模式识别实验报告

姓　　名：范红乐
班　　级：计算机学院20250718班
学　　号：2025E8007382043

1 实验概述

1.1 实验内容

实现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。分析不同隐含节点数对性能的影响不同学习率对收敛速度和稳定性的影响；比较批量更新与单样本更新的性能差异。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

1.2 实验数据

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]

2 算法实现

2.1 实验参数

```
├── data_loader.py
├── network.py          # 三层神经网络结构
├── main.py             # 不同条件下的对比实验
├── comparison.py       # 单样本与批量更新的性能对比
└── imgs/               # 实验结果图
```

要求	说明
网络结构	三层神经网络（输入-隐含-输出）
隐含层激活函数	双曲正切：$f(x)=tanh(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}}$
输出层激活函数	Sigmoid：$f(x)=\frac{1}{1+e^{-x}}$
损失函数	平方误差：$E=\frac{1}{2}\sum_j(t_j-z_j)^2$
更新方式	单样本更新 single；批量更新 batch
数据预处理	将3类数据做归一化处理，并分别设置类别标签为 [1, 0, 0]、[0, 1, 0]、[0, 0, 1] 训练集与验证集比例默认为 7:3

2.2 前向传播

- 隐含层输入：$net_h=\sum_i w_{ih}x_i$
- 隐含层输出：$y_h=tanh(net_h)$
- 输出层输入：$net_j=\sum_h w_{hj}y_h$
- 输出层输出：$z_j=sigmoid(net_j)$
- 计算误差：$E=\frac{1}{2}\sum_j(t_j-z_j)^2$

2.3 反向传播

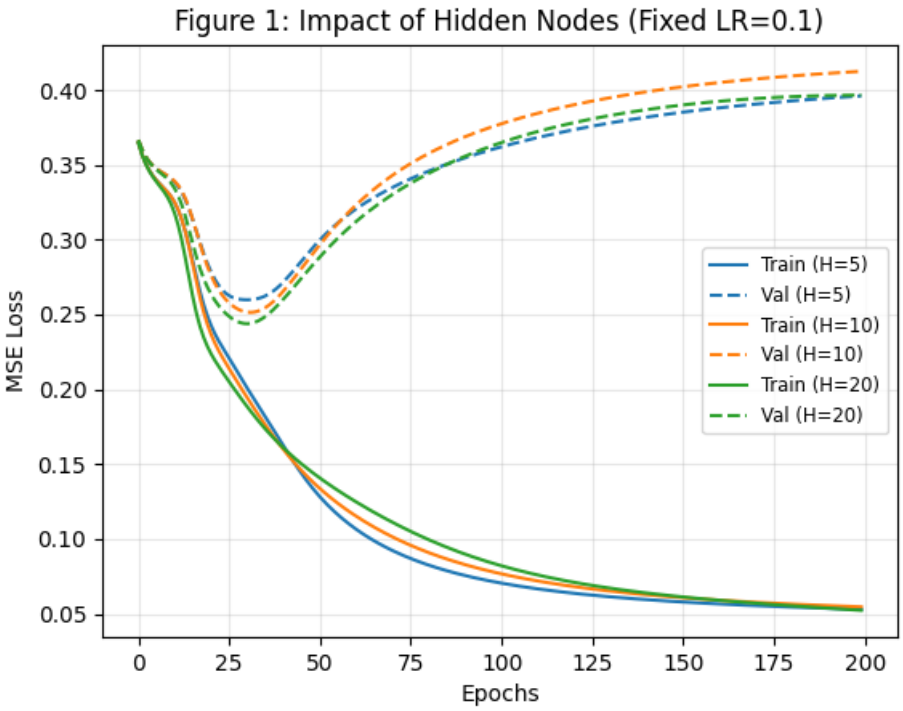
- 输出层误差信号：$\delta_j=\frac{-\partial E}{\partial net_j^k}=f'(net_j^k)(t_j^k-z_j^k)$ 其中 $f'(net_j^k)=sigmoid'(net_j^k)$
- 隐含层误差信号：$\delta_h=\frac{-\partial E}{\partial net_h^k}=f'(net_h^k)\sum_j w_{hj}\delta_j^k$ 其中 $f'(net_h^k)=tanh'(net_h^k)$
- 输出层权重更新：$w_{hj}\leftarrow w_{hj}+\eta\cdot\delta_j\cdot y_h$
- 隐含层权重更新：$w_{ih}\leftarrow w_{ih}+\eta\cdot\delta_h\cdot x_i$
- 两种权重更新方式：
  - 单样本更新：随机梯度下降，每输入一个样本就立即更新权重
  - 批量更新：积累所有样本的梯度，每轮结束后统一更新权重

