(Dahl et al., 2010)

Method	PER
Stochastic Segmental Models	36.0%
Conditional Random Field	34.8%
Large-Margin GMM	33.0%
CD-HMM	27.3%
Augmented conditional Random Fields	26.6%
Recurrent Neural Nets	26.1%
Bayesian Triphone HMM	25.6%
Monophone HTMs	24.8%
Heterogeneous Classifiers	24.4%
Deep Belief Networks(DBNs)	**23.0%**
Triphone HMMs discriminatively trained w/ BMMI	22.7%
Deep Belief Networks with mcRBM feature extraction	**20.5%**

Convolutional Neural Network

- 卷积神经网络是人工神经网络的一种，已成为当前语音分析和图像识别领域的研究热点
- 1962年Hubel和Wiesel通过对猫视觉皮层细胞的研究，提出了感受野(receptive field)的概念
- 1984年日本学者Fukushima基于感受野概念提出的神经认知机(neocognitron)可以看作是卷积神经网络的第一个实现网络，也是感受野概念在人工神经网络领域的首次应用
- CNN为识别二维形状而特殊设计的一个多层感知器
- 每层由多个二维平面组成，而每个平面由多个独立神经元组成
- 对平移、比例缩放、倾斜或者其他形式的变形具有高度不变性

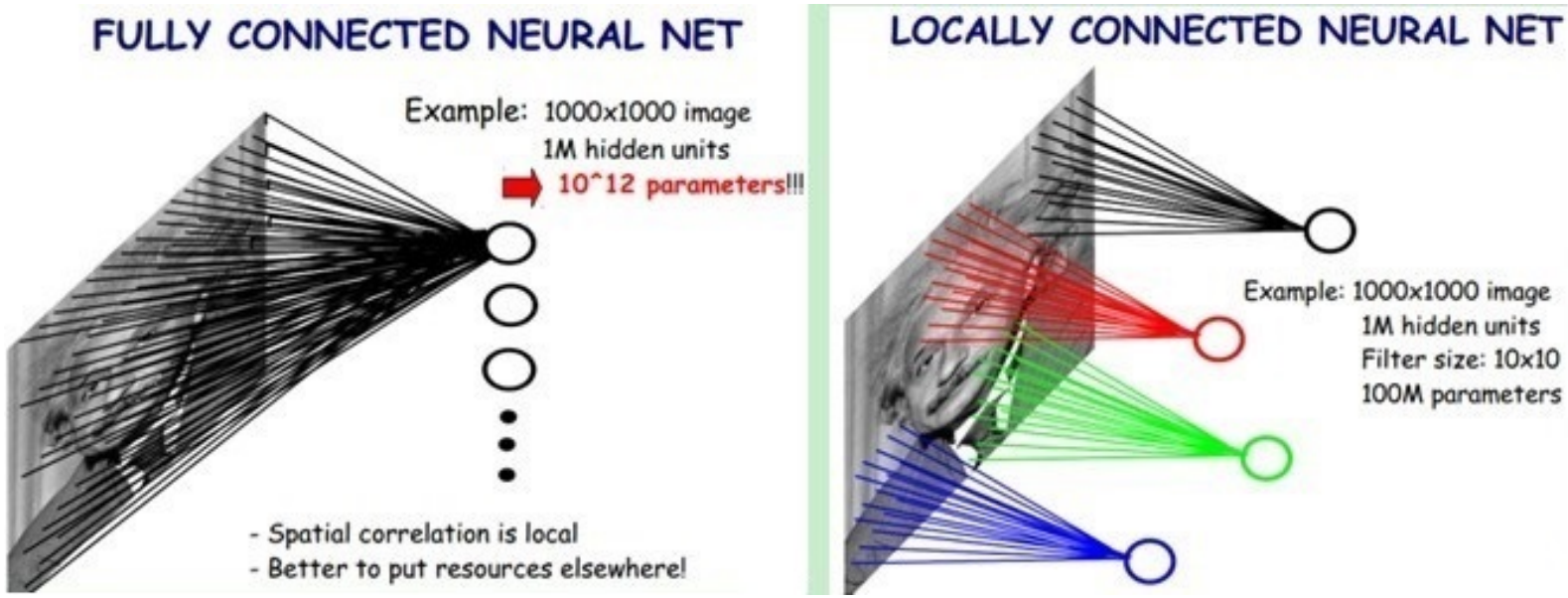
经典的LeNet5结构图



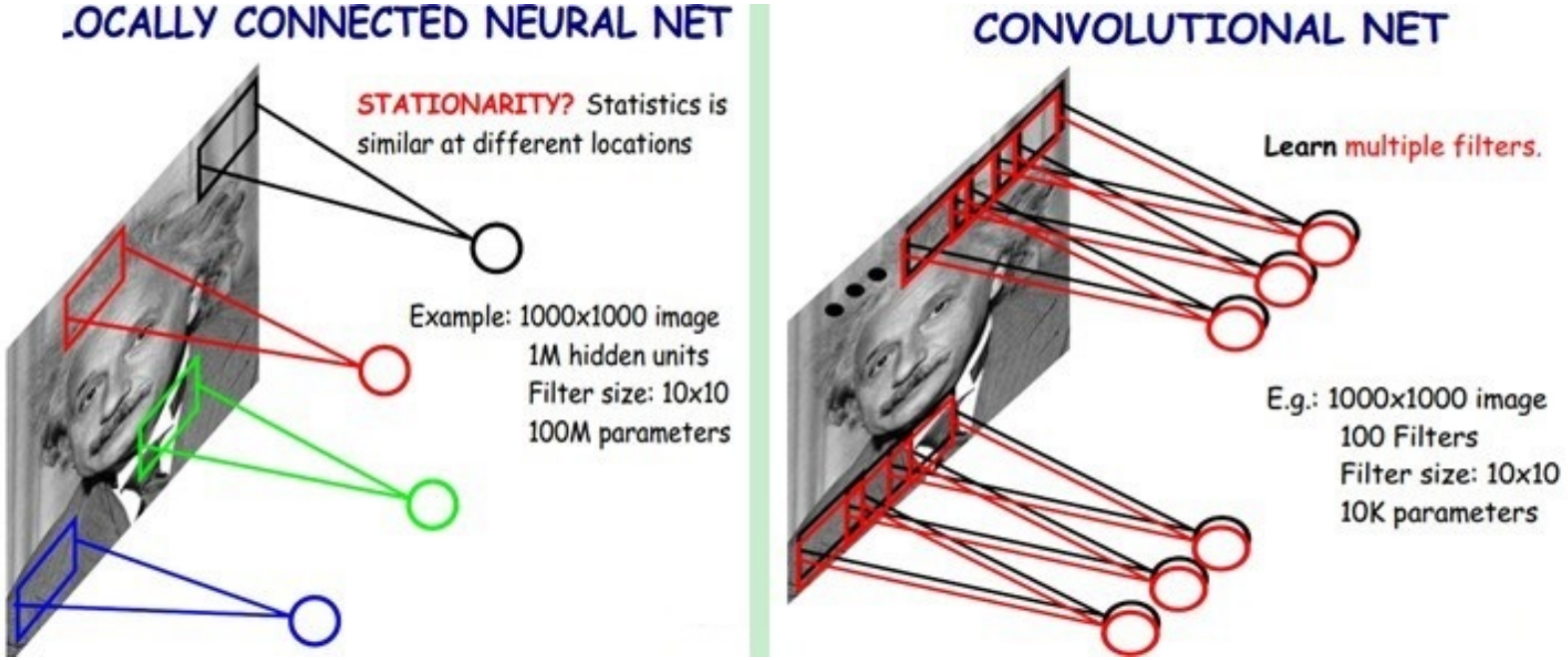
- 每输入一张32 × 32大小的图片，就输出一个84维的向量，这个向量即我们提取出的特征向量
- 网络的C1层是由6张28 × 28大小的特征图构成，其来源是我们用6个5 × 5大小的patch对32 × 32大小的输入图进行convolution得到，28 = 32 − 5 + 1，其中每次移动步伐为1 个像素
- 对2 × 2个像素进行pooling，S2层变成了6张14 × 14大小的特征图
- C3层也是一个卷积层，同样通过16个5 × 5大小的patch去卷积层S2，得到只有10 × 10大小的特征
- 对2 × 2个像素进行pooling，S4层变成了16张5 × 5大小的特征图
- *C*层为特征提取层（卷积层），每个神经元的输入与前一层的局部感受野相连，并提取该局部的特征；*S*层是特征映射层（采样层），网络的每个计算层由多个特征映射组成，每个特征映射为一个平面，平面上所有神经元的权值相等

