计算机图像处理大作业实验报告

报告人： 何峙

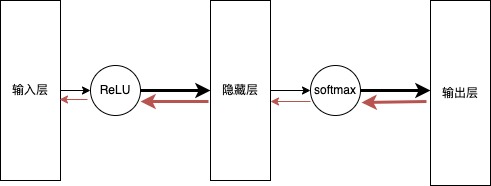
学号： 21215122

专业：大数据与人工智能

实验步骤：

1. **全连接神经网络**

本实验主要实现一个两层的全连接神经网络，基本架构如下所示：



其中黑色箭头为前向传播，红色箭头为反向传播（线条粗细只是用来区分不同层的关系）。

* 前向传播即输入向量进行线性变换后接一个非线性变换的运算，本实验使用ReLU和Softmax函数作为非线性变换，过程如下：

（1）

（2）

（3）

（4）

（5）计算损失函数，本实验使用交叉熵作为损失函数：



反复套用（1）、（2）、（3）步，可搭建更多层的全连接神经网络。

* 反向传播，计算Loss损失函数，Softmax函数、ReLU函数，还有线性函数Z = W·X +B各函数分别对W，对B和对X的偏导数，其目的是根据下式更新权重W和B：





其中F为各层对应使用的函数，lr为学习率，本实验使用随机梯度下降方法（SGD）。

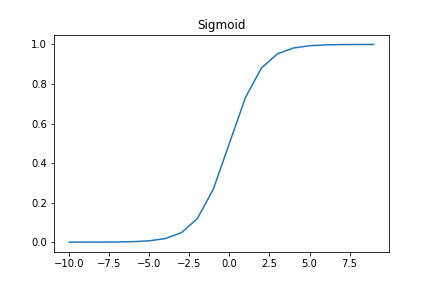
综上，不断的使用前向传播——后向传播——前向传播——后向传播——…，迭代到一定的轮数Loss函数收敛到某个值，即可停止网络的训练，至此完成全连接神经网络权值的学习。

本实验还进行了三个小试验：

* 激活函数Sigmoid、ReLU和Leaky ReLU的讨论

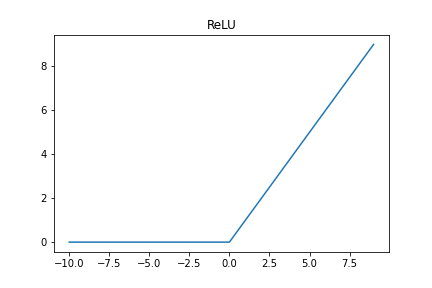
对比三个函数：

（1）Sigmoid



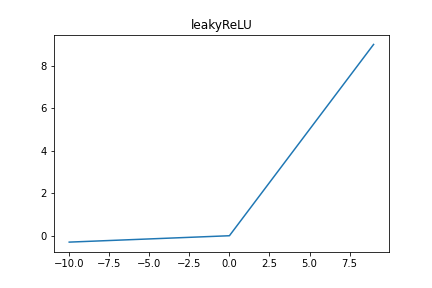
靠近0的值导数较大，趋于负无穷和正无穷时，导数越趋于0。所以使用其作为激活函数时，在0附近，有较好的激活性，但在正负无穷区域容易发生梯度消失现象。

