神经网络入门、反向传播算法（基于一个简单的神经网络）

覃雄派

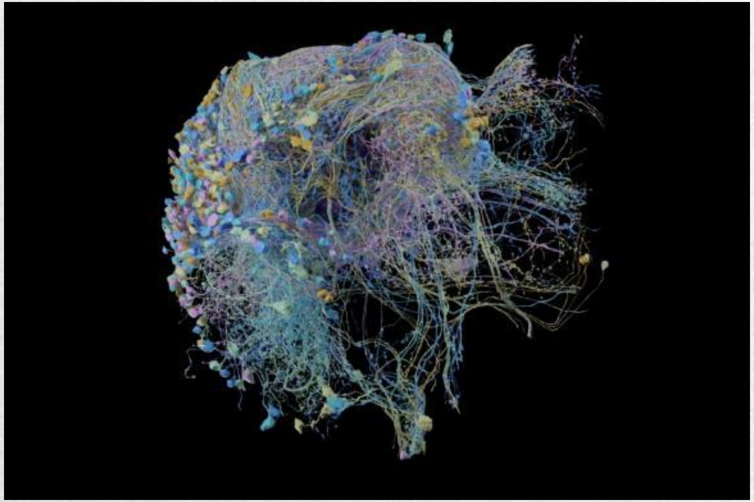
神经网络入门、反向传播算法（基于一个简单的神经网络）

提纲

- 神经网络入门
  - 神经元
  - 前向反馈多层神经网络
  - 循环神经网络/卷积神经网络/注意力机制
- 通过一个简单的神经网络
  - 前向传导
  - 反向传播

神经网络入门、反向传播算法（基于一个简单的神经网络）

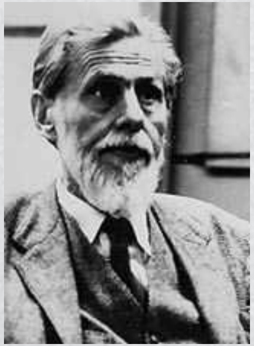
- 人工神经网络(Neural Network)
  - 是模仿人类神经系统特征，进行分布式并行信息处理的数学模型。

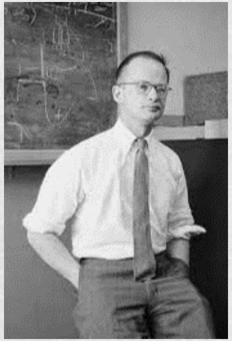


Google publishes largest ever high-resolution map of brain connectivity.2020

神经网络入门、反向传播算法（基于一个简单的神经网络）

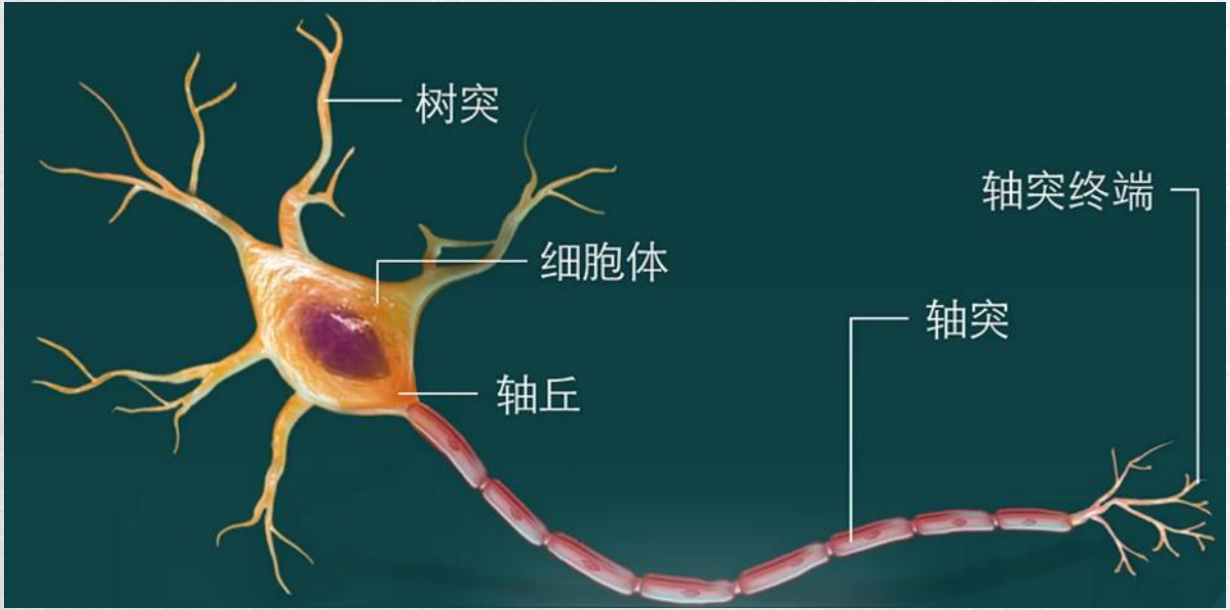
- 人工神经网络(Neural Network)
  - 人工神经网络技术可以追溯到20世纪40年代
    - 1943年，沃伦.麦考洛克(Warren McCulloch)与沃尔特.皮茨(Walter Pitts)首次提出了神经元的数学模型
    - 1958年，心理学家弗兰克.罗森布拉特(Frank Rosenblatt)提出了感知机(Perceptron)的概念，在神经元的结构中加入了训练修正参数的机制
    - 完成了人工神经网络基本原理的构建





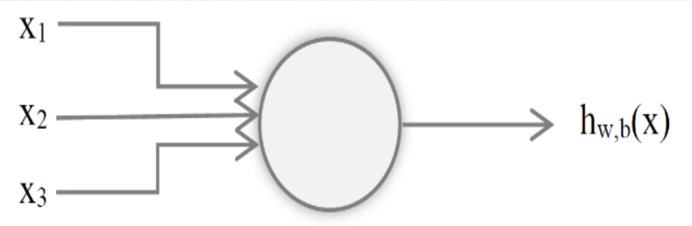
神经网络入门、反向传播算法（基于一个简单的神经网络）

- 人工神经网络(Neural Network)
  - 神经元



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 人工神经网络(Neural Network)
  - 人工神经元信息处理过程
  - 首先把前端(模仿神经元的树突)收集到的输入信号，进行加权求和，再通过一个激活函数转换成输出，传送出去(模仿神经元的轴突)
  - 图中展示了一个简单的神经元；其中，x_1、x_2、x_3为神经元的输入
    - 神经元的输出通过$h_{w,b}(x)=f(\sum_{i=1}^{3} w_i x_i + b)$函数来计算
    - f:R→R称为激活函数。