3 分析结论

3.1 单样本更新

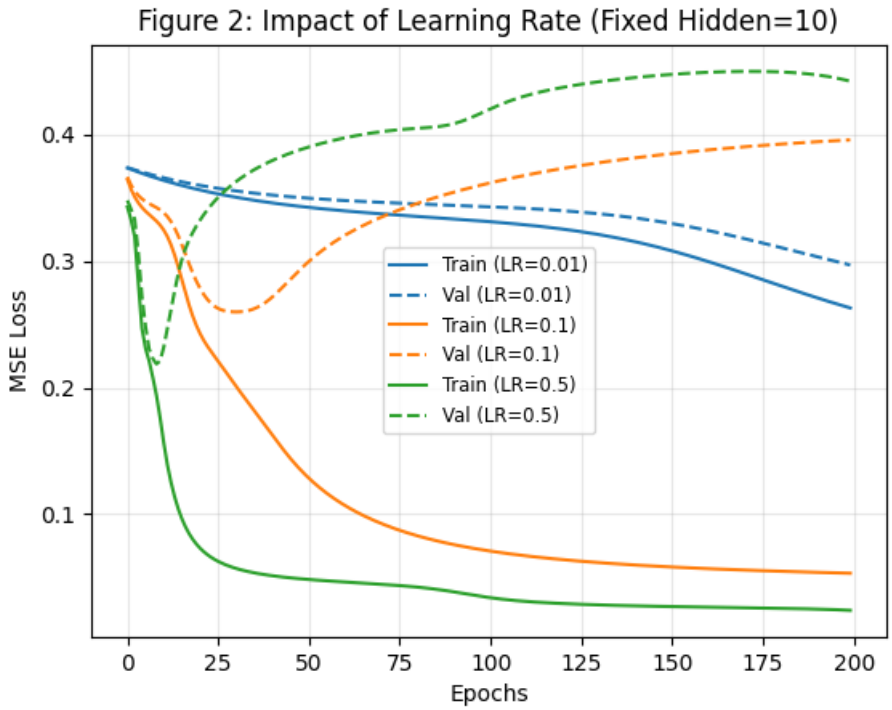
设置学习率为0.1，分别测试隐含层节点数为5、10、20时对训练效果的影响。

由 `Figure 1 (Single)` 可知，在epochs=25之前，随着隐含节点数增大，训练损失函数的斜率变大；但在epochs=30附近及之后，所有训练集误差很小，而所有验证集误差很大，呈“V”字形，即出现了过拟合现象，说明当前0.1的学习率对单样本更新来说太大了



