手写数字字符识别实验报告

赵启元 F1803012 518030910289

一、 实验简介

本学期的课程上介绍了神经网络及其他多种机器学习算法。本次实验将使用 C++，实现两种不同的神经网络：单层神经网络（即多重分类与 Logistic 回归）和多层前向神经网络（使用反向传播算法训练），并比较两者对于人类手写字符识别的精确度和效率。

本次实验所使用的人类手写字符集来自 MNIST 手写数字字符数据集（测试数据官网：http://yann.lecun.com/exdb/mnist/），其中已经划分好了训练集和测试集。

本次的实验硬件环境是一台 CPU 为 Intel(R) i7-8750H CPU @ 2.20GHz，含 8GB 内存的个人笔记本电脑。软件环境是 G++ 6.3.0。

二、 准备工作

在正式进行实验之前，需要做一些预处理。

（一）图像读取

MNIST 的图像文件保存在一个二进制文件中，因此只需要使用编程语言自带的二进制文件读写操作就可以实现对图像及其标签的读写。在 C++ 中，此次使用的是自带的 cstdio 库。

（二）数值计算

神经网络的前向传播和反向传播均需要线性代数的相关数值计算和数值优化方法。在 Python 中，可以直接使用 numpy 库完成相关计算。在 C++中，尽管没有现成的方法，但仍可以自行实现。虽然和专门的数值计算库在各个方面都会有差距，但这样的实践可以帮助更好的理解神经网络相关算法的数学原理。

（三）评价机制

对于神经网络性能的评价有诸多标准，这里选取 F1 值为标准，即

$$\mathrm{F1} = 2\frac{precision \cdot recall}{precision + recall}$$

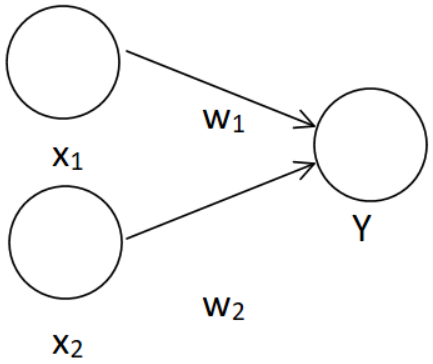
其中$\mathrm{precision} = \frac{TP}{TP+FP}$，$\mathrm{recall} = \frac{TP}{TP+FN}$。这里将会对每一个数字

进行 F1 值的计算。除了 F1 值之外，训练时间也将作为评价标准之一。

三、 单层神经网络

(一) 理论简介

单层感知机是最简单的一类神经网络。它只有一个输入层和一个输出层，没有中间层。



如图，这张图中就展示了一个有着两个输入神经元和一个输出神经元的感知机。

利用单层感知机可以实现简单的二元分类，即逻辑回归（Logistic Regression）。而使用 One vs All 的方法可以实现多元分类，即：如果有K个类，可以训练K个单层感知机，每一个单层感知机都将第i个类K_i作为正类，而将其他类作为负类训练。训练完成后，对于每一个测试用例，可以对所有K个分类器运行一遍，分类器中给出的结果最大的那一个所代表的类别就是对于这个测试用例的预测。

(二) 代码实现

单层感知机对手写字符识别的 C++描述版本在项目文件夹 One-Layer NN 下。本次代码实现中，神经元的激活函数为经典的 sigmoid 函数。使用的训练方法是 mini-batch 随机梯度下降法。使用的误差函数是交叉熵（cross-entropy）误差函数。

(三) 实验结果

实验结果如下：（每次固定选取前 2000 个测试样例）

使用每次从训练集中随机选取 100 个样例的 mini-batch 梯度下降方法，总共进行 100 次梯度下降，学习率$\alpha = 0.1$的情况下，有：（以下是代码输出）

```
Training Time: 15903ms
```

```
For index 0:
F1 score: 0.664207
For index 1:
F1 score: 0.852174
For index 2:
F1 score: 0.664688
For index 3:
F1 score: 0.666667
For index 4