基于MLP的MNIST分类实验

计81 肖光烜 2018011271

超参数设定

以下所有实验均采用一种超参数设定：

```
config = {
    'learning_rate': 0.05,
    'weight_decay': 0.0,
    'momentum': 0.0,
    'batch_size': 100,
    'max_epoch': 100,
    'disp_freq': 200,
    'test_epoch': 1
}
```

Hinge Loss 的间隔阈值设为5.

用时计算

由于所有实验都是进行100 epochs迭代，且每步迭代都会进行测试，因此最终测得时间为100个epoch的训练测试总时间。

单隐层MLP

模型设置

```
Image(768 dim) -> Linear(768:64) -> Actf -> Linear(64:10) -> Loss
```

隐层维数设定为64，激活函数有三种选择，损失函数有三种选择，因此共有九种可能的实验模式。

实验结果

以下为最后一个epoch训练后的模型在训练集、测试集的性能以及总耗时。

测试集

准确率（**%**）

激活函数\损失函数	**EuclideanLoss (MSE)**	**SoftmaxCrossEntropyLoss**	**HingeLoss**
Relu	96.6900	97.5800	97.6100
Sigmoid	94.8300	96.8100	97.4900
Gelu	96.5600	97.6400	**97.9100**

Loss

激活函数\损失函数	EuclideanLoss (MSE)	SoftmaxCrossEntropyLoss	HingeLoss
Relu	**0.046947**	0.008452	0.509331
Sigmoid	0.061824	0.010878	0.510150
Gelu	0.051419	**0.007997**	**0.421297**

训练集

准确率（**%**）

激活函数\损失函数	EuclideanLoss (MSE)	SoftmaxCrossEntropyLoss	HingeLoss
Relu	97.4917	99.5083	99.7217
Sigmoid	94.9233	97.4850	99.0350
Gelu	97.0067	99.5533	**99.8817**

Loss

激活函数\损失函数	EuclideanLoss (MSE)	SoftmaxCrossEntropyLoss	HingeLoss
Relu	**0.041176**	0.002166	0.049773
Sigmoid	0.062175	0.009140	0.183448
Gelu	0.047750	**0.001940**	**0.026761**

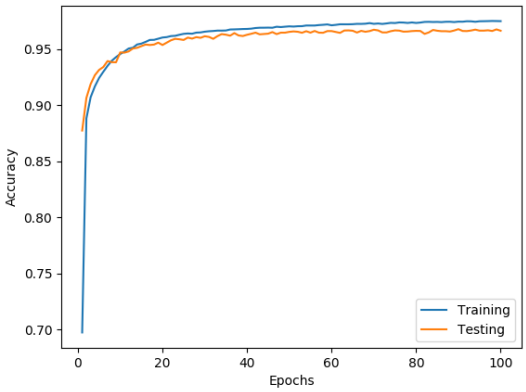
耗时（秒）

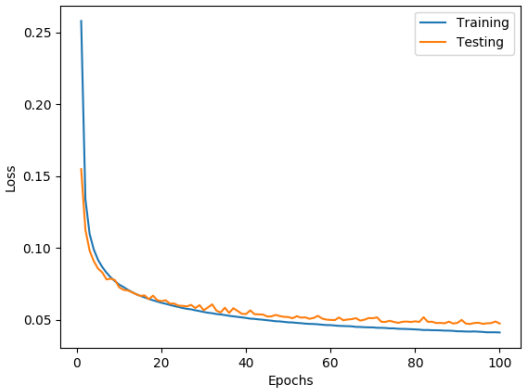
激活函数\损失函数	EuclideanLoss (MSE)	SoftmaxCrossEntropyLoss	HingeLoss
Relu	**84.91**	97.67	93.05
Sigmoid	123.80	133.62	133.56
Gelu	302.22	307.89	309.02