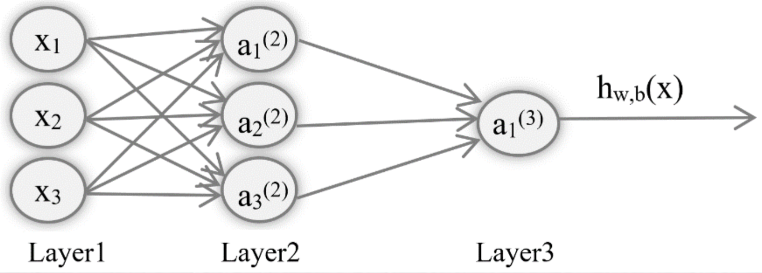
$$f(x) = \text{sigmoid}(x) = \frac{1}{1+e^{-x}};$$

$$f(x) = \tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

神经网络入门、反向传播算法（基于一个简单的神经网络）

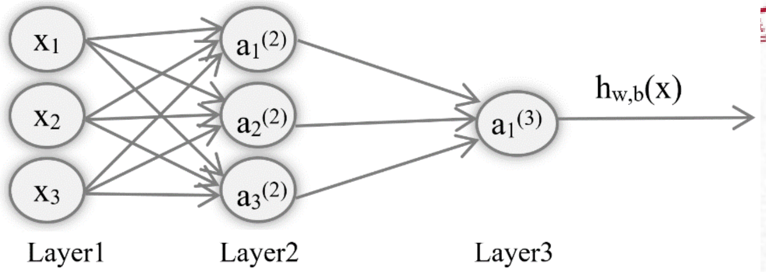
神经网络入门、反向传播算法（基于一个简单的神经网络）

- 人工神经网络(Neural Network)
  - 带一个隐藏层的简单的神经网络
    - 最简单的神经网络是前馈(Feed Forward)神经网络；它是一个多层网络，在这个神经网络中，每一层的节点仅和下一层的节点相连
  - 下图是一个简单的神经网络
    - 该神经网络最左边的一层，称为输入层，最右边的一层称为输出层
    - 中间的节点组成独立的一层，称为隐藏层；之所以称为隐藏层，是因为我们不能从训练样本上观察到它们的取值



神经网络入门、反向传播算法（基于一个简单的神经网络）

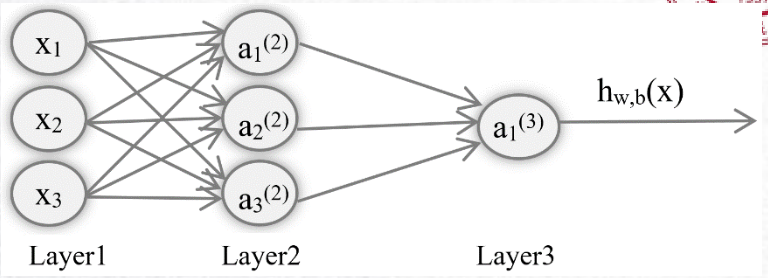
- 人工神经网络(Neural Network)
  - 带一个隐藏层的简单的神经网络
  - 前向传导
  - 计算输出值的过程，称为前向传导
    - $W_{ij}^{(l)}$表示第l层第j单元与第l+1层第i单元之间的连接参数，也就是连接线上的权重，$b_i^{(l)}$表示第l层第i单元的偏置项，也就是激活函数的常量部分；$a_i^{(l)}$表示第l层第i单元的激活值(即输出值)，当l=1的时候，$a_i^{(1)}=x_i$



$$a_1^{(2)} = f\left(w_{11}^{(1)}x_1 + w_{12}^{(1)}x_2 + w_{13}^{(1)}x_3 + b_1^{(1)}\right)$$
$$a_2^{(2)} = f\left(w_{21}^{(1)}x_1 + w_{22}^{(1)}x_2 + w_{23}^{(1)}x_3 + b_2^{(1)}\right)$$
$$a_3^{(2)} = f\left(w_{31}^{(1)}x_1 + w_{32}^{(1)}x_2 + w_{33}^{(1)}x_3 + b_3^{(1)}\right)$$
$$h_{w,b}(x) = a_1^{(3)} = f(w_{11}^{(2)}a_1^{(2)} + w_{12}^{(2)}a_2^{(2)} + w_{13}^{(2)}a_3^{(2)} + b_1^{(2)})$$

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 人工神经网络(Neural Network)
  - 带一个隐藏层的简单的神经网络
  - 前向传导
  - 计算输出值的过程，称为前向传导
    - 矩阵形式



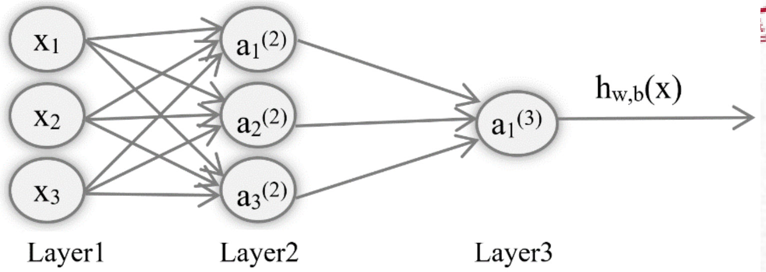
$$\begin{Bmatrix} {a_1}^{(2)} \\ {a_2}^{(2)} \\ {a_3}^{(2)} \end{Bmatrix} = \sigma\left(\begin{Bmatrix} {w_{11}}^{(1)} & {w_{12}}^{(1)} & {w_{13}}^{(1)} & {b_1}^{(1)} \\ {w_{21}}^{(1)} & {w_{22}}^{(1)} & {w_{23}}^{(1)} & {b_2}^{(1)} \\ {w_{31}}^{(1)} & {w_{32}}^{(1)} & {w_{32}}^{(1)} & {b_3}^{(1)} \end{Bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{Bmatrix}\right),$$

$$\{{a_1}^{(3)}\} = \sigma\left(\{{w_{11}}^{(2)} \quad {w_{12}}^{(2)} \quad {w_{13}}^{(2)} \quad {b_1}^{(2)}\} \begin{Bmatrix} {a_1}^{(2)} \\ {a_2}^{(2)} \\ {a_3}^{(2)} \\ 1 \end{Bmatrix}\right)$$

σ为传导函数

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 人工神经网络(Neural Network)
  - 带一个隐藏层的简单的神经网络
  - 反向传播
    - 计算误差，修正权重参数
  - 损失函数，均方差(此处只考虑一个样本)
    - $e = \frac{1}{2}\sum(x_i{}^l - y^{(i)})^2$
      - $x_i{}^l$是第l层的实际输出，$y^{(i)}$则是期望的输出(即实际值)，也就是训练样本里的对应$x^{(i)}$的输出
    - 采用梯度下降方法来修改权重
    - 对于某个神经元的某个权重的更新，采用公式$\Delta w_i = -\alpha \frac{\partial E}{\partial w_i}$，
      - E为输出误差；w_i为输入到该神经元的第i个连接的权重；α为学习率