（2）ReLU



由其图像可知，大于0时，梯度为常数，不会出现梯度消失。但小于0时，梯度为0，这是神经元不会被激活，即不会被训练。

（3）LeakyReLU

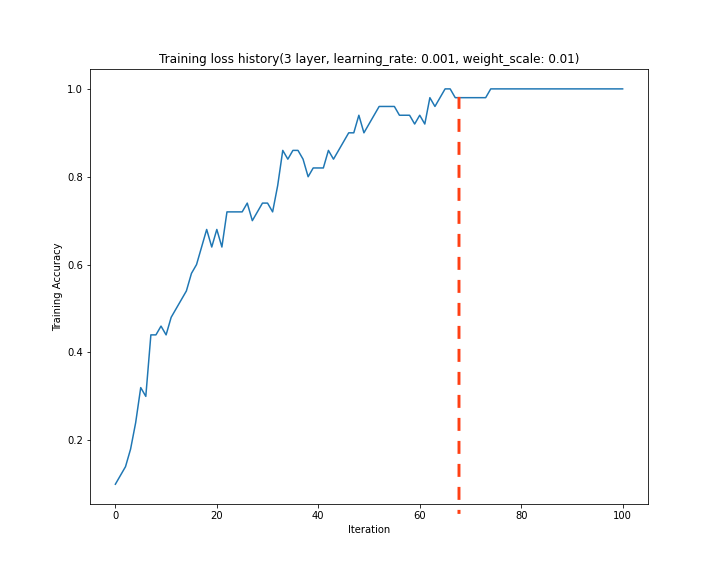


大于0的时候与ReLU类似，会保留一个比较小的负的值，使得此时梯度也不会消失。

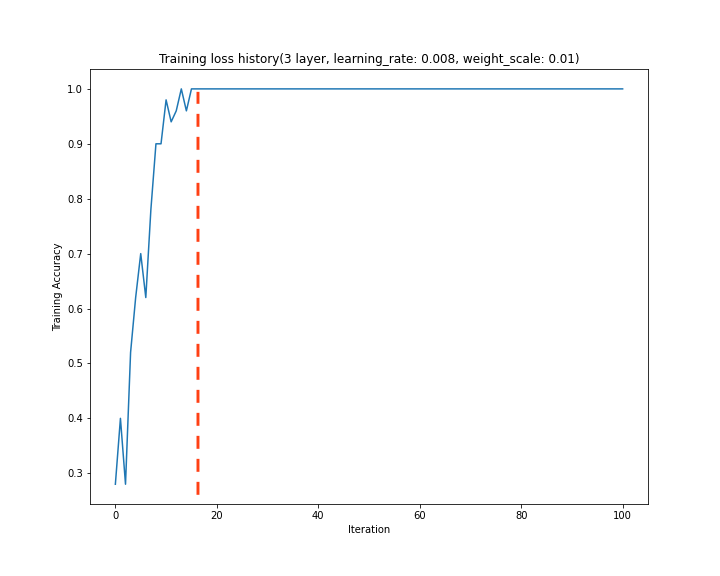
其实被实验使用LeakyReLU也可以，只是效果不会太明显。

* 使用3层全连接网络在20轮内、用50个样本、使训练准确度达到100%

最初使用的学习率为1e-3，发现20轮里无法达到100%的训练准确度，便尝试迭代100轮，结果如下：



发现大概在70~72轮，网络训练准确度才能达到100%，猜测是迭代速度过慢所致，于是尝试将学习率增大为8e-3，可达到要求：a

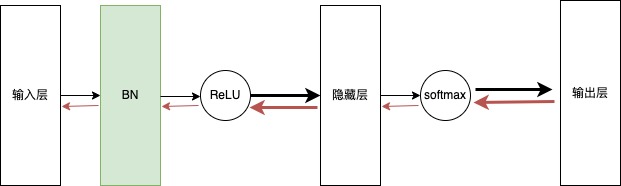


* 使用5层全连接网络在20轮内、用50个样本、使训练准确度达到100%：

随着网络深度的增加，参数调整比之前难度增加。这里调节学习率后，发现影响不大，调节weight\_scale发现对结果影响比较大，需要经过多次调整后才能达到100%准确率。说明对于单纯增加网络深度，其权值的调整会变得特别艰难。

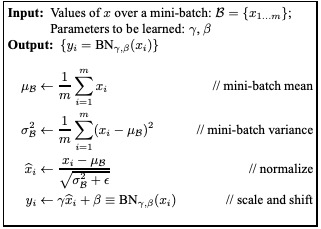
1. **归一化**

当我们训练深度神经网络的时候，不断的网络层的处理也会使得原始分布发生改变。更严重得使，随着权重得不断更新，每一层得输入特征的分布也会不断地发生漂移。归一化的引入使得数据经过网络层后继续保持均值为0方差为1的分布，基本架构如下：

