**Pattern Recognition**

University of Chinese Academy of Sciences

Fall 2023

Shiming Xiang, Gaofeng Meng

# Homework 4

**Chenkai GUO**

2023.12.10

1. *Consider a three-layer network for classification with $n_H$ nodes in hidden layer, and $c$ nodes in output layer. The patterns (also say samples) are in $d$ dimensional space. The activation function (or transfer function) for the nodes in the hidden layer is the sigmoid function. Differently, the nodes in the output layer will employ the following softmax operation as their activation function:*

$$z_j = \frac{e^{net_j}}{\sum_{m=1}^{c} e^{net_m}},\ j = 1, 2, \cdots, c$$

*where $net_j$ stands for the weighted sum at the $j-th$ node in the output layer. Please derive the learning rule under the back propagation framework if the criterion function for each sample is the sum of the squared errors, that is* （即分析每一层权重的更新方法）*:*

$$J(\mathbf{w}) = \frac{1}{2}\sum_{j=1}^{c}(t_j - z_j)^2$$

注意：本题只需要推导出单个样本对权重更新的贡献即可（因为多个样本只是简单地相加）

由题可得*:*

$$\begin{aligned}
J(\mathbf{w}) &= \frac{1}{2}\sum_{j=1}^{c}(t_j - z_j)^2 = \frac{1}{2}\sum_{j=1}^{c}\left(t_j - softmax\big(net_j\big)\right)^2 \\
&= \frac{1}{2}\sum_{j=1}^{c}\left\{t_j - softmax\left(\sum_{h=1}^{n_H} w_{hj}y_h\right)\right\}^2 \\
&= \frac{1}{2}\sum_{j=1}^{c}\left\{t_j - softmax\left(\sum_{h=1}^{n_H} w_{hj}\cdot sigmoid\left(net_h\right)\right)\right\}^2 \\
&= \frac{1}{2}\sum_{j=1}^{c}\left\{t_j - softmax\left(\sum_{h=1}^{n_H} w_{hj}\cdot sigmoid\left(\sum_i w_{ih}x_i\right)\right)\right\}^2
\end{aligned}$$

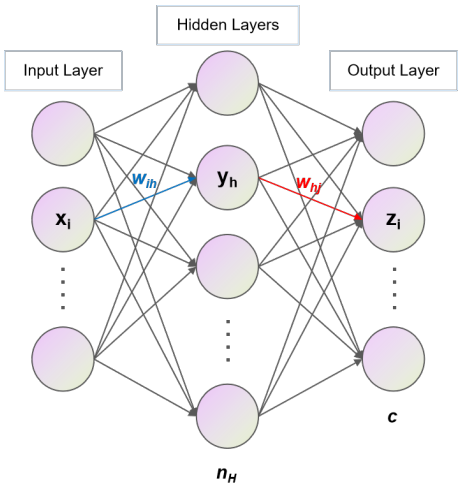


Fig. 1: Corresponding Network Architecture in Question

接下来对单样本 $x_i$ 进行权重更新：

$$\begin{aligned}\Delta w_{hj} = \frac{\partial J(\mathbf{w})}{\partial w_{hj}} &= -\eta \sum_{k=1}^{c} \frac{\partial J(\mathbf{w})}{\partial z_j^k} \frac{\partial z_j^k}{\partial net_j^k} \frac{\partial net_j^k}{\partial w_{hj}} \\ &= \eta \sum_{k=1}^{c} (t_j - z_j) \cdot f^{'}(net_j^k) \cdot y_h \\ &= \eta \sum_{k=1}^{c} (t_j - z_j) \cdot \frac{\sum_{n\neq k}^{c} e^{net_n}}{(\sum_{m=1}^{c} e^{net_m})^2} \cdot y_h\end{aligned}$$

$$\begin{aligned}\Delta w_{ih} = \frac{\partial J(\mathbf{w})}{\partial w_{ih}} &= -\eta \sum_{k=1}^{c} \sum_{r=1}^{n_H} \frac{\partial J(\mathbf{w})}{\partial z_j^k} \frac{\partial z_j^k}{\partial net_j^k} \frac{\partial net_j^k}{\partial y_h^r} \frac{\partial y_h^r}{\partial net_h^r} \frac{\partial net_h^r}{\partial w_{ih}} \\ &= \eta \sum_{k=1}^{c} \sum_{r=1}^{n_H} (t_j - z_j) \cdot f^{'}(net_j^k) \cdot w_{hj}^r \cdot f^{'}(net_h^k) \cdot x_i \\ &= \eta \sum_{k=1}^{c} \sum_{r=1}^{n_H} (t_j - z_j) \cdot \frac{\sum_{n\neq k}^{c} e^{net_n}}{(\sum_{m=1}^{c} e^{net_m})^2} \cdot w_{hj}^r \cdot \frac{e^{-net_h^r}}{(1+e^{-net_h^r})^2} \cdot x_i\end{aligned}$$

