南开大学
Nankai University

南　　开　　大　　学

网 络 空 间 安 全 学 院

深度学习平时作业

MLP 实验报告

穆禹宸　2012026

年级：2020 级

专业：信息安全、法学双学位班

指导教师：侯淇彬

2023 年 6 月 23 日

目录

一、 原始 MLP 网络结构

(一) 网络结构

```
Net(
  (fc1): Linear(in_features=784, out_features=100, bias=True)
  (fc1_drop): Dropout(p=0.2, inplace=False)
  (fc2): Linear(in_features=100, out_features=80, bias=True)
  (fc2_drop): Dropout(p=0.2, inplace=False)
  (fc3): Linear(in_features=80, out_features=10, bias=True)
)
```

基础的 MLP 网络为一个三层的网络，其映射关系如上面的结构所示。

(二) 实验结果

图 1: **基础 mlp 的损失**

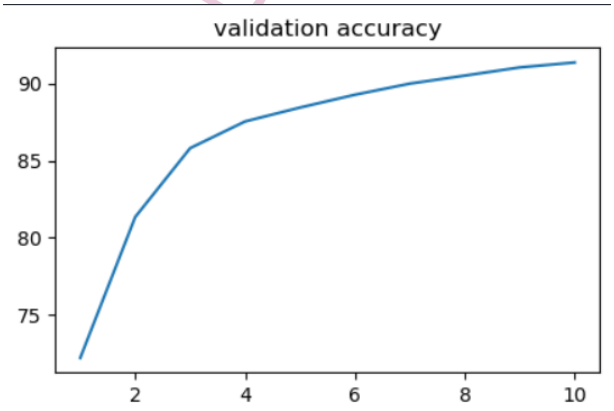

图 2: **基础 mlp 的准确率**

可以看出前馈神经网络的训练取得了良好的效果。

二、 实验过程

对于调整网络结构和超参数的过程，这里的步骤较多。

首先我调整了网络的结构。

对于网络结构，我分别尝试了不同的层数。其中，下面的这一种结构是我实验之后效果最好的。

```
MLPNet(
  (fc1): Linear(in_features=784, out_features=512, bias=True)
  (fc1_drop): Dropout(p=0.2, inplace=False)
  (fc2): Linear(in_features=512, out_features=256, bias=True)
  (fc2_drop): Dropout(p=0.2, inplace=False)
  (fc3): Linear(in_features=256, out_features=128, bias=True)
  (fc3_drop): Dropout(p=0.2, inplace=False)
  (fc4): Linear(in_features=128, out_features=10, bias=True)
)
```

