模式识别：作业 #4

201928014629008
牛李金梁

2019 年 11 月 23 日

第一部分：计算与证明

Question 1

(a)

记第 k 个样本 $\boldsymbol{x}^k = \left[1, x_1^k, x_2^k, \cdots, x_d^k\right]^T$。设隐含层有 n 个神经元，输入层到隐含层的连接权重为 $(d+1)\times n$ 维的矩阵 W^H，其中某个元素记为 w_{ih}；输出层有 c 个神经元，隐含层到输出层的连接权重为 $(n)\times c$ 维的矩阵 W^c，其中某个元素记 w_{hj}。

在前馈过程中，隐含层 h 结点的输出为 $y_h^k = f_1\left(net_h^k\right) = f\left(\sum_i w_{ih}x_i^k\right)$，$f$ 为 sigmoid 函数；输出层 j 结点的输出为 $z_j^k = f_2\left(net_j^k\right) = f\left(\sum_h w_{hj}y_h^k\right) = f_2\left(\sum_h w_{hj}f_1\left(\sum_i w_{ih}x_i^k\right)\right)$，$f$ 为 softmax 函数。

误差函数为

$$\begin{aligned} E(w)^k = J(w)^k &= \frac{1}{2}\sum_j \left(t_j^k - z_j^k\right)^2 \\ &= \frac{1}{2}\sum_j \left(t_j^k - f\left(net_j^k\right)\right)^2 \\ &= \frac{1}{2}\sum_j \left(t_j^k - f\left(\sum_h w_{hj}f\left(net_h^k\right)\right)\right)^2 \end{aligned}$$

隐含层到输出层的连接权重调节量：

$$\begin{aligned} \Delta w_{hj} = -\eta\frac{\partial E}{\partial w_{hj}} &= -\eta\sum_k \frac{\partial E}{\partial net_j^k}\frac{\partial net_j^k}{\partial w_{hj}} \\ &= \eta\sum_k \left(t_j^k - z_j^k\right) f'\left(net_j^k\right) y_h^k \\ &= \eta\sum_k \delta_j^k y_h^k \end{aligned}$$

其中，$\delta_j^k = -\frac{\partial E}{\partial net_j^k} = f'\left(net_j^k\right)\left(t_j^k - z_j^k\right)$，$f$ 为 softmax 函数。softmax 的导数为雅可比矩阵，此处需要的是矩阵主对角线上的元素。设 softmax 主对角线上的第 j 个元素是 S_j，有 $S_j{}' = S_j\left(1 - S_j\right)$。

因此，$\Delta w_{hj} = \eta\sum_k \left(t_j^k - z_j^k\right) S_j\left(1 - S_j\right) y_h^k$，其中，$S_j = \frac{\exp\left(net_j^k\right)}{\sum_i \exp\left(net_i^k\right)}$

输入层到隐含层的连接权重调节量：

$$\begin{aligned}
\Delta w_{ih} = -\eta\frac{\partial E}{\partial w_{ih}} &= -\eta\sum_{k,j}\frac{\partial E}{\partial z_j^k}\frac{\partial z_j^k}{\partial w_{ih}} \\
&= \eta\sum_{k,j}\left(t_j^k - z_j^k\right)\frac{\partial z_j^k}{\partial w_{ih}} \\
&= \eta\sum_{k,j}\left(t_j^k - z_j^k\right)\frac{\partial z_j^k}{\partial net_j^k}\frac{\partial net_j^k}{\partial w_{ih}} \\
&= \eta\sum_{k,j}\left(t_j^k - z_j^k\right)f'\left(net_j^k\right)\frac{\partial net_j^k}{\partial y_h^k}\frac{\partial y_h^k}{\partial w_{ih}} \\
&= \eta\sum_{k,j}\left(t_j^k - z_j^k\right)f'\left(net_j^k\right)w_{hj}\frac{\partial y_h^k}{\partial w_{ih}} \\
&= \eta\sum_{k,j}\left(t_j^k - z_j^k\right)f'\left(net_j^k\right)w_{hj}\frac{\partial y_h^k}{\partial net_h^k}\frac{\partial net_h^k}{\partial w_{ih}} \\
&= \eta\sum_{k,j}\left(t_j^k - z_j^k\right)f'\left(net_j^k\right)w_{hj}f'\left(net_h^k\right)x_i^k \\
&= \eta\sum_{k}\delta_h^k x_i^k
\end{aligned}$$

