PRML-Chap5

Neural Networks

周方铭

2019.12.25

目录

- 1. 前馈神经网络

①函数形式

- 2. 网络训练
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化

②最大似然确定参数

非线性最优化

求导

- 3. 误差反向传播
  - 误差函数导数（梯度）的计算/例子/反向传播的效率

③BP

高效!

推广

- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

④

⑤正则化

⑥混合密度网络

⑦贝叶斯

目录

- 1. 前馈神经网络
- 2. 网络训练
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播
  - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

线性模型

$\Downarrow$ 推广

神经网络

- 函数结构
- 网络扩展
- 权空间
- 近似能力

线性模型

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right)$$

- $\phi_j(x)$ 固定基函数
- ω 模型系数
- $f(*)$ 激活函数
  - 回归：恒等函数
  - 分类：非线性激活函数

推广线性模型 -> 前馈神经网络

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right)$$

- $\phi_j(x)$ 基函数也能调节
- 嵌套使用上面这个公式，先用它计算基函数，再用它计算输出

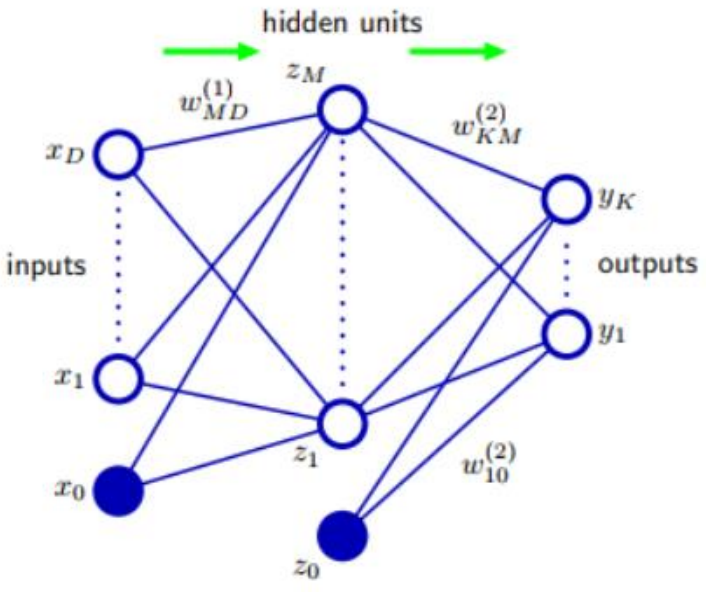


Figure: 5.1

前馈神经网络

第一层

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i$$

- a_j 加权和
- ω_{ji} ·第一层权重
- ω_{j^0} 第一层偏置项b

$$z_j = h(a_j)$$

- $h(*)$ 非线性激活函数

第二层

$$a_k = \sum_{j=0}^{M} w_{kj}^{(2)} z_j$$

- a_k 加权和
- ω_{kj} 第一层权重
- ω_{k0} 第一层偏置项b

$$y_k = \sigma(a_k)$$

- y_k 最终输出
- $\sigma(*)$ 输出激活函数

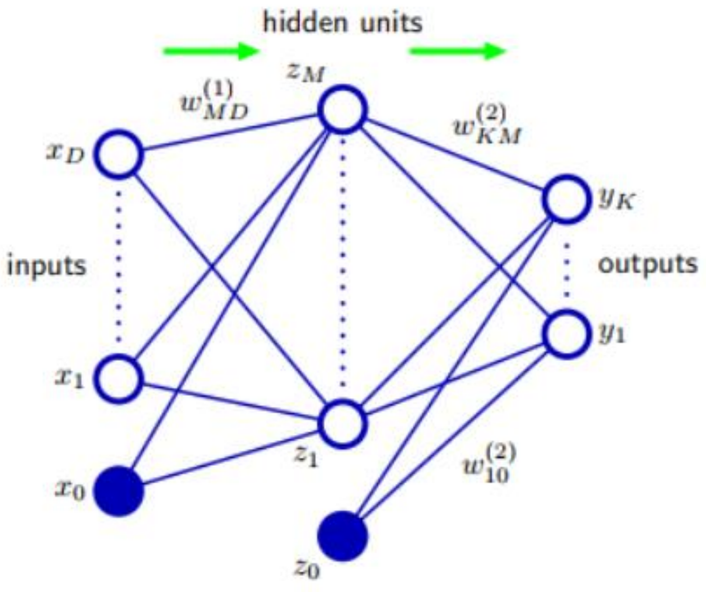


Figure: 5.1

前馈神经网络

完整的两层神经网络模型

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\underbrace{\sum_{j=0}^{M} w_{kj}^{(2)}\, h\left(\underbrace{\sum_{i=0}^{D} w_{ji}^{(1)} x_i}_{a_j}\right)}_{a_k}\right)$$