****

数据经过全连接层后，在流入激活函数之前，先进行batch normalization（上图绿色部分）处理。在增加更多层网络时，可以套用这种模式，按需进行batch normalization。

Batch normalization流程如下：



# （图1, 来自“batch normalization”论文（插入脚注《Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

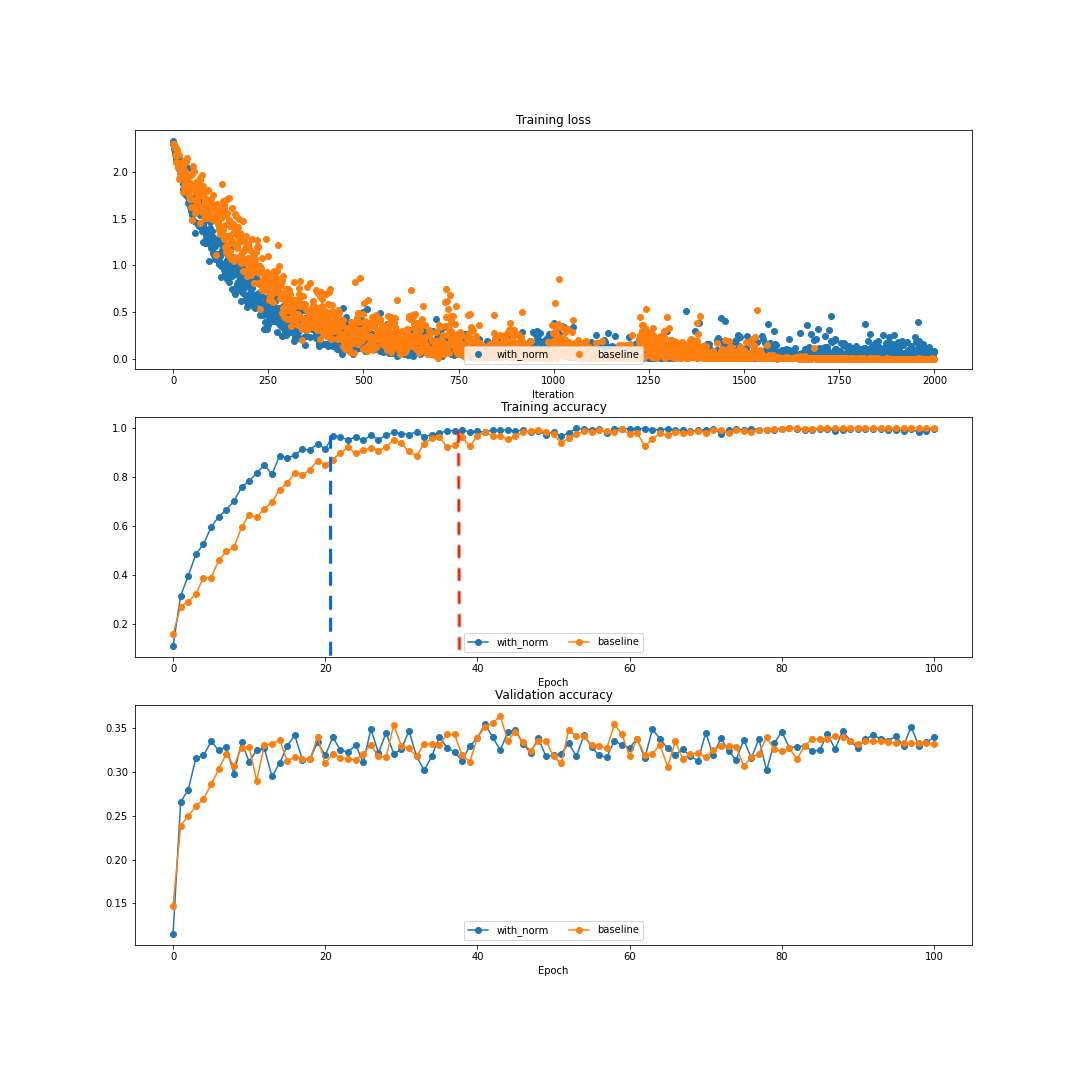
》<https://arxiv.org/abs/1502.03167>）的算法说明）

