模式识别第四次作业

孙浩淼 201928013229100

Email: sunhm15@gmail.com

1 问题一

1.1 分析每一层梯度

设 f 表示全连接层，g 表示 sigmoid 激活层，S 表示 Softmax 激活层，J 表示平方误差损失函数。且 x_i 表示神经网络输入，net_h 表示第一层全连接层输出（未激励），x_h 表示隐层结果，net_j 表示网络输出（未激励），y_j 表示 softmax 输出，z 表示标签，E 表示网络的损失。

每一个函数定义如下：

$$J(t,z): \quad y = \frac{1}{2}\sum_{j=1}^{c}(t_j - z_j)^2 \tag{1}$$

$$S(y): \quad y_i = \frac{e^{x_i}}{\sum\limits_{j=1}^{c} e^{x_j}} \tag{2}$$

$$g(y): \quad y = \frac{1}{1+e^{-x}} \tag{3}$$

$$f(y): \quad y = Wx + b \tag{4}$$

网络的链接方式如下：

$$x_h = g(W_{ih}x_i + b_h), \quad net_j = W_{hj}x_h + b_j, \quad y_j = S(net_j), \quad L = J(y_j, z) \tag{5}$$

则每一个函数的梯度计算如下：

$$J(w): \quad \frac{\partial y}{\partial x} = x - z \tag{6}$$

$$S(x): \quad \frac{\partial y_j}{\partial x_i} = \begin{cases} y_j(1-y_j) & i == j \\ -y_j y_i & i \neq j \end{cases} \tag{7}$$

$$g(x): \quad \frac{\partial y}{\partial x} = y(1-y) \tag{8}$$

$$f(x): \quad \frac{\partial y_j}{\partial x_i} = w_{ij}, \quad \frac{\partial y_j}{\partial w_{ij}} = x_i, \quad \frac{\partial y_j}{\partial b_j} = 1 \tag{9}$$

因此，网络输出层的求导过程如下:

$$\frac{\partial E}{\partial y} = y - z \tag{10}$$

$$\frac{\partial E}{\partial net_j} = \sum_{m=1}^{c} \frac{\partial y_m}{\partial net_j} \frac{\partial E}{\partial y_m} = y_j(y_j - z_j) - \sum_{m=1}^{c} y_j y_m (y_m - z_m) \tag{11}$$

第二个隐层对应梯度如下:

$$\Delta w_{hj} = \eta \frac{\partial E}{\partial w_{hj}} = \eta \sum_k \frac{\partial E}{\partial net_j^k} \frac{\partial net_j^k}{\partial w_{hj}} = \eta \sum_k x_h^k \left[y_j^k (y_j^k - z_j^k) - \sum_{m=1}^{c} y_j^k y_m^k (y_m^k - z_m^k) \right] \tag{12}$$

$$\Delta b_j = \eta \frac{\partial E}{\partial b_j} = \eta \sum_k \frac{\partial E}{\partial net_j^k} \frac{\partial net_j^k}{\partial b_j} = \eta \sum_k \left[y_j^k (y_j^k - z_j^k) - \sum_{m=1}^{c} y_j^k y_m^k (y_m^k - z_m^k) \right] \tag{13}$$

$$\frac{\partial E}{\partial x_h} = \sum_{j=1}^{c} \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial x_h} = \sum_{k,j} w_{hj} \left[y_j^k (y_j^k - z_j^k) - \sum_{m=1}^{c} y_j^k y_m^k (y_m^k - z_m^k) \right] \tag{14}$$

