计算机视觉第一次作业实验报告

一、代码架构（参数与函数功能）

1. 解析读取数据使用 python 的 struct 库进行，即 data_fetch_preprocessing() 函数，在其中将图片数据归一化，并将训练集标签设为 10*1 维向量，其隶属类值为 0.91，其他类值为 0.01，最后返回四个数组 x_train, y_train, x_test, y_test，x 代表图片数据，y 代表标签数据，train 代表训练数据，test 代表测试数据。

2. MLP 代表网络结构，其中传入参数可根据参数查找参数种类自行设定，参数包括网络结构，学习率（最大、最小），正则化参数 lamda，epoch，batch 等等。

其中函数 leaky_relu 与 dleaky_relu 分别代表激活函数与其导数，softmax 函数代表输出分类函数，loss 函数计算 loss 值。

forward()与 backwards()分别代表前向传播与反向传播过程，其运算过程都通过矩阵运算得到。

train()函数代表训练过程，共训练 epoch 轮，每轮训练 iteration 个随机 batch 大小的批量数据，最后返回网络矩阵参数与训练曲线数据矩阵。

calculate()函数通过输入数据计算当前网络前向传播预测后的 loss 和分类精确率。

compute_eta_t() 函数用于计算一个下降周期内的学习率值矩阵，compute_learnrate()用于计算所有 epoch 内的变化学习率值，其中通过 Ti 可设置重启时刻（注：其和应等于 epoch 数，以确保学习率个数足够）。

predict()函数通过前向传播计算预测准确率。

3. randomst 与 randomit 分别通过传入数量，最小值，最大值而生成随机小数、整数向量，以为参数查找提供随机参数。

4. train_the_model()函数基于随机参数训练生成 basenum 个不同模型，每训练完成一个模型就计算输出它的测试精度，并在每轮计算展示最优模型参数与准确率，在模型数等于 basenum 后输出保存最优模型的 config 与网络矩阵参数。