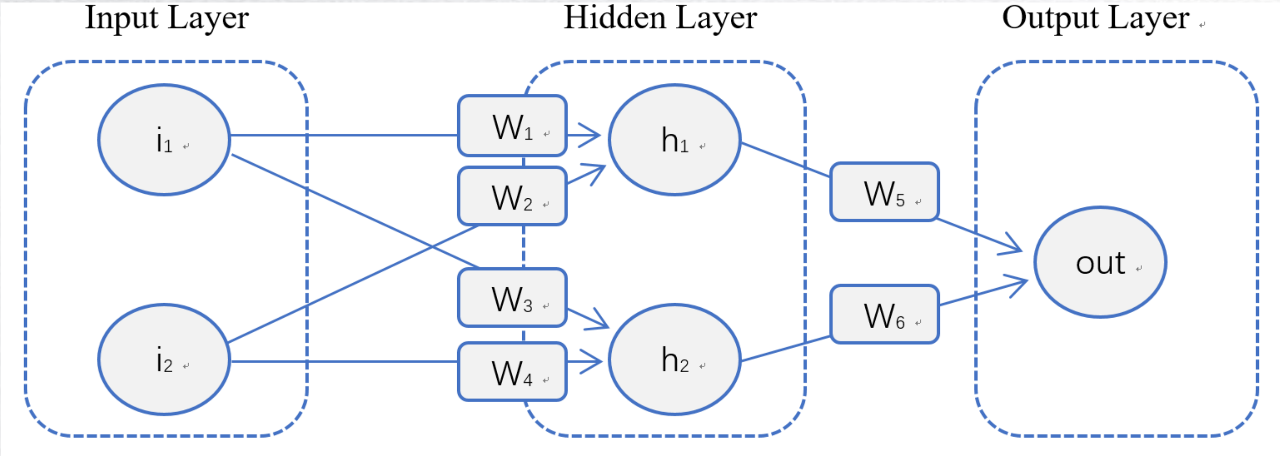


下文通过实例，掌握前向传导、反向传播过程

神经网络入门、反向传播算法（基于一个简单的神经网络）

神经网络入门、反向传播算法（基于一个简单的神经网络）

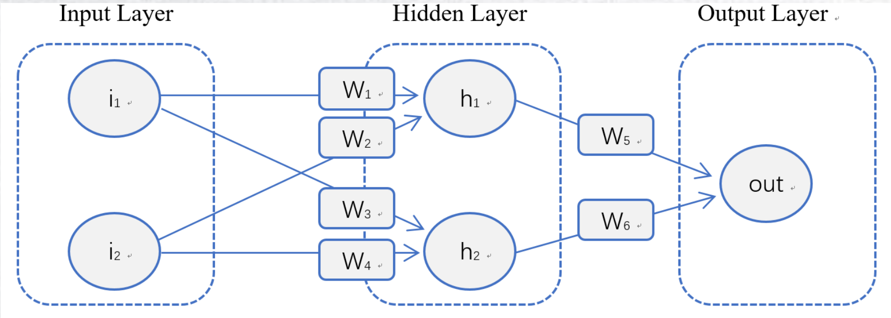
- 通过一个简单的神经网络了解反向传播算法
  - 该网络的输入层有2个神经元，隐藏层有2个神经元，输出层有1个神经元，如图所示
    - 图中同时标注了各个神经元之间的连接权重，比如i_1和h_1的连接权重是w_1



为了简单起见，这里的传导函数不做任何非线性变换，每个神经元把输入直接作为输出

神经网络入门、反向传播算法（基于一个简单的神经网络）

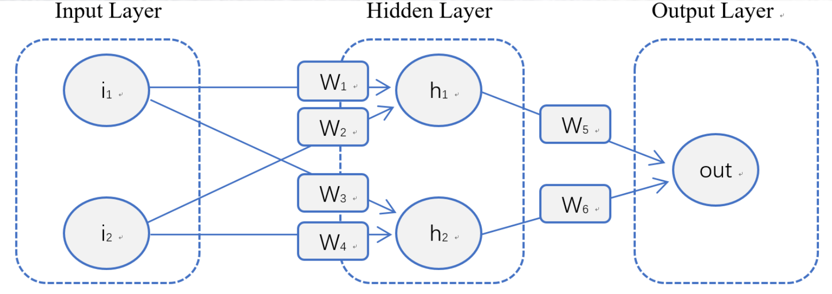
- 通过一个简单的神经网络了解反向传播算法
  - 前向传导
  - 开始，我们给神经网络一组随机的权重组合，比如w_1=0.11，w_2=0.21，w_3=0.12，w_4=0.08，w_5=0.14，w_6=0.15
  - 训练集只有一个样本(有多个样本的训练方法是类似的)
    - 输入为向量<2, 3>，输出为向量<1>
  - $h_1 = i_1 w_1 + i_2 w_2;$
  - $h_2 = i_1 w_3 + i_2 w_4;$
  - $out = h_1 w_5 + h_2 w_6$



神经网络入门、反向传播算法（基于一个简单的神经网络）

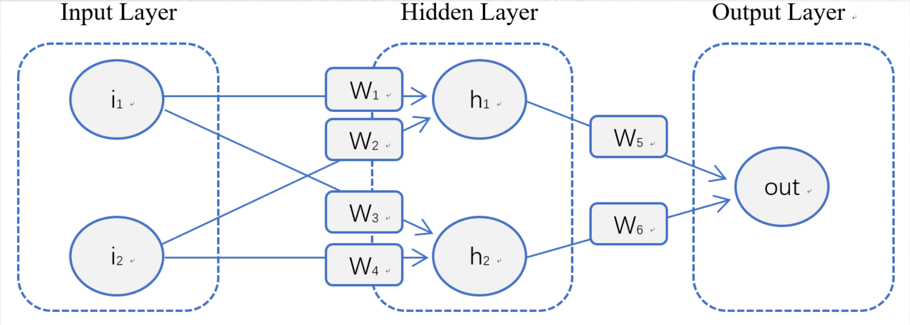
- 通过一个简单的神经网络了解反向传播算法
  - 前向传导
  - – 开始，我们给神经网络一组随机的权重组合，比如w_1=0.11，w_2=0.21，w_3=0.12，w_4=0.08，w_5=0.14，w_6=0.15
  - – 训练集只有一个样本(有多个样本的训练方法是类似的)
    - 输入为向量<2, 3>，输出为向量<1>

$$[h_1\ h_2] = [\mathrm{i}_1 \quad i_2]\begin{bmatrix} \mathrm{w}_1 & \mathrm{w}_3 \\ \mathrm{w}_2 & \mathrm{w}_4 \end{bmatrix}$$
$$= [2 \quad 3]\begin{bmatrix} 0.11 & 0.12 \\ 0.21 & 0.08 \end{bmatrix} = [0.85 \quad 0.48]$$
$$[out] = [h_1\ h_2]\begin{bmatrix} \mathrm{w}_5 \\ \mathrm{w}_6 \end{bmatrix} = [0.85 \quad 0.48]\begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix}$$
$$= 0.119 + 0.072 = [0.191]$$