其中，$\delta_h^k = -\frac{\partial E}{\partial net_h^k} = f'\left(net_h^k\right)\sum_j w_{hj}\delta_h^k$，$\delta_j^k = f'\left(net_j^k\right)\left(t_j^k - z_j^k\right)$。

sigmoid 函数的导数 $S_h{}' = S_h\left(1 - S_h\right)$，于是，$\Delta w_{ih} = \sum_j w_{hj}\delta_j^k S_h\left(1 - S_h\right)x_i^k$，$S_h = \frac{1}{1+\exp\left(-net_h^k\right)}$

(b)

通过 (a) 中的推导可以看出，反向传播算法将各层间的连接权重分隔开，使之相互独立，传播时输出层的误差沿着前馈网络的反方向逐层更新层间的连接权重，从而避免了深层网络中复合函数链式求导中过长的链。这一算法对简化神经网络优化过程有着重要的意义。

Question 2

计算步骤：

- ☐ 初始化网络。通常使用随机初始化的方法。
- ☐ 输入训练样本，每个样本为一个 d 维向量。
- ☐ 计算映射层的权重向量和输入向量的距离。$d_j = \sqrt{\sum_{i=1}^{d}\left(x_i - w_{ij}\right)^2}$。
- ☐ 选择与权重向量距离最小的神经元作为胜出神经元 j^*，并给出其邻接神经元集合 $h\left(., j^*\right)$
- ☐ 调整胜出神经元和其邻接神经元的权重，按下式更新：

$$\Delta w_{ij} = \eta h\left(j, j^*\right)\left(x_i - w_{ij}\right)$$

$$w_{ij}\left(t+1\right) = w_{ij}\left(t\right) + \Delta w_{ij}$$

□ 检查是否达到预设要求，是则结束算法，否则进行迭代。

算法流程图：

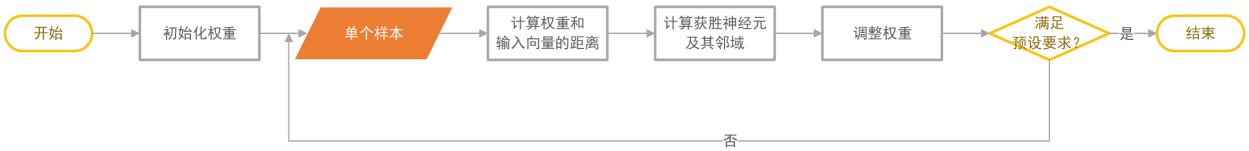

图 1：自组织算法流程图

Question 3

(1)

输入 图像大小：400×400

第一卷积层 滤波器大小：5×5

结点图像个数：20

图像大小：396×396

pooling 后图像大小：198×198

包含偏置的训练权重个数：$5 \times 5 \times 20 + 20 = 520$

第二卷积层 滤波器大小：$3 \times 3 \times 20$

结点图像个数：30

滤波器总数：30

图像大小：196×196

pooling 后图像大小：98×98

包含偏置的训练权重个数：$20 \times 3 \times 3 \times 30 + 30 = 5430$

第三卷积层 滤波器大小：$3 \times 3 \times 30$

结点图像个数：20

滤波器总数：20

图像大小：96×96

pooling 后图像大小：48×48

包含偏置的训练权重个数：$30 \times 3 \times 3 \times 20 + 20 = 5420$

第四卷积层 滤波器大小：$3 \times 3 \times 10$

结点图像个数：10

滤波器总数：10

特征图尺度：46×46

pooling 后图像大小：23×23

包含偏置的训练权重个数：$20 \times 3 \times 3 \times 10 + 10 = 1810$

全连接层 输入图像大小：23×23

输入图像个数：10

输出结点个数：10

包含偏置的训练权重个数：$10 \times 23 \times 23 \times 10 + 10 = 52910$

采用权值共享和局部连接的总权重个数为 65290。

采用全连接网络：

输入层到第一隐层：$400 \times 400 \times 396 \times 396 \times 20 + 396 \times 396 \times 20 = 501814336320$
第一隐层到第二隐层：$198 \times 198 \times 20 \times 196 \times 196 \times 30 + 196 \times 196 \times 30 = 903637670880$
第二隐层到第三隐层：$98 \times 98 \times 30 \times 96 \times 96 \times 20 + 96 \times 96 \times 20 = 53106462720$
第三隐层到第四隐层：$48 \times 48 \times 20 \times 46 \times 46 \times 10 + 46 \times 46 \times 10 = 975073960$
第四隐层到输出层：$23 \times 23 \times 10 \times 10 + 10 \times 10 = 53000$
总权重个数约为 1.459×10^{12}。