训练曲线

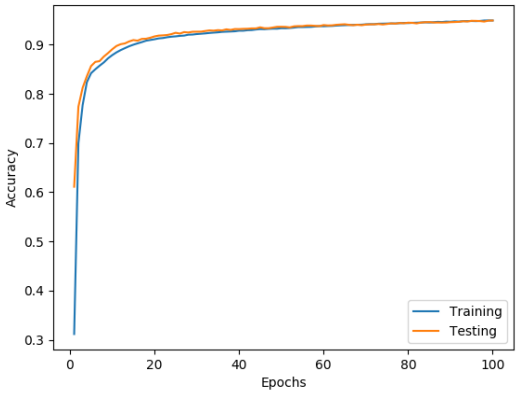
Euclidean Loss

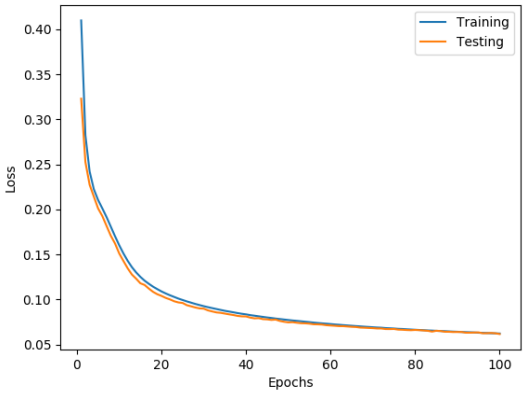
Relu



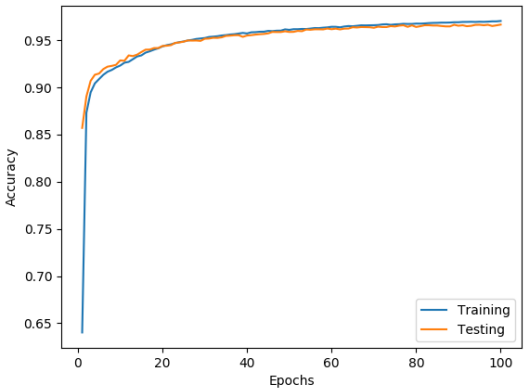


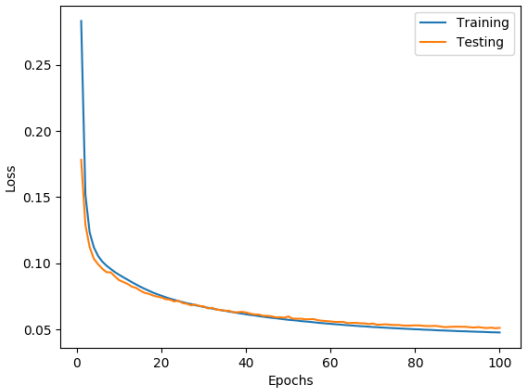
Sigmoid





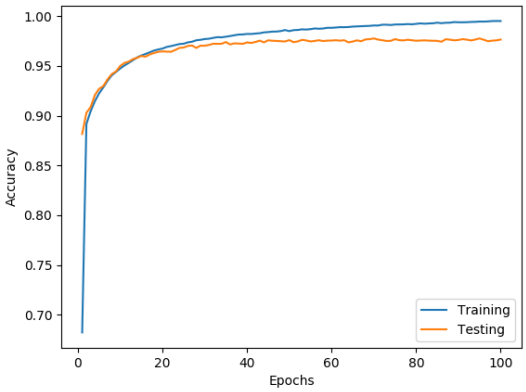
Gelu

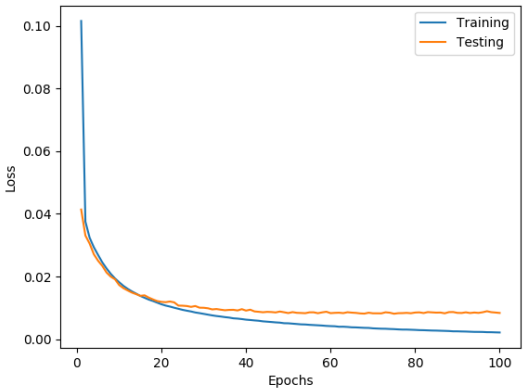




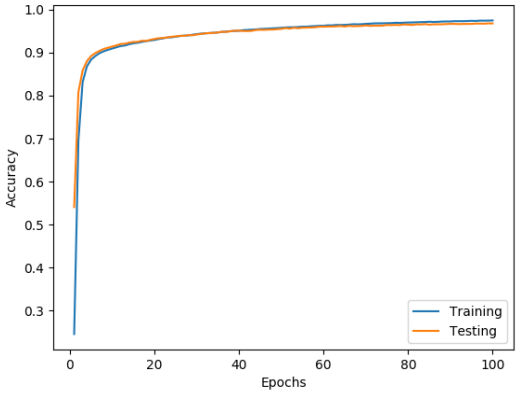
Softmax Cross-Entropy Loss

