基于 MLP 的 MNIST 分类实验

计 81 卢星宇 2018011280

超参数

为控制变量，整个实验流程超参数设置不变，如下图所示

```
config = {
    'learning_rate': 0.003,
    'weight_decay': 0.0,
    'momentum': 0.0,
    'batch_size': 100,
    'max_epoch': 200,
    'disp_freq': 50,
    'test_epoch': 5
}
```

单隐层 MLP 网络实验

网络结构

单层网络依次由 linear 层(768:128)，activate Function，linear 层(128:10)，loss Function 组成。通过组合 3 种激活函数和 3 种损失函数，共有九种网络。

训练集最终准确率

激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	94.9033%	96.3333%	98.3667%
Sigmoid	91.3767%	91.7017%	95.8917%
Gelu	95.6983%	95.9533%	98.785%

训练集最终 Loss

激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	0.0681859	0.0132722	0.2961626
Sigmoid	0.1052926	0.0289748	0.8937044
Gelu	0.0671430	0.0141443	0.21225007

测试集最终准确率

激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	94.38%	95.74%	97.41%
Sigmoid	91.69%	91.90%	95.54%
Gelu	95.61%	95.60%	97.82%

测试集最终 Loss

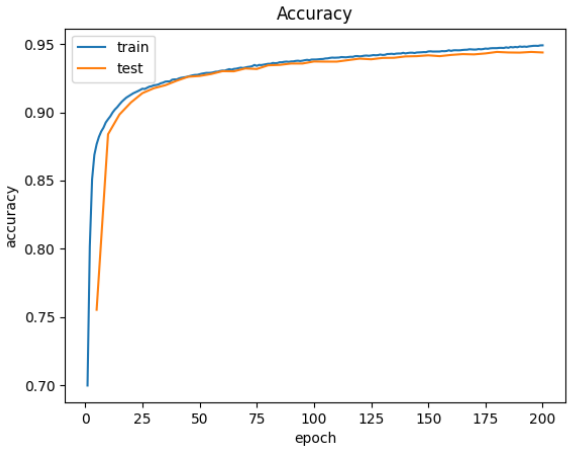
激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	0.0697157	0.0141394	0.5501007
Sigmoid	0.1028543	0.0283136	0.9826091
Gelu	0.0683297	0.0151582	0.4638956

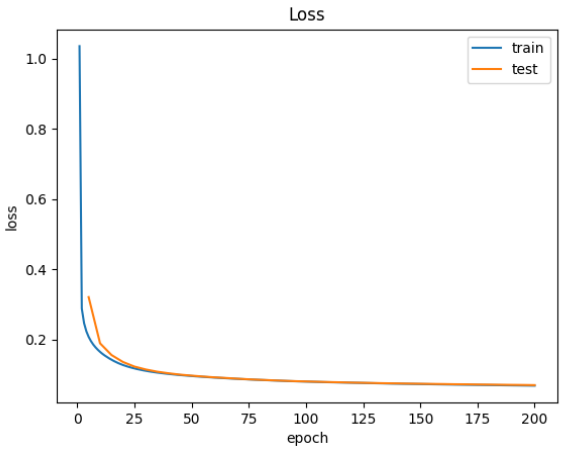
训练耗时（秒）

激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	335.82	395.33	427.46
Sigmoid	343.21	365.84	397.37
Gelu	601.32	622.89	623.32