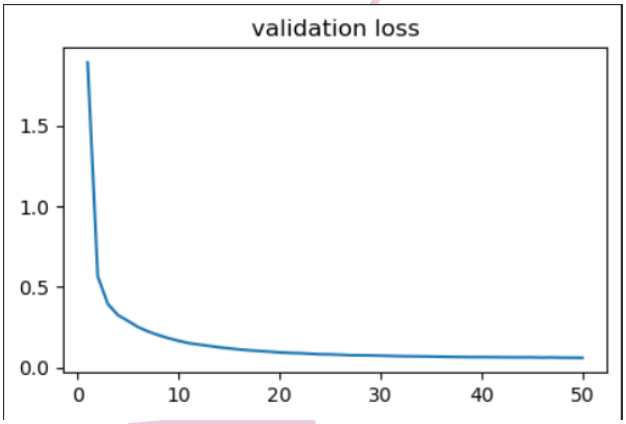

图 3: **调整后 mlp 的损失**

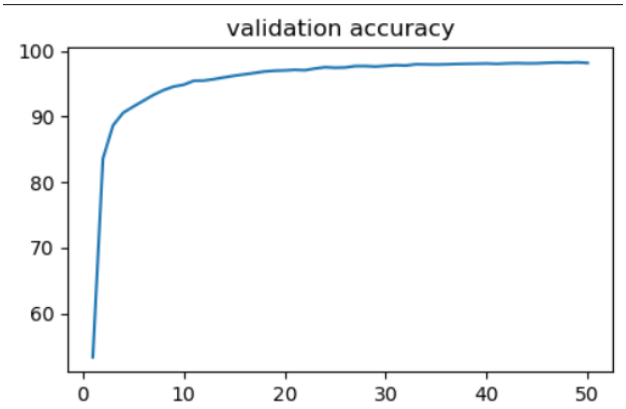

图 4: **调整后 mlp 的准确率**

这里我们看到，当层数变深的时候，网络取得了更好的效果，最终的准确率达到了 98.35% 的水平。这是我使用简单的 MLP 所取得的最好的效果。

而后，我尝试继续加深网络，我实现了五层的网络，其结构如下：

```
MLPNet(
  (fc1): Linear(in_features=784, out_features=512, bias=True)
  (fc1_drop): Dropout(p=0.2, inplace=False)
  (fc2): Linear(in_features=512, out_features=256, bias=True)
  (fc2_drop): Dropout(p=0.2, inplace=False)
  (fc3): Linear(in_features=256, out_features=128, bias=True)
  (fc3_drop): Dropout(p=0.2, inplace=False)
  (fc4): Linear(in_features=128, out_features=84, bias=True)
  (fc4_drop): Dropout(p=0.2, inplace=False)
  (fc5): Linear(in_features=84, out_features=10, bias=True)
)
```

然后得到了如下的结果:

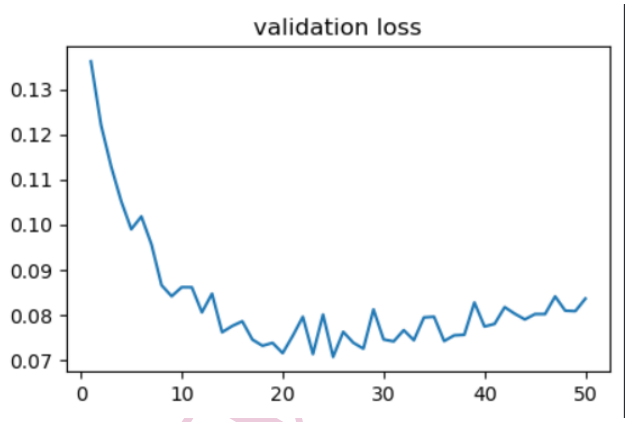

图 5: **调整后 mlp 的损失**

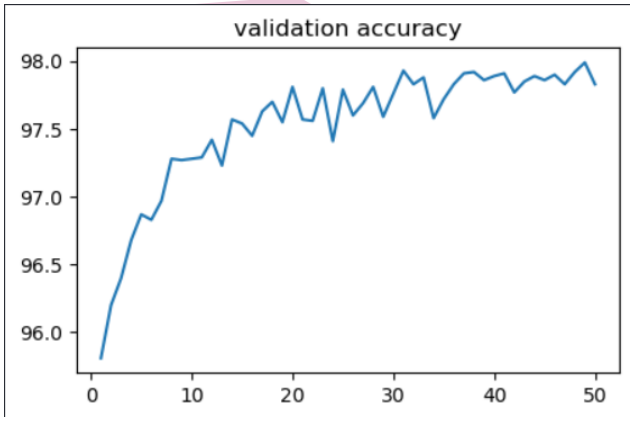

图 6: **调整后 mlp 的准确率**

这里我们可以看到，我们的网络的表现并没有刚才的网络得到的结果好，没有达到 98% 甚至在最后好像表现出来了一定的过拟合的现象。

这里我认为是出现了一些梯度消失和梯度爆炸的现象，导致了网络出现退化，因此五层的神经网络并不如四层和三层的效果好，然而由于数据集本身简单，因此仍然与最好的表现相近，如果是更加复杂的数据集，那么可能表现会更差。也就是说，在不采取后面要讲的残差网络的情况下，更深的网络并不会带来更好的效果。