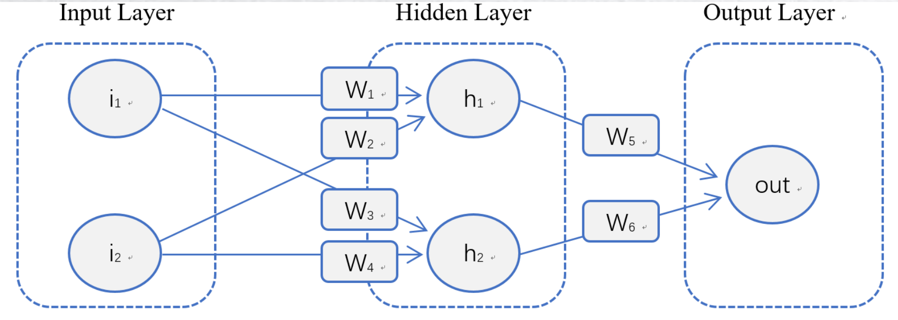
神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差
    - 神经网络的输出是0.191，而预期的输出是1，两者的误差计算如下(此处使用均方误差)：
    - $\text{Error} = \frac{1}{2}(0.191 - 1.0)^2$=0.327



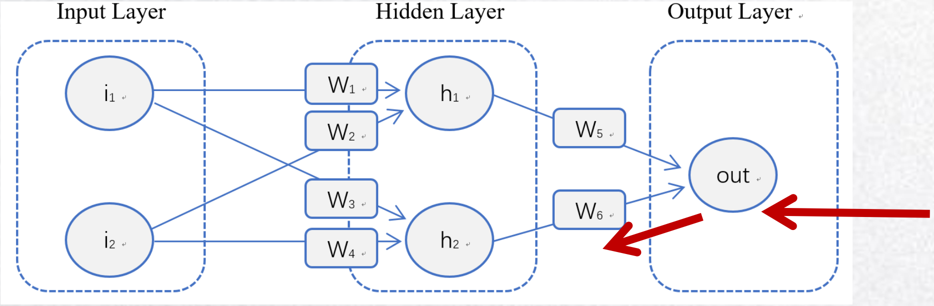
神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 反向传播算法的基础是梯度下降算法
    - 为了对网络连接的权重进行调整，需要计算误差相对于每个权重的梯度；首先有
  - Prediction = out $= h_1 w_5 + h_2 w_6 = (i_1 w_1 + i_2 w_2) w_5 + (i_1 w_3 + i_2 w_4) w_6$
  - 梯度下降的基本公式为：W = w -η $\frac{\partial Error}{\partial w}$
    - 为了调整w_6，我们需要计算误差针对w_6的梯度，然后代入$w_6 = w_6 - \eta \frac{\partial Error}{\partial w_6}$



神经网络入门、反向传播算法（基于一个简单的神经网络）

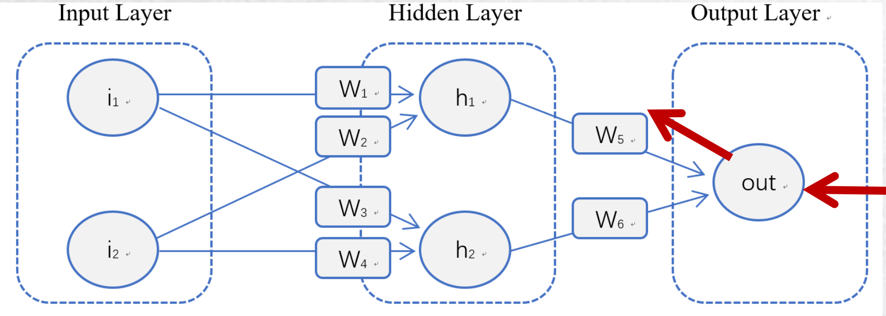
- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - $\frac{\partial Error}{\partial w_6} = \frac{\partial Error}{\partial prediction} \frac{\partial prediction}{\partial w_6} = \frac{\partial \frac{1}{2}(prediction - actual)^2}{\partial prediction} \frac{(i_1 w_1 + i_2 w_2) w_5 + (i_1 w_3 + i_2 w_4) w_6}{\partial w_6} =$
  - $2\frac{1}{2}(prediction - actual)(i_1 w_3 + i_2 w_4) = (prediction - actual) h_2 = \Delta h_2$
    - 备注：$h_2 = i_1 w_3 + i_2 w_4$,
    - Δ为$(prediction - actual)$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - w_5的更新公式为$w_5 = w_5 - \eta\frac{\partial Error}{\partial w_5}$，而$\frac{\partial Error}{\partial w_5}$=$\Delta h_1$

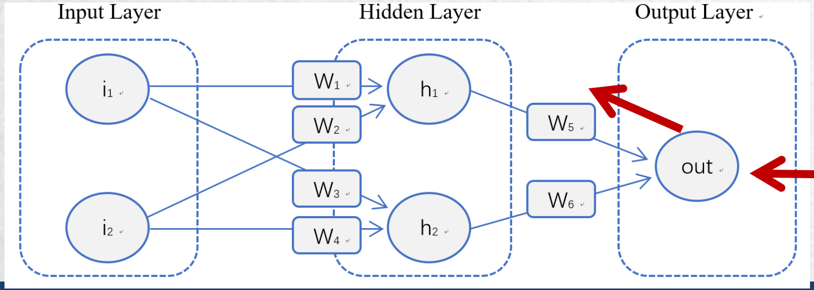
课堂练习，请推导$\frac{\partial Error}{\partial w_5}$



神经网络入门、反向传播算法（基于一个简单的神经网络）

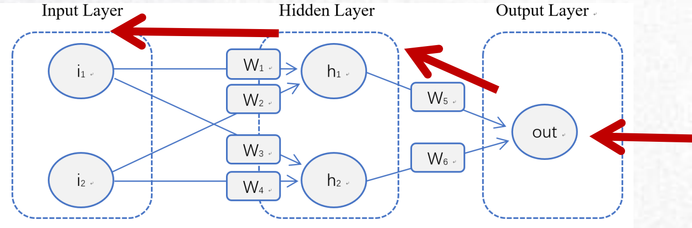
- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - $\frac{\partial Error}{\partial w_6} = \frac{\partial Error}{\partial prediction}\frac{\partial prediction}{\partial w_5}$
  - $= \frac{\partial \frac{1}{2}(prediction-actual)^2}{\partial prediction}\frac{(i_1w_1+i_2w_2)w_5+(i_1w_3+i_2w_4)w_6}{\partial w_5} =$
  - $2\frac{1}{2}(prediction - actual)(i_1w_1 + i_2w_2) = (prediction - actual)h_1 = \Delta h_1$
    - 备注：$h_1 = i_1w_1 + i_2w_2$,
    - Δ为$(prediction - actual)$