不采用权值共享：

输入层到第一隐层：$5 \times 5 \times 396 \times 396 \times 20 + 396 \times 396 \times 20 = 105304320$
第一隐层到第二隐层：$3 \times 3 \times 20 \times 196 \times 196 \times 30 + 196 \times 196 \times 30 = 208598880$
第二隐层到第三隐层：$3 \times 3 \times 30 \times 96 \times 96 \times 20 + 96 \times 96 \times 20 = 49950720$
第三隐层到第四隐层：$3 \times 3 \times 20 \times 46 \times 46 \times 10 + 46 \times 46 \times 10 = 3829960$
第四隐层到输出层：$23 \times 23 \times 10 \times 10 + 10 \times 10 = 53000$
总权重个数约为 3.67×10^{8}。

与二者相比，采用权值共享和局部连接的总权重个数基本可以忽略。

(2)

max pooling 的前向传播是舍去其他元素，把区域内最大的值传给后一层。反向传播则把梯度直接传给前一层该区域的某一个像素，而其他像素为 0。确定这个像素需要记录下 pooling 时取的是哪个像素，即最大值所在位置。

(3)

能改变网络结构的因素：

- ☐ 改变网络层数
- ☐ 采用不同大小的滤波器
- ☐ 激励函数的选择
- ☐ 池化操作的选择
- ☐ 能量函数的选择

第二部分：计算机编程

环境为 python3.7
依赖库：numpy、matplotlib
运行方式：直接在 cmd 中执行相应 py 文件

Question 1

使用 Batch perception 算法训练判别函数，首先要对样本进行规范化增广，然后利用这些样本训练线性判别函数。

具体的训练过程需要将每一步错分的点相加并乘以步长来修正权向量 $\boldsymbol{a}$，如此迭代直至完全分开或修正值很小（以应对线性不可分的情况）。

实验中，取初始权向量 $\boldsymbol{a}=\mathbf{0}$，$\eta=1$，$\theta=0.01$，结果如下图：

图 2: Batch perception 算法结果

问题 (a) 中分类 ω_1 和 ω_2 的结果如左图所示，判别函数左上判别为 ω_1，右下判别为 ω_2，收敛步数为 23。

问题 (b) 中分类 ω_3 和 ω_2 的结果如右图所示，判别函数左上判别为 ω_3，右下判别为 ω_2，收敛步数为 16。

Question 2

Ho-Kashyap 算法通过最小化误差来优化 b，并通过 $a=Y^{+}b$ 计算对应的 a，进而得到一组 a,b 使 $Ya=b>0$。

搜索到最优解是比较困难的，需要设定最大迭代次数和收敛误差。本程序设定步长为 0.8，最大迭代次数为 2000，收敛误差为 0.01，若所有样本的误差小于 b_{min}（设定为 0.001）或迭代 2000 次，则停止迭代。

ω_1 和 ω_3 的分类结果如下图中左图所示，决策面左上方为 ω_3，右下方为 ω_1。可以看到有 2 个分类错误点，并且输出了 No solution found，说明该问题是线性不可分的。

ω_2 和 ω_4 的分类结果如下图中右图所示，决策面左下方为 ω_4，右上方为 ω_2。可以看出该问题是线性可分的，迭代次数为 899。

图 3: Ho-Kashyap 算法结果

Question 3

使用 MSE 多类扩展方法对 4 个类别进行分类。

具体的算法实现中，主要是要构造 $\boldsymbol{Y}$，并利用 $\hat{W}=(\hat{X}\hat{X}^T+\lambda I)^{-1}\hat{X}Y^T$ 来计算 $\hat{W}$ 确定分类区域，程序中设定 $\lambda=0.01$。

根据决策规则 $x\in\omega_i, g_i(x)=\max g_j(x), j=1,\ldots,4$ 可以绘制出决策区域（本程序直接利用点采样确定了决策区域，而非计算线性判别函数），

图中红色为 ω_1 决策区域，黄色为 ω_2 决策区域，绿色为 ω_3 决策区域，蓝色为 ω_4 决策区域。并以实心圆标记训练样本，x 标记测试样本，通过统计可以得到测试样本的分类正确率为 100%。

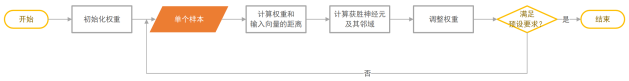

图 4：决策区域示意图