在确定了四层网络之后，我对每一个全连接层的维度做了很多探索，以下是一个例子。

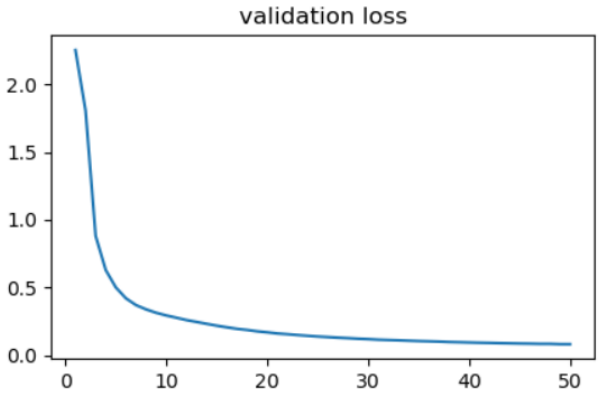

图 7: **调整后 mlp 的损失**

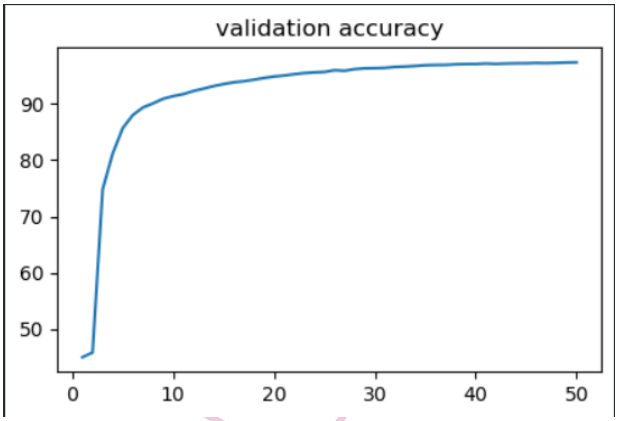

图 8: **调整后 mlp 的准确率**

三、 实验心得——参数和结构调整

首先，对于神经网络，我们知道其在层数极深的时候会出现网络退化的现象，这是因为出现了梯度消失和梯度爆炸等现象，导致其在层数更深的时候效果更差，因此，在选择网络的层数的时候，不能选择非常深的神经网络。对于本次实验的简单数据集来说，只需要 2-4 层即可，对于五层及以上的网络来说，会造成较差的效果。

对于全连接层的超参数选择：

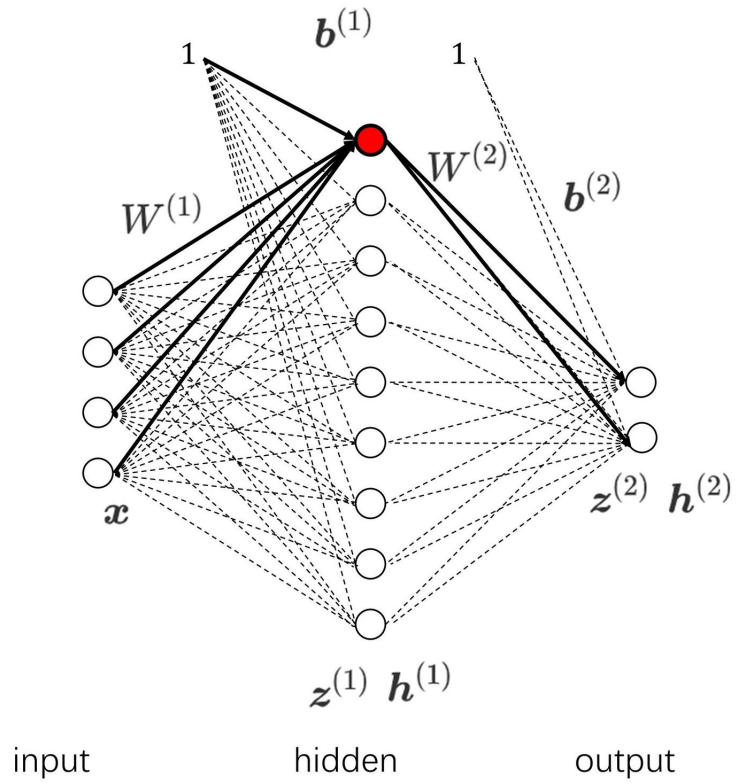

图 9: **全连接网络**

[1]

从上图我们可以看出，全连接层的本质上是一个映射，即把一个维度的向量空间之中的一切向量映射到另一个维度的向量空间之中，因此我们实际上可以得到一个结论就是 MLP 网络是在逐步地将原始向量空间（取决于图片大小）映射到最后的分类结果的空间之中。由于数据集较为简单，各项实验的大致结果都差不多，没有巨大的差别，至少都能达到 95% 的正确率这个级别。不过这里需要注意的基本原理是，全连接层的输入和输出维度不应该有过大的差距，这样才能保证每一层都能够保留足够多而且足够重要的信息。在保证这一原则的基础上，我们发现我们的空间维度在逐步减小，以我得到的最优结果为例：是从 784->512->256->128->10 这一过程，因此网络可以很好的学习到内容。当然，选择 2 的倍数也是一个小的技巧，可以得到一些更好的效果，不过对于更加复杂的模型来说应该考虑与其他参数配合进行调节。另外，我也做了其他尝试，比如我发现全连接层的参数选择并不合理的时候，也会出现很差的结果，比如如果我们把 out_features 设置得比映射之前的维度更大，这就有点儿类似于 GAN 的生成器，这样不光没有学习到足够的内容，还加入了一定的噪音，对于 MLP 是不利的，这样我们不会取得良好的效果。