5. save_the_model, draw_the_curve 函数，功能如其名所示。

6. PCA 通过传入主成分数 n，将原数组降维至（xnum，3），以利于后续作图。

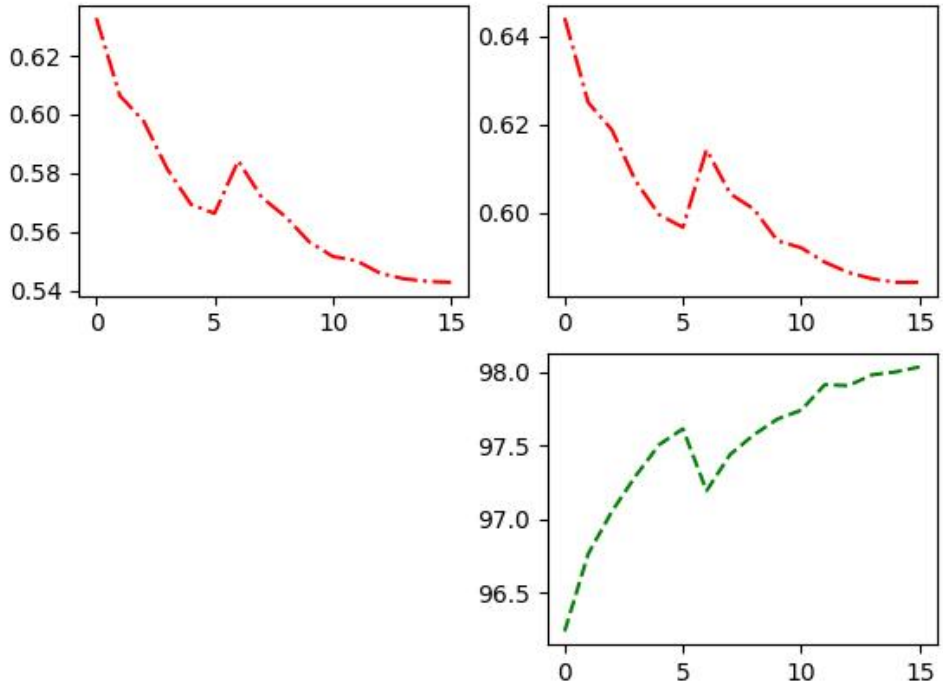
7. load_the_model 函数通过路径，读取并返回对应模型。

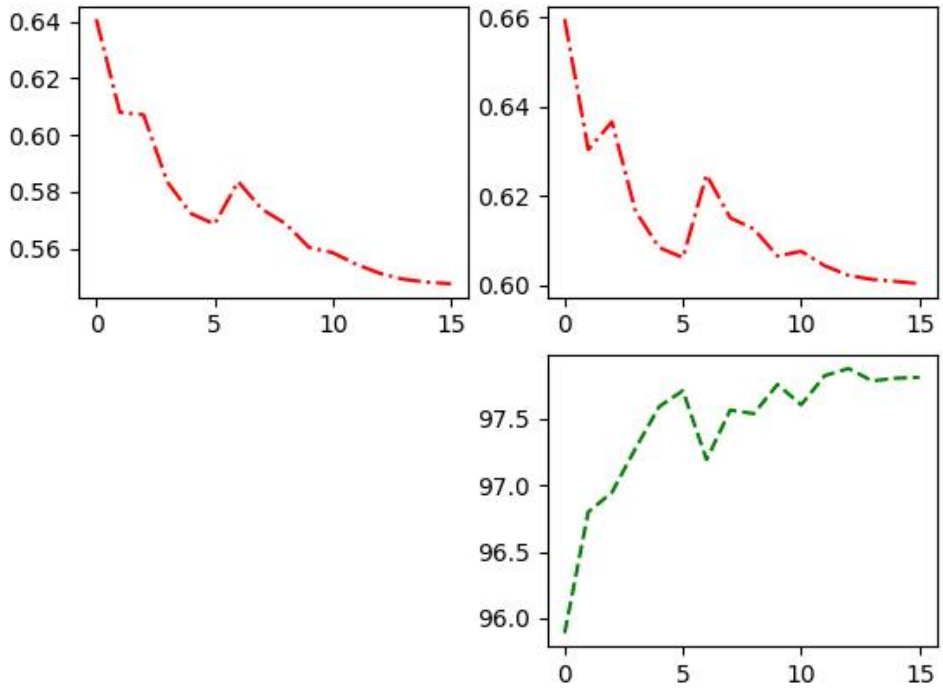
二、基于传统神经网络的各种优化措施的作用效果

1. 学习率下降策略，可不断逼近极优点，观察未加学习率下降与添加学习率下降策略，前者对初始学习率要求较高，若设置不合理，很容易产生不收敛或准确率较低的情况，而学习率下降对初始学习率要求不是很高，因为会逐渐下降至合适值，设置最小学习率可防止过拟合，有更好的适应性和训练收敛效果，离极优点更近。在此基础上，若增加重启过程，则可以帮助网络跳出鞍点，向最优点靠近，故网络学习率策略使用余弦退火策略，限制死总的 epoch 数，尝试各种重启时刻组合，查看训练网络的测试精度。