2. 请对反向传播算法的训练步骤进行总结；结合三层网络给出不超过三个有关权重更新的公式，并用文字描述所述公式的含义；指出哪些因素会对网络的性能产生影响。

由于结合三层网络给出不超过三个有关权重更新的公式，因此下面采用随机反向传播算法。随机反向传播算法的训练步骤一般如下：给定隐藏层节点数/维度 $n_H$，初始化的权重 $w$，学习率 $\eta$，训练终止条件（即损失函数阈值）$\theta$，算法开始讲随机抽取样本，对该样本计算网络输出值和损失函数值，然后按照 *Eq.1* 反向传播更新网络各层参数，其中，$w_{hj}$ 的更新公式代表原权重值减去损失函数对隐含层节点

的梯度，$w_{ih}$ 的更新公式代表原权重值减去损失函数对输入层节点的梯度，沿着梯度方向进行更新便可是网络收敛到损失函数最小值处，该更新过程的收敛条件即为损失函数值小于给定的终止阈值 $\theta$，最后输出更新结束的权重 $w = \{w_{hj}, w_{ih}\}$

**Algorithm 1** Stochastic Back-propagation

1: Initialization: $n_H, w, \eta$, criterion $\theta$, $k = 0$
2: **repeat**
3: $k \leftarrow k + 1 \ (mod\ n)$
4: $x_k$, randomly chosen a sample (pattern)
5: Update $w_{hj} \leftarrow w_{hj} + \eta\epsilon_j^k y_h^k, w_{ih} \leftarrow w_{ih} + \eta\epsilon_j^k x_i^k$ (Eq.1)
6: **until** $\|\nabla J(\mathbf{w})\| < \theta$

**Output:** $w$

*3.* 计算编程题

本题使用的数据如下所示

第一类 10 个样本（三维空间）：
[ 1.58, 2.32, -5.8], [ 0.67, 1.58, -4.78], [ 1.04, 1.01, -3.63],
[-1.49, 2.18, -3.39], [-0.41, 1.21, -4.73], [1.39, 3.16, 2.87],
[ 1.20, 1.40, -1.89], [-0.92, 1.44, -3.22], [ 0.45, 1.33, -4.38],
[-0.76, 0.84, -1.96]

第二类 10 个样本（三维空间）：
[ 0.21, 0.03, -2.21], [ 0.37, 0.28, -1.8], [ 0.18, 1.22, 0.16],
[-0.24, 0.93, -1.01], [-1.18, 0.39, -0.39], [0.74, 0.96, -1.16],
[-0.38, 1.94, -0.48], [0.02, 0.72, -0.17], [ 0.44, 1.31, -0.14],
[ 0.46, 1.49, 0.68]

第三类 10 个样本（三维空间）：
[-1.54, 1.17, 0.64], [5.41, 3.45, -1.33], [ 1.55, 0.99, 2.69],
[1.86, 3.19, 1.51], [1.68, 1.79, -0.87], [3.51, -0.22, -1.39],
[1.40, -0.44, -0.92], [0.44, 0.83, 1.97], [ 0.25, 0.68, -0.99],
[ 0.66, -0.45, 0.08]

(*1*) 请编写两个通用的三层前向神经网络反向传播算法程序，一个采用批量方式更新权重，另一个采用单样本方式更新权重。其中，隐含层结点的激励函数采用双曲正切函数，输出层的激励函数采用 *sigmoid* 函数。目标函数采用平方误差准则函数。

