1前论

1.1优点

1. 自学习和自适应
2. 非线性 现实世界是一个非线性的复杂系统，人脑也是，
3. 鲁棒性 局部损坏只会削弱神经网络，而不会产生灾难性错误
4. 计算的并行性 每个神经元独立处理信息
5. 存储的分布性 知识不是存储于某一处，而是分布在所有连接权值中

1.2历史

- 1949
  - Hebb
  - 《The Organization of Behavior》
  - （人脑）学习假说：两个神经元之间的重复激活，将使其连接权值得到加强
- 1952
  - Ashby
- 1957
  - Rosenblatt
  - Perception，感知机收敛定理
  - Widrow-Hoff法则：最小均方误差原理（LMS）
- 1969
  - Minsky&Papert
  - 《Perception》
  - （数学）感知机处理能力有限，无法解决异或；多层感知机无法克服这种局限性。神经网络进入萧条
- 1972
  - Teuvo&James
    - 用于记忆的神经网络

    - Amari
        - 神经元附加模型
    - Wilson&Cowan
    - （数学）兴奋和抑制模型神经元空间局部化的群体动力学耦合非线性微分方程
- 1977
    - Anderson Silverstein
    - 盒中脑（BSB）模型
- 1980
    - Grosserg
    - ART理论：包括自下而上的认知层和自上而下的生成层
- 1982
    - Hopfield
    - 引入含有对称突触连接的反馈网络，为大量的物理学家进入神经网络铺平道路，第一次清楚阐述了如何在动态的稳定网络中存储信息
    - 1983
        - 模拟退火算法
- 1988
- Linsker RBF 径向基网络
- 1990s Vapnik SVM 以一种自然方式包含了VC维数

2神经元的结构

2.1Linear neurons

$y = b + \sum_i x_i w_i$

2.2Binary Threshold neuros

$$z = \sum_i x_i w_i$$

$$y = \begin{cases} 1 & if z >= 0 \\ 0 & otherwize \end{cases}$$

2.3Rctified Linear Neuros

$$z = b + \sum_i x_i w_i$$

$$y = \begin{cases} z & if z > 0 \\ 0 & otherwise \end{cases}$$

2.4Sigmoid neurons

$$z = b + \sum_i x_i w_i$$

$$y = \frac{1}{1+e^{-z}}$$

- Sigmoid函数还有一种是tanh(z)

2.5 Stochastic binary neurons

$$z = b + \sum_i x_i w_i$$

$$p(s=1) = \frac{1}{1+e^{-z}}$$

- 这种神经元依概率输出0或1

3 网络结构

3.1 Feed-forward neural networks

3.2 Recurrent networks

- 同一个layer之间互相关联
- They can have very complicated dynamics, and this can make them very difficult to train.

3.3 Symmetrically connected networks

- Hopfield：无hidden units
- Boltzmann machines:有hidden units

4 学习的过程

- 字母意义
  - n 表示training set中的第n个
  - i 表示某个神经元的weight vector 中第i个
  - j 表示第j个神经元

4.1 delta rule：linear的情况

- 平方形式的残差
  $E = 0.5 * \sum_{n \in training} (t_n - y_n)^2$
- 微分
  $\frac{\partial E}{\partial w_i} = 0.5 * \sum_n \frac{\partial y_n}{\partial w_i} \frac{dE_n}{dy_n} = -\sum_n x_{in}(t_n - y_n)$
- 改变weight
  $\Delta w_i = -\varepsilon \frac{\partial E}{\partial w_i}$

4.2 delta rule: sigmoid的情况

1. sigmoid的神经元是这样的
   $z = b + \sum_i x_i w_i$
   $y = \frac{1}{1+e^{-z}}$
2. sigmoid神经元的特点
   $\frac{dy}{dz} = y(1-y)$
3. sigmoid神经元的学习过程
   $\frac{\partial E}{\partial w_i} = -\sum_n x_{in} y_n (1-y_n)(t_n - y_n)$

4.3 误差向前传播

4.3.1 变量定义

I表示中间层，J表示输出层
n表示第n个case

v表示输出，u表示输入，故而v=g(u)，其中 g()是神经元内的函数

学习过程$w_{ij}(n+1)=\Delta w_{ij}(n)+w_{ij}(n)$

4.3.2 对于第n个case,J层情况

$$\frac{\partial e(n)}{\partial w_{ij}}=\frac{\partial e}{\partial e_j}\frac{\partial e_j}{\partial v_j}\frac{\partial v_j}{\partial u_j}\frac{\partial u_j}{\partial w_{ij}}$$

