成都大学
CHENGDU UNIVERSITY

《人工智能导论》
实验指导书

IntrotoAI Labscript by YYANG

杨耀如

成都大学 电子信息与电气工程学院

实验三　神经网络

一、实验目的

掌握用 MATLAB 实现神经网络功能的方法，加深对神经网络前向和反向传播的理解。

二、实验内容

编写 MATLAB 代码，实现利元神经网络自动识别手写数字的功能。

三、实验设备

1、计算机

2、MATLAB 软件

四、实验原理

1、代码模版

本实验使用的 MATLAB 代码模版共包含以下文件：

（1） lab3.m 本文件会指引你完成本实验

（2） lab3data.mat 手写数字数据集

（3） lab3weights.mat 神经网络的参数集

（4） datademonstrate.m 数据集可视化函数

（5） fmincg.m 实现求函数最小值功能

（6） sgmd.m sigmoid 函数

（7） numericalgrad.m 数值化计算梯度

（8） gradcheck.m 检验计算所得梯度准确性的函数

（9） initialiseweights.m 初始化权重的函数

（10） forecast.m 神经网络预测函数

（11） sgmdgrad.m 计算 sigmoid 函数梯度的函数

（12） rdinitweights.m 使用随机数初始话权重

（13） nncfunction.m 神经网络的代价函数

其中，（11）-（13）需要补全

2、神经网络模型表示

本实验使用的神经网络模型如图 1 所示。该网络有 3 层，一个输入层，一个隐藏层，一个输出层。本实验使用的每幅输入图像由$20 \times 20 = 400$个像素点组成，每个像素点由一个像素值表示，因此，神经网络的输入层由 400 个神经元组成。

本实验中神经网络的权重$(\theta^{(1)}, \theta^{(2)})$由 lab3weights.mat 文件给出，会由 lab3.m 导入矩阵 Theta1 和 Theta2 中。网络参数的维度与神经网络的隐藏层和输出层的神经元个数吻合。

3、前向传播与代价函数

完成 nncfunction.m 文件，实现神经网络代价函数的计算功能。

神经网络的代价函数如下式：

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log((h_\theta(x^{(i)}))_k) - (1-y_k^{(i)})\log(1-(h_\theta(x^{(i)}))_k)\right]$$

Photo AI Labscript by YYANG

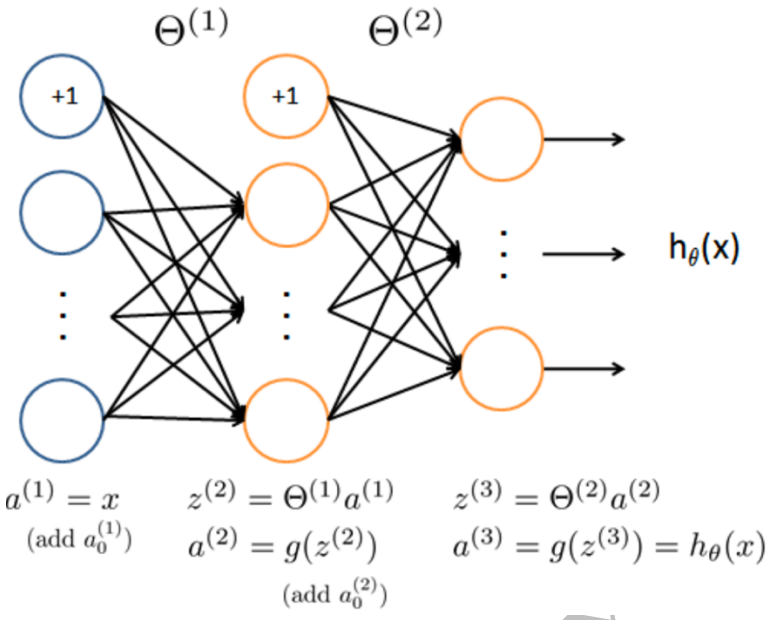

图 1 神经网络结构

式中$h_\theta(x^{(i)})$如何计算见图 1，$K = 10$为总的标签数。注意$h_\theta(x^{(i)}) = a_k^{(3)}$是第 k 个输出神经元的的激活函数值。