课堂练习，请推导$\frac{\partial Error}{\partial w_5}$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 我们需要继续计算$\frac{\partial Error}{\partial w_1}$，以便对$w_1$进行更新。
  - $\frac{\partial Error}{\partial w_1} = \frac{\partial Error}{\partial prediction} \frac{\partial prediction}{\partial h_1} \frac{\partial h_1}{\partial w_1}$
  - $= \frac{\partial \frac{1}{2}(prediction-actual)^2}{\partial prediction} \frac{\partial (h_1 w_5 + h_2 w_6)}{\partial h_1} \frac{\partial (i_1 w_1 + i_2 w_2)}{\partial w_1}$
  - $= 2\frac{1}{2}\left(prediction - actual\right) w_5 i_1 = \Delta w_5 i_1$



神经网络入门、反向传播算法（基于一个简单的神经网络）

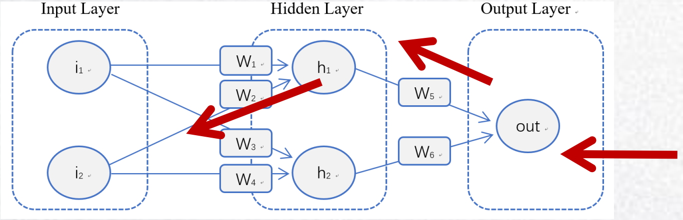
- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 同样道理，$\frac{\partial Error}{\partial w_2}=\Delta w_5 i_2$, $\frac{\partial Error}{\partial w_3}=\Delta w_6 i_1$, $\frac{\partial Error}{\partial w_4}=\Delta w_6 i_2$

课堂练习，请推导 $\frac{\partial Error}{\partial w_2}$ $\frac{\partial Error}{\partial w_3}$ $\frac{\partial Error}{\partial w_4}$

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 我们需要继续计算$\frac{\partial Error}{\partial w_2}$，以便对$w_2$进行更新。
  - $\frac{\partial Error}{\partial w_2} = \frac{\partial Error}{\partial prediction}\frac{\partial prediction}{\partial h_1}\frac{\partial h_1}{\partial w_2}$
  - $= \frac{\partial \frac{1}{2}(prediction-actual)^2}{\partial prediction}\frac{\partial (h_1 w_5+h_2 w_6)}{\partial h_1}\frac{\partial (i_1 w_1+i_2 w_2)}{\partial w_2}$
  - $=2\frac{1}{2}\left(prediction - actual\right)w_5 i_2=\Delta w_5 i_2$

课堂练习，请推导 $\frac{\partial Error}{\partial w_2}$ $\frac{\partial Error}{\partial w_3}$ $\frac{\partial Error}{\partial w_4}$

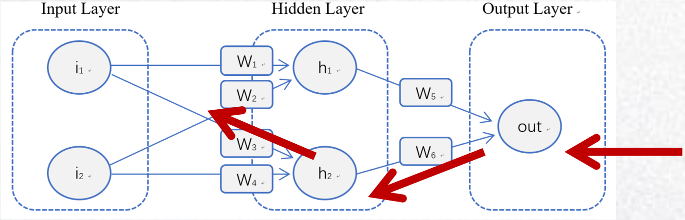


神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 我们需要继续计算$\frac{\partial Error}{\partial w_3}$，以便对$w_3$进行更新。
  - $\frac{\partial Error}{\partial w_3} = \frac{\partial Error}{\partial prediction} \frac{\partial prediction}{\partial h_2} \frac{\partial h_1}{\partial w_3}$
  - $= \frac{\partial \frac{1}{2}(prediction - actual)^2}{\partial prediction} \frac{\partial (h_1 w_5 + h_2 w_6)}{\partial h_2} \frac{\partial (i_1 w_3 + i_2 w_4)}{\partial w_3}$
  - $= 2\frac{1}{2}\left(prediction - actual\right) w_6 i_1 = \Delta w_6 i_1$

课堂练习，请推导

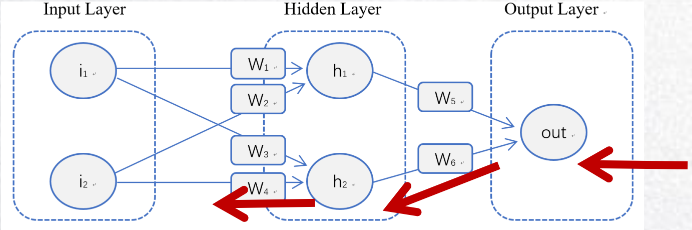
$\frac{\partial Error}{\partial w_2}$ $\frac{\partial Error}{\partial w_3}$ $\frac{\partial Error}{\partial w_4}$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 我们需要继续计算$\frac{\partial Error}{\partial w_4}$，以便对$w_4$进行更新。
  - $\frac{\partial Error}{\partial w_4} = \frac{\partial Error}{\partial prediction}\frac{\partial prediction}{\partial h_2}\frac{\partial h_1}{\partial w_4}$
  - $= \frac{\partial \frac{1}{2}(prediction - actual)^2}{\partial prediction}\frac{\partial (h_1 w_5 + h_2 w_6)}{\partial h_2}\frac{\partial (i_1 w_3 + i_2 w_4)}{\partial w_4}$
  - $= 2\frac{1}{2}\left(prediction - actual\right) w_6 i_2 = \Delta w_6 i_2$

课堂练习，请推导$\frac{\partial Error}{\partial w_2}\frac{\partial Error}{\partial w_3}\frac{\partial Error}{\partial w_4}$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
    - 参数修正公式列表
    - $w_6 = w_6 - \eta\Delta h_2$
    - $w_5 = w_5 - \eta\Delta h_1$
    - $w_4 = w_4 - \eta\Delta w_6 i_2$
    - $w_3 = w_3 - \eta\Delta w_6 i_1$
    - $w_2 = w_2 - \eta\Delta w_5 i_2$
    - $w_1 = w_1 - \eta\Delta w_5 i_1$

$$\frac{\partial Error}{\partial w_6} = \Delta h_2$$

$$\frac{\partial Error}{\partial w_5} = \Delta h_1$$

$$\frac{\partial Error}{\partial w_4} = \Delta w_6 i_2$$

$$\frac{\partial Error}{\partial w_3} = \Delta w_6 i_1$$

$$\frac{\partial Error}{\partial w_2} = \Delta w_5 i_2$$

$$\frac{\partial Error}{\partial w_1} = \Delta w_5 i_1$$

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差的反向传播
  - 矩阵形式
    - $w_6 = w_6 - \eta\Delta h_2$
    - $w_5 = w_5 - \eta\Delta h_1$
    - $w_4 = w_4 - \eta\Delta w_6 i_2$
    - $w_3 = w_3 - \eta\Delta w_6 i_1$
    - $w_2 = w_2 - \eta\Delta w_5 i_2$
    - $w_1 = w_1 - \eta\Delta w_5 i_1$