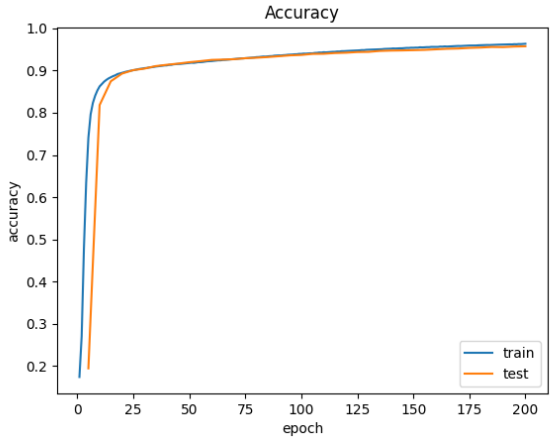
训练曲线

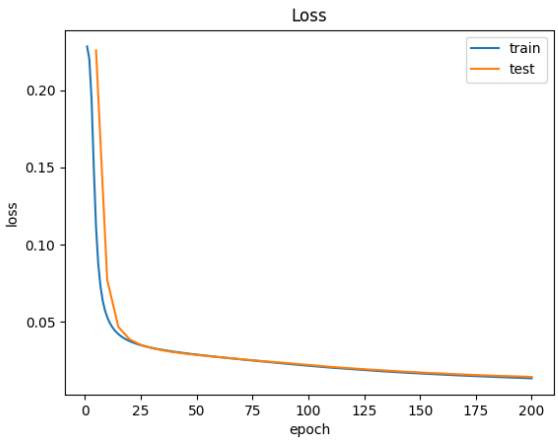
Relu+ EuclideanLoss



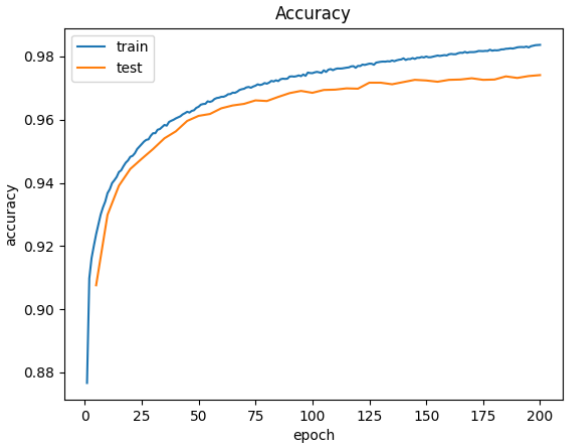


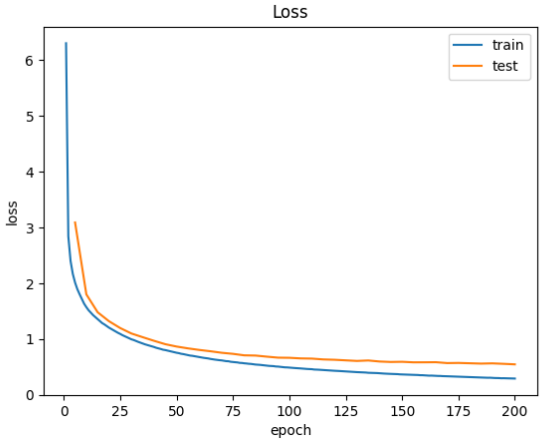
Relu+ SoftmaxCrossEntropyLoss



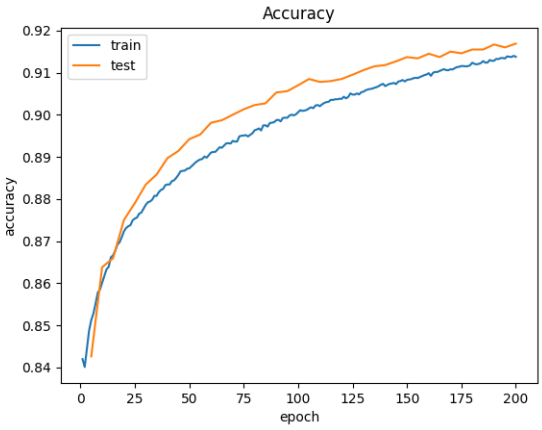


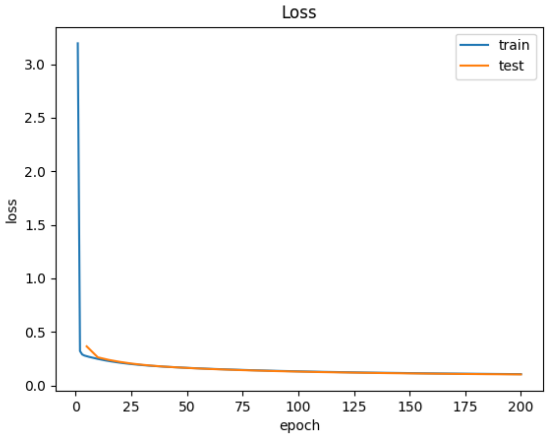
Relu+HingeLoss



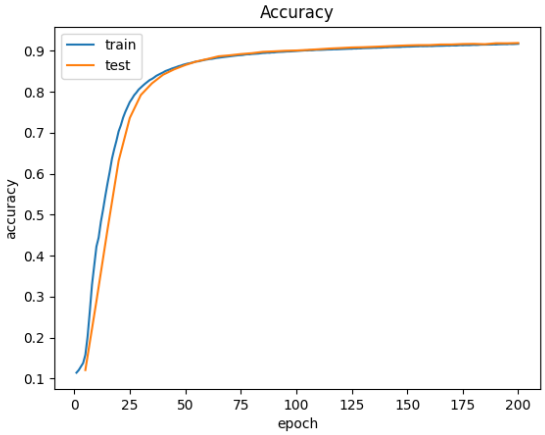


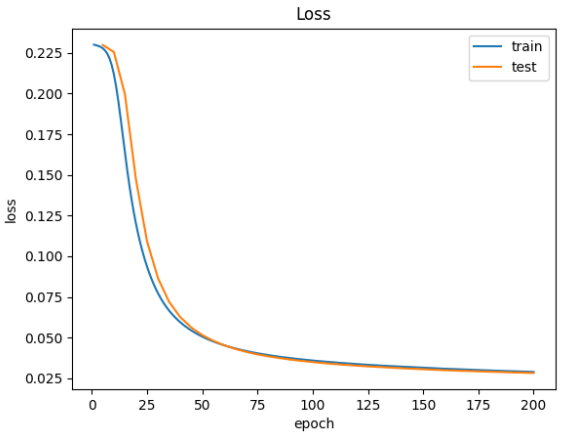
Sigmoid+ EuclideanLoss



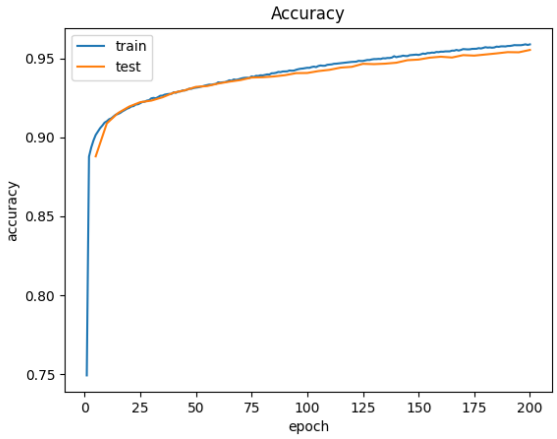


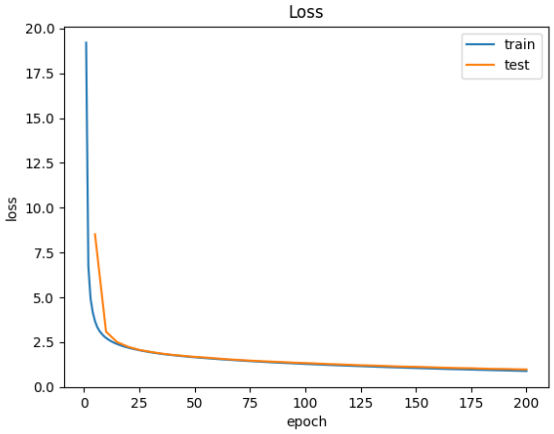
Sigmoid+ SoftmaxCrossEntropyLoss



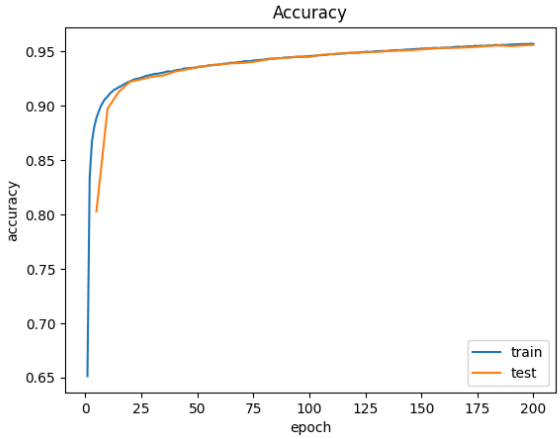


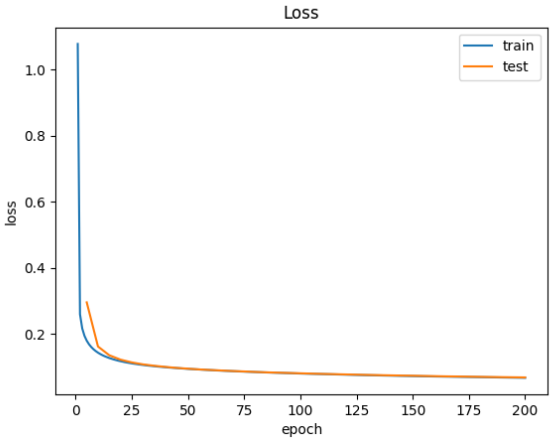
Sigmoid+HingeLoss



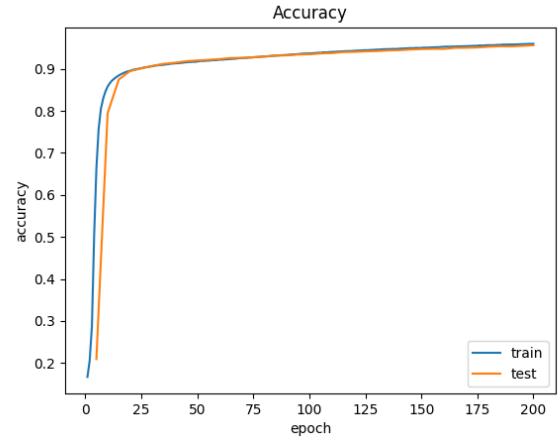


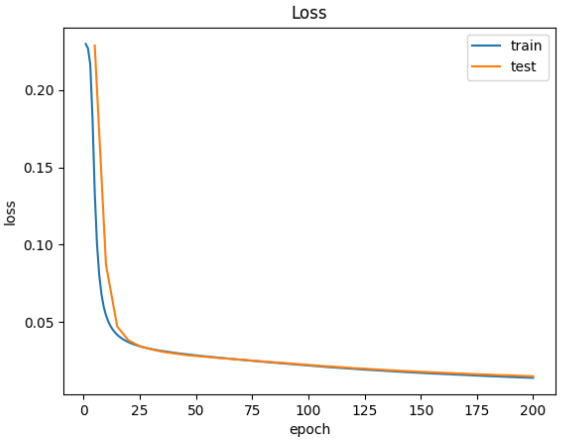