Forward Propagation

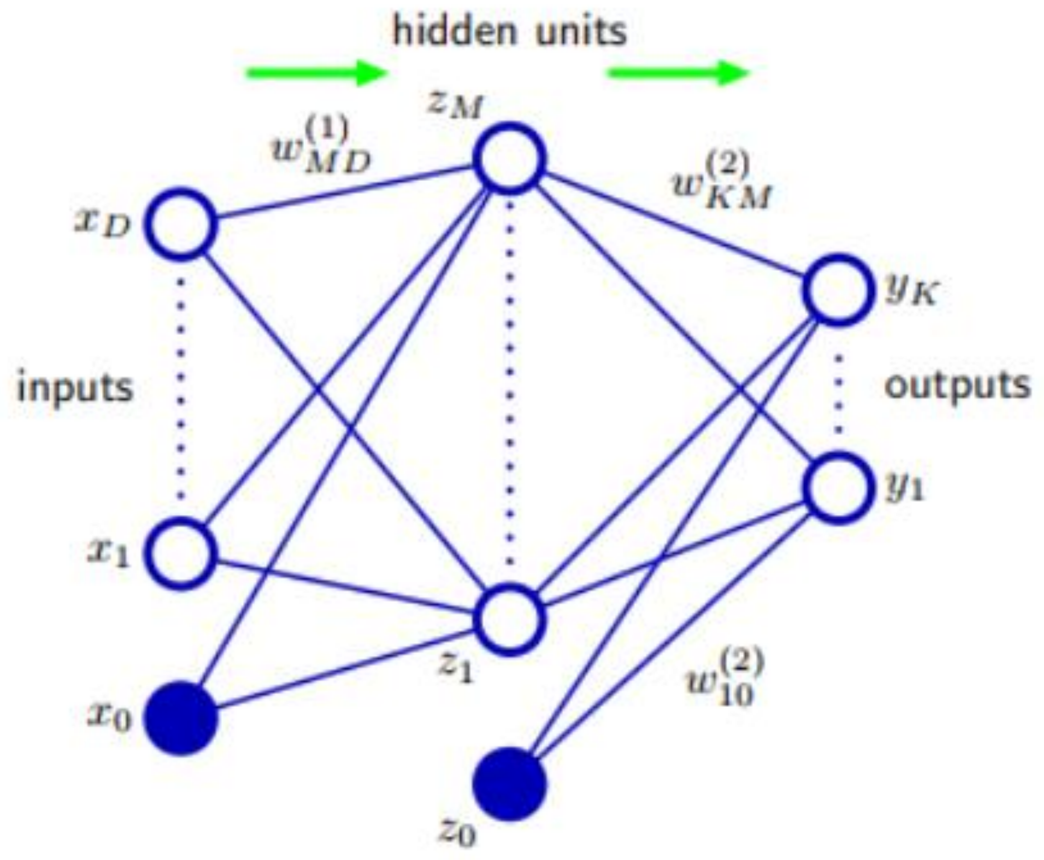


Figure: 5.1

前馈神经网络

模型的拓展

- 增加隐层层层数 替换激活函数
- 跨层$skip-\text{layer}$连接
- 稀疏网络

权空间的对称性

- M 个隐含层单元
- $M!$ 隐含层单元排列组合
- 2^M 变换符号（$h(*)$为奇函数)
- 任意一个权向量都是$M!\,2^M$个对称权向量中的一个

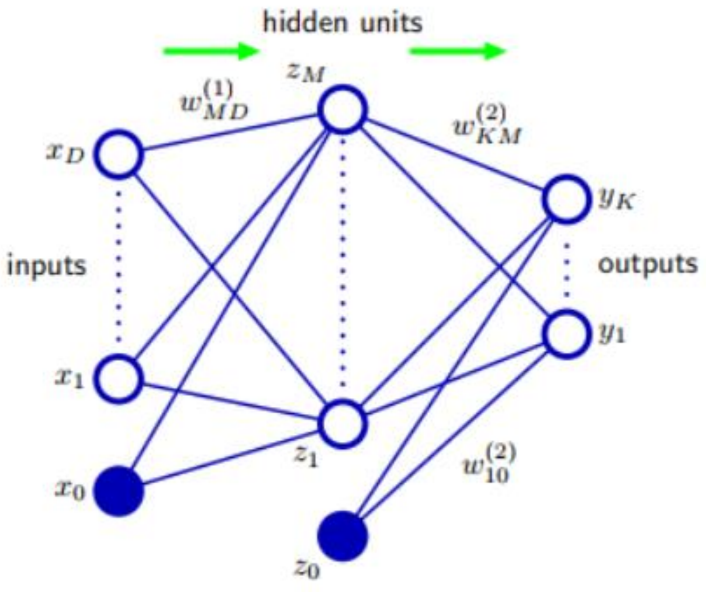
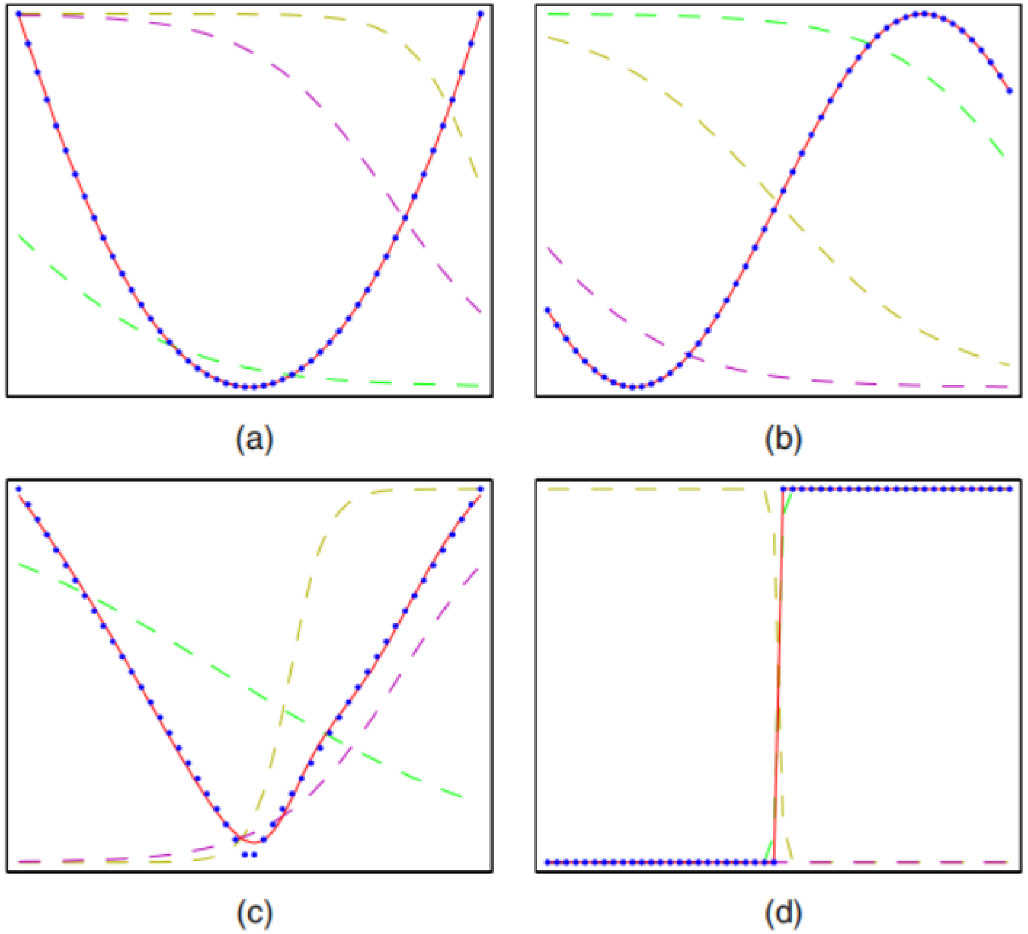


Figure: 5.1

前馈神经网络

Universal Approximators

- (a) $f(x) = x^2$
- (b) $f(x) = \sin(x)$
- (c) $f(x) = |x|$
- (d) $f(x) = H(x)$



目录

- 1. 前馈神经网络
- 2. 网络训练
    - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播
    - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
    - Jacobian矩阵
    - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

线性模型

$\Downarrow$ 推广

神经网络

- 函数结构
- 网络扩展
- 权空间
- 近似能力

MLE → $E(w)$, $\sigma(x)$ (分类/回归)

最优化(最小化) → 局部二次近似

→ 梯度下降

目录

- 1. 前馈神经网络
- 2. 网络训练
  - 极大似然方法（回归/分类问题）
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播
  - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