局部感受野和权值共享

- 通过感受野和权值共享减少了神经网络需要训练的参数的个数
- 在网络的输入是多维图像时表现的更为明显，使图像可以直接作为网络的输入，避免了传统识别算法中复杂的特征提取和数据重建过程

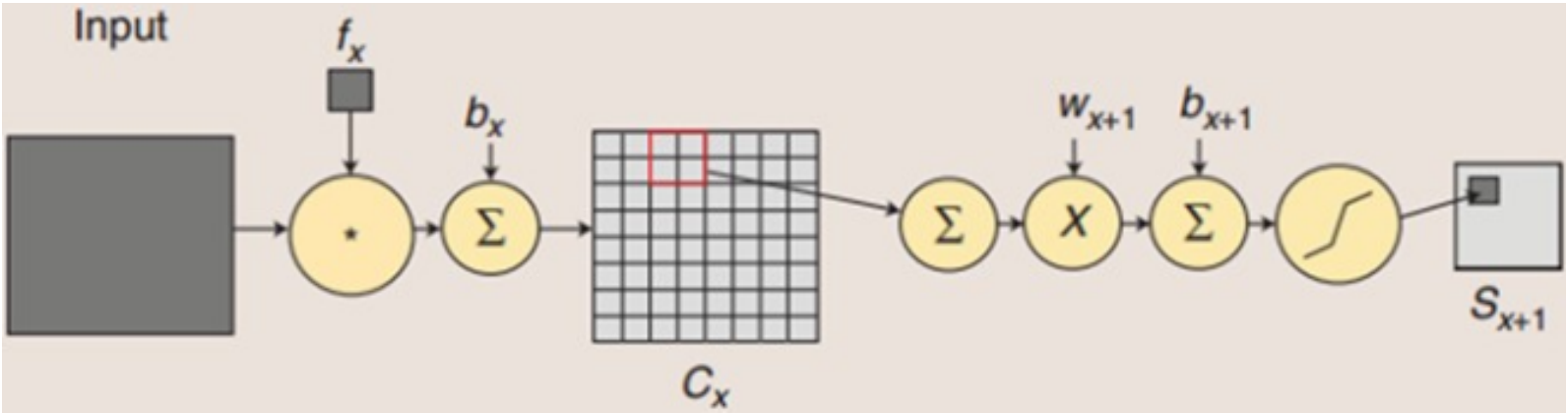


- 图中有1000×1000像素的图像，有1百万个隐层神经元，全连接的话，就有10^{12} 个权值参数
- 假如局部感受野是10×10，隐层每个感受野只需要和这10×10的局部图像相连接，所以只需要10^8 个连接（参数）
- 每一个神经元存在100个连接权值参数，如果每个神经元的这些参数是相同的，也就是说每个神经元用的是同一个卷积核去卷积图像，这样两层间的连接只有100 个参数



- 假设一种filter提取图像的一种特征，有100种filters，每种filter的参数不一样，去卷积图像就得到对图像的不同特征的放映，称之为Feature Map，100个Feature Map组成了一层神经元
有$100 \times 100 = 10000$个参数，见图右：不同的颜色表达不同的滤波器

