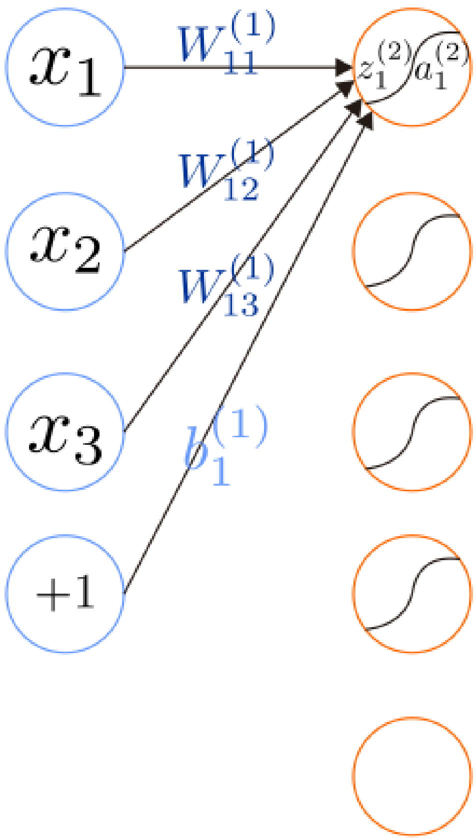
这是一篇学习UFLDL反向传导算法的笔记，按自己的思路捋了一遍，有不对的地方请大家指点。

首先说明一下神经网络的符号：
1. n_l 表示神经网络的层数。
2. s_l 表示第 l 层神经元个数，不包含偏置单元。
3. $z_i^{(l)}$ 表示第 l 层第 i 个神经元的输入；$a_i^{(l)}$ 表示第 l 层第 i 个神经元的输出。
4. $W_{ij}^{(l)}$ 表示第 l 层第 j 个神经元连接到第 $l+1$ 层第 i 个神经元的权重，因此权值矩阵 W 的维数为 $s_{l+1} \times s_l$



第二层各神经元的计算方法如下：

$$a_1^{(2)} = f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)})$$
$$a_2^{(2)} = f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)})$$
$$a_3^{(2)} = f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)})$$
$$a_4^{(2)} = f(W_{41}^{(1)}x_1 + W_{42}^{(1)}x_2 + W_{43}^{(1)}x_3 + b_4^{(1)})$$

我们可以将其向量化表示：

$$z^{(2)} = W^{(1)}x + b^{(1)}$$
$$a^{(2)} = f(z^{(2)})$$

这里的矩阵W的具体形式为：

$$W_{4\times 3} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \\ W_{31}^{(1)} & W_{32}^{(1)} & W_{33}^{(1)} \\ W_{41}^{(1)} & W_{42}^{(1)} & W_{43}^{(1)} \end{bmatrix}$$

第2层的神经元个数为4，第1层神经元的个数为3，因此为 4×3 维的矩阵。

代价函数

对于单个样本我们将神经网络的代价函数定义为：

$$J(W, b; x, y) = \frac{1}{2} \|h_{W,b}(x) - y\|^2$$

对所有 K 个样本，神经网络的总的代价函数(这也是批量的由来)为：

$$\begin{aligned} J(W, b) &= [\frac{1}{K}\sum_{k=1}^{K} J(W, b; x^{(k)}, y^{(k)})] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}} (W_{ji}^{(l)})^2 \\ &= [\frac{1}{K}\sum_{k=1}^{K} (\frac{1}{2}\|h_{W,b}(x^{(k)}) - y^{(k)}\|^2)] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}} (W_{ji}^{(l)})^2 \end{aligned}$$

使用批量梯度下降算法寻求神经网络的最优参数

我们使用批量梯度下降算法寻求神经网络的最优参数 $W^{(l)}, b^{l}$。
我们先来看对于 第 $l+1$ 层第 i 个神经元来说，第 l 层第 j 个神经元的权值可按如下方式迭代更新：

$$\begin{aligned} W_{ij}^{(l)} &= W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b) \\ &= W_{ij}^{(l)} - \alpha \left[\left(\frac{1}{K}\sum_{k=1}^{K} \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b; x^{(k)}, y^{(k)}) \right) + \lambda W_{ij}^{(l)} \right] \end{aligned}$$