模型训练 – 极大似然方法

最小化平方和误差函数

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2.$$

模型训练 – 极大似然方法

回归问题 - 一元目标变量 t

假定目标变量服从高斯分布

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$

极大似然函数

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \mathbf{w}, \beta).$$

取负对数，求其极小

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi)$$

确定参数w， min E(w)

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2$$

确定参数β

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}_{\mathrm{ML}}) - t_n\}^2.$$

模型训练 – 极大似然方法

回归问题 – 多元目标变量 **t**

假定目标变量服从高斯分布

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(\mathbf{t}|\mathbf{y}(\mathbf{x}, \mathbf{w}), \beta^{-1}\mathbf{I}\right).$$

确定参数w， min E(w)

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} \|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2.$$

确定参数β

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{NK}\sum_{n=1}^{N} \|\mathbf{y}(\mathbf{x}_n, \mathbf{w}_{\mathrm{ML}}) - \mathbf{t}_n\|^2$$

模型训练 – 极大似然方法

分类问题 - 二分类

- 单一目标变量 t
  - $t = 1$ 类别C1
  - $t = 0$ 类别C2

- 激活函数Sigmoid

$$y = \sigma(a) \equiv \frac{1}{1 + \exp(-a)}$$

- $y(\vec{x}, \vec{\omega}) \qquad = p(C_1|\vec{x})$
- $1 - y(\vec{x}, \vec{\omega}) = p(C_2|\vec{x})$

目标变量t的条件概率分布

$$p(t|\mathbf{x}, \mathbf{w}) = y(\mathbf{x}, \mathbf{w})^t \{1 - y(\mathbf{x}, \mathbf{w})\}^{1-t}$$

交叉熵误差函数

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

模型训练 – 极大似然方法

分类问题 - K个独立的二分类

- K个输出， $t_k \in \{0,1\}, k = 1, \dots, K$

- 激活函数Sigmoid

$$y = \sigma(a) \equiv \frac{1}{1 + \exp(-a)}$$

目标变量t的条件概率分布

$$p(t|\mathbf{x}, \mathbf{w}) = y(\mathbf{x}, \mathbf{w})^t \{1 - y(\mathbf{x}, \mathbf{w})\}^{1-t}$$

目标向量$\vec{t}$的条件概率分布

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \prod_{k=1}^{K} y_k(\mathbf{x}, \mathbf{w})^{t_k} [1 - y_k(\mathbf{x}, \mathbf{w})]^{1-t_k}.$$

交叉熵误差函数

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

交叉熵误差函数

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \sum_{k=1}^{K} \{t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln(1 - y_{nk})\}$$

模型训练 – 极大似然方法

分类问题 – 多分类

- $t_k \in \{0,1\}$
- 1-of-K
- $y_k(\vec{x},\vec{\omega}) = p(t_k = 1, \vec{x})$

- 激活函数softmax

$$y_k(\mathbf{x},\mathbf{w}) = \frac{\exp(a_k(\mathbf{x},\mathbf{w}))}{\sum_j \exp(a_j(\mathbf{x},\mathbf{w}))}$$

目标变量t的条件概率分布

$$p(t|\mathbf{x},\mathbf{w}) = y(\mathbf{x},\mathbf{w})^t \{1 - y(\mathbf{x},\mathbf{w})\}^{1-t}$$

目标向量$\vec{t}$的条件概率分布

$$p(\mathbf{t}|\mathbf{x},\mathbf{w}) = \prod_{k=1}^{K} y_k(\mathbf{x},\mathbf{w})^{t_k} [1 - y_k(\mathbf{x},\mathbf{w})]^{1-t_k}$$

交叉熵误差函数

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1-t_n)\ln(1-y_n)\}$$

交叉熵误差函数

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} \{t_{nk} \ln y_{nk} + (1-t_{nk})\ln(1-y_{nk})\}$$

交叉熵误差函数

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{kn} \ln y_k(\mathbf{x}_n,\mathbf{w})$$

模型训练 – 极大似然方法

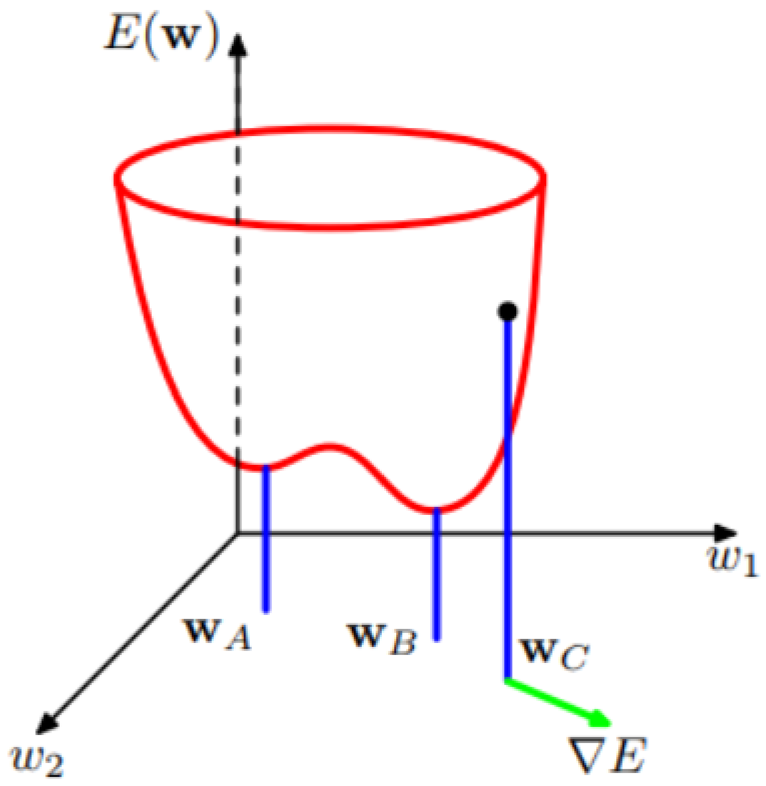
总结

- 问题（分类/回归）类型 —— 输出单元激活函数 误差函数
  - 回归： 线性输出激活函数 平方和误差函数
  - （多个独立的）二分类：sigmoid 交叉熵误差函数
  - 多分类：softmax激活函数 交叉熵误差函数

模型训练 – 参数最优化

误差函数最小化

- 误差函数的最小值点出现在梯度为0的点
  - $\nabla E = 0$
- 误差函数 E 与权重w是高度非线性的
- 存在许多极大值、极小值、鞍点（$\nabla E = 0$）
- 每个极小值点是$M!\,2^M$个等价点中的一个
  - 局部极小值
  - 全局最小值
- 无法找到$\nabla E = 0$的解析解
  - 迭代法 $\omega^{(\tau+1)} = \omega^{(\tau)} + \Delta w^{(\tau)}$
  - $\Delta w^{(\tau)}$的选择取决于算法
  - 使用梯度