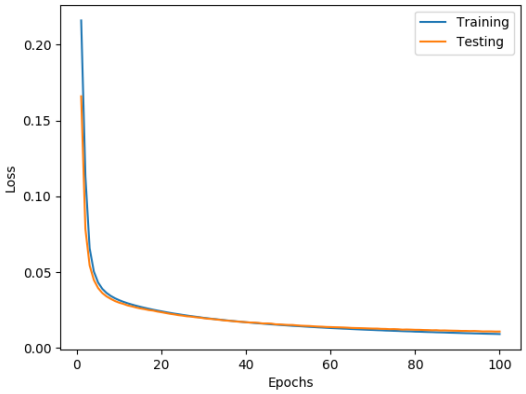
Relu



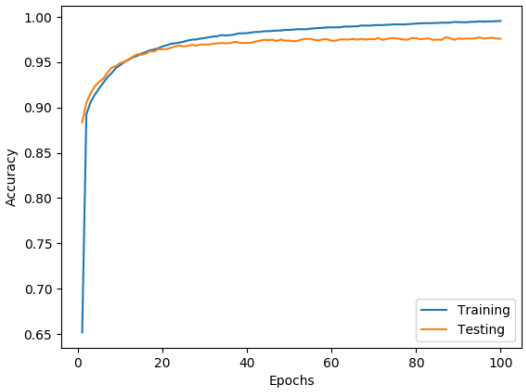


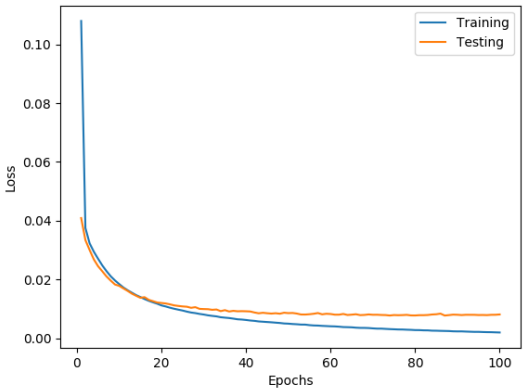
Sigmoid





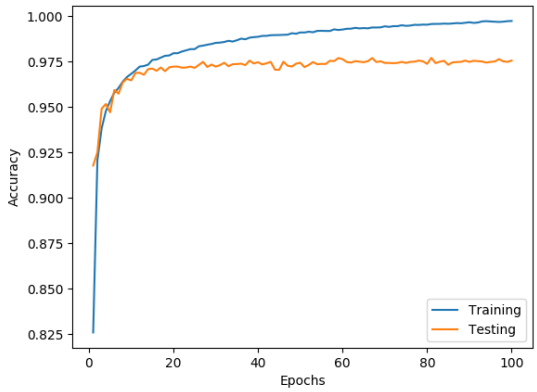
Gelu

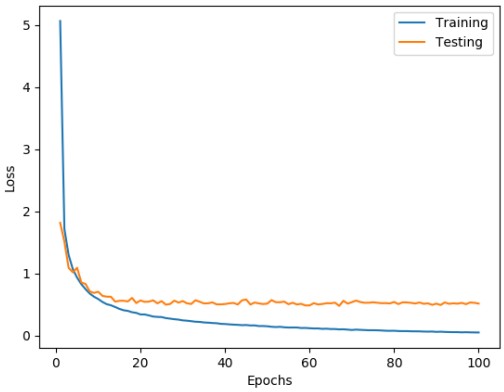




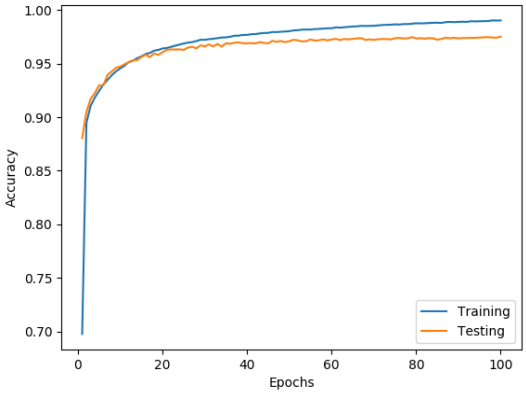
Hinge Loss

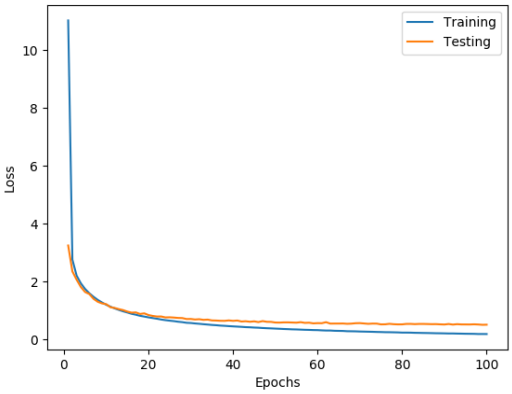
Relu



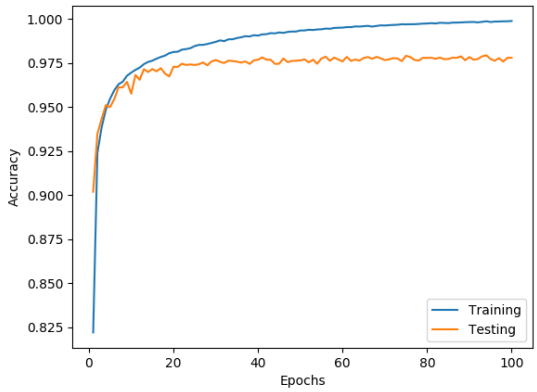


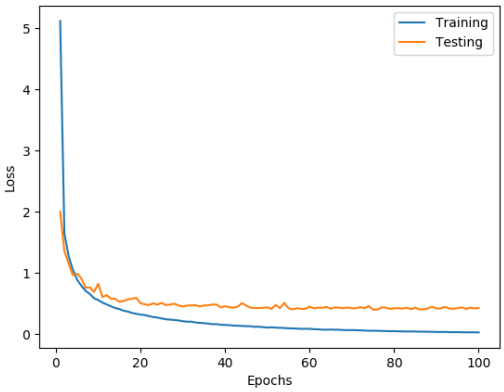
Sigmoid





Gelu





讨论

综合以上实验结果和训练曲线可以看出，在当前的实验设置和实现下：

采用不同激活函数

- 分类效果：$Gelu > Relu > Sigmoid$
- 计算复杂度：$Gelu > Sigmoid > Relu$
- 收敛速度：$Gelu \approx Relu > Sigmoid$

采用不同损失函数

- 分类效果：$HingeLoss \approx SoftmaxCrossEntropy > EuclideanLoss$
- 计算复杂度：$HingeLoss \geq SoftmaxCrossEntropy > EuclideanLoss$
- 收敛速度：$HingeLoss > SoftmaxCrossEntropy \approx EuclideanLoss$