当你完成代价函数的代码部分后，lab3.m 会调用 nncfunction。你能从命令窗口的打印信息看到代价值约为.287629。

为了防止过拟合，通常可以在代价函数后加入正则项来达到正则化代价函数的目的：

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log((h_\theta(x^{(i)}))_k) - (1-y_k^{(i)})\log(1-(h_\theta(x^{(i)}))_k)\right] + \frac{\lambda}{2m}\left[\sum_{j=1}^{25}\sum_{k=1}^{400}(\Theta_{j,k}^{(1)})^2 + \sum_{j=1}^{10}\sum_{k=1}^{25}(\Theta_{j,k}^{(2)})^2\right].$$

改进你的代价函数代码，实现代价函数的正则化，需要注意的是，权重参数中与引入的偏差项关联的系数不应出现在正则项中，对举证 Theta1 和 Theta2 来说，就是它们的第一列。

当你完成代价函数正则化的代码部分后，lab3.m 会调用 nncfunction。当$\lambda = 1$时，你能从命令窗口的打印信息看到代价值约为.383770。

4、参数随机初始化

在训练神经网络时，使用随机值初始化参数十分重要。一个有效的随机初始化策略是将当前层的参数$\theta^{(l)}$均匀地随机初始化为范围$[-\epsilon_{init}, \epsilon_{init}]$中的值。在本实验中，可以使用$\epsilon_{init} = .12$。这一范围内的取值会使参数值保持较小的情况下提升学习的效率。

完成 rdinitweights.m 文件中的初始化权重θ的功能，编辑该文件，加入如下代码片段：

```
% Randomly initialize the weights to small values
epsilon_init = 0.12;
W = rand(L_out, 1 + L_in) * 2 * epsilon_init - epsilon_init;
```

5、反向传播

反向传播，如图 2 所示，其原理在于，对于给定的训练样本$(x^{(t)}, y^{(t)})$，首先使用前向

传播计算输出值$h_\theta(x)$。然后对于第l层中的第j个节点，计算“误差值”$\delta_j^{(l)}$，这一误差值表征了当前节点对于输出结果的误差贡献了多少份额。

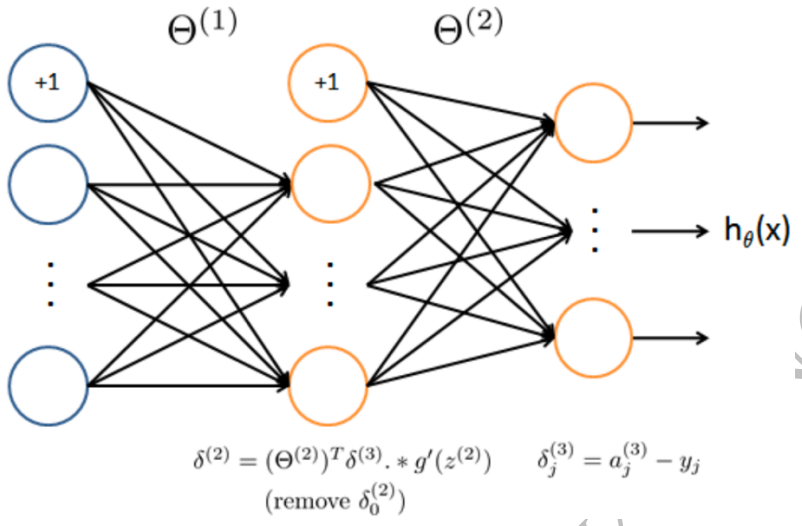

图 2 误差反向传播原理

反向传播算法总结如下：

（1） 将输入层的值$a^{0(1)}$设置为第 t 个悬链样本$x^{(t)}$，如图一所示进行前向传播，计算第 2 层和第 3 层的$(z^{(2)}, a^{(2)}, z^3, a^{(3)})$；

