1.2 关于正向传播中函数值溢出问题；

在神经网络的正向传播中，常会在正、反向传播中遇到与指数相关的激活函数或导数的运算。在这种情况下，若函数对应输入值较大（正）/小（负），如某一全连接神经网络的某一层使用了 $Sigmoid$ 激活函数：

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

而该层神经元的前一层神经网络的神经元数量较大，导致该层的某个神经元有一个很小的输入，例如-308；这种情况下，虽然直观上很容易得到 $f(-308)$ 的值应几乎为 0; 但对计算机来说需要计算 e^{308} 此时其结果很可能已经超过所定义浮点数的最大值，导致计算结果异常。

我们考虑以下的计算方式：

Sigmoid:

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

Sigmoid 导函数：

$$f'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$$
$$= \frac{e^x}{(1+e^x)^2}$$

这里记：

$$f_1(x) = g(-x) \tag{1}$$

$$f_2(x) = g(x) \tag{2}$$

于是，当 x 是正数时使用（1）式计算，若 x 是负数则使用（2）式计算；进一步考虑 x 是 $n \times m$ 的矩阵时的情形:

首先，使用 $Relu(x)$ 将矩阵 x 中的正负数分离，并分别按如上规则计算：

$$Relu(x) \to f_1(x)$$

$$-Relu(-x) \to f_2(x)$$

注意到，使用 $Relu$ 函数分离正/负数时，被分离的负/正数将化为 0，故对于结果：

$$f_1\big(relu(x)\big) + f_2\big(-relu(-x)\big)$$

将会比实际值大 $f_1(0) = f_2(0)$ 因此，实际值应当为：

$$result = f_1\big(relu(x)\big) + f_2\big(-relu(-x)\big) - \frac{1}{2}$$

对于 sigmoid 函数的导函数，实际值应为：

$$result = f_1\big(relu(x)\big) + f_2\big(-relu(x)\big) - \frac{1}{4}$$

Softmax:

$$f(x) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

令 $y_i = x_i - c, i = 1,2,\dots,n$，于是

$$f(y_i) = \frac{e^{x_i - c}}{\sum_j e^{x_j - c}}$$

$$= \frac{e^{-c} \cdot e^{x_i}}{e^{-c} \cdot \sum_j e^{x_j}}$$

$$= \frac{e^{x_i}}{\sum_j e^{x_j}}$$

$$= f(x_i)$$

即对于向量 x，将其所有元素减去一个常量，其对应 softmax 函数的输出值不变；于是可以考虑令：

$$c = \max(x)$$

这种情况下，$x_i - c \le 0, \forall x_i \in x$;