Gelu+ EuclideanLoss



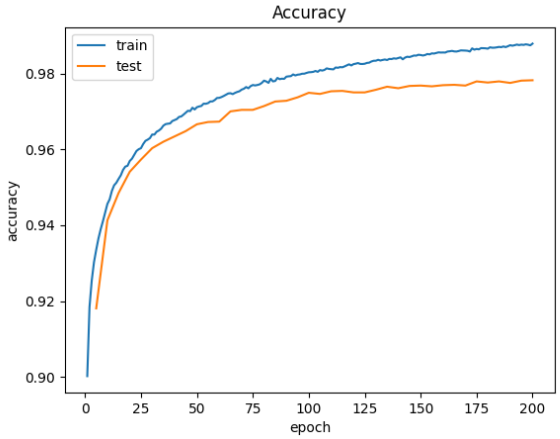


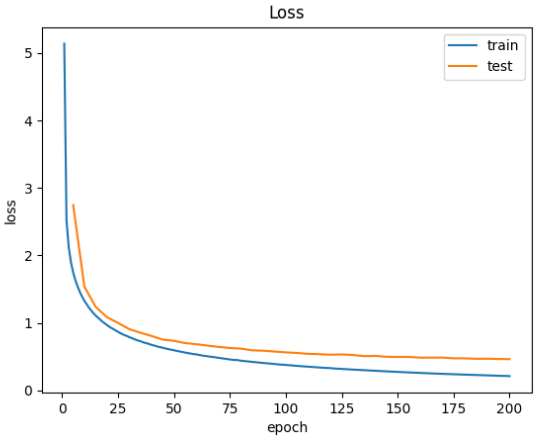
Gelu+ SoftmaxCrossEntropyLoss





Gelu+HingeLoss





简答题 2：比较不同的激活函数和损失函数

答：

对于激活函数:

从训练时间来看，Gelu > Sigmoid > Relu，

从收敛性来看，Gelu≈Relu>Sigmoid，

从准确率来看，Gelu>Relu>Sigmoid

对于损失函数：

从训练时间来看，HingeLoss > SoftmaxCrossEntropyLoss > EuclideanLoss，

从收敛性来看，HingeLoss > SoftmaxCrossEntropyLoss > EuclideanLoss，

从准确率来看，HingeLoss > SoftmaxCrossEntropyLoss > EuclideanLoss，

简答题 3：最好的激活函数

答：从准确率来看，Gelu 激活函数准确率最高，收敛速度最快，虽然计算复杂性较高，但我认为 Gelu 是最好的激活函数。

我认为 Gelu 的优点在于，Sigmoid 由于导数值较小，存在梯度较小使得收敛缓慢的问题。而 Relu 虽然导数值在大于零时恒为 1，但简单粗暴的置零操作很可能会导致处理过程的信息丢失，而 Gelu 同时避免了 Relu 和 Sigmoid 的缺点，相当于 dropout，zoneout Relu 的组合，虽然在计算量上出现了增长，但可以有效地避免过拟合问题，而且针对正态分布构造的激活函数在直觉上对该任务适应性非常好，分类效果好了很多。