类似的，对于 第 $l+1$ 层第 i 个神经元来说，第 l 层的偏置单元的权值可按如下方式迭代更新：

$$\begin{aligned} b_i^{(l)} &= b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W, b) \\ &= b_i^{(l)} - \alpha \left[\frac{1}{K}\sum_{k=1}^{K} \frac{\partial}{\partial b_i^{(l)}} J(W, b; x^{(k)}, y^{(k)}) \right] \end{aligned}$$

我们现在的目的是求出以下两个式子就可以对参数进行迭代了：

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b; x^{(k)}, y^{(k)})$$
$$\frac{\partial}{\partial b_i^{(l)}} J(W, b; x^{(k)}, y^{(k)})$$

又我们知道第 $l+1$ 层第 i 个神经元的输入 $z_i^{(l+1)}$ 可以由以下式子计算：

$$z_i^{(l+1)} = \sum_{j=1}^{s_l} W_{ij}^{(l)} a_j^{(l)} + b_i^{(l)}$$

再进一步的对上面的式子进行变形：

$$\begin{aligned}
&\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b; x^{(k)}, y^{(k)}) \\
&= \frac{\partial J(W, b; x^{(k)}, y^{(k)})}{\partial z_i^{(l+1)}} \cdot \frac{\partial z_i^{(l+1)}}{\partial W_{ij}^{(l)}} \\
&= \frac{\partial J(W, b; x^{(k)}, y^{(k)})}{\partial z_i^{(l+1)}} \cdot a_j^{(l)}
\end{aligned}$$

同样的，对于 $b_i^{(l)}$ 的偏导数：

$$\begin{aligned}
&\frac{\partial}{\partial b_i^{(l)}} J(W, b; x^{(k)}, y^{(k)}) \\
&= \frac{\partial J(W, b; x^{(k)}, y^{(k)})}{\partial z_i^{(l+1)}} \cdot \frac{\partial z_i^{(l+1)}}{\partial b_i^{(l)}} \\
&= \frac{\partial J(W, b; x^{(k)}, y^{(k)})}{\partial z_i^{(l+1)}}
\end{aligned}$$

残差的定义

接下来我们定义：

$$\delta_i^{(l)} = \frac{\partial}{\partial z_i^{(l)}} J(W, b; x^{(k)}, y^{(k)})$$

为第k个样本在第l层第i个神经元上产生的残差。再次回顾我们的参数更新公式：
对于 $W_{ij}^{(l)}$ 我们有：

$$
\begin{aligned}
W_{ij}^{(l)} &= W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b) \\
&= W_{ij}^{(l)} - \alpha \left[\left(\frac{1}{K} \sum_{k=1}^{K} \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b;x^{(k)},y^{(k)}) \right) + \lambda W_{ij}^{(l)} \right] \\
&= W_{ij}^{(l)} - \alpha \left[\left(\frac{1}{K} \sum_{k=1}^{K} \frac{\partial J(W,b;x^{(k)},y^{(k)})}{\partial z_i^{(l+1)}} \cdot a_j^{(l)} \right) + \lambda W_{ij}^{(l)} \right] \\
&= W_{ij}^{(l)} - \alpha [(\frac{1}{K} \sum_{k=1}^{K} \delta_i^{(l+1)} \cdot a_j^{(l)}) + \lambda W_{ij}^{(l)}]
\end{aligned}
$$

类似的，对于 $b_i^{(l)}$ 我们有：

$$
\begin{aligned}
b_i^{(l)} &= b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W,b) \\
&= b_i^{(l)} - \alpha \frac{1}{K} \sum_{k=1}^{K} \frac{\partial}{\partial b_i^{(l)}} J(W,b;x^{(k)},y^{(k)}) \\
&= b_i^{(l)} - \alpha \frac{1}{K} \sum_{k=1}^{K} \frac{\partial J(W,b;x^{(k)},y^{(k)})}{\partial z_i^{(l+1)}} \\
&= b_i^{(l)} - \alpha \frac{1}{K} \sum_{k=1}^{K} \delta_i^{(l+1)}
\end{aligned}
$$

