Neural Network 的实现

一、数据

从 http://yann.lecun.com/exdb/mnist/ 下载 MNIST 数据集。

数据文件	说明
train-images-idx3-ubyte	**images 训练集，60000 个样本，每个样本包含 28x28 无符号整数**
train-labels-idx1-ubyte	训练集的标签，60000 个，标签是 0-9 之间的数字
t10k-images-idx3-ubyte	Images 测试集，10000 个样本，每个样本包含 28x28 无符号整数
t10k-labels-idx1-ubyte	测试集的标签，10000 个，标签是 0-9 之间的数字

二、实验过程

	网络结构	迭代	学习率	准确率	说明
1	784->200->10	1	0.5	0.8968	
2	784->200->10	1	0.7	0.871	增大学习率后，效果反而不好
3	784->200->10	1	0.3	0.9391	降低学习率后，效果变好
4	784->200->10	1	0.1	0.9490	降低学习率后，效果变好
5	784->200->10	1	0.01	0.9186	继续降低 lr 后，效果反而没有 0.1 的效果好
6	784->300->10	1	0.05	0.9475	0.01-0.1 的学习率比较好些
7	784->250->10	13	0.05	0.971	增加迭代轮次，效果更好
8	784->600->10	9	0.1	0.9712	增加隐藏层节点数量，效果好一点
9	784->1024->10	28	0.05	0.9751	增加隐藏层节点数量，效果好些
10	784->500->300->10	20	0.05	0.9412	增加网络层次，效果没有变好
11	784->500->150->10	19	0.1	0.9269	降低第 2 个 hidden 层的节点数量，效果没有变好

实验调优主要考虑的方面：

1. 增加网络的宽度：通过增加隐藏节点的数量，来增大网络宽度，实验效果确实有增加，参考 7 和 9；
2. 增加网络的深度：通过增加网络的层数，参考 9 和 10，但是效果并没有变好，准确度反而降低了一些；
3. 对样本做归一化处理：由于 MNIST 样本像素值是从 0-255，所以采用 Z=(X*0.99/255)+0.01 来处理样本；
4. 初始化：初始化 weight 和 bias 采用 random 初始化；
5. 后续可以再考虑正则化、或者其他激活函数；

三、代码说明

cpp 文件	说明
NNetwork.h	NNetwork 类的头文件
NNetwork.cpp	提供 train 方法用于训练样本，提供 predict 方法用于预测结果，基于 Eigen 的矩阵运算
MNistLoader.cpp	用于加载 MNIST 数据集
main.cpp	执行 train 和 test，对源数据归一化，并统计准确度

GCC 普通编译后，执行速度很慢，需要加上 -O2 参数来编译，这样速度明显变快。