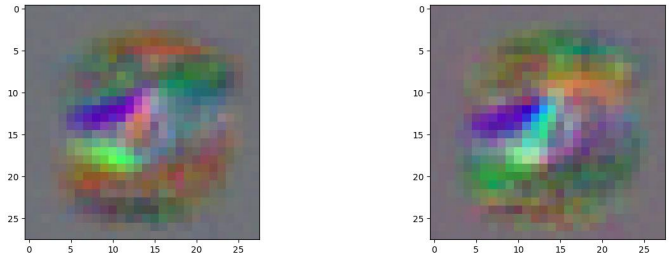
发现本数据集特征似乎不存在峰，只需学习率不断下降即可得到很好的结果（在 batch=1 时测试发现），但当加入优化 SGD 后，重启时刻则需要详细考虑，若重启时刻过多，则模型来不及收敛到较好情况，若重启时刻位置靠后，也容易打乱其之前训练的较好的结果，若总 epoch 数过多，会导致模型训练时间过长，浪费成本。故在不断测试后，选取 epoch=16，Ti=[2,4,10]的方案，开始时迅速重启可以帮助还未收敛的模型迅速收敛，中间时刻的重启帮助网络模型跳出鞍点，最后 6 个 epoch 的长时间段帮助模型逐渐收敛。

2. L2 正则化中较好的正则化强度不仅可以帮助模型提高精度，还有利于加快模型收敛速度。下图展示了输入层数据维度 784，隐藏层 128，输出层 10，learningratemin=0.0193，learningratemax=0.5423，batch=32，在 lamda=0（未正则化，上图）与 lamda=1（正则化，下图）后的训练 loss，验证 loss 与验证 acc。未正则化测试准确率为 98.24%，正则化准确率为 98.26%。二者曲线下降趋势大致相同，但添加正则化后的训练 loss 更大，防止其过拟合，且在相同 epoch 时间内，其验证集准确率更高，受重启影响更大，更有可能跳出鞍点，故加入 L2 正则化有利于神经网络的训练。





下图为二者的权重矩阵降维归一化可视化图（左图，未正则化；右图，正则化）：



可以发现正则化虽一定程度上影响网络结构，但起主要作用的区域节点仍然变化不大，且分布大致以同样的趋势，体现出神经网络同参数的统一架构。同时可以看到右图中明亮区域占比更大，即其起主要作用的节点更少，即其网络结构更简单，这也正是正则化的优点之一。

3. 优化 SGD 设置随机 batch 个数据进行训练，相较于原先每次一个数据的效率而言，其训练速度更快，因为每次学习到的特征更多，速度大概为 batch=1，epoch=10 与 batch=64，epoch=16 的总耗时大致相同。但在最后训练精度上差距不大，未加 SGD 最高精度 98.31%，加 SGD 最高精度 98.33%，具体参数可以在 githua 网站上看到。

4. 参数查找

开始时查找的参数为（batch，lrmin，lrmax），固定其他参数 hidelayernum=128，lamda=0，以及其他参数，查找 100 个模型结果可在 GitHub 文件 result 文件夹中 hdnum=128 文件夹中的 txt 文件查看，精确度最高为 98.33%，普遍模型精确率位于 97.55%以上，且方差较小，精确率普遍较高。

后来发现作业要求为隐藏层神经元数量，学习率与正则化强度，故代码为此三个参数的查找结果。

查找发现下面由于需要改变网络结构，故其不同模型精确率差别较大，通过这三个参数查找得到最高精确度为 98.26%，且方差较大。也可能由于查找时间不同，第一类型的查找大致十小时，第二类型查找大致 10 个模型。

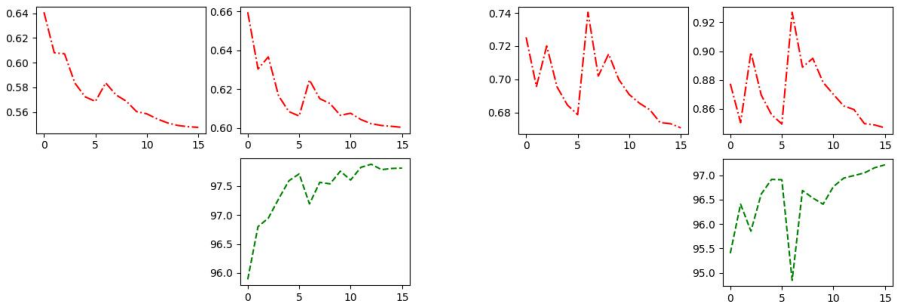
batch 的设置，对第二种查找影响很大，根据测试，大致 50-120 的 batchsize 训练得到的测试精度较高。