误差函数曲面

模型训练 – 局部二次近似

在$\widehat{\omega}$点泰勒展开

误差函数近似:

$$E(\mathbf{w}) \simeq E(\widehat{\mathbf{w}}) + (\mathbf{w} - \widehat{\mathbf{w}})^{\mathrm{T}}\mathbf{b} + \frac{1}{2}(\mathbf{w} - \widehat{\mathbf{w}})^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \widehat{\mathbf{w}})$$

$$\mathbf{b} \equiv \nabla E|_{\mathbf{w}=\widehat{\mathbf{w}}}$$

$$(\mathbf{H})_{ij} \equiv \left.\frac{\partial E}{\partial w_i \partial w_j}\right|_{\mathbf{w}=\widehat{\mathbf{w}}}$$

梯度近似:

$$\nabla E \simeq \mathbf{b} + \mathbf{H}(\mathbf{w} - \widehat{\mathbf{w}})$$

模型训练 – 局部二次近似

设ω^*点为局部最小值

误差函数近似：

$$E(\mathbf{w}) = E(\mathbf{w}^\star) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^\star)^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}^\star)$$

先讲**结论**：

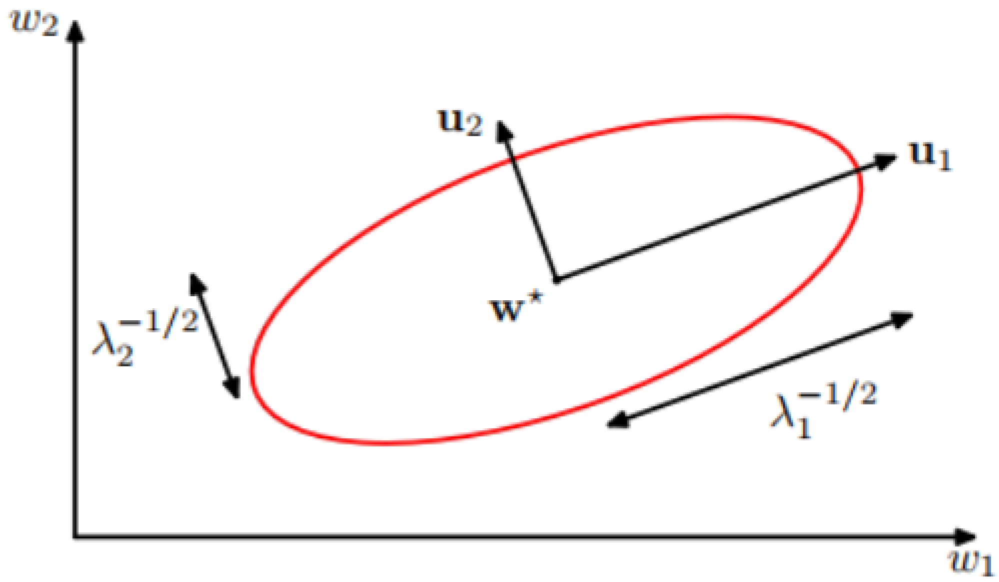
- 驻点$\widehat{\omega^*}$（$\nabla E = 0$）是局部最小值的条件为：
  - ω^*处海森矩阵是正定矩阵
  - $\left.\frac{\partial^2 E}{\partial w^2}\right|_{w^*} > 0$ （权空间为一维）

模型训练 – 局部二次近似

设ω^*点为局部最小值

误差函数近似:

$$E(\mathbf{w}) = E(\mathbf{w}^\star) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^\star)^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}^\star)$$



海森矩阵的特征向量u_i是相互正交的，且构成了完备集

$$\mathbf{H}\mathbf{u}_i = \lambda_i \mathbf{u}_i \qquad \mathbf{w} - \mathbf{w}^\star = \sum_i \alpha_i \mathbf{u}_i.$$

代入误差函数近似中，得

$$E(\mathbf{w}) = E(\mathbf{w}^\star) + \frac{1}{2}\sum_i \lambda_i \alpha_i^2$$

- 所有特征值$\lambda_i > 0 \Leftrightarrow$ H 正定

结论:

- 驻点$\widehat{\omega^*}$（$\nabla E = 0$）是局部最小值的条件为:
  - ω^*处海森矩阵是正定矩阵
  - $\left.\frac{\partial^2 E}{\partial w^2}\right|_{w^*} > 0$ （权空间为一维）

模型训练 – 梯度下降最优化

二次近似寻找局部最小值所需的计算复杂度： $O(w^3)$

- 误差曲面由b和H所决定，共$\frac{w(w+1)}{2}+w$个参数： $O(w^2)$
- 每次求值需要 $O(w)$个步骤

误差反向传播寻找局部最小值所需的计算复杂度： $O(w^2)$

- 计算$O(w)$次梯度
- 每次计算$O(w)$步

模型训练 – 梯度下降最优化

梯度下降法 (处理整个数据集)

优缺点:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E\left(\mathbf{w}^{(\tau)}\right)$$

- $\eta > 0$ 学习率

随机梯度下降法 (在线处理)

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n(\mathbf{w}^{(\tau)}).$$

- $\eta > 0$ 学习率

共轭梯度法、拟牛顿法…

MLE → E(w), σ(x) (分类/回归)

最优化(最小化) → 局部二次近似

→ 梯度下降

目录

- 1. 前馈神经网络
- 2. 网络训练
  - 极大似然方法（回归/分类问题）
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播
  - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

目录

- 1. 前馈神经网络
- 2. 网络训练
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播
  - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

误差反向传播

误差函数 E 最小化的迭代过程：

- **1. 计算E关于w的导数（梯度）**
  - BP算法提供了一个高效的方法
- 2. 导数用于计算权值的调整
  - 例如梯度下降法

误差反向传播

- 适用于 多种网络 多种误差函数
- 适用于 计算其他类型的导数，例如 Jacobian矩阵、Hessian矩阵

误差反向传播

误差函数是所有数据样本误差的总和，考虑$\nabla E_n(\omega)$

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}).$$

一个简单BP —— 线性模型

$$y_k = \sum_i w_{ki} x_i$$

误差函数

$$E_n = \frac{1}{2}\sum_k (y_{nk} - t_{nk})^2, \quad \text{where} \quad y_{nk} = y_k(\mathbf{x}_n, \mathbf{w}).$$