设置隐含层节点数为10，分别测试学习率为0.01、0.1、0.5时对训练效果的影响。

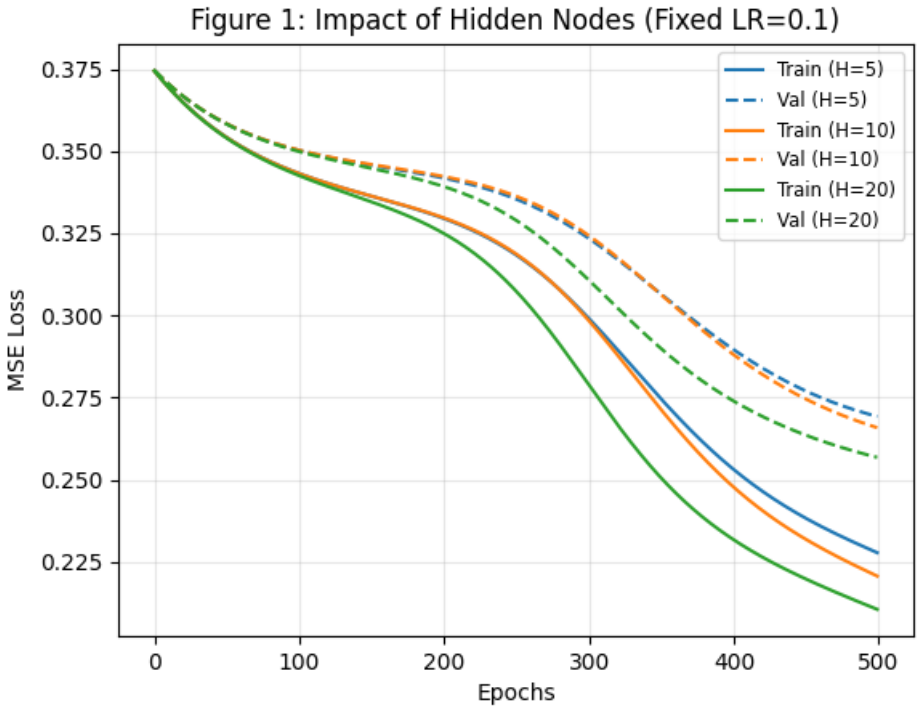
由 `Figure 2(Single)` 可知，学习率为0.01相较于0.1和0.5，收敛速度很慢；学习率为0.1和0.5时，训练初期，收敛速度很快，但随着训练的继续，很快都出现了过拟合现象