CNN训练过程



- **卷积过程：**用一个可训练的滤波器fx去卷积一个输入的图像（第一阶段是输入的图像，后面的阶段就是Feature Map 了），然后加一个偏置b_x，得到卷积层C_x。
- **下采样过程：**每邻域n个像素通过pooling步骤变为一个像素，然后通过标量W_{x+1}加权，再增加偏置b_{x+1}，然后通过一个sigmoid激活函数，产生一个大概缩小n 倍的特征映射图S_{x+1}。

卷积层前向

$$x_j^l = f\left(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l\right)$$

其中，M_j是输入map的集合，卷积channel；k_{ij}^l是卷积核，也即是权值。

采样层前向

$$x_j^l = f\left(\beta_j^l down\left(x_j^{l-1}\right) + b_j^l\right)$$

卷积层后向

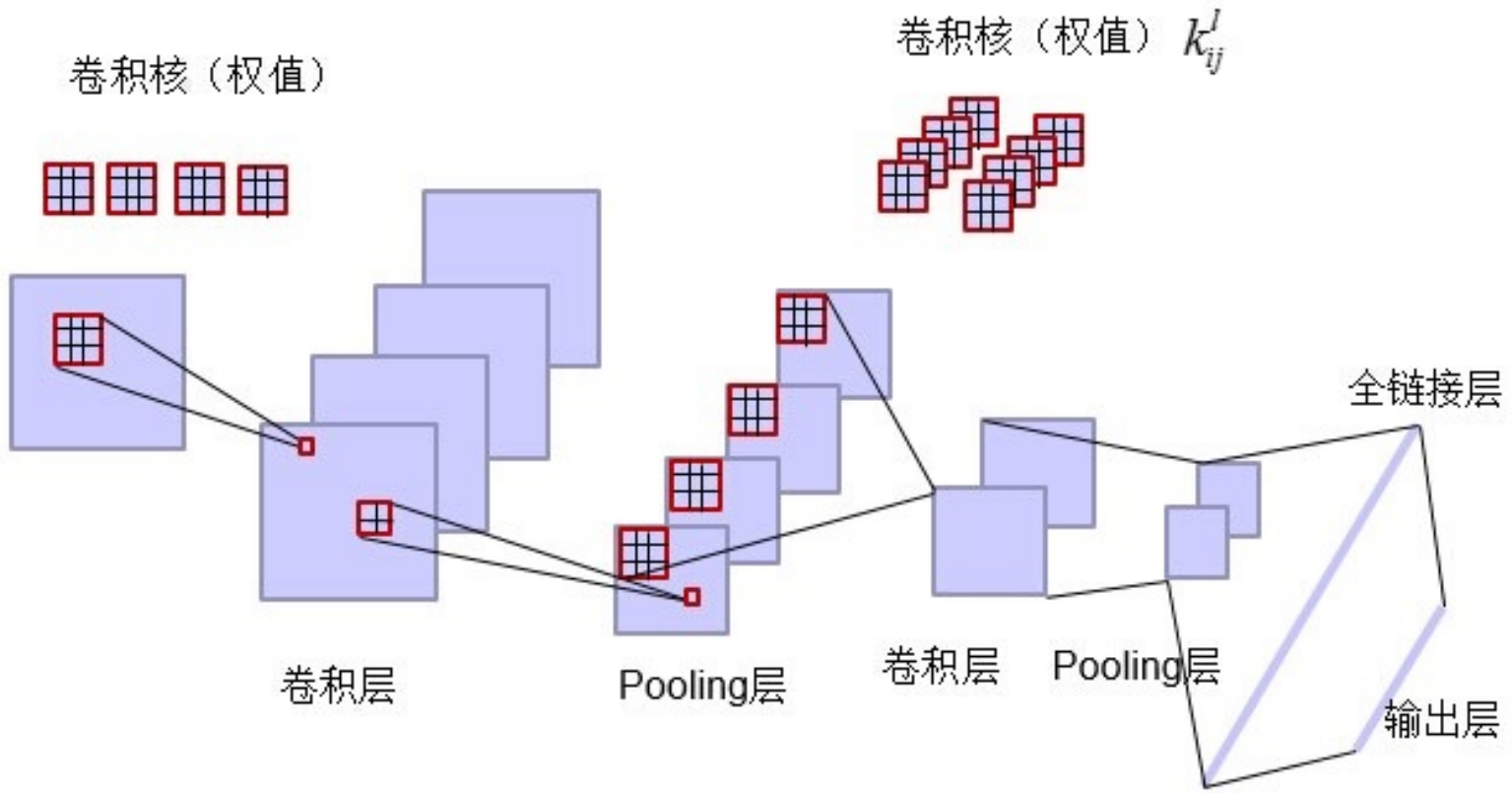
$$\delta_j^l = \beta_j^{l+1}\left(f'(u_j^l) \circ up(\delta_j^{l+1})\right)$$