参数γ和β一起参与训练。

这里进行了四个小实验：

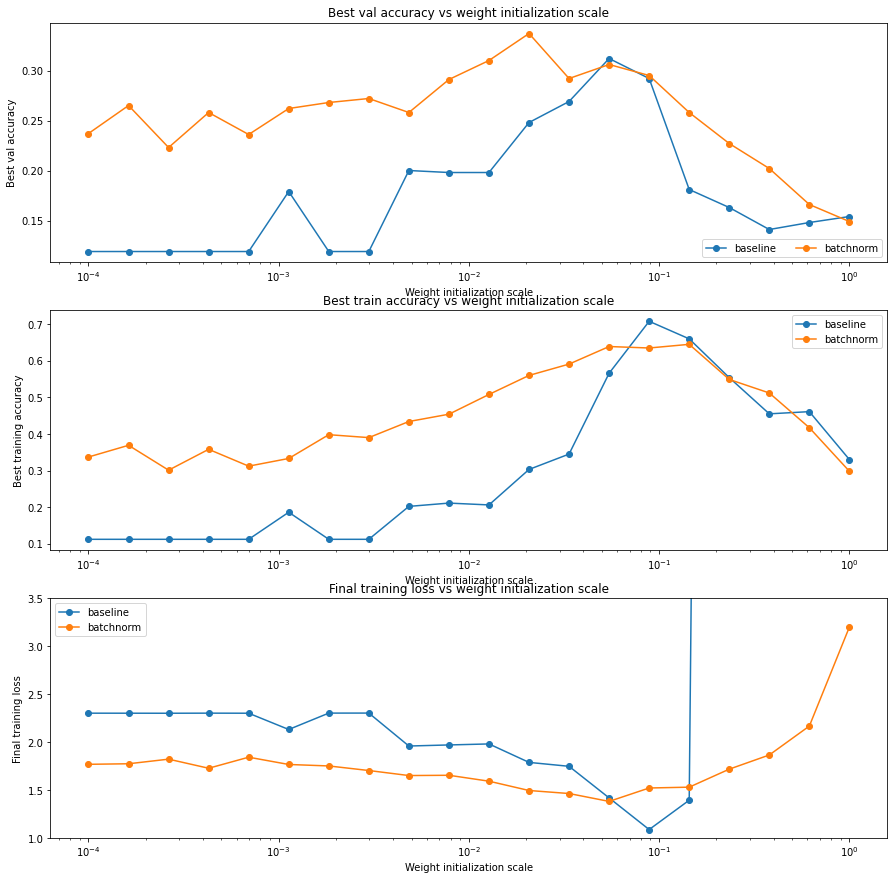
* 使用batch normalization和不使用batch normalization的区别