- 关于第一项
  $e=0.5\sum e_j^2$
  于是$\frac{\partial e}{\partial e_j}=e_j=t_j-v_j^J$
- 关于第二项
  $e_j=t_j-v_j^J$
  于是$\frac{\partial e_j}{\partial v_j}=-1$
- 关于第三项
  $v_j^J=g(u_j^J)$
  于是$\frac{\partial v_j}{\partial u_j}=g^1(u_j^J)$
- 关于第四项
  $u_j^J=\sum_{i\in I}w_{ij}v_i^I$
  于是$\frac{\partial u_j}{\partial w_{ij}}=v_i^I$

4.3.3 对于第n个case，J层学习

$$\Delta w_{ij}(n)=-\eta\frac{\partial e(n)}{\partial w_{ij}}$$

- 用链式法则拆分

$$\Delta w_{ij}(n) = -\eta \frac{\partial e}{\partial e_j} \frac{\partial e_j}{\partial v_j} \frac{\partial v_j}{\partial u_j} v_i^I$$

- 定义梯度为：

$$\delta_j^J = \frac{\partial e}{\partial e_j} \frac{\partial e_j}{\partial v_j} \frac{\partial v_j}{\partial u_j}$$

于是:
$$\Delta w_{ij}(n) = -\eta \delta_j^J v_i^I$$

2.3.4 对于第n个case，l层学习

(l层之前为M层)
$$\Delta w_{mi}^I = -\eta \frac{\partial e}{\partial w_{mi}} v_m^M$$

- 用链式法则拆分

$$\frac{\partial e}{\partial w_{mi}} = \frac{\partial e}{\partial v_i^I} \frac{\partial v_i^I}{\partial u_i^I} \frac{\partial u_i^I}{\partial w_{mi}}$$

- 用梯度形式改写

$$\Delta w_{mi}^I = -\eta \delta_i^I v_m^M$$

- 其中梯度为：

$$\delta_i^I = \frac{\partial e}{\partial v_i} \frac{\partial v_i}{\partial u_i}$$

经计算，其中，

$$\frac{\partial e}{\partial v_i^I} = \sum_{j \in J} \delta_j^J w_{ij}$$

4.4 关于Linear

纯linear 也服从误差向前传播，并且注意多层linear网络与单层linear网络没有区别（证明）

4.5 关于softmax output function

unit 的结构

$$y_i = \frac{e^{z_i}}{\sum_{j \in group} e^{z_j}}$$

于是

$$\frac{\partial y_i}{\partial z_i} = y_i(1 - y_i)$$

对应的cost function

这时，cost function 不应当是误差平方和了，而是这样

$$C = -\sum_j t_j log y_j$$

此时$\frac{\partial C}{\partial z_i} = y_i - t_i$

以二值为例：
$y = \frac{1}{1+e^{-z}}$
如果cost function是误差平方和$E = 0.5(y-t)^2$
那么$\frac{dE}{dz} = (y-t)y(1-y)$
这时，如果y接近0或接近1，那么学习速度将会非常小
合适的cost function是这样
$E = -tlog(y) - (1-t)log(1-y)$,因为这时$\frac{dE}{dz} = y - t$

模型总结：
1、二值时，模型就是logistics回归
2、多值时，似乎是最大熵模型

5 训练方式

5.1 更新频率

How often to update the weights

- Online ： after each training case
- Full batch:after a full sweep
- Mini-batch:after a small sample of training data

5.2 学习率

- fixed learning rate

- Adapt the global learning rate
- Adapt the learning rate on each connection
- Don't use steepest descent

overfitting

- 原因
  - data不好
  - target values unreliable
    - sampling error （eg: rectangle）
    - regularities:不知道是真实regularities还是sampling error
- Ways to reduce overfitting
  - Weight-decay:keep weights small
  - Weight-sharing: insisting that many of the weights have the same value
  - early stopping
  - model averaging:train lots of different nenual nets. And average them
  - Bayesian fitting of nrnual nets: A fancy form of model averaging
  - Dropout：emitting hidden layer
  - Generative pre-trainning: （后面会讲到）
    - lec 7 继续这个话题

关于认知

The feature theory

- A concept is a set of semantic features.
- 因此knowledge以a vector of feature来表示

The structuralist theory

- The meaning of concept lies in its relationships to other concept
- 因此knowledge以relational graph 来表示