双隐层MLP

模型设置

```
Image(768 dim) -> Linear(768:128) -> Actf -> Linear(128:64) -> Actf ->
Linear(64:10) -> Loss
```

由单隐层的实验可以看出，无论使用哪一种激活函数，EuclideanLoss损失函数都不太行，因此在双隐层MLP实验中抛弃这一种损失函数；激活函数中可见Sigmoid函数怎么都不太行，于是抛弃这种激活函数。因此进行$2 \times 2 = 4$次实验。

实验结果

括号内标明与相应单层MLP的对比。

测试集

准确率（**%**）

激活函数\损失函数	**SoftmaxCrossEntropyLoss**	**HingeLoss**
Relu	97.7800 (+0.2000)	97.5900 (-0.2000)
Gelu	**97.8400 (+0.2000)**	97.5400 (-0.3700)

Loss

激活函数\损失函数	**SoftmaxCrossEntropyLoss**	**HingeLoss**
Relu	0.009969	0.679308
Gelu	0.009332	0.709859

训练集

准确率（**%**）

激活函数\损失函数	**SoftmaxCrossEntropyLoss**	**HingeLoss**
Relu	**99.9967 (+0.4884)**	99.7917 (-0.0700)
Gelu	99.9567 (+0.4034)	99.8033 (-0.0784)

Loss

激活函数\损失函数	SoftmaxCrossEntropyLoss	HingeLoss
Relu	0.000159	0.031574
Gelu	0.000339	0.029386

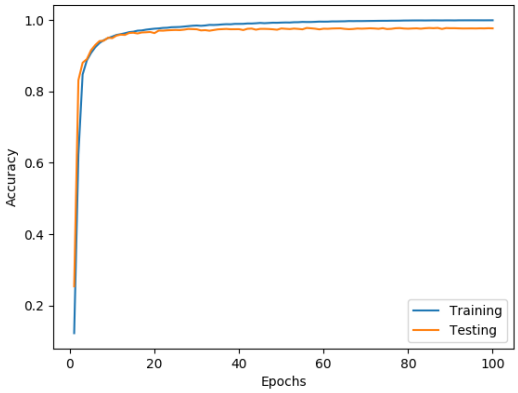
耗时（s）

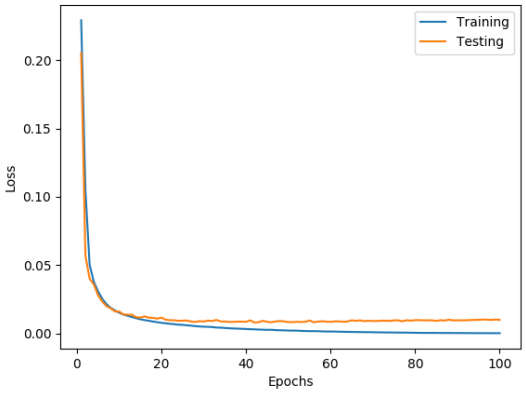
激活函数\损失函数	SoftmaxCrossEntropyLoss	HingeLoss
Relu	139.63 (+41.96)	141.44 (+48.39)
Gelu	1545.10 (+1237.21)	1595.43 (+1286.41)