双隐层 MLP 网络实验

网络结构

双层网络依次由 linear 层(768:256)，Relu Function，linear 层(256:64)，Activate Function，linear 层（64:10），loss Function 组成。通过组合 3 种激活函数和 3 种损失函数，共有九种网络。

训练集最终准确率

激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	97.32%	97.5417%	99.8750%
Sigmoid	93.38%	91.9933%	99.05%
Gelu	96.4500%	97.375%	99.8617%

训练集最终 Loss

激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	0.0392526	0.0088063	0.0263332
Sigmoid	0.0707241	0.0280716	0.1744394
Gelu	0.0520509	0.0093170	0.0272725

测试集最终准确率

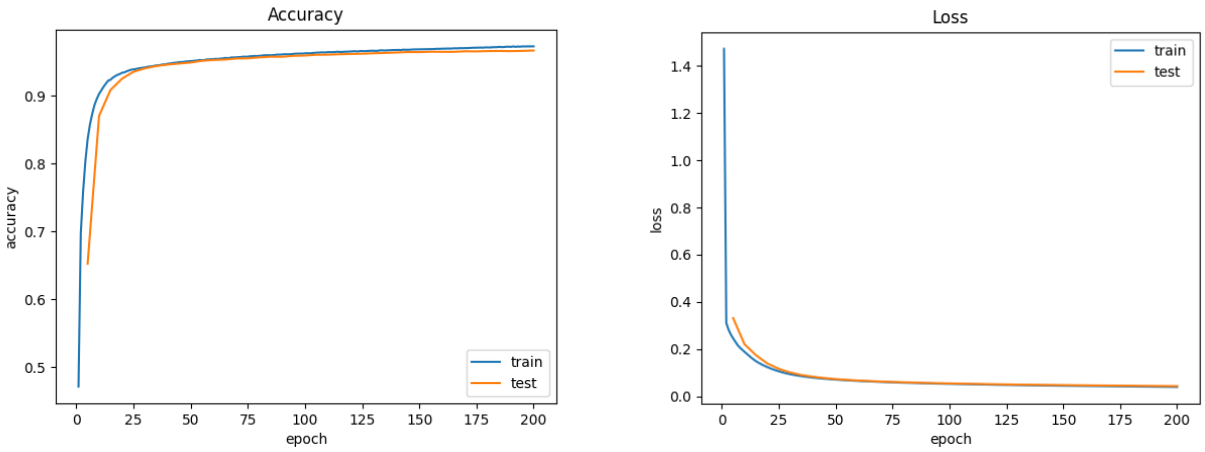
激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	96.70%	96.66%	97.79%
Sigmoid	93.34%	91.74%	97.68%
Gelu	96.03%	96.71%	97.89%