第一个隐层对应梯度如下:

$$\begin{aligned} \Delta w_{ih} &= \eta \frac{\partial E}{\partial w_{ih}} = \eta \sum_k g'(net_h^k) \frac{\partial E}{\partial x_h} \frac{\partial net_h^k}{\partial w_{ih}} \\ &= \eta \sum_{k,j} x_i^k x_h^k (1 - x_h^k) w_{hj} \left[y_j^k (y_j^k - z_j^k) - \sum_{m=1}^{c} y_j^k y_m^k (y_m^k - z_m^k) \right] \end{aligned} \tag{15}$$

$$\begin{aligned} \Delta w_{ih} &= \eta \frac{\partial E}{\partial b_j} = \eta \sum_k g'(net_h^k) \frac{\partial E}{\partial x_h} \frac{\partial net_h^k}{\partial b_h} \\ &= \eta \sum_{k} x_h^k (1 - x_h^k) w_{hj} \left[y_j^k (y_j^k - z_j^k) - \sum_{m=1}^{c} y_j^k y_m^k (y_m^k - z_m^k) \right] \end{aligned} \tag{16}$$

其中，$g'(net_h^k) = x_h^k(1 - x_h^k)$ 为激活函数的导数。

1.2 总结

首先，计算误差对网络输出的导数，之后利用 δ 规则计算对权重的梯度，即

$$\Delta w_{ih} = \eta \sigma_j^k y_h^k = \eta f'(net_h^k) \Delta_h^k y_h^k$$

最后利用链式法则逐层反传梯度。

$$\Delta_h^k = \sum_j w_{hj} \delta_j^k$$

影响反向传播算法的主要因素有：隐藏层节点个数、激励函数、损失函数选择、初始参数选取、学习率 η 等。

2 问题二

2.1 计算步骤

S1 网络初始化（随机初始）

S2 输入向量

S3 计算映射层的权重向量与输出层的距离 d_j：

$$d_j = \sqrt{\sum_{i=1}^{d}(x_i - w_{ij})^2}$$

S4 计算并选择与权重向量距离最小的神经元，将其作为胜出神经元 j^*，并给出其临近神经元集合 $h(., j^*)$

S5 更新权重

$$\Delta w_{ij} = \eta h(j, j^*)(x_i - w_{ij}), \quad h(j, j^*) = \exp\left(-\|j - j^*\|^2/\sigma^2\right), \quad w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}$$

S6 检查是否达到预先设定的要求，否则返回 **S2**

2.2 算法框图

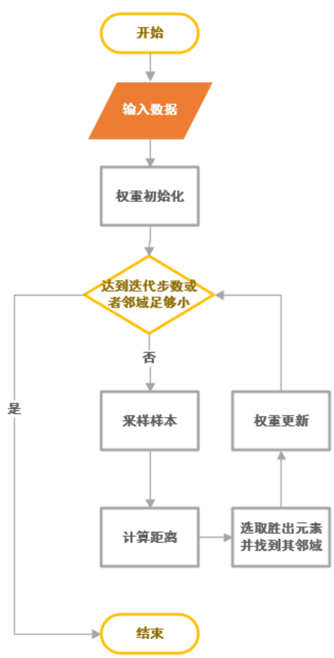


图 1 算法框图

3 问题三

3.1 权重数量计算

Layer	Input Size	Filter/Stride	Output Size	num of Params
Conv1	400×400×3	5×5/1	400×400×20	$5*5*3*20=1500$
Pool1	400×400×20	2×2/2	200×200×20	0
Conv2	200×200×20	3×3/1	200×200×30	$3*3*20*30=5400$
Pool2	200×200×30	2×2/2	100×100×30	0
Conv3	100×100×30	3×3/1	100×100×20	$3*3*30*20=5400$
Pool3	100×100×20	2×2/2	50×50×20	0
Conv4	50×50×20	3×3/1	50×50×10	$3*3*20*10=1800$
Pool4	50×50×10	2×2/2	25×25×10	0
FC	6250	-	10	$6250*10=62500$

表 1 参数估计

总参数数目为，

$$1500+5400+5400+1800+62500=76600$$

3.2 反传

$$\frac{\partial f}{\partial K(i,j)}=\frac{\partial f}{\partial K'(i/2,j/2)}*M(i,j)$$