梯度

$$\frac{\partial E_n}{\partial \omega_{k_i}} = (y_{n_k} - t_{nk}) x_{n_i}$$

误差反向传播

一个一般的前馈网络

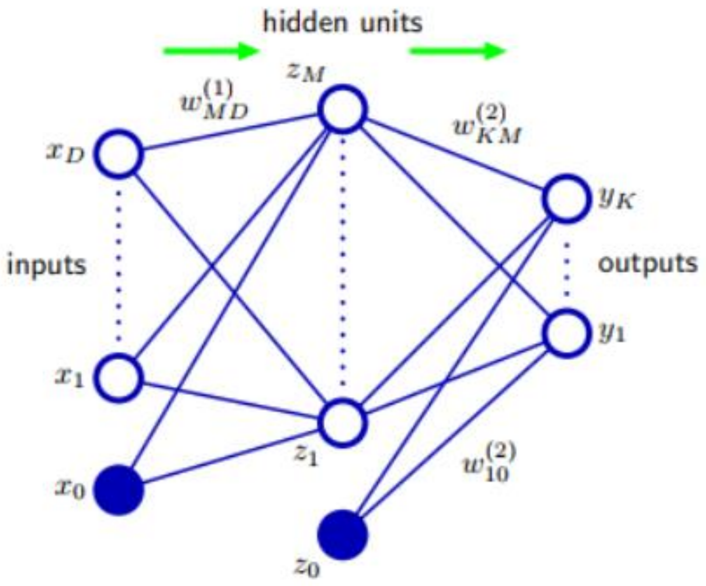
$$a_j = \sum_i w_{ji} z_i \qquad \frac{\partial a_j}{\partial w_{ji}} = z_i$$

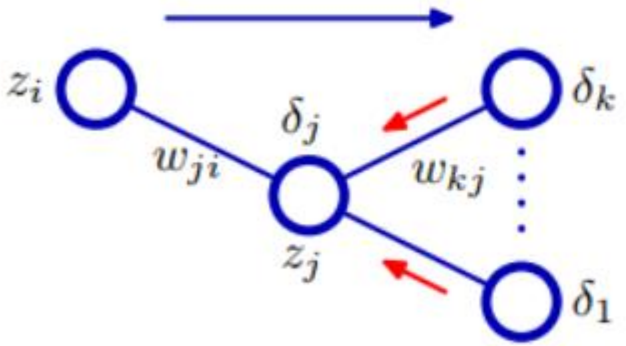
$$z_j = h(a_j)$$

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}}$$

引入 $\delta_j \equiv \frac{\partial E_n}{\partial a_j}$ ，δ称为误差

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i.$$





误差反向传播

计算误差 δ

$a_k = \sum w_{kj} z_j$

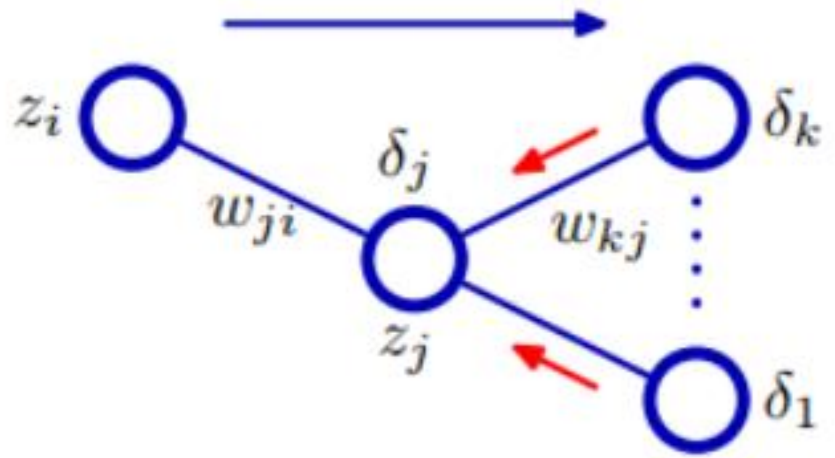
$$\delta_k = y_k - t_k$$

$$a_j = \sum_i w_{ji} z_i$$

$$z_j = h(a_j)$$

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j}$$

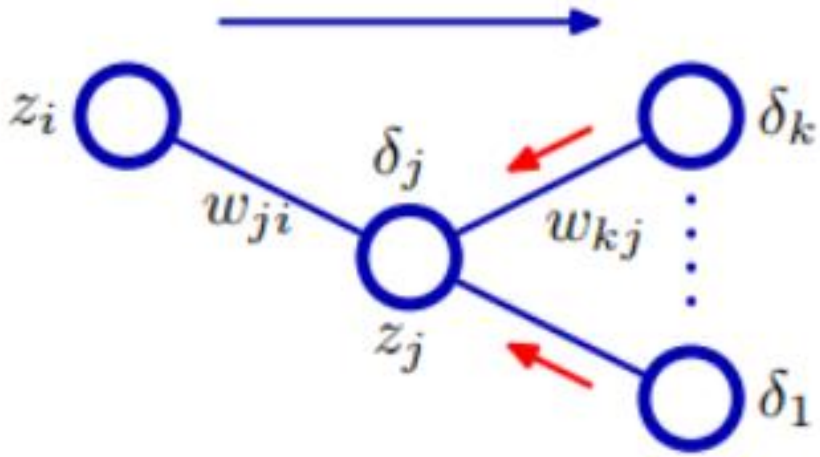
$$\delta_j = h'(a_j) \sum_k w_{kj} \delta_k$$



误差反向传播

误差反向传播的计算步骤

1. 正向传播，计算出所有a和z
2. 计算所有输出单元的 δ_k
3. 反向传播 δ
4. 计算导数 $\dfrac{\partial E_n}{\partial \omega_{ji}} = \delta_j z_i$
5. （批处理） $\dfrac{\partial E}{\partial w_{ji}} = \sum_n \dfrac{\partial E_n}{\partial w_{ji}}$.