三、不同精确度的模型 loss、acc 曲线与权重可视化对比（所有模型与图片均在网盘模型中）

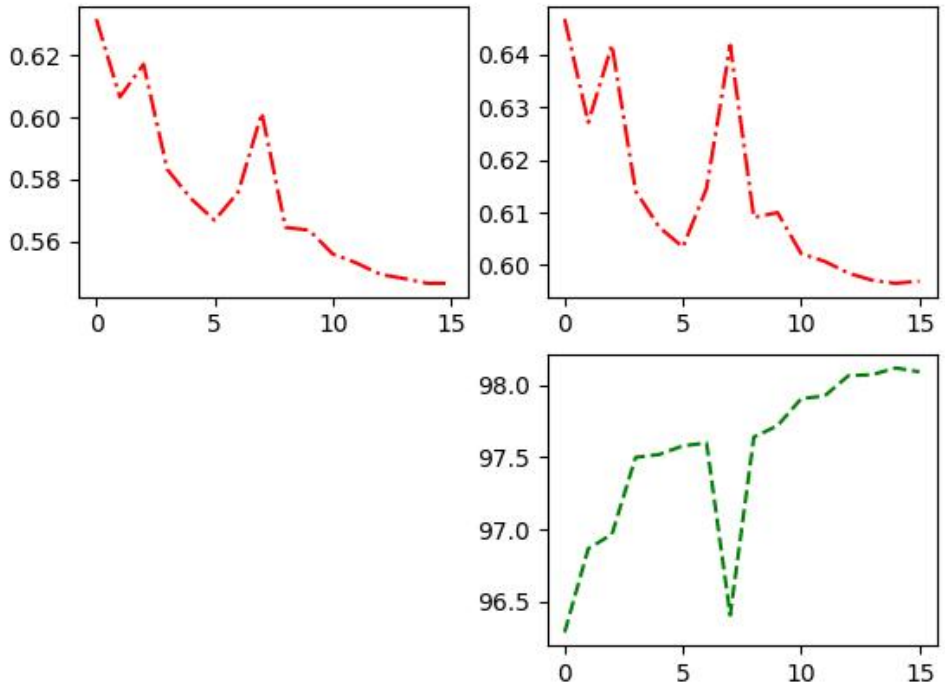
1. 参数：config=[128, 1, 0.0193, 0.5423]，测试准确率：98.26%