（2） 对于每一个位于输出层的神经元，令：

$$\delta_k^{(3)} = (a_k^{(3)} - y_k)$$

其中$y_k \in \{0,1\}$表示当前训练样本是否属于类k $(y_k = 1)$，或是属于其他的类；

（3） 对于隐藏层$l = 2$，令：

$$\delta^{(2)} = \left(\Theta^{(2)}\right)^T \delta^{(3)} .* g'(z^{(2)})$$

（4） 按照下式更新各层梯度的累积，注意跳过$\delta_0^{(2)}$：

$$\Delta^{(l)} = \Delta^{(l)} + \delta^{(l+1)}(a^{(l)})^T$$

（5） 通过将梯度累积除以$\frac{1}{m}$，获得未正则化的神经网络代价函数的梯度：

$$\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)} = \frac{1}{m}\Delta_{ij}^{(l)}$$

6、梯度检查

numericalgrad.m 中实现了梯度的数值化计算，在 lab3.m 的下一步中，会调用 gradcheck.m，该文件会生成一个小型神经网络和一些数据，该小型神经网络和数据会被用来测试梯度的计算是否正确，运行这一步，在命令窗口可以观察到，数值计算梯度的误差小于$1e-9$。

7、正则化神经网络

在考虑到正则化的情况下，可以对梯度计算公式做如下修改：

IntrotoAI Labscript by Y

$$\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)} = \frac{1}{m} \Delta_{ij}^{(l)} \qquad \text{for } j = 0$$

$$\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)} = \frac{1}{m} \Delta_{ij}^{(l)} + \frac{\lambda}{m} \Theta_{ij}^{(l)} \qquad \text{for } j \geq 1$$

按照这个思路修改 nncfunction 中的 grad 的计算，当完成后，lab3.m 会执行梯度检查来检查增加正则项后的数值计算梯度，在命令窗口可以观察到，数值计算梯度的误差仍旧小于$1e-9$。

8、使用 fmincg 学习参数

在实现了神经网络的代价函数和梯度的梯度后，lab3.m 执行的下一步是使用 fmincg 来学习出一组好的模型参数。在训练结束后，lab3.m 会报告训练准确率，在代价函数实现正确的情况下，准确率约为 95.3%左右（由于使用随机值初始化策略，准确率可能有 1%左右的出入）。增加神经网络的训练轮次和调整正则化参数λ的值可能会得到正确率更高的模型，可自行尝试。

9、隐藏层可视化

继续运行 lab3.m 会调用 datademonstarte.m 来将隐藏层的信息可视化为 25 个图像，如图 3 所示，每个图像于隐藏层的一个神经元相对应。通过观察图像，可以发现，隐藏层直观上的作用类似于一个检测笔画和输入图像中其他固定模式的探测器。

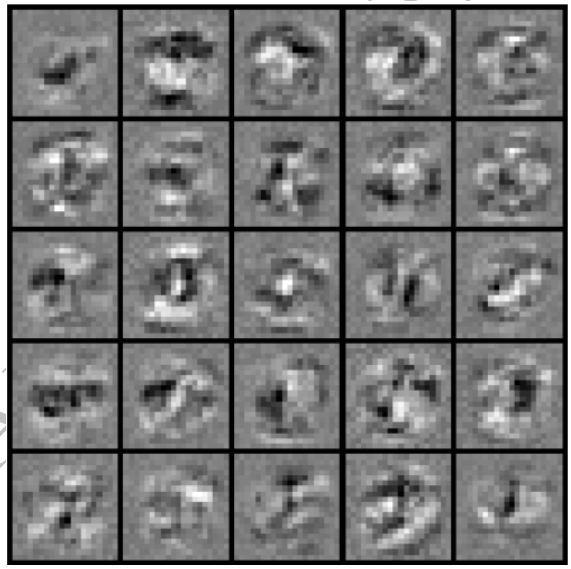

图 3 隐藏层可视化结果

五、实验要求

1、完成四.1（11）-（13）的 MATLAB 代码；

2、绘制必要的图像；

3、分析实验结果。