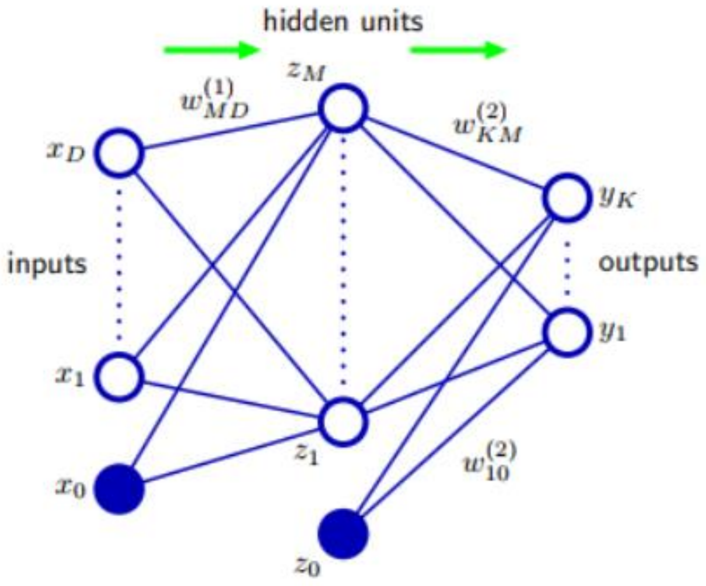


误差反向传播

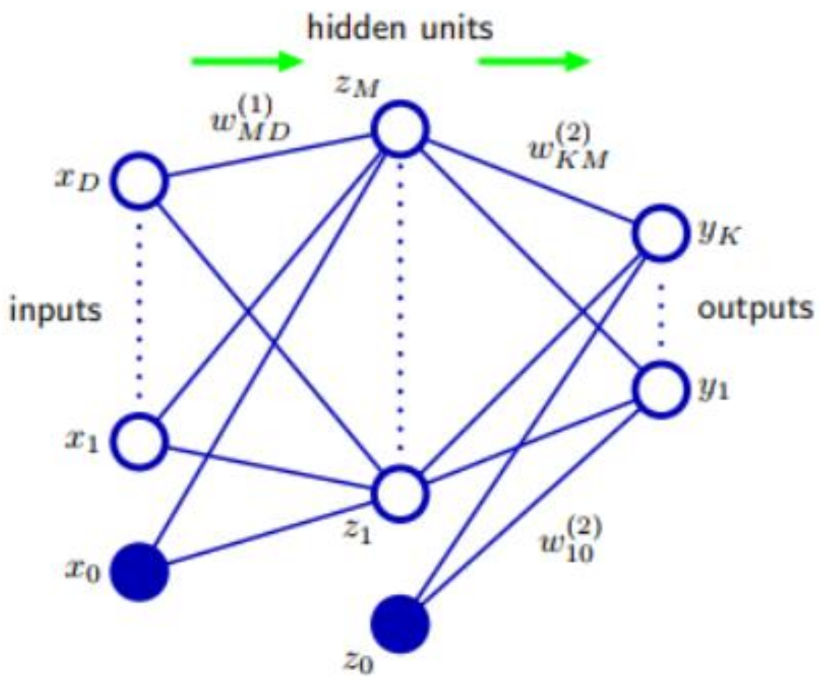
误差反向传播的一个具体例子

1. 两层神经网络
2. 平方和误差函数
3. 输出单元线性激活函数，即 $y_R = a_k$
4. 隐含层激活函数： $\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$

$$h'(a) = 1 - h(a)^2.$$



误差反向传播



误差反向传播的一个具体例子

$$E_n = \frac{1}{2}\sum_{k=1}^{K}(y_k - t_k)^2$$

$$\left.\begin{aligned} a_j &= \sum_{i=0}^{D} w^{(1)}_{ji} x_i \\ z_j &= \tanh(a_j) \\ y_k &= \sum_{j=0}^{M} w^{(2)}_{kj} z_j. \end{aligned}\right\} \text{前向传播}$$

$$\delta_k = y_k - t_k.$$

$$\left.\delta_j = (1 - z_j^2)\sum_{k=1}^{K} w_{kj}\delta_k.\right\} \text{误差反传}$$

$$\frac{\partial E_n}{\partial w^{(1)}_{ji}} = \delta_j x_i, \qquad \frac{\partial E_n}{\partial w^{(2)}_{kj}} = \delta_k z_j.$$

误差反向传播

反向传播的**效率**

- 整体计算复杂度 $O(w)$
- 权值w的数量一般总大于节点数（非常稀疏的网络除外）
- 计算复杂度取决于:

$$a_j = \sum_i w_{ji} z_i \qquad \delta_j = h'(a_j) \sum_k w_{kj} \delta_k$$

用 数值计算方法 计算误差函数导数方法的效率 – 有限差法

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{E_n(w_{ji}+\epsilon) - E_n(w_{ji})}{\epsilon} + O(\epsilon) \qquad \frac{\partial E_n}{\partial w_{ji}} = \frac{E_n(w_{ji}+\epsilon) - E_n(w_{ji}-\epsilon)}{2\epsilon} + O(\epsilon^2)$$

- 正向传播 $O(w)$
- 每个权值必须被单独施加扰动 $O(w^2)$

目录

- 1. 前馈神经网络
- 2. 网络训练
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播 BP
  - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

近似计算H

误差反向传播的推广

Jacobian矩阵 – 输出关于输出的导数

$$J_{ki} = \frac{\partial y_k}{\partial x_i}$$

$$\Delta y_k \simeq \frac{\partial y_k}{\partial x_i} \Delta x_i$$

将对x求导转换为对a求导

$$\frac{\partial y_k}{\partial x_i} = \sum_j \frac{\partial y_k}{\partial a_j} \frac{\partial a_j}{\partial x_i} = \sum_j w_{ji} \frac{\partial y_k}{\partial a_j}$$

对a求导的递推公式

$$\begin{aligned} \frac{\partial y_k}{\partial a_j} &= \sum_l \frac{\partial y_k}{\partial a_l} \frac{\partial a_l}{\partial a_j} \\ &= h'(a_j) \sum_l w_{lj} \frac{\partial y_k}{\partial a_l} \end{aligned}$$

输出层

$$\frac{\partial y_k}{\partial a_j} = \delta_{kj} \sigma'(a_j)$$

$$\frac{\partial y_k}{\partial a_j} = \delta_{kj} y_k - y_k y_j$$

误差反向传播的推广

Jacobian矩阵 – 输出关于输出的导数

- Jacobian 的数值计算方法

$$\frac{\partial y_k}{\partial x_i} = \frac{y_k(x_i+\epsilon) - y_k(x_i-\epsilon)}{2\epsilon} + O(\epsilon^2)$$

误差反向传播的推广

Hessian矩阵 – 误差函数的二阶导数

$$\frac{\partial^2 E}{\partial w_{ji} \partial w_{lk}}$$

- 作用:
  - 一些用来训练神经网络的非线性最优化算法是基于误差曲面的二阶性质的，这些性质由海森矩阵控制

  - 对于训练数据的微小变化，Hessian矩阵构成了快速训练前馈神经网络的基础

  - Hessian矩阵的逆矩阵用来鉴别神经网络中不要的权值，用于网络剪枝

  - Hessian矩阵的贝叶斯神经网络的拉普拉斯近似的核心
    - 逆矩阵用来确定训练过的神经网络的预测分布
    - 行列式用来计算模型证据
    - 特征值确定了超参数的值

误差反向传播的推广

Hessian矩阵 – 误差函数的二阶导数

$$\frac{\partial^2 E}{\partial w_{ji} \partial w_{lk}}$$

计算效率

计算方法	计算结果	计算效率			
对角近似	近似的H （只含对角线元素）	$O(w)$			
外积近似	$H \approx \sum_{n=1}^{N} b_n b_n^T$	$O(w^2)$			
外积近似	H^T				
有限差	近似的H	$O(w^2)$			
精确计算	H	$O(w^2)$			
快速乘法	$v^T H$	$O(w)$			