3.2 批量更新

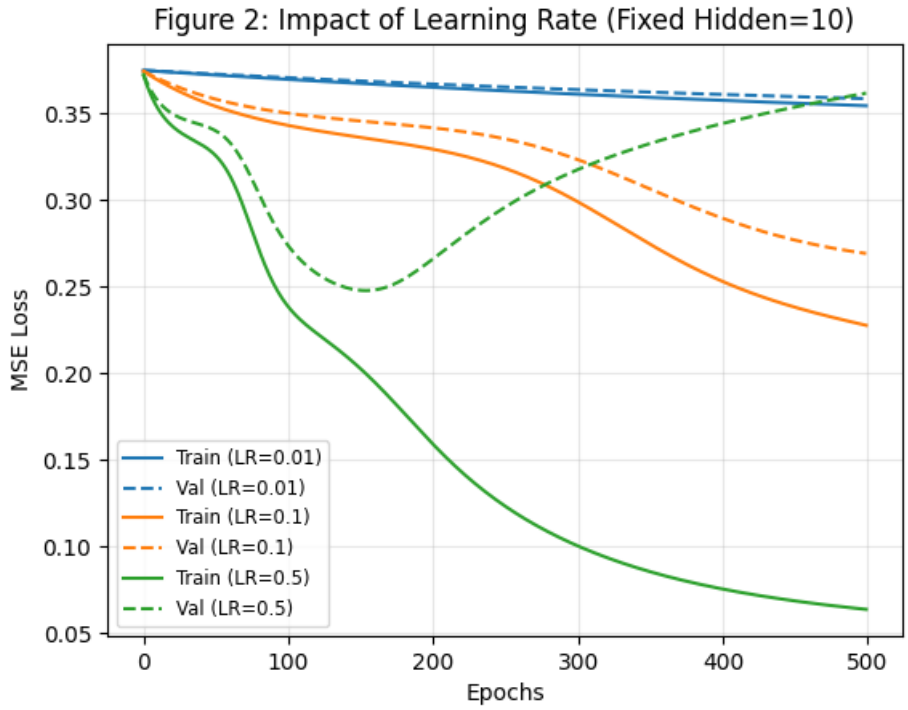
设置学习率为0.1，分别测试隐含层节点数为5、10、20时对训练效果的影响。

由 `Figure 1 (Batch)` 可知，隐含节点数为20时训练集损失函数下降最快，说明增加隐含节点数能够加快收敛速度



设置隐含层节点数为10，分别测试学习率为0.01、0.1、0.5时对训练效果的影响。

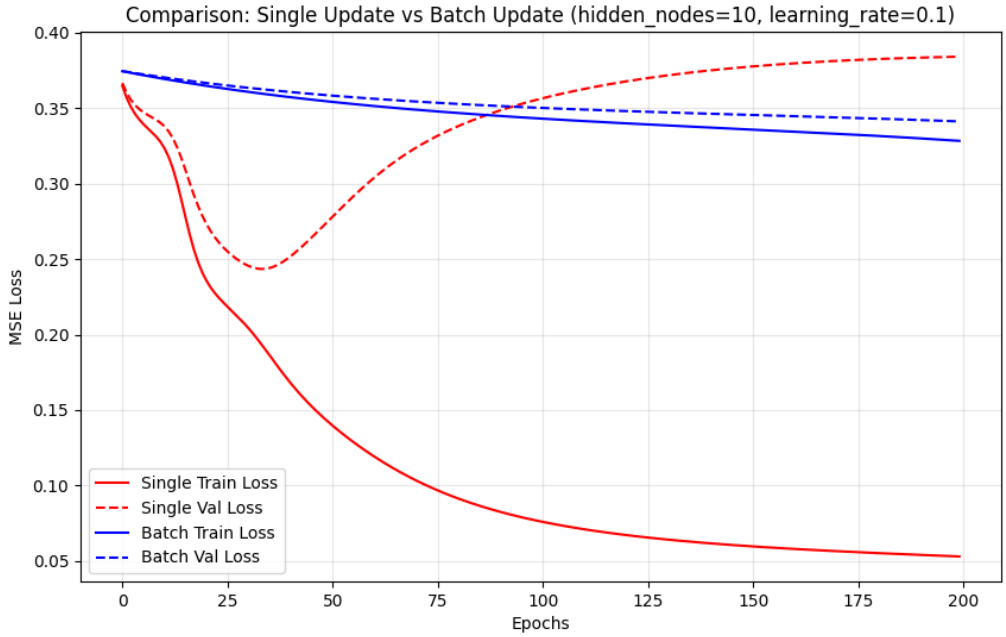
由 `Figure 2(Batch)` 可知，当学习率为0.01时，步长太小，收敛速度很慢，几乎是一条平缓的直线；学习率为0.1时，下降趋势比较明显，收敛相对比较稳定；当学习率为0.5时，收敛速度明显加快，但在epochs=150附近验证集损失函数出现“V”字形，说明步长过大，进入了过拟合状态



3.3 单样本更新 vs 批量更新

由图可知，单样本更新时损失函数的曲线有明显震荡，说明单样本更新容易受到个别样本噪声的影响；而批量更新利用全样本梯度的均值，代表整体的趋势，所以其损失函数的曲线较为平滑

单样本更新相较于批量更新，收敛速度更快，但由于当前学习率为0.1，对单样本更新来说步长过大，导致模型过快地陷入局部极小值，引发过拟合现象，在30轮之后权重更新完全跑偏，失去泛化能力；而0.1的学习率对于批量更新来说又太小，收敛速度非常缓慢



3.4 实验结论

由实验结果可知，随着隐含层节点的增加，训练效果有所提升，更多的节点意味着网络有更多的权重参数，能够拟合更加复杂的非线性决策边界，但节点过多时可能会把非规律的样本噪声学习进去，导致过拟合，降低模型的泛化能力

学习率决定了梯度下降的步长，过大容易导致模型在损失函数的谷底反复横跳，极容易快速陷入局部最小值，引发过拟合；过小时收敛速度太慢，容易处于欠拟合状态

单样本更新更容易受到个别样本噪声的影响，而批量更新使用所有样本梯度的平均值抵消了噪声，对真实梯度的估计更加稳定