如下图所示，可见使用batch normalization时算法收敛的更快（垂直蓝线标记），收敛的也更稳定：



* batch normalization与初始化权重的联系

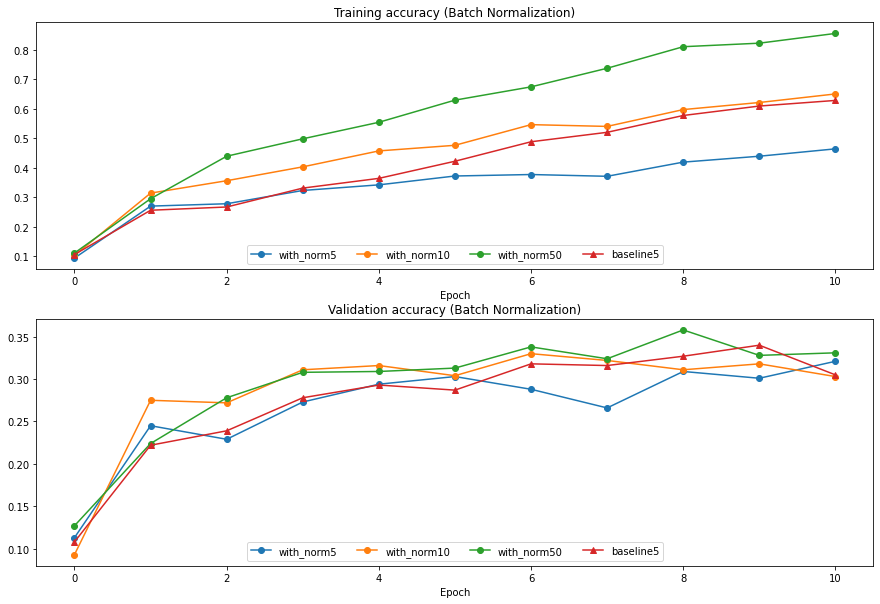
本实验定义一个8层的神经网络，然后分别比较不同的权重初始化参数下，带batch normalization和不带batch normalization时的网络的性能差异，结果如下图：



可见，batch normalization使得网络的训练对网络参数初始化变得不那么敏感。若不带batch normalization，对于权重初始化过小，则参数分布逐渐集中在0附近，导致回传的梯度乘以参数之后变得非常小。对于权重初始化过大，则参数分布逐渐两极化，出现梯度消失现象。

* batch normalization与batch size的联系

初始化一个带BN层的6层神经网络，然后使用不同batchsize大小的参数进行训练。对比如下图：

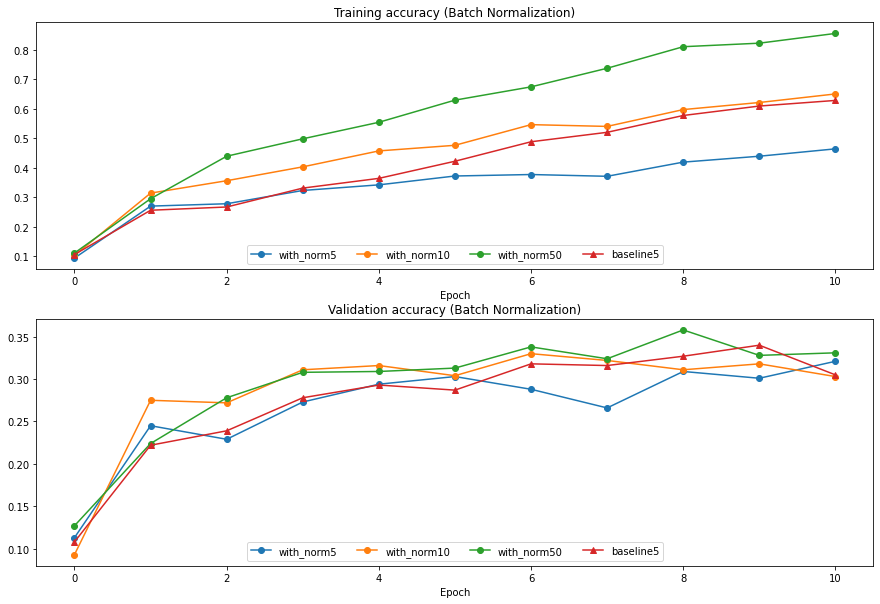


可见，随着batchsize的增加，模型收敛的更快，说明batch normalization层适合大的batchsize，因为大的批量使得样本batch得均值和方差估计得更准确。