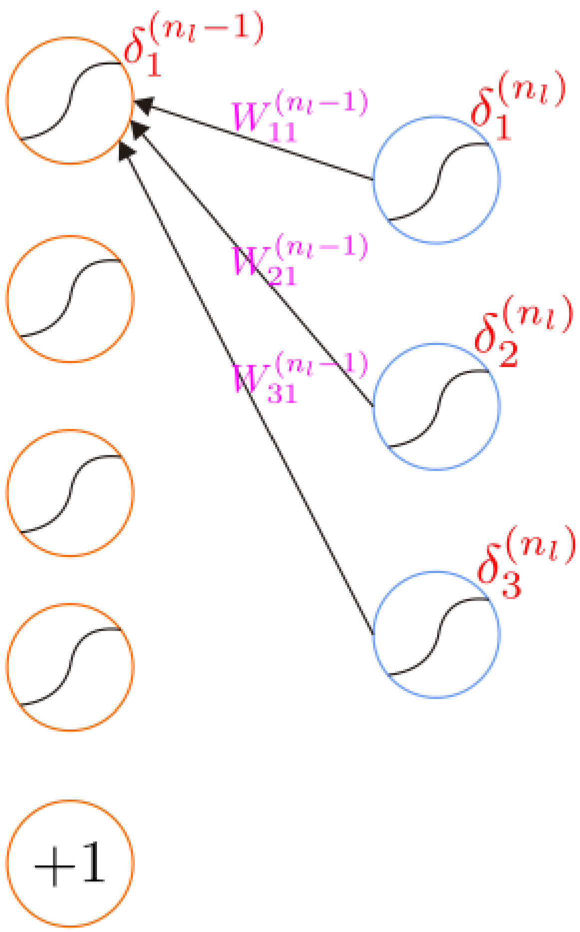
现在的核心问题只剩下一个了，这个残差该如何求？
我们先计算最后一层第 i 个神经元上的残差，这里为了简单起见，不再指定为第 k 个样本。

$$
\begin{aligned}
\delta_i^{(n_l)} &= \frac{\partial}{\partial z_i^{(n_l)}} J(W,b;x,y) \\
&= \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \| h_{W,b}(x) - y \|^2 \\
&= \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \sum_{j=1}^{s_{n_l}} (y_j - a_j^{(n_l)})^2 \\
&= \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \sum_{j=1}^{s_{n_l}} (y_j - f(z_j^{(n_l)}))^2 \\
&= -(y_i - f(z_i^{(n_l)})) f'(z_i^{(n_l)})
\end{aligned}
$$

然后计算倒数第二层即第 $n_l - 1$ 层第 i 个神经元的残差：

$$
\begin{aligned}
\delta_i^{(n_l-1)} &= \frac{\partial}{\partial z_i^{(n_l-1)}} J(W,b;x,y) \\
&= \frac{\partial}{\partial z_i^{(n_l-1)}} \frac{1}{2}\sum_{j=1}^{s_{n_l}} (y_j - a_j^{(n_l)})^2 \\
&= \frac{1}{2}\sum_{j=1}^{s_{n_l}} \frac{\partial}{\partial z_i^{(n_l-1)}} (y_j - f(z_j^{(n_l)}))^2 \\
&= \sum_{j=1}^{s_{n_l}} -(y_j - f(z_j^{(n_l)})) \frac{\partial}{\partial z_i^{(n_l-1)}} f(z_j^{(n_l)}) \\
&= \sum_{j=1}^{s_{n_l}} -(y_j - f(z_j^{(n_l)})) f'(z_j^{(n_l)}) \frac{\partial z_j^{(n_l)}}{\partial z_i^{(n_l-1)}} \\
&= \sum_{j=1}^{s_{n_l}} \delta_j^{(n_l)} \frac{\partial}{\partial z_i^{(n_l-1)}} \sum_{q=1}^{s_{n_l}} W_{jq}^{(n_l-1)} f(z_q^{(n_l-1)}) \\
&= \sum_{j=1}^{s_{n_l}} W_{ji}^{(n_l-1)} \delta_j^{(n_l)} f'(z_i^{(n_l-1)})
\end{aligned}
$$