测试集最终 Loss

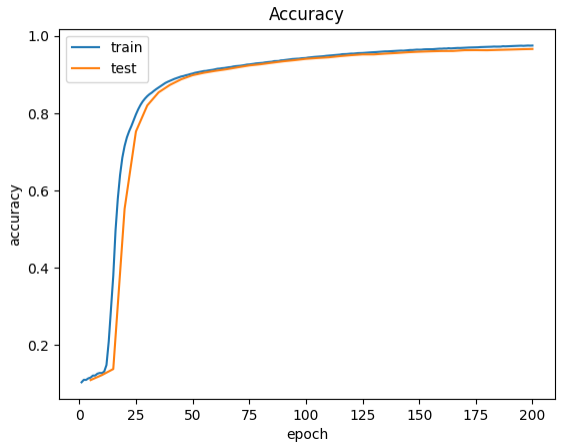
激活函数\损失函数	EuclideanLoss	SoftmaxCrossEntropyLoss	HingeLoss
Relu	0.0427176	0.0112311	0.4149707
Sigmoid	0.0695033	0.0287777	0.4653703
Gelu	0.0543731	0.0115954	0.3969557

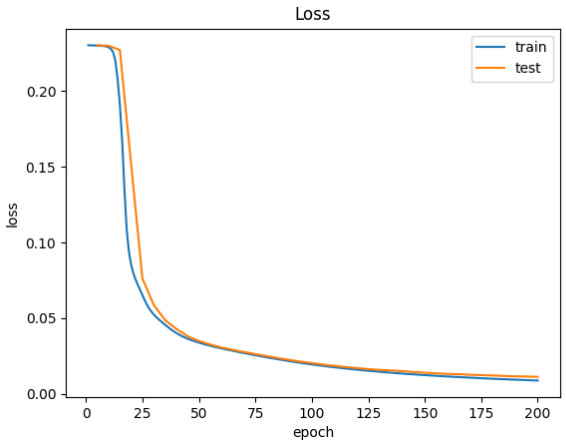
训练曲线

Relu+ EuclideanLoss

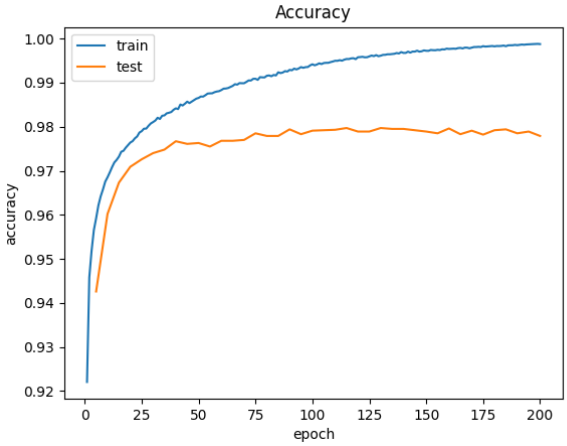


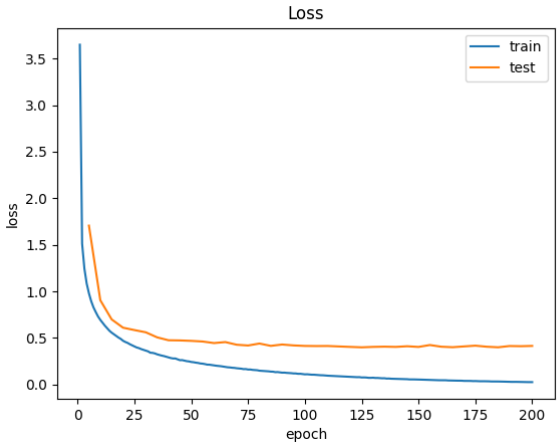
Relu+ SoftmaxCrossEntropyLoss



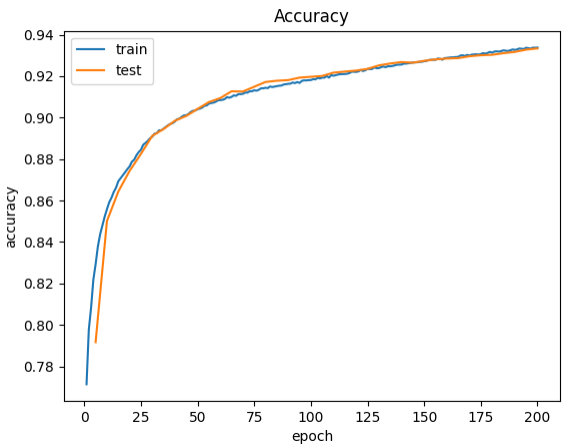


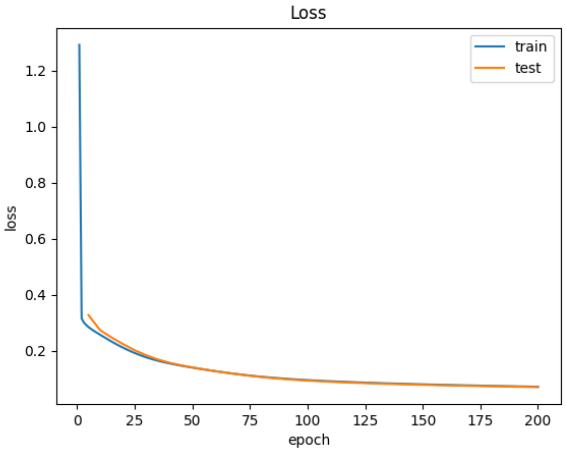
Relu+HingeLoss



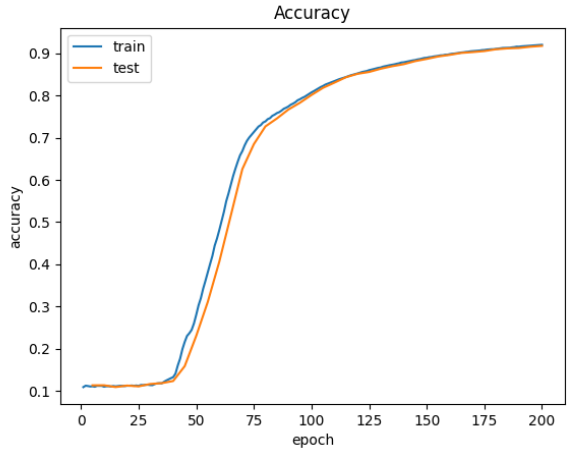


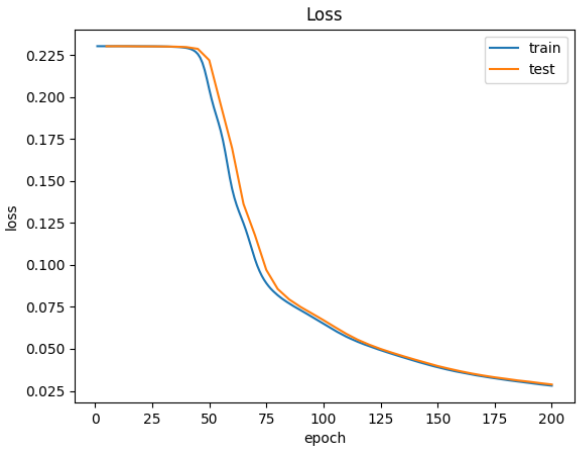