误差反向传播的推广

$$a_j = \sum_i w_{ji} z_i$$

$$z_j = h(a_j)$$

$$\frac{\partial E_n}{\partial \omega_{ji}} = \frac{\partial E_n}{a_j} z_i$$

$$\frac{\partial E_n}{\partial a_j} = h'(a_j) \Sigma \omega_{k_j} \frac{\partial E_n}{\partial a_k}$$

Hessian矩阵 – 对角近似 （$O(w)$）

$$E = \sum_n E_n.$$

$$\frac{\partial^2 E_n}{\partial w_{ji}^2} = \frac{\partial^2 E_n}{\partial a_j^2} z_i^2$$

$$\frac{\partial^2 E_n}{\partial a_j^2} = h'(a_j)^2 \sum_k \sum_{k'} w_{kj} w_{k'j} \frac{\partial^2 E_n}{\partial a_k \partial a_{k'}} + h''(a_j) \sum_k w_{kj} \frac{\partial E^n}{\partial a_k}$$

$$\frac{\partial^2 E_n}{\partial a_j^2} = h'(a_j)^2 \sum_k w_{kj}^2 \frac{\partial^2 E_n}{\partial a_k^2} + h''(a_j) \sum_k w_{kj} \frac{\partial E_n}{\partial a_k}$$

误差反向传播的推广

$$\nabla E = \sum_{n=1}^{N} (y_n - t_n) \nabla y_n$$

Hessian矩阵 – 外积近似 （$O(w^2)$）

平方差误差函数，恒等激活函数

$$E = \frac{1}{2} \sum_{n=1}^{N} (y_n - t_n)^2$$

$$\mathbf{H} = \nabla\nabla E = \sum_{n=1}^{N} \nabla y_n \nabla y_n + \sum_{n=1}^{N} (y_n - t_n) \nabla\nabla y_n$$

$$\mathbf{H} \simeq \sum_{n=1}^{N} \mathbf{b}_n \mathbf{b}_n^{\mathrm{T}}$$

交叉熵误差函数，sigmoid 激活函数

$$\mathbf{H} \simeq \sum_{n=1}^{N} y_n (1 - y_n) \mathbf{b}_n \mathbf{b}_n^{\mathrm{T}}$$

误差反向传播的推广

$$\left(\mathbf{M}+\mathbf{v}\mathbf{v}^{\mathrm{T}}\right)^{-1}=\mathbf{M}^{-1}-\frac{\left(\mathbf{M}^{-1}\mathbf{v}\right)\left(\mathbf{v}^{\mathrm{T}}\mathbf{M}^{-1}\right)}{1+\mathbf{v}^{\mathrm{T}}\mathbf{M}^{-1}\mathbf{v}}$$

Hessian矩阵 – 利用外积近似的结果求逆矩阵

$$\mathbf{H}_N=\sum_{n=1}^{N}\mathbf{b}_n\mathbf{b}_n^{\mathrm{T}}$$

$$\mathbf{H}_{L+1}=\mathbf{H}_L+\mathbf{b}_{L+1}\mathbf{b}_{L+1}^{\mathrm{T}}$$

$$\mathbf{H}_{L+1}^{-1}=\mathbf{H}_L^{-1}-\frac{\mathbf{H}_L^{-1}\mathbf{b}_{L+1}\mathbf{b}_{L+1}^{\mathrm{T}}\mathbf{H}_L^{-1}}{1+\mathbf{b}_{L+1}^{\mathrm{T}}\mathbf{H}_L^{-1}\mathbf{b}_{L+1}}.$$

$$H_0=\alpha I$$

计算的实际是 $H+\alpha I$ 的逆矩阵

误差反向传播的推广

Hessian矩阵 – 有限差近似

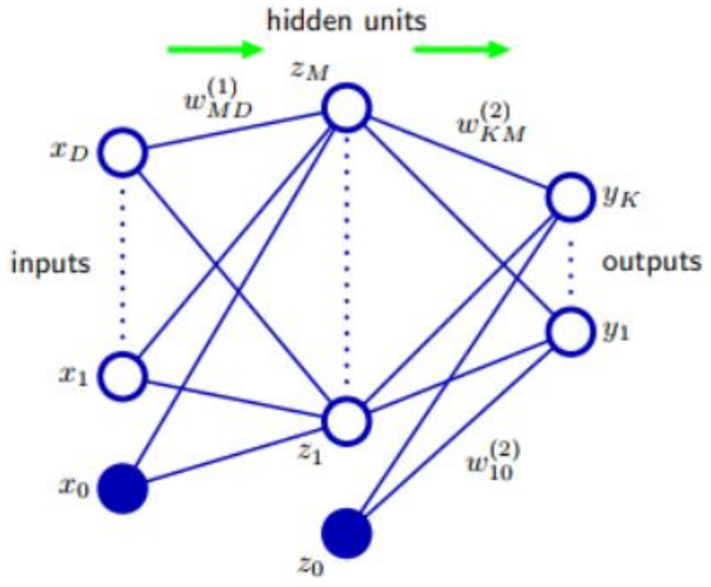
$$\frac{\partial^2 E}{\partial w_{ji}\partial w_{lk}} = \frac{1}{4\epsilon^2}\{E(w_{ji}+\epsilon, w_{lk}+\epsilon) - E(w_{ji}+\epsilon, w_{lk}-\epsilon) - E(w_{ji}-\epsilon, w_{lk}+\epsilon) + E(w_{ji}-\epsilon, w_{lk}-\epsilon)\} + O(\epsilon^2).$$

$(O(w^3))$

$$\frac{\partial^2 E}{\partial w_{ji}\partial w_{lk}} = \frac{1}{2\epsilon}\left\{\frac{\partial E}{\partial w_{ji}}(w_{lk}+\epsilon) - \frac{\partial E}{\partial w_{ji}}(w_{lk}-\epsilon)\right\} + O(\epsilon^2).$$

$(O(w^2))$

误差反向传播的推广



Hessian矩阵 – 精确计算 （$O(w^2)$）　—— 以二层网络为例

定义: $\delta_k = \dfrac{\partial E_n}{\partial a_k}$, $\quad M_{kk'} \equiv \dfrac{\partial^2 E_n}{\partial a_k \partial a_{k'}}$

- 两个权值都在第二层

$$\frac{\partial^2 E_n}{\partial w^{(2)}_{kj} \partial w^{(2)}_{k'j'}} = z_j z_{j'} M_{kk'}.$$

- 两个权值都在第一层

$$\frac{\partial^2 E_n}{\partial w^{(1)}_{ji} \partial w^{(1)}_{j'i'}} = x_i x_{i'} h''(a_{j'}) I_{jj'} \sum_k w^{(2)}_{kj'} \delta_k + x_i x_{i'} h'(a_{j'}) h'(a_j) \sum_k \sum_{k'} w^{(2)}_{k'j'} w^{(2)}_{kj} M_{kk'}$$