采样层后向

$$\delta_j^l = f'(u_j^l) \circ conv2\left(\delta_j^{l+1}, rot180(k_j^{l+1}), 'full'\right)$$

卷积层后向

$$\frac{\partial E}{\partial K_{ij}^l} = rot180\left(conv2(x_i^{l-1}, rot180(\delta_j^l), 'valid')\right)$$


卷积核（权值）
卷积核（权值） k_{ij}^{l}
全链接层
卷积层
Pooling层
卷积层
Pooling层
输出层

CNN的优点

- 避免了显式特征抽取，隐式地从训练数据中进行学习；
- 同一特征映射面上的神经元权值相同，从而网络可以并行学习，降低了网络的复杂性；
- 采用时间或者空间的子采样结构，可以获得某种程度的位移、尺度、形变鲁棒性；
- 输入信息和网络拓扑结构能很好的吻合，在语音识别和图像处理方面有着独特优势。

在图像识别上的应用

< Caltech 256 >

# of training images	30	60
Griffin et al. [2]	34.10	-
vanGemert et al., PAMI 2010	27.17	-
ScSPM [Yang et al., CVPR 2009]	34.02	40.14
LLC [Wang et al., CVPR 2010]	41.19	47.68
Sparse CRBM [Sohn et al., ICCV 2011]	42.05	47.94

实验在Caltech 256数据集上，利用单特征识别，Sparse CRBM性能最优。

在音频识别上的应用

- Speaker identification

TIMIT Speaker identification	Accuracy
Prior art (Reynolds, 1995)	99.7%
Convolutional DBN	100.0%

- Phone classification

TIMIT Phone classification	Accuracy
Clarkson et al. (1999)	77.6%
Petrov et al. (2007)	78.6%
Sha & Saul (2006)	78.9%
Yu et al. (2009)	79.2%
Convolutional DBN	80.3%
Transformation-invariant RBM (Sohn et al., ICML 2012)	81.5%

在视频识别上的应用

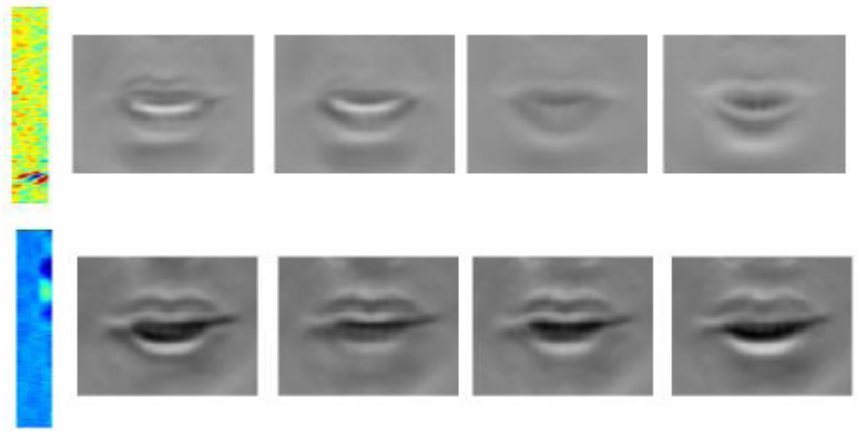
Video Activity recognition (Hollywood 2 benchmark)