Sigmoid+ EuclideanLoss



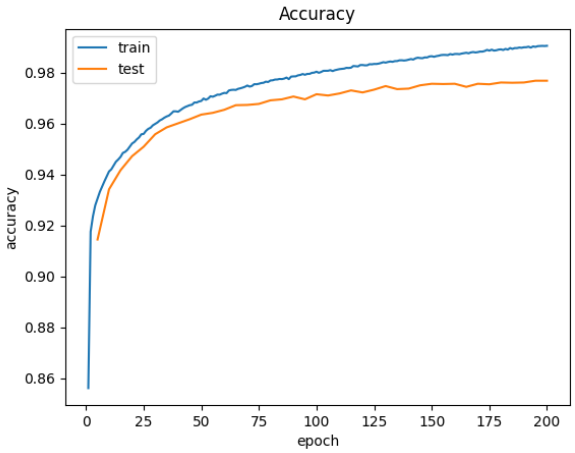


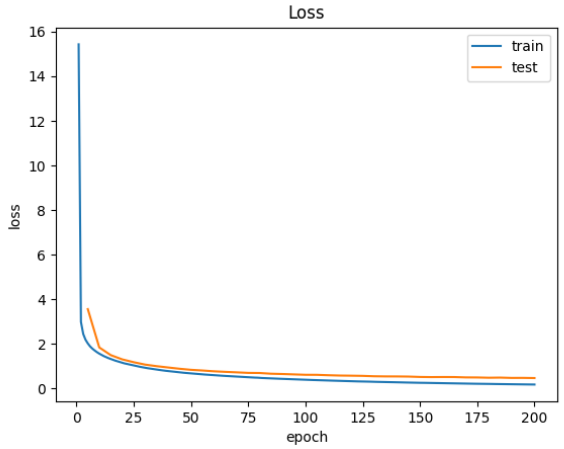
Sigmoid+ SoftmaxCrossEntropyLoss



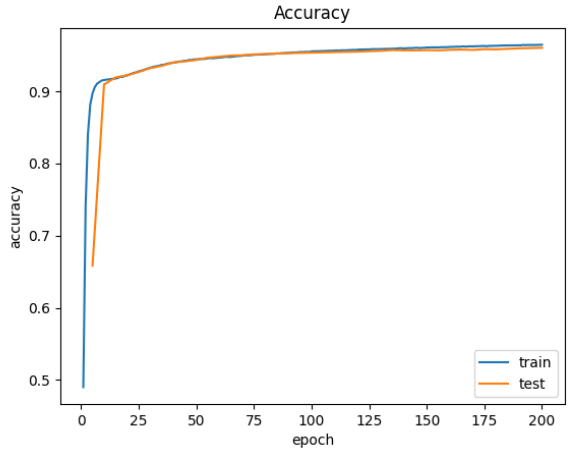


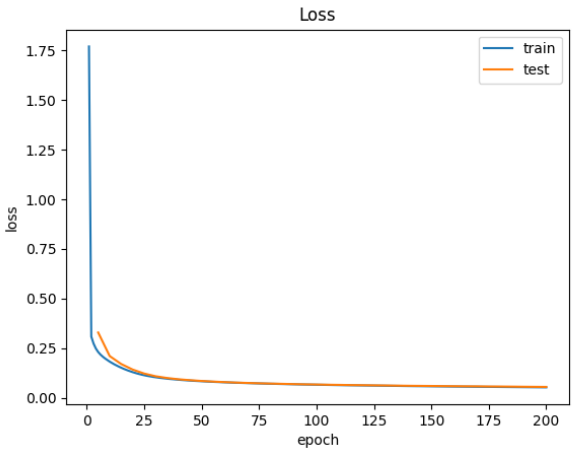
Sigmoid+HingeLoss



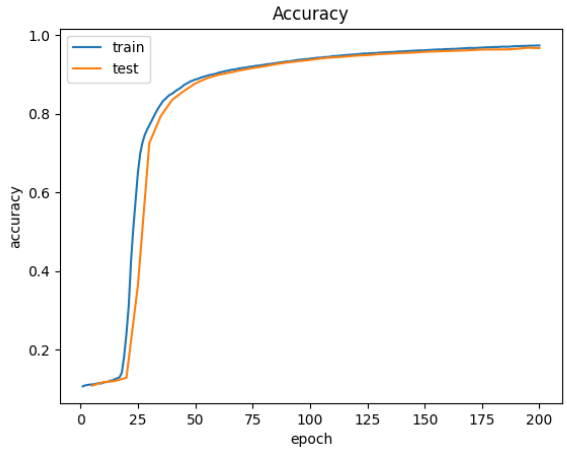


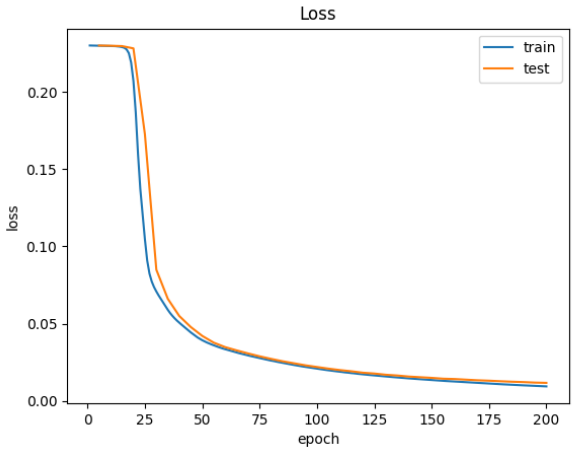
Gelu+ EuclideanLoss



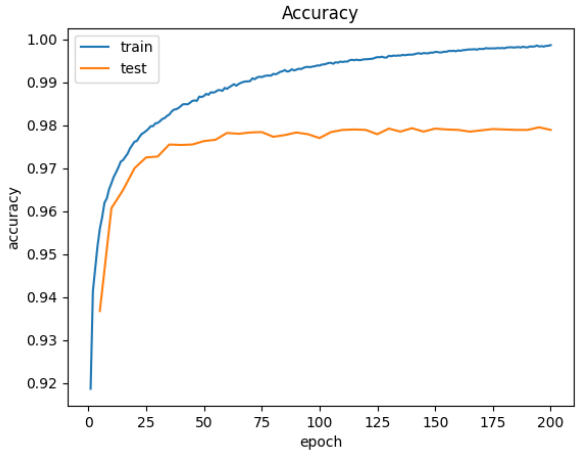


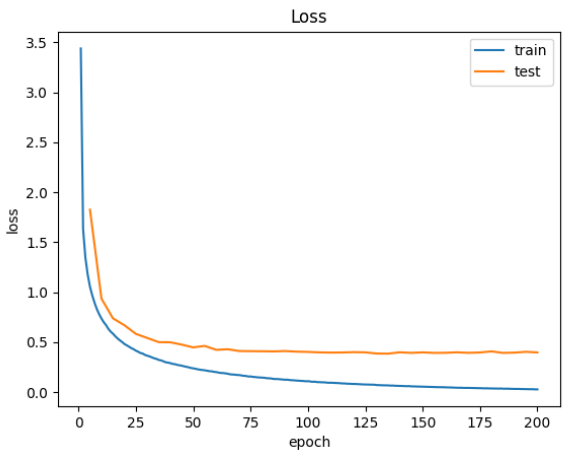
Gelu+ SoftmaxCrossEntropyLoss





Gelu+HingeLoss





单双层网络对比

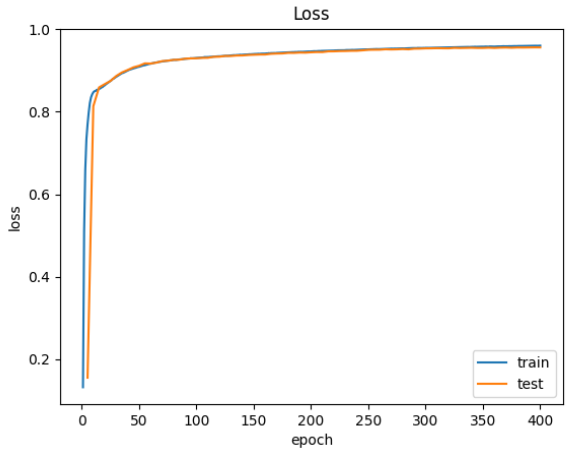