下面是残差传播的示意图：



从这里可以看出紧挨着的两层神经元之间的残差是有关系的，这也是反向传播的由来。更一般的，可以将上述关系表述为：

$$\delta_i^{(l)} = \sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} f'(z_i^{(l)})$$

再再次回顾我们的参数更新公式：

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \left[\left(\frac{1}{K} \sum_{k=1}^{K} \delta_i^{(l+1)} \cdot a_j^{(l)} \right) + \lambda W_{ij}^{(l)} \right]$$
$$b_i^{(l)} = b_i^{(l)} - \alpha \frac{1}{K} \sum_{k=1}^{K} \delta_i^{(l+1)}$$

我们需要先计算输出层神经元的残差，然后一级一级的计算前一层的神经元的残差，利用这些残差就可以更新神经网络参数了。

向量化表示

这里我们尝试将上述结果表示成向量或矩阵的形式，比如我们希望能一次性更新某一层神经元的权值和偏置，而不是一个一个的更新。
$\delta_i^{(l+1)}$ 表示的是第 $l+1$ 层第 i 个神经元的残差，那么整个第 $l+1$ 层神经元的偏差是多少呢？

$$\delta_i^{(l+1)} = \sum_{j=1}^{s_{l+2}} W_{ji}^{(l+1)} \delta_j^{(l+2)} f'(z_i^{(l+1)})$$

从而得到：

$$\delta^{(l+1)} = (W^{(l+1)})^T \delta^{(l+2)} \cdot f'(z^{(l+1)})$$

注：这里的 $\cdot$ 是指点乘，即对应元素相乘，$\delta^{(l+1)}$ 是一个 $s_{l+1} \times 1$ 维的列向量。

$a_j^{(l)}$ 表示第 l 层第 j 个神经元的输出，因此整个第 l 层的神经元的输出可用 $a^{(l)}$ 表示，是一个 $s_l \times 1$ 维的列向量。

因此对于矩阵 $W^{(l)}$ 来说，我们记：

$$\nabla_{W^{(l)}} J(W, b; x, y) = \delta^{(l+1)} (a^{(l)})^T$$

我们将 $\Delta W^{(l)}$ 初始化为 0，然后对所有 K 个样本将它们的 $\nabla_{W^{(l)}} J(W, b; x, y)$ 累加到 $\Delta W^{(l)}$ 中去：

$$\Delta W^{(l)} := \Delta W^{(l)} + \nabla_{W^{(l)}} J(W, b; x, y)$$

然后更新一次$W^{(l)}$：

$$W^{(l)} = W^{(l)} - \alpha[(\frac{1}{K}\Delta W^{(l)}) + \lambda W^{(l)}]$$

这里再强调一下：上式中的 $\Delta W^{(l)}$ 是所有 K 个样本的 $\delta^{(l+1)}(a^{(l)})^T$ **累加**和，如果希望做随机梯度下降了或者是mini-batch，这里就不用把所有样本的残差加起来了。

类似的，令：

$$\nabla_{b^{(l)}} J(W, b; x, y) = \delta^{(l+1)}$$

我们将 $\Delta b^{(l)}$ 初始化为 0， 然后对所有 K 个样本将它们的 $\nabla_{b^{(l)}} J(W, b; x, y)$ 累加到 $\Delta b^{(l)}$ 中去

$$\Delta b^{(l)} := \Delta b^{(l)} + \nabla_{b^{(l)}} J(W, b; x, y)$$

于是有：

$$b^{(l)} = b^{(l)} - \alpha[\frac{1}{K}\Delta b^{(l)}]$$

同样的，上式中的 $\Delta b^{(l)}$ 是所有 K 个样本的 $\delta^{(l+1)}$ **累加**和。

小结

上面的推导过程尝试把所有的步骤都写出来了，个人感觉比UFLDL上的教程更为详尽，只要你耐心看总能看得懂的。当然这篇文章有些细节并未作说明，比如惩罚因子的作用，为什么没有对偏置进行规则化，激活函数的选择等，这些都可以在UFLDL中找到答案，对应的链接在下面的参考中给出。

参考

[1] 神经网络
[2] 反向传导算法