而对于 epoch、batch_size、learning rate 等超参数，我认为其中 epoch 的调节是比较简单的，只需要在算力足够的情况下尽可能的保持更大的轮数，这样才能够得到更好的迭代效果。而对于 batch_size，理论上如果内存或者显存更大，也许将全部数据集进行计算是更好的选择，因为空间不够大才会分批训练，但是当我们必须采取分批训练的时候，选择合适的批次大小也许可以适当的引入一些噪声，这样可以取得一定的防止过拟合的效果，也许并不比全部数据集一起训练的结果更差。而对于学习率来说，其调节应该保持一个较小的范围，在算力充足的情况下，小的学习率才会达到最优效果，不然会在最后效果的附近反复“震荡”，却始终达不到最好的结果，不过过于小也不可行，那样会发现模型收敛非常缓慢甚至不收敛，这样的结果我也得到过。因此公认的范围是在 0.01-0.001 这一范围之内比较好，不过还是需要根据具体的情况作出更多调整。

四、 实现 ResMLP

我还实现了 RESMLP，这个模型引入了残差结构。

(一) 网络结构

其网络结构如下所示:

```
ResMLP(
  (in_layer): Linear(in_features=784, out_features=256, bias=True)
  (hidden_layers): ModuleList(
    (0): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.2, inplace=False)
    )
    (1): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.2, inplace=False)
    )
    (2): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.2, inplace=False)
    )
    (3): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.2, inplace=False)
    )
    (4): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.2, inplace=False)
    )
  )
  (out_layer): Linear(in_features=256, out_features=10, bias=True)
)
```

NKU

(二) 实验结果

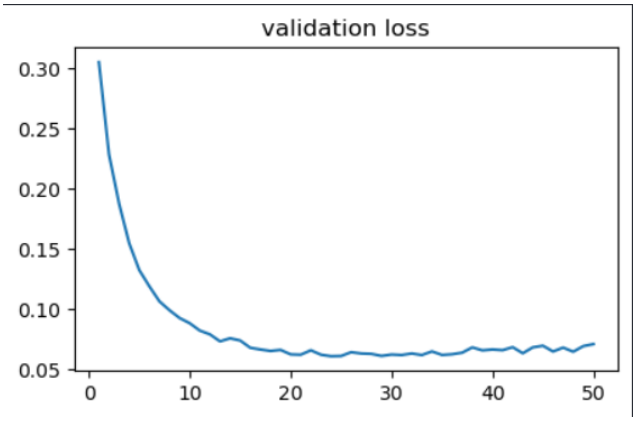

图 10: **RESmlp 的损失**

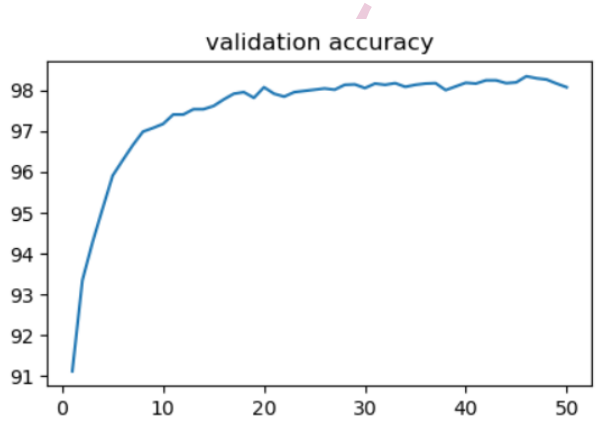

图 11: **RESmlp 的准确率**

这个模型我进行了一些简单的调参工作，不过发现并没有特别明显的提升，虽然其最终效果达到了 98.56%，比之前简单的 MLP 取得的效果要好，但是并不明显，猜测可能与数据集过于简单有关。

参考文献

[1] https://www.bing.com/images/search?view=detailV2&ccid=HjGrFQdX&id=066363B72AED59C304256CB6366C89DDDE9FC39A&thid=OIP.HjGrFQdXYglLc5JUui8ncwHaHp&mediaurl=https%3a%2f%2fpic1.zhimg.com%2fv2-572d6814335bee9e23d408926b73f2b8_1440w.jpg%3fsource%3d172ae18b&exph=1488&expw=1440&q=%e5%85%a8%e8%bf%9e%e6%8e%a5&simid=608049833559734406&FORM=IRPRST&ck=A8A10B8A1CB692CE8DA34832E227C282&selectedIndex=2&ajaxhist=0&ajaxserp=0.

NKU