- 每层一个权值

$$\frac{\partial^2 E_n}{\partial w^{(1)}_{ji} \partial w^{(2)}_{kj'}} = x_i h'(a_{j'}) \left\{ \delta_k I_{jj'} + z_j \sum_{k'} w^{(2)}_{k'j'} H_{kk'} \right\}$$

误差反向传播的推广

Hessian矩阵 – 快速乘法（$O(w)$）

$$\mathbf{v}^{\mathrm{T}}\mathbf{H} = \mathbf{v}^{\mathrm{T}}\nabla(\nabla E)$$

定义运算：$\mathrm{R}\{\} = \nu^T\nabla$

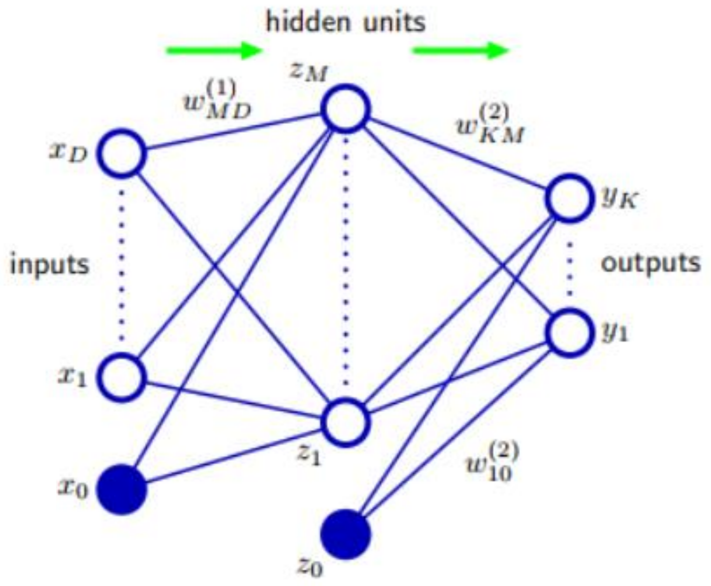
$$\mathcal{R}\{\mathbf{w}\} = \mathbf{v}.$$

先说结论

$$\mathcal{R}\left\{\frac{\partial E}{\partial w_{kj}}\right\} = \mathcal{R}\{\delta_k\}z_j + \delta_k\mathcal{R}\{z_j\}$$

$$\mathcal{R}\left\{\frac{\partial E}{\partial w_{ji}}\right\} = x_i\mathcal{R}\{\delta_j\}.$$

误差反向传播的推广



Hessian矩阵 – 快速乘法（$O(w)$）　——　以二层网络为例

- $\mathrm{R}\{\} = \nu^T$

正向传播:

$$a_j = \sum_i w_{ji} x_i \qquad \mathcal{R}\{a_j\} = \sum_i v_{ji} x_i$$

$$z_j = h(a_j) \qquad \mathcal{R}\{z_j\} = h'(a_j)\mathcal{R}\{a_j\}$$

$$y_k = \sum_j w_{kj} z_j \qquad \mathcal{R}\{y_k\} = \sum_j w_{kj}\mathcal{R}\{z_j\} + \sum_j v_{kj} z_j$$

反向传播:

$$\frac{\partial E}{\partial w_{kj}} = \delta_k z_j \qquad \mathcal{R}\left\{\frac{\partial E}{\partial w_{kj}}\right\} = \mathcal{R}\{\delta_k\} z_j + \delta_k \mathcal{R}\{z_j\}$$

$$\frac{\partial E}{\partial w_{ji}} = \delta_j x_i \qquad \mathcal{R}\left\{\frac{\partial E}{\partial w_{ji}}\right\} = x_i \mathcal{R}\{\delta_j\}.$$

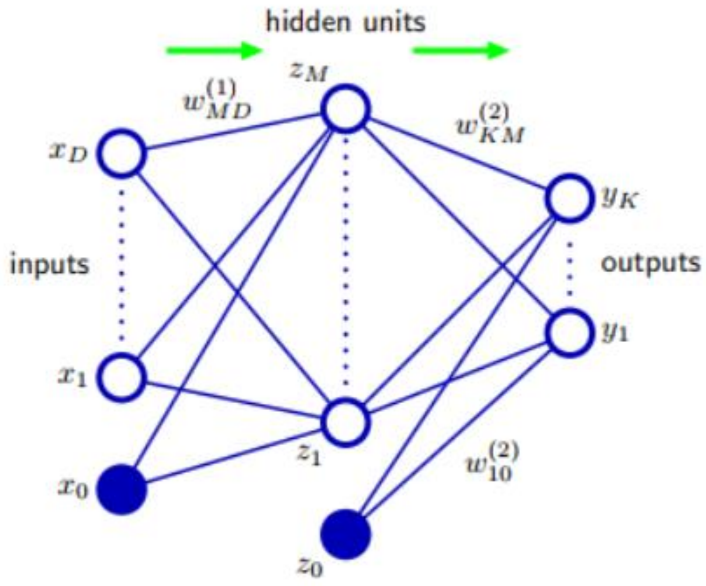
误差反向传播的推广

Hessian矩阵 – 快速乘法（$O(w)$）　　——　以二层网络为例

- $\mathrm{R}\{\} = \nu^T$

$$\mathcal{R}\left\{\frac{\partial E}{\partial w_{kj}}\right\} = \mathcal{R}\{\delta_k\}z_j + \delta_k\mathcal{R}\{z_j\}$$

$$\mathcal{R}\left\{\frac{\partial E}{\partial w_{ji}}\right\} = x_i\mathcal{R}\{\delta_j\}.$$



计算完整的Hessian矩阵：

- 令$\nu = (0,0,\ldots,1,\cdots,0)$的形式，每次算出一列（$O(w^2)$）

目录

- 1. 前馈神经网络
- 2. 网络训练
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化
- 3. 误差反向传播 BP
  - 误差函数导数（梯度）的计算/例子/反向传播的效率
- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 $\underbrace{\text{对角近似/外积近似/逆矩阵/有限差近似}}_{\text{近似计算}H}$/$\underbrace{\text{精确计算/快速乘法}}$

目录

- 1. 前馈神经网络

①函数形式

- 2. 网络训练
  - 参数最优化/局部二次近似/使用梯度信息/梯度下降最优化

②最大似然确定参数

非线性最优化

求导

- 3. 误差反向传播
  - 误差函数导数（梯度）的计算/例子/反向传播的效率

③BP

高效！

推广

- 4. 误差反向传播的推广
  - Jacobian矩阵
  - Hessian矩阵 对角近似/外积近似/逆矩阵/有限差近似/精确计算/快速乘法

④

⑤正则化

⑥混合密度网络

⑦贝叶斯