训练曲线

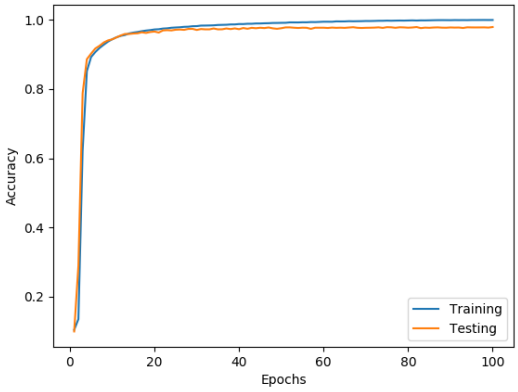
Softmax Cross-Entropy Loss

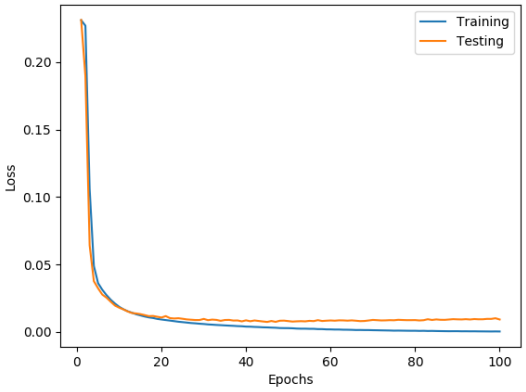
Relu





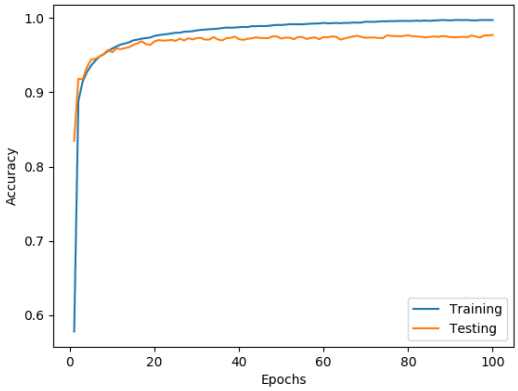
Gelu

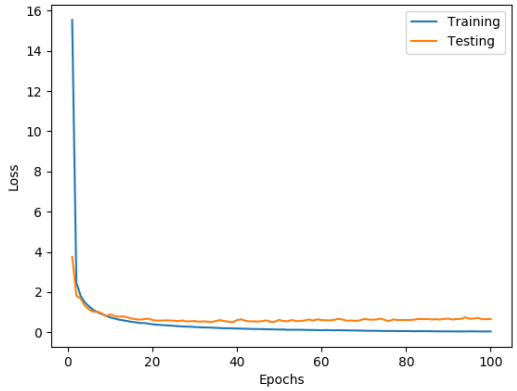




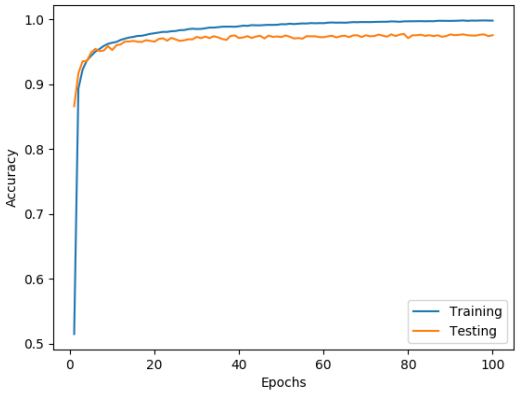
Hinge Loss

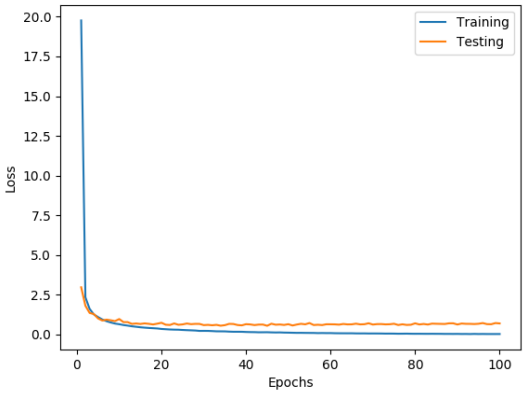
Relu





Gelu





讨论

通过双层MLP的实验结果和训练曲线可以看出，加深MLP层数在这个简单的数字分类任务上并不能起到很好的加强效果，反而由于参数量的增加导致模型收敛缓慢、训练缓慢和过拟合等问题。

实验总结

通过此次实验我熟练掌握了MLP的结构，通过调参和对比不同激活函数和损失函数的实验结果掌握了这些常见激活函数和损失函数的性质，最后对比深层和浅层MLP的实验效果得知模型并非越深越好，而是要根据任务选择和设计合适的模型。