BP 算法

Li Liang*

1 BP 算法

对训练例 (x_k, y_k)，假定神经网络的输出为 $\hat{y_k}$，则神经网络在 (x_k, y_k) 上的均方误差为：

$$E_k = \frac{1}{2}(\hat{y_k} - y_k)^2 \tag{1}$$

其中，

$$\hat{y_k} = f(\sum_{h=1}^{q} wb - \theta) \tag{2}$$

其中，w 为隐层与输出层之间的连接权，b 为隐层的输出，θ 为输出层的阈值，q 为隐层的神经元个数。$f = \frac{1}{1+e^{-x}}$ 为 Sigmoid 函数，具有 $\frac{\partial f}{\partial x} = f(x)(1 - f(x))$ 性质。

BP 算法是基于梯度下降策略，以目标的负梯度方法对参数进行调整。下面以 w 的求解为例讲解过程。

$$\begin{aligned} w &\leftarrow w + \Delta w \\ \Delta w &= -\eta \frac{\partial E_k}{\partial w} \end{aligned} \tag{3}$$

其中，

$$\frac{\partial E_k}{\partial w} = \frac{\partial E_k}{\partial \hat{y_k}} \frac{\partial \hat{y_k}}{\partial (\sum_{h=1}^{q} wb - \theta)} \frac{\partial (\sum_{h=1}^{q} wb - \theta)}{\partial w} \tag{4}$$

*https://github.com/leeliang/

其中，

$$\begin{aligned}
&\frac{\partial E_k}{\partial \hat{y}_k} = (\hat{y}_k - y_k) \\
&\frac{\partial \hat{y}_k}{\partial(\sum_{h=1}^{q} wb - \theta)} = \frac{\partial f}{\partial x} = \hat{y}_k(1 - \hat{y}_k) \\
&\frac{\partial(\sum_{h=1}^{q} wb - \theta)}{\partial w} = b
\end{aligned} \tag{5}$$

则，

$$\Delta w = \eta \hat{y}_k(1 - \hat{y}_k)(y_k - \hat{y}_k)b \tag{6}$$

类似可得 $\Delta\theta, \Delta v, \Delta\gamma$，$\Delta v, \Delta\gamma$ 分别为输入层到隐层的连接权和隐层的阈值。

上面介绍的标准 BP 算法每次仅针对一个训练样本更新连接权和阈值。如果类似推导出基于累计误差最小化的更新规则就得到了累计误差逆传播。一般来说，标准 BP 算法参数更新非常频繁，对不同样本可能出现“抵消”现象。累计误差在下降一定程度后，进一步下降会非常缓慢，这时标准 BP 往往会更快的获得较好的解。

由于 BP 神经网络的强大表示能力，经常遭遇过拟合。有两种策略常用来缓和过拟合：

- 早停：将数据分为训练集和验证集，若训练集误差降低但验证机误差升高，则停止训练，同时返回具有最小验证集误差的结果；
- 正则化。