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \eta\Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \begin{bmatrix} w_5 & w_6 \end{bmatrix}$$

$$= \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta \begin{bmatrix} i_1 w_5 & i_1 w_6 \\ i_2 w_5 & i_2 w_6 \end{bmatrix}$$

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 拿实际数据进行反向传播过程计算

1

使用初始权重，正向传播与误差

$$[h_1\ h_2] = [2 \quad 3]\begin{bmatrix}0.11 & 0.12\\ 0.21 & 0.08\end{bmatrix} = [0.85 \quad 0.48]$$

$$[out] = [0.85 \quad 0.48]\begin{bmatrix}0.14\\ 0.15\end{bmatrix} = 0.119 + 0.072 = [0.191]$$

Δ=0.191 – 1.0=-0.809

注意可以通过 https://hmkcode.com/netflow/验证

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 拿实际数据进行反向传播过程计算

1

$$[h_1\ h_2] = [2 \quad 3]\begin{bmatrix}0.11 & 0.12\\0.21 & 0.08\end{bmatrix} = [0.85 \quad 0.48]$$

$$[out] = [0.85 \quad 0.48]\begin{bmatrix}0.14\\0.15\end{bmatrix} = 0.119 + 0.072 = [0.191]$$

修正w5和w6

$$\begin{bmatrix}w_5\\w_6\end{bmatrix}=\begin{bmatrix}w_5\\w_6\end{bmatrix}\text{-}\eta\Delta\begin{bmatrix}h_1\\h_2\end{bmatrix}$$

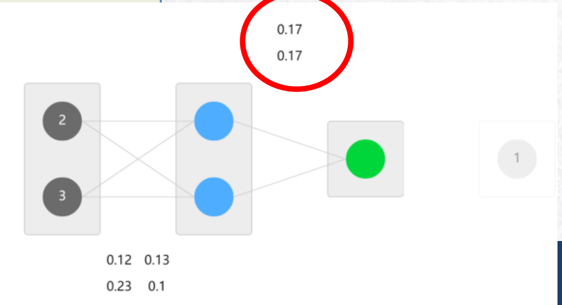
$$\begin{bmatrix}w_1 & w_3\\w_2 & w_4\end{bmatrix} = \begin{bmatrix}w_1 & w_3\\w_2 & w_4\end{bmatrix}\text{-}\eta\Delta\begin{bmatrix}i_1\\i_2\end{bmatrix}[w_5 \quad w_6]$$
$$=\begin{bmatrix}w_1 & w_3\\w_2 & w_4\end{bmatrix}\text{-}\eta\Delta\begin{bmatrix}i_1w_5 & i_1w_6\\i_2w_5 & i_2w_6\end{bmatrix}$$

$$\Delta\text{=}0.191 - 1.0\text{=-}0.809$$

$$\eta\text{=}0.05$$

$$\begin{bmatrix}w_5\\w_6\end{bmatrix} = \begin{bmatrix}0.14\\0.15\end{bmatrix} - 0.05(-0.809)\begin{bmatrix}0.85\\0.48\end{bmatrix} = \begin{bmatrix}0.14\\0.15\end{bmatrix} + 0.04045\begin{bmatrix}0.85\\0.48\end{bmatrix}$$

$$= \begin{bmatrix}0.14\\0.15\end{bmatrix}+\begin{bmatrix}0.0344\\0.0194\end{bmatrix}=\boxed{\begin{bmatrix}\mathbf{0.1744}\\\mathbf{0.1694}\end{bmatrix}}$$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 拿实际数据进行反向传播过程计算

1

修正
w1,w2,w3,w4

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} 0.11 & 0.12 \\ 0.21 & 0.08 \end{bmatrix} - 0.05(-0.809)\begin{bmatrix} 2 \\ 3 \end{bmatrix}\begin{bmatrix} 0.14 & 0.15 \end{bmatrix}$$

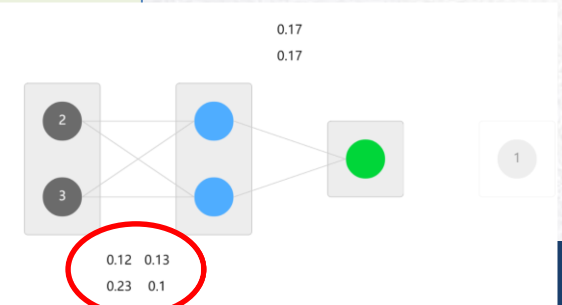
$$= \begin{bmatrix} 0.11 & 0.12 \\ 0.21 & 0.08 \end{bmatrix} + 0.04045\begin{bmatrix} 0.28 & 0.30 \\ 0.42 & 0.45 \end{bmatrix}$$

$$= \begin{bmatrix} 0.11 & 0.12 \\ 0.21 & 0.08 \end{bmatrix} + \begin{bmatrix} 0.011 & 0.012 \\ 0.017 & 0.018 \end{bmatrix} = \boxed{\begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix}}$$

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \eta\Delta\begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta\begin{bmatrix} i_1 \\ i_2 \end{bmatrix}\begin{bmatrix} w_5 & w_6 \end{bmatrix}$$

$$= \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta\begin{bmatrix} i_1 w_5 & i_1 w_6 \\ i_2 w_5 & i_2 w_6 \end{bmatrix}$$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 利用调整过的权重
  - 重新进行前向传导

2

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \eta\Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \begin{bmatrix} w_5 & w_6 \end{bmatrix}$$

$$= \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta \begin{bmatrix} i_1 w_5 & i_1 w_6 \\ i_2 w_5 & i_2 w_6 \end{bmatrix}$$

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} \mathbf{0.1744} \\ \mathbf{0.1694} \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} == \begin{bmatrix} \mathbf{0.12} & \mathbf{0.13} \\ \mathbf{0.23} & \mathbf{0.10} \end{bmatrix}$$

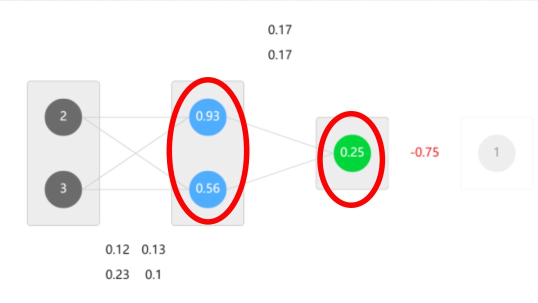
$$[h_1\ h_2] = [2 \quad 3] \begin{bmatrix} \mathbf{0.12} & \mathbf{0.13} \\ \mathbf{0.23} & \mathbf{0.10} \end{bmatrix}$$

$$= [0.93 \quad 0.56]$$

$$[out] = [0.93 \quad 0.56] \begin{bmatrix} \mathbf{0.1744} \\ \mathbf{0.1694} \end{bmatrix} = [0.2571]$$

$$[h_1\ h_2] = [i_1 \quad i_2] \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix}$$

$$[out] = [h_1\ h_2] \begin{bmatrix} w_5 \\ w_6 \end{bmatrix}$$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 利用调整过的权重，重新进行前向传导
  - 新的预测值为0.2571，比起上一个预测值0.191，更加接近目标值1
  - 利用训练样本，不断迭代前向传导和反向传播过程
    - 直到误差接近0或者到达某个阈值之下