2. 参数：config=[121, 65, 0.0851, 0.4919]，测试准确率：97.76%

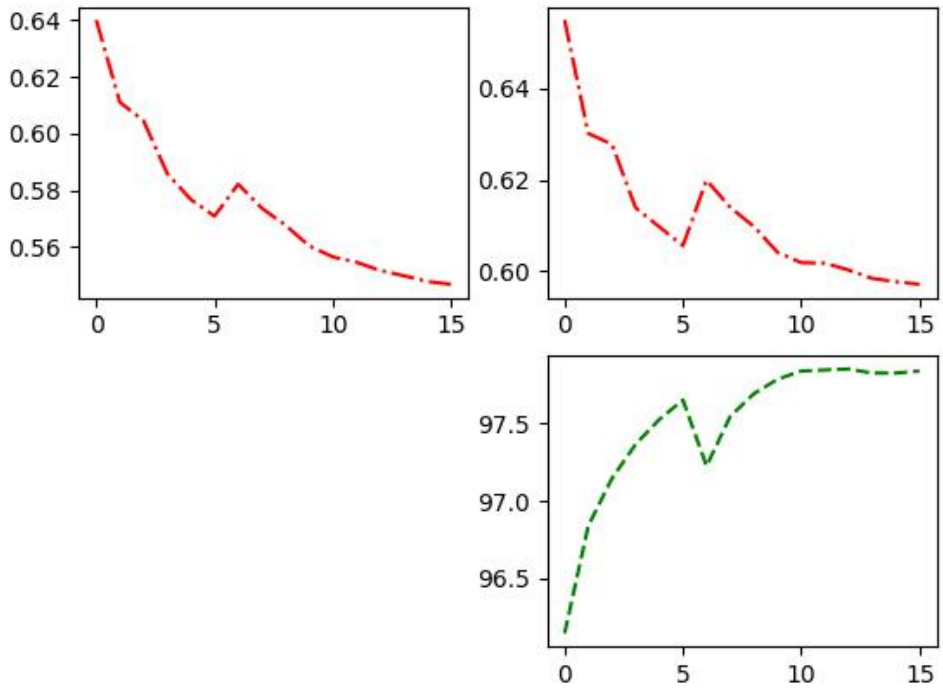
（1） 曲线对比（左 1 右 2）：



可以看到，右图中其准确率以及 loss 曲线受到学习率重启影响过大，导致其最终收敛结果不好，即学习率重启不但没帮助其跳出鞍点，反而使其退化，重新训练，故产生该现象的主要原因可能为其正则化强度太大，若将正则化 lamda 设为 1，可以发现其准确率变为：98.12%，提高了很多，其 loss 曲线如下：

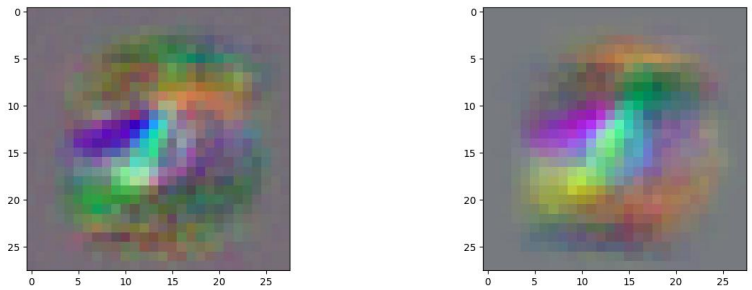


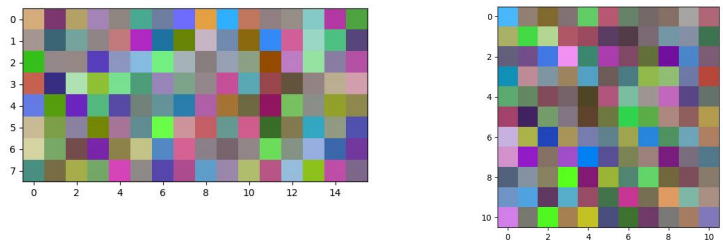
可以看到准确率图像确实减小少了许多，但第二次重启仍对其有较大影响，故接下来可以考虑减小最大学习率，以使其重启时准确率下降更少。将 lrmax 由 0.4919 设为 0.4，config=[121, 1, 0.0851, 0.4]，训练模型准确率为 98.15%，准确率确实得到了提高，其 loss 曲线如下：



其准确率在学习率第二次重启时下降更少，验证了之前的推论以及降低最大学习率措施的合理性。

（2） 二者的权重可视化（较大-较小准确率）：





可以发现，准确率较大的 w1 矩阵可视化图中，其较亮区域有紫色群，青蓝色群

以及橙色群，而准确率较小的 w1 矩阵可视化图中，较亮区域颜色较为单一且集中，即准确率较大的其主要成分较多，成分多样，更多的主要节点参与，故其模型准确率较高，而单一主要成分的模型准确率低，w2 可视化都较杂乱得不到明显结论。

四、网址链接

github repo 链接：https://github.com/theWaWang/l

上述模型百度网盘分享链接（需解压）：

https://pan.baidu.com/s/1Ix7DMfrGeHci-Uopf8GFRw

提取码：bOhm