案例1：预测下一个word
见附件

案例**2**：图像中的物体识别（**LEC5**）

困难

- segmentation
  - to tell which pieces is parts of the same object
  - parts of an object can be hidden behind other object
- Lighting
- Deformation
  - 各个部位变形，仍然是这个东西
- 不同的造型，相同的功能，仍然叫一个名字

    - 例如，椅子
- Viewpoint
    - dimension hop

解决方案:针对**viewpoint**

- redundant invariant features
- put a box around the object and use normalized pixels.
- replicated features with pooling. (convolutional neural nets)CNN 卷积神经网络
- use a hierarchy of parts that have explicit poses relative to the camera

针对**dimension-hopping**

解决方案： The judicious normalization approach
具体方法： put a box around the object and use it as a coordinate frame
优点：解决 translation, totation, scale, shear, stretch
然而，找到合适的box比较困难：

- segmentation errors
- occlusion（？遮光）
- unusual orientation(?为什么呢)
- 事实上，人脑是先识别再旋转的

首先要recognize the shape to get the box right.

- 解决box的方案： the brute force normalization
  具体方法： train阶段，用竖直的、切分良好的样本。test阶段遍历box所有的可能性

CNN模型

to constrain $w_1 = w_2$
we need $\Delta w_1 = \Delta w_2$
use $\frac{\partial E}{\partial w_1} + \frac{\partial E}{\partial w_2}$

传统神经网络采用back propagation
但是，

- 层数太深，残差传播到最前面的层已经变得太小，，误差校正信号越来越小，出现所谓gradient diffusion。
- 局部最小值（随机初始化导致的）
- 只能做有监督学习
  deep learning 用layer-wise pre-training
  deep learning的训练包括两步：训练和调优

1. 训练： 逐层训练单层神经元，这样每次按单层神经元训练
2. 调优： 所有层训练完后，用wake-sleep 算法调优
   1. 向上的权重用于“认知”，向下的权重用于“+生成”，wake-sleep 算法用来使认知和生成达成一致
   2. wake阶段： 认知过程，通过外界的特征和向上的权重（认知权重）产生每一层的抽象表示（结点状态），并且使用梯度下降修改层间的下行权重（生成权重）。也就是“如果现实跟我想象的不一样，改变我的权重使得我想象的东西就是这样的”。
      3. sleep阶段：生成过程，通过顶层表示（醒时学得的概念）和向下权重，生成底层的状态，同时修改层间向上的权重。也就是“如果梦中的景象不是我脑中的相应概念，改变我的认知权重使得这种景象在我看来就是这个概念”。

deep learning训练过程具体如下：

1）使用自下上升非监督学习（就是从底层开始，一层一层的往顶层训练）：

采用无标定数据（有标定数据也可）分层训练各层参数，这一步可以看作是一个无监督训练过程，是和传统神经网络区别最大的部分（这个过程可以看作是feature learning过程）：

具体的，先用无标定数据训练第一层，训练时先学习第一层的参数（这一层可以看作是得到一个使得输出和输入差别最小的三层神经网络的隐层），由于模型capacity的限制以及稀疏性约束，使得得到的模型能够学习到数据本身的结构，从而得到比输入更具有表示能力的特征；在学习得到第n-1层后，将n-1层的输出作为第n层的输入，训练第n层，由此分别得到各层的参数；

2）自顶向下的监督学习（就是通过带标签的数据去训练，误差自顶向下传输，对网络进行微调）：

基于第一步得到的各层参数进一步fine-tune整个多层模型的参数，这一步是一个有监督训练过程；第一步类似神经网络的随机初始化初值过程，由于DL的第一步不是随机初始化，而是通过学习输入数据的结构得到的，因而这个初值更接近全局最优，从而能够取得更好的效果；所以deep learning效果好很大程度上归功于第一步的feature learning过程。

- 按结构分类
- 前向网络
  - 单层前向
    - 单层感知机
    - 线性网络
  - 多层前向
    - 多层感知机
    - 径向基
- 反馈网络

- Hopfield网络
- Elman网络
- 按学习分类
  - 有监督学习
  - BP，径向基，Hopfield
  - 无监督学习
  - 自组织神经网络

- 竞争神经网络

此外，还有：
Hebb学习规则
纠错学习规则（用的多）
随机学习规则（Boltzmann机）
竞争学习规则

感知机模型