2

$$[h_1\ h_2] = [2 \quad 3]\begin{bmatrix} \mathbf{0.12} & \mathbf{0.13} \\ \mathbf{0.23} & \mathbf{0.10} \end{bmatrix}$$

$$= [0.93 \quad 0.56]$$

$$[out] = [0.93 \quad 0.56]\begin{bmatrix} \mathbf{0.1744} \\ \mathbf{0.1694} \end{bmatrix} = \boxed{[0.2571]}$$

预测值
更接近1

old out
[0.191]

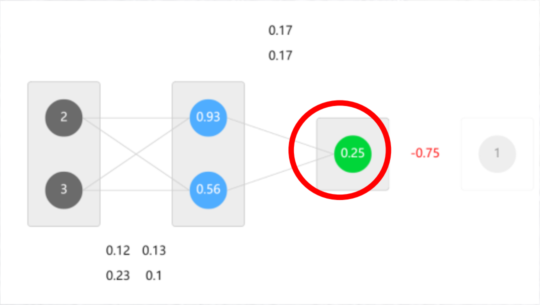
神经网络入门、反向传播算法（基于一个简单的神经网络）

- 请课堂练习反向传播和参数修正
  - 目前预测值为0.2571
  - 实际值为1
  - 那么误差为-0.7429

2

课堂练习：
1.误差
2.反向传播
3.前向传导，计算新误差

$$[h_1\ h_2] = [2 \quad 3]\begin{bmatrix} \mathbf{0.12} & \mathbf{0.13} \\ \mathbf{0.23} & \mathbf{0.10} \end{bmatrix}$$
$$= [0.93 \quad 0.56]$$
$$[out] = [0.93 \quad 0.56]\begin{bmatrix} \mathbf{0.1744} \\ \mathbf{0.1694} \end{bmatrix} = \boxed{[0.2571]}$$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差
  - 拿实际数据进行反向传播过程计算

2

修正w5和w6

$$\begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} \mathbf{2} \\ \mathbf{3} \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} \mathbf{0.12} & \mathbf{0.13} \\ \mathbf{0.23} & \mathbf{0.10} \end{bmatrix}$$

$$\begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.93} \\ \mathbf{0.56} \end{bmatrix}$$

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \eta\Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \begin{bmatrix} w_5 & w_6 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta \begin{bmatrix} i_1 w_5 & i_1 w_6 \\ i_2 w_5 & i_2 w_6 \end{bmatrix}$$

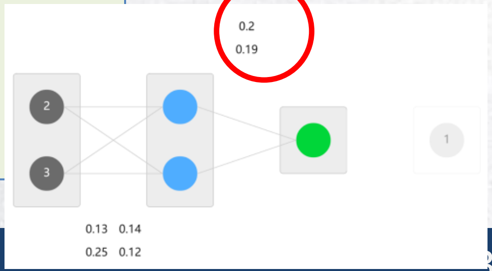
$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} \mathbf{0.1744} \\ \mathbf{0.1694} \end{bmatrix}$$

$\Delta=0.2571-1.0=-0.7429$

$\eta=0.05$

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0.1744 \\ 0.1694 \end{bmatrix} - 0.05(-0.7429)\begin{bmatrix} 0.93 \\ 0.56 \end{bmatrix} = \begin{bmatrix} 0.1744 \\ 0.1694 \end{bmatrix} + 0.037145\begin{bmatrix} 0.93 \\ 0.56 \end{bmatrix}$$

$$= \begin{bmatrix} 0.1744 \\ 0.1694 \end{bmatrix} + \begin{bmatrix} 0.034545 \\ 0.02080 \end{bmatrix} = \begin{bmatrix} \mathbf{0.2089} \\ \mathbf{0.1902} \end{bmatrix}$$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差
  - 拿实际数据进行反向传播过程计算

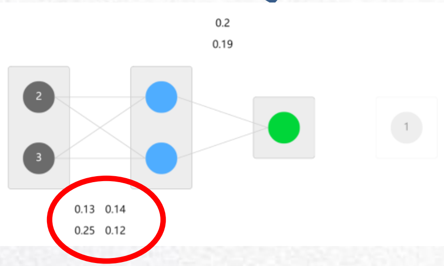
2

修正
w1,w2,w3,w4

$$\begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} \mathbf{2} \\ \mathbf{3} \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} == \begin{bmatrix} \mathbf{0.12} & \mathbf{0.13} \\ \mathbf{0.23} & \mathbf{0.10} \end{bmatrix}$$

$$\begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.93} \\ \mathbf{0.56} \end{bmatrix}$$



$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} \mathbf{0.1744} \\ \mathbf{0.1694} \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix} - 0.05(-0.7429)\begin{bmatrix} 2 \\ 3 \end{bmatrix}\begin{bmatrix} 0.1744 & 0.1694 \end{bmatrix}$$

$$= \begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix} + 0.037145\begin{bmatrix} 0.28 & 0.30 \\ 0.42 & 0.45 \end{bmatrix}$$

$$= \begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix} + \begin{bmatrix} 0.0104 & 0.0111 \\ 0.0156 & 0.0167 \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & \mathbf{0.14} \\ \mathbf{0.246} & \mathbf{0.1167} \end{bmatrix}$$

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \eta\Delta\begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta\begin{bmatrix} i_1 \\ i_2 \end{bmatrix}\begin{bmatrix} w_5 & w_6 \end{bmatrix}$$

$$= \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \eta\Delta\begin{bmatrix} i_1 w_5 & i_1 w_6 \\ i_2 w_5 & i_2 w_6 \end{bmatrix}$$

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 误差
  - 利用调整过的权重
  - 重新进行前向传导

2

$[h_1\ h_2] = [2 \quad 3]\begin{bmatrix} 0.13 & 0.14 \\ 0.246 & 0.1167 \end{bmatrix} =$
$[0.998 \quad 0.63]$