Method	Accuracy
Hessian + ESURF [Williems et al 2008]	38%
Harris3D + HOG/HOF [Laptev et al 2003, 2004]	45%
Cuboids + HOG/HOF [Dollar et al 2005, Laptev 2004]	46%
Hessian + HOG/HOF [Laptev 2004, Williems et al 2008]	46%
Dense + HOG / HOF [Laptev 2004]	47%
Cuboids + HOG3D [Klaser 2008, Dollar et al 2005]	46%
Unsupervised feature learning (our method)	**52%** ←

Unsupervised feature learning significantly improves on the previous state-of-the-art.

在多模态学习中的应用

- Visualization of learned filters



Audio(spectrogram) and Video features learned over 100ms windows

- Results: AVLetters Lip reading dataset

Method	Accuracy
Prior art (Zhao et al., 2009)	58.9%
Multimodal deep autoencoder (Ngiam et al., 2011)	**65.8%**

总结

- Deep Learning对图像、语音这种特征不明显（需要手工设计且很多没有直观物理含义）的问题，能够在大规模训练数据上取得更好的效果
- 目前关注点还是从机器学习的领域借鉴一些可以在Deep Learning使用的方法，特别是降维领域，通过压缩感知理论对高维数据进行降维，使得非常少的元素的向量就可以精确的代表原来的高维信号
- 很多方法缺少严谨的理论，例如一些算法是否收敛，收敛速度有多快等
- 有效的可并行训练算法
- 如何合理充分利用Deep Learning来增强传统学习算法的性能
- .....

参考文献: *SVM*

- Cortes C, Vapnik V. Support-vector networks[J]. Machine learning, 1995, 20(3): 273-297.
- Wu Y, Liu Y. Robust truncated hinge loss support vector machines[J]. Journal of the American Statistical Association, 2007, 102(479).
- Liu, Yufeng and Wu, Yichao. Flexible Large Margin Classifiers. High-dimensional Data Analysis (T. T. Cai and X. Shen, eds),2010, 39-71, World Scientific, New Jersey.
- Liu Y, Zhang H H, Wu Y. Hard or soft classification? large-margin unified machines[J]. Journal of the American Statistical Association, 2011, 106(493): 166-177.
- Liu Y, Yuan M. Reinforced multicategory support vector machines[J]. Journal of Computational and Graphical Statistics, 2011, 20(4): 901-919.
- Wu Y, Liu Y. Adaptively weighted large margin classifiers[J]. Journal of Computational and Graphical Statistics, 2013, 22(2): 416-432.
- Zhang C, Liu Y. Multicategory angle-based large-margin classification[J]. Biometrika, 2014, 101(3): 625-640.

参考文献: *DL*

- Hinton G, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural computation, 2006, 18(7): 1527-1554.
- Bengio Y, Lamblin P, Popovici D, et al. Greedy layer-wise training of deep networks[J]. Advances in neural information processing systems, 2007, 19: 153.
- Poultney C, Chopra S, Cun Y L. Efficient learning of sparse representations with an energy-based model[C]//Advances in neural information processing systems. 2006: 1137-1144.
- Bengio Y, Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, 2009.
- Vincent P, Larochelle H, Bengio Y, and Manzagol, P. Extracting and composing robust features with denoising autoencoders. ICML, 2008.
- Lee H, Ekanadham C, and Ng A.Y, Sparse deep belief net model for visual area V2. NIPS, 2008.
- LeCun Y, Boser B, Denker J.S, Henderson D, Howard R.E, Hubbard W, and Jackel, L.D Backpropagation applied to handwritten zip code recognition. Neural Computation, 1989,1:541 – 551.

谢谢！