其中，K(i,j) 表示 pooling 前的第 i 行第 j 列参数，K'(i/2,j/2) 表示 pooling 后其对应的像素位置，M(i,j) 表示是否为 2*2 方格中的最大值，只有最大值的位置反传，其他地方不反传。

3.3 结构改变

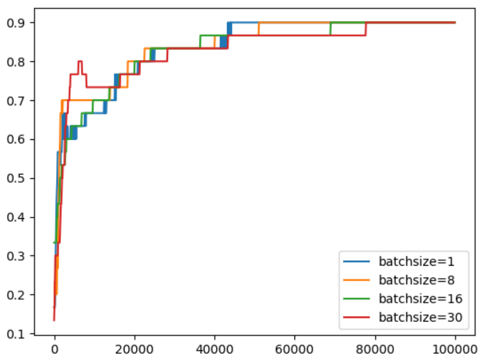
1) 5*5 换成 2 个 3*3

2) 增加卷积层个数

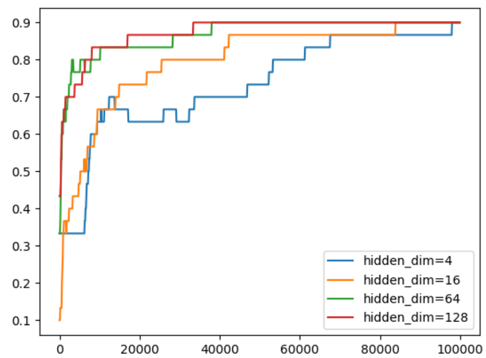
3）改变每层的 channel 数目

4 编程题

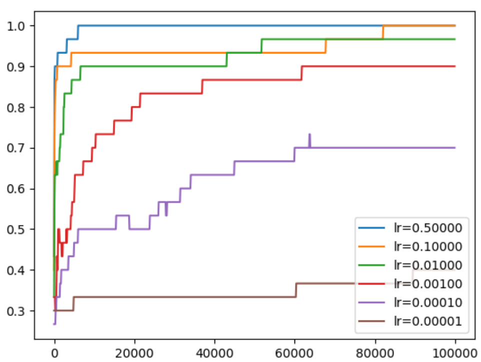
代码在 run.py 中，分别实现了多种因素的比较，实验结果如图 2 所示。



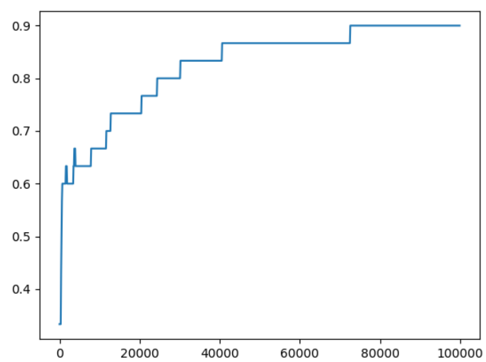
(a) 批处理大小对实验精度的影响



(b) 隐藏层数目对实验精度的影响



(c) 迭代步长对实验精度的影响



(d) 训练轮数对实验精度的影响

图 2 实验结果

4.1 批处理大小对实验精度的影响

根据图 2(a) 可以看到每一次处理的数据越多，梯度越稳定，单样本在初期精度会产生较大的震荡，收敛结果相同。

4.2 隐藏层数目对实验精度的影响

实验结果如图 2(b) 所示，可以看出隐藏层数目过少的时候收敛较为困难，而隐藏层过多的时候又会造成过拟合的现象（本题中由于实验数据过少，没有区分训练集测试集，因此显示不出该现象）

4.3 迭代步长对实验精度的影响

实验结果如图 2(c) 所示，增大迭代步长可以有效的提高收敛速度，但是过大的迭代步长可能使得模型难于收敛。

4.4 训练轮数对实验精度的影响

实验结果如图 2(d) 所示，随着训练轮数的增多，模型的精度也在不断地提高，但是在一定迭代次数之后，精度不再变化（又称为模型收敛了）。