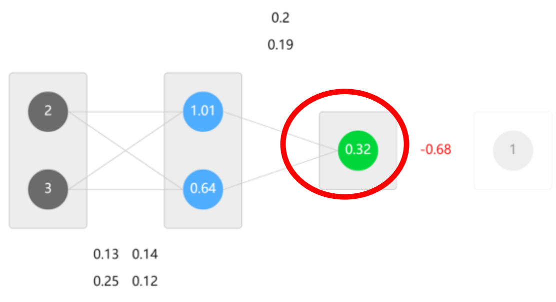
$[out] = [0.998 \quad 0.63]\begin{bmatrix} 0.2089 \\ 0.1902 \end{bmatrix} = 0.2085 +$
$0.1198 = [0.3283]$

$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0.2089 \\ 0.1902 \end{bmatrix}$

$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.14 \\ 0.246 & 0.1167 \end{bmatrix}$

$[h_1\ h_2] = [i_1 \quad i_2]\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix}$

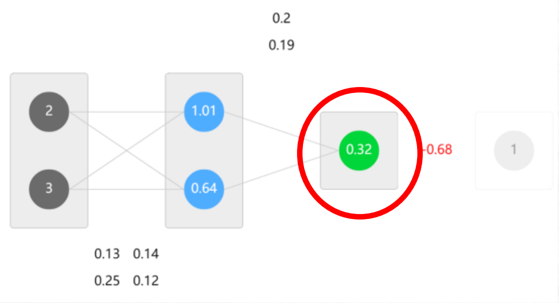
$[out] = [h_1\ h_2]\begin{bmatrix} w_5 \\ w_6 \end{bmatrix}$



神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 利用调整过的权重，重新进行前向传导
  - 新的预测值为0.3283，比起上一个预测值0.2571，更加接近目标值1
  - 利用训练样本，不断迭代前向传导和反向传播过程
    - 直到误差接近0或者到达某个阈值之下

2



$$[h_1\ h_2] = [2 \quad 3]\begin{bmatrix} \mathbf{0.13} & \mathbf{0.14} \\ \mathbf{0.246} & \mathbf{0.1167} \end{bmatrix} = [0.998 \quad 0.63]$$

$$[out] = [0.998 \quad 0.63]\begin{bmatrix} \mathbf{0.2089} \\ \mathbf{0.1902} \end{bmatrix} = 0.2085 + 0.1198 = \boxed{[0.3283]}$$

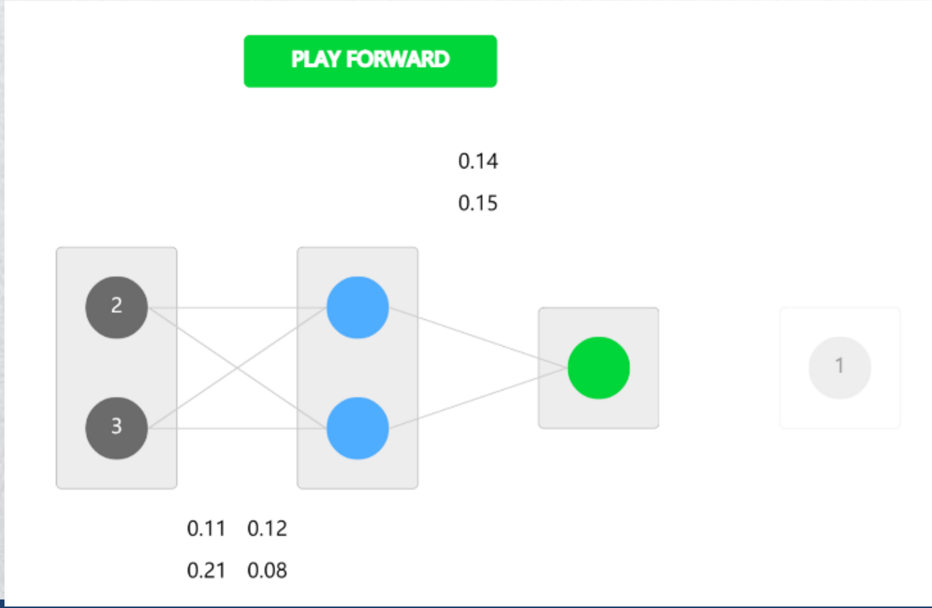
预测值更接近1

[0.2571]

神经网络入门、反向传播算法（基于一个简单的神经网络）

神经网络入门、反向传播算法（基于一个简单的神经网络）

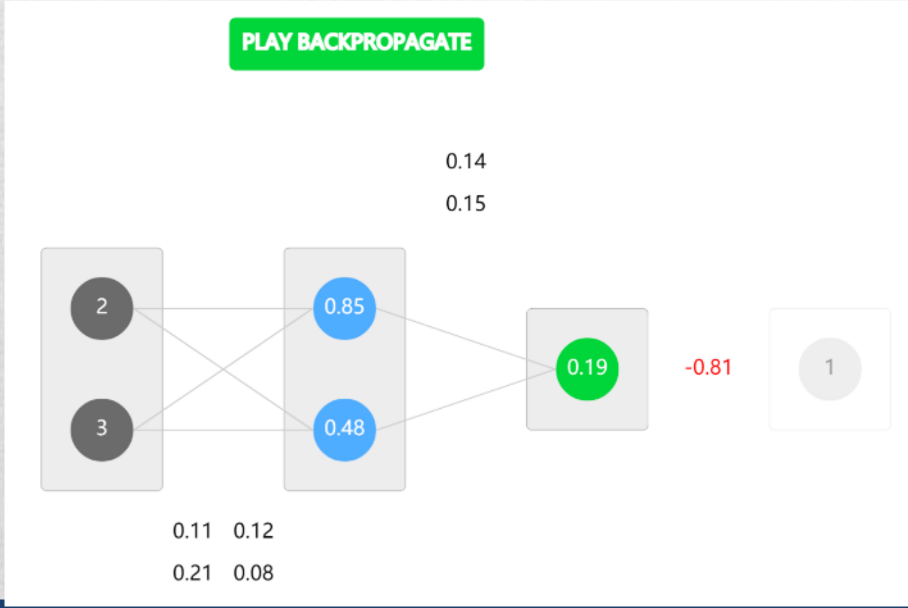
- 通过一个简单的神经网络了解反向传播算法
  - 展示训练过程
  - https://hmkcode.com/netflow/



Play forward
Play back propagation
观察参数、误差的变化

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 通过一个简单的神经网络了解反向传播算法
  - 展示训练过程
  - https://hmkcode.com/netflow/



Play forward
Play back propagation
观察参数、误差的变化

神经网络入门、反向传播算法（基于一个简单的神经网络）

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 展望
  - 为了简单起见，前文的实例
    - 传导函数不做任何非线性变换，每个神经元把输入直接作为输出
    - 实际应用中的网络层数更多、每层的神经元数量更多，结构更加复杂，意味着更多的参数，求导和训练的过程代价更大

神经网络入门、反向传播算法（基于一个简单的神经网络）

神经网络入门、反向传播算法（基于一个简单的神经网络）

- 利用矩阵求导求解梯度更新式

请参考下一个PPT