(*2*) 请利用上面的数据验证你写的程序，分析如下几点：

*(a)* 隐含层不同结点数目对训练精度的影响；

*(b)* 观察不同的梯度更新步长对训练的影响，并给出一些描述或解释；

*(c)* 在网络结构固定的情况下，绘制出目标函数随着迭代步数增加的变化曲线。

原始数据为三维数据，以下为对数据的三维可视化和二维投影可视化的展示：

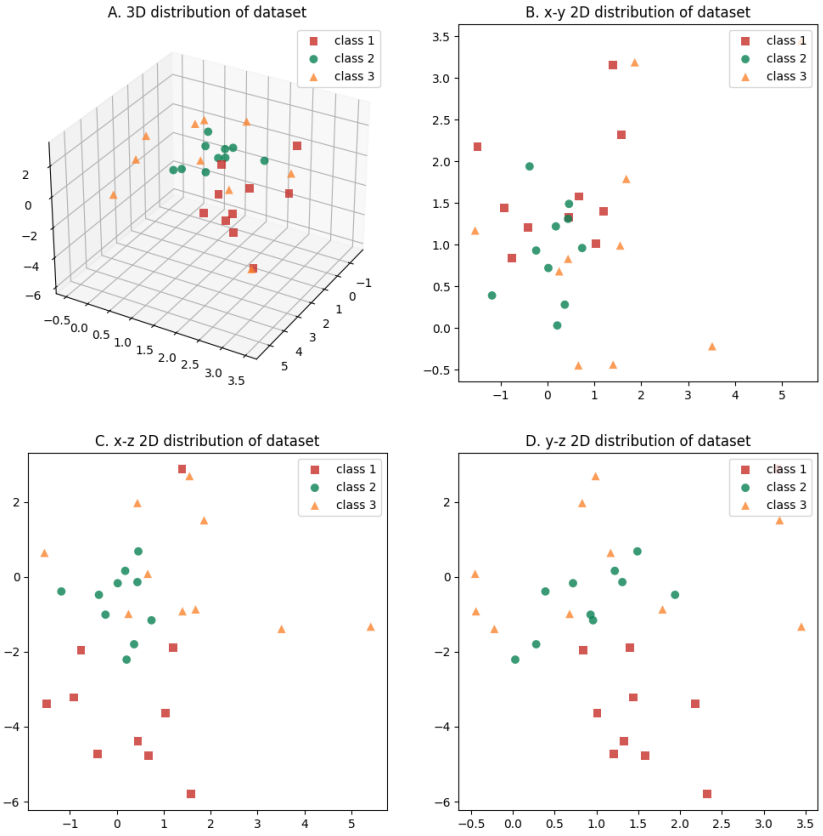


Fig. 2: Origin Data Distribution

(*1*) 由以下训练结果可以得出：固定学习率 ($lr = 0.01$)，*Fig.3* 和 *Fig.4* 分别展示了 *batch size* 为 *1* 和 *4* 时的训练场景，可以发现训练收敛情况基本相似，在训练效果上基本不相上下，但对时间复杂度的研究发现，模型训练相同 *epochs* 下，*batch size* 为 1 的平均训练时间是 *batch size* 为 4 的平均训练时间的 3.37 倍，该结果指导我们在训练模型过程中，选择合适的 *batch size* 可以加速模型的收敛。

Table 1: **Training time of different batch size**

| Training Session | Batch size=1(s) | Batch size=4(s) |
|---|---|---|
| 1 | 10.7085998 | 3.4579558 |
| 2 | 12.3710005 | 3.5295649 |
| 3 | 10.9132735 | 3.8749950 |
| 4 | 11.6097395 | 3.0624950 |
| 5 | 12.2431125 | 3.2197549 |
| Average time | 11.5691452 | 3.4289531 |

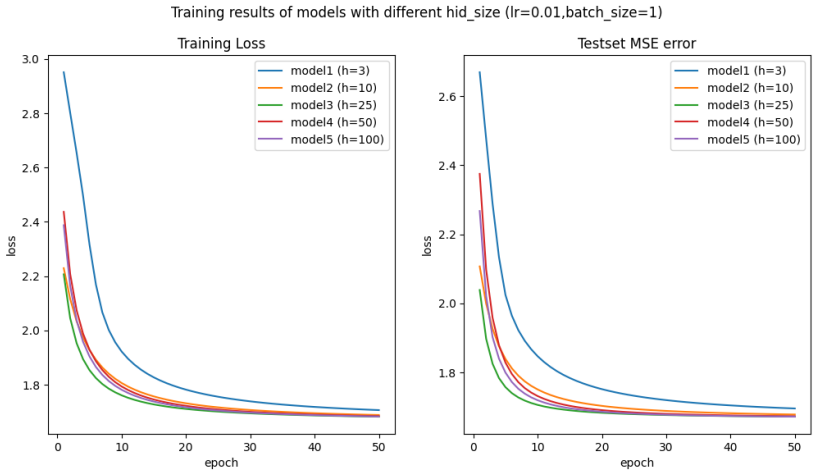


Fig. 3: Linear Network (lr=0.01, batch size=1)