* 使用Layer normalization

batch normalization是对batch内样本的每个特征做归一化，而layer normalization是对一个样本中所有特征做归一化。

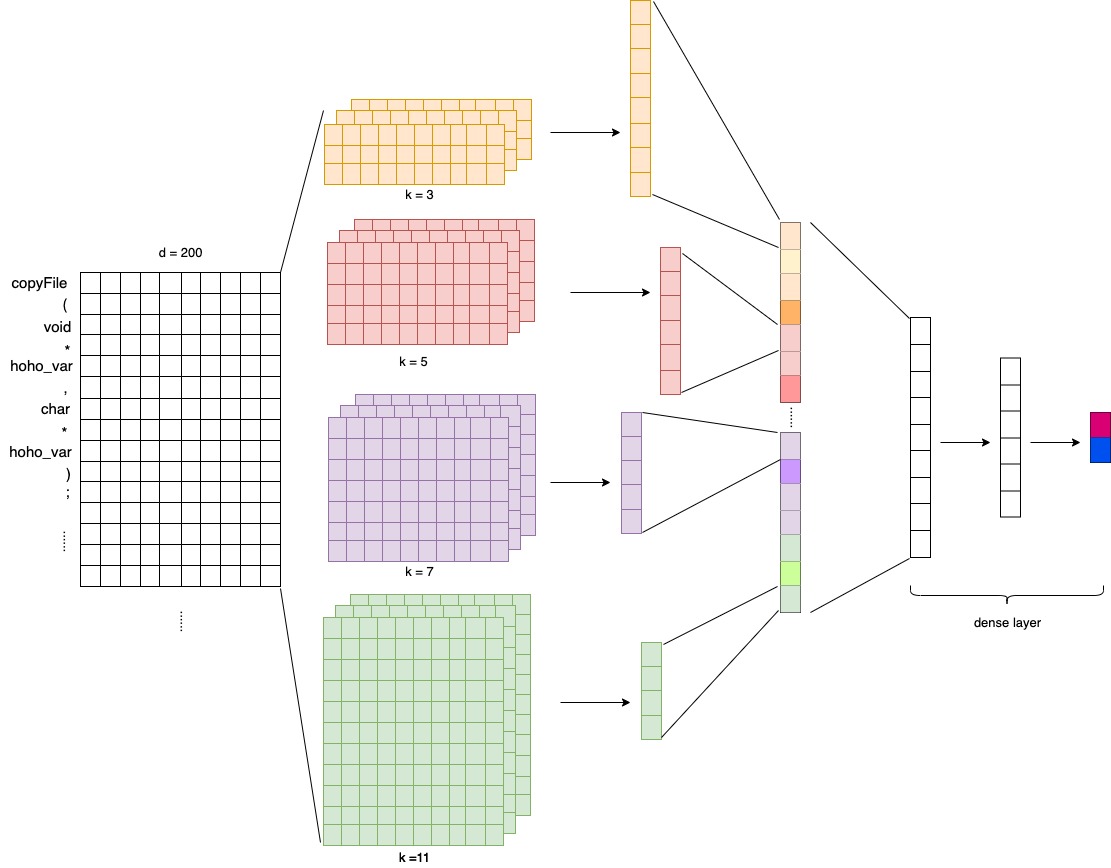
本实验主要比较不用batch size对使用layer normalization性能的影响，如下图：



可以发现，batch size对layer normalization的影响比batch normalization影响小。

1. **CNN**

卷积神经网络基本架构为卷积层、池化层、全连接层等，如下图（这里现实了4种不同大小的卷积核）：

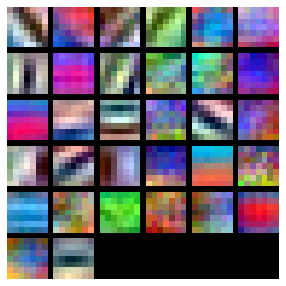
****

本实验搭建一个三层神经网络，关键是卷积计算和池化计算。

1. 卷积计算即每个卷积核与图像每个像素的乘积的加和
2. 池化计算是对卷积计算的结果进行取最大化或平均值的运算。

其余操作（前向运算，反向传播，等）跟实验1的全连接层类似，这里不再累述。

最后可视化卷积核，可见其可以提前图像的各类特征，如横竖轮廓、色彩饱和度，等。

****