从最终结果准确率来看，大部分双层网络相较于单层网络有更好的准确率，最终的 loss 也更低，问题在于训练的时间显著增加，而准确率的增长在 1%以下。可见对于此任务单层网络已经能取得较好的准确率，增加隐藏层层数收益并不大，反而增加了训练的时间成本

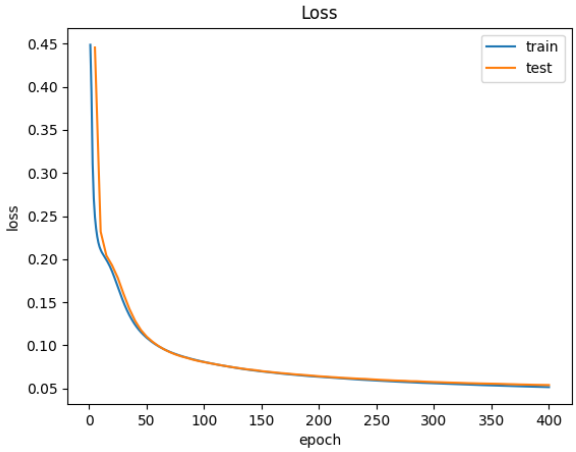
超参数调整

针对准确率最低的 Sigmoid+ EuclideanLoss 网络，试图通过调整参数进行优化。

max epoch 调整

由于 200 个 epoch 之后仍观察到准确率上升，故增加 max epoch 数目，观察最终的准确率，在训练了 400 个 epoch 之后，准确率无明显上升，在训练集上达到 96.05%，在测试集上达到了 95.67%





通过调整 max epoch，使得模型训练更加充分。但训练一段时间之后，可能出现过拟合现象，继续训练取得的效果几乎可以忽略不计，对准确率的提升效果不大，所以在 400 个 epoch 之后停止训练。

learning rate 调整

将 learning rate 调整为 0.01，0.02， 0.03，0.06，0.09，0.12，分别进行实验，获得的准确率如下表所示

Learning rate	Train accuracy	Test accuracy
0.01	94.26%	94.28%
0.02	95.91%	95.42%
0.03	96.67%	96.10%
0.06	97.52%	96.69%
0.09	97.85%	96.73%
0.12	96.68%	95.91%

learning rate 决定每次参数调整的步长大小，如果 learning rate 过小，会导致模型收敛缓慢，训练时间漫长，而 learning rate 过大会导致容易错过最优的参数，训练准确率和 loss 跳动幅度和方向不固定，在上述实验中，最终结论是 learning rate 在 0.09 最为适宜，这比最初设置的 0.003 大了 30 倍，可见由于 sigmoid 激活函数梯度较小的特性，应该选取较大的 learning rate 来提高训练速度，节省时间。也可能是由于初始设置的 learning rate 过小。

将 loss 函数改为 HingeLoss， learning rate 设为 0.09，经 200 个 epoch 之后，在训练集和测试集上分别取得了 100%和 98%的准确率，远超之前的结果，由此可以看出，针对不同的超参数，不同激活函数和损失函数的表现会有很大不同，更改了学习率可以让之前并不起眼的 sigmoid 激活函数获得出色的结果。