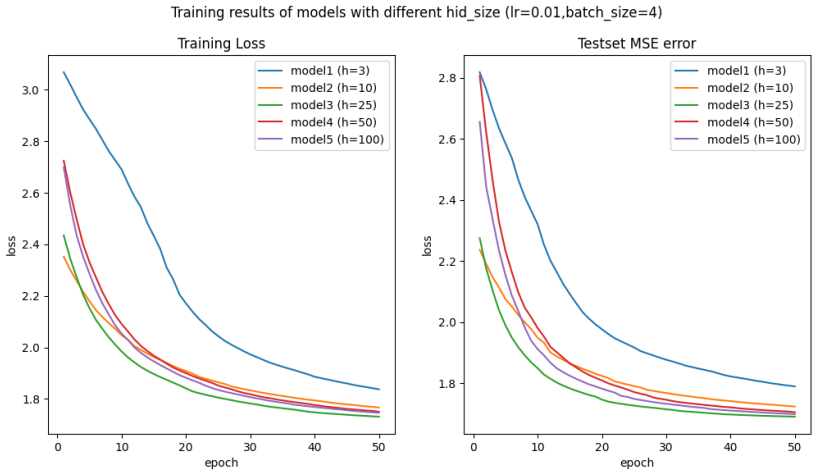


Fig. 4: Linear Network (lr=0.01, batch size=4)

(*2.a*) 固定学习率和 *batch size*，本人测试了 *5* 种 *hidden layer size*，即 $hid\_size = \{3, 10, 25, 50, 100\}$ 来研究改变隐含层节点数目对训练效果的影响，由 *Fig.3-6* 可看出，随着隐含层节点数目，模型的收敛速度有小幅度的提升，在训练精度上可看出具有一定的提升的效果，但过高的隐含层节点数目会提升模型的计算复杂度，有存在过拟合的风险，甚至模型效果降低（*Fig.4* 中，在 $lr = 0.01, batchsize = 4$ 时，采用 25 个隐含层节点数目的模型效果优于所有其他模型），选择合适的隐含层节点数目可以提升模型的精度。

(*2.b*) 固定 $batchsize = 4$ 和隐含层节点数目（$hidsize = 25$），如 *Fig.7* 所示，提高学习率对提高模型收敛速度有较大的影响，对模型训练精度影响较小（从该数据集训练结果出发，实际上选择一个好的学习率在复杂数据模型训练有重要的作用，能有助于有效收敛到高精度极小值），选择较高的学习率可以减少模型训练的时间成本，提升训练效率。

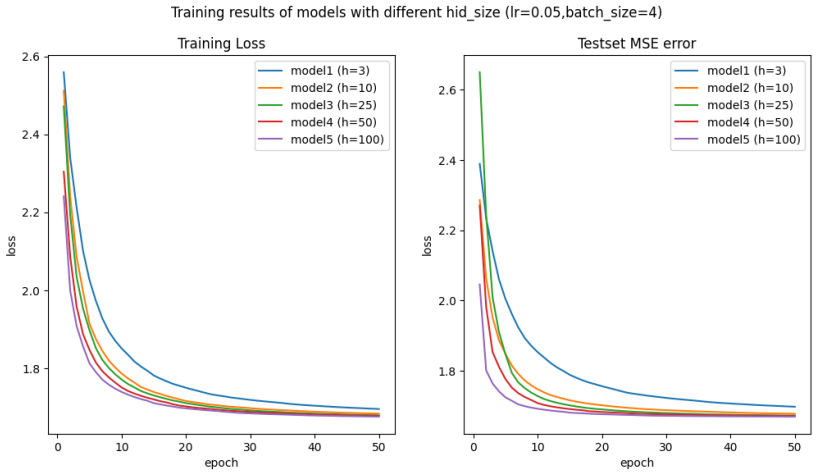


Fig. 5: Linear Network with Different Hidden Size (lr=0.05, batch size=4)

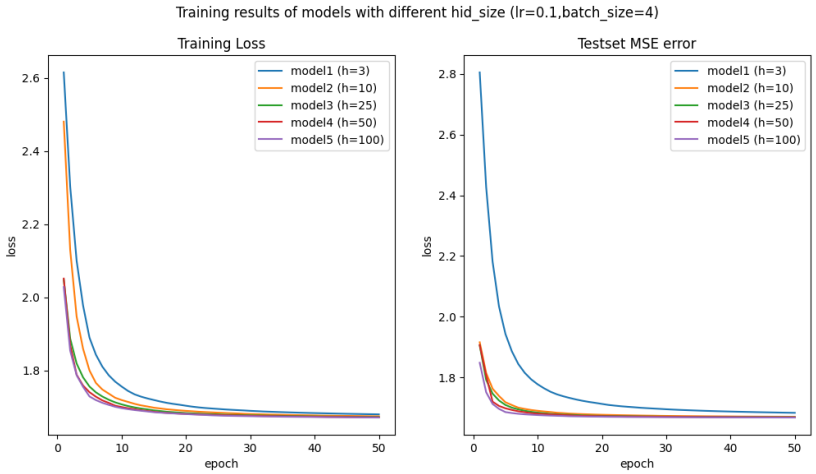


Fig. 6: Linear Network with Different Hidden Size (lr=0.1, batch size=4)

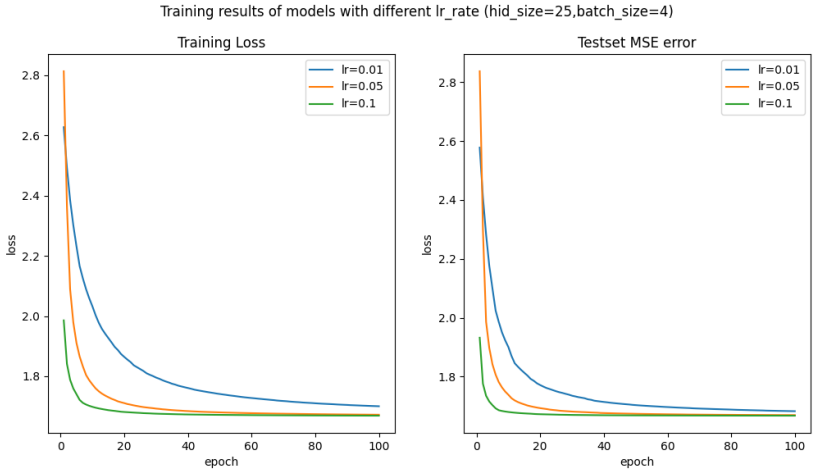


Fig. 7: Influence of Different lr rate (hid size=25, batch size=4)

***Supplementary Figures:***

代码见附件(*homework4.py*)

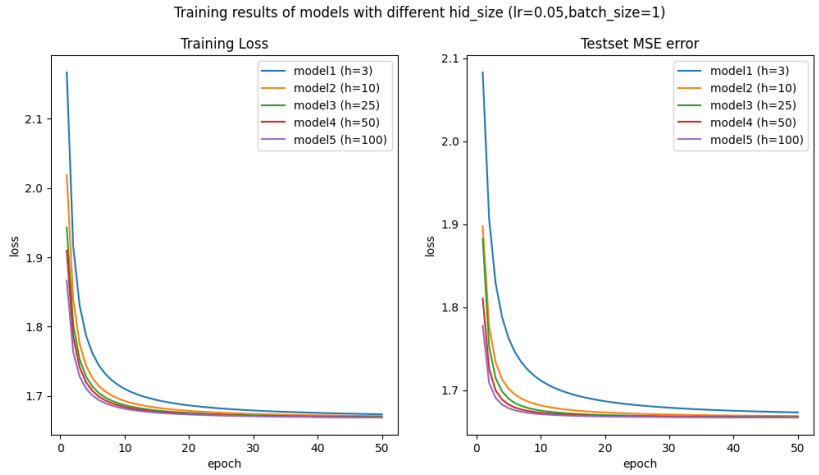


Fig. 8: Linear Network with Different Hidden Size (lr=0.05, batch size=1)

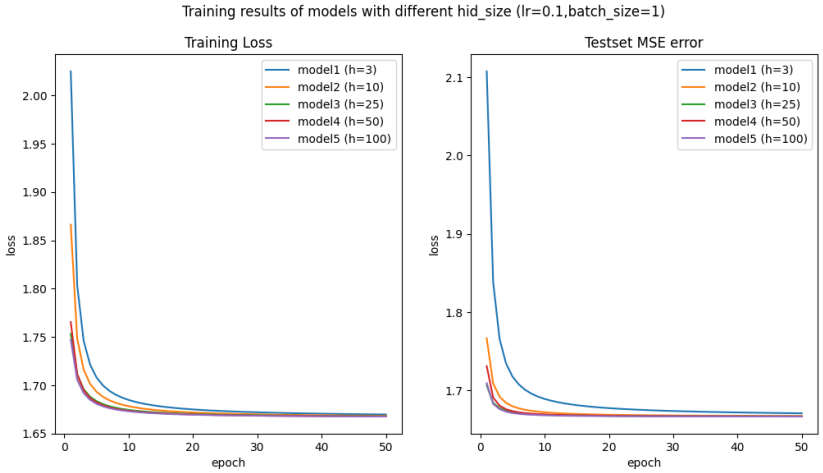


Fig. 9: Linear Network with Different Hidden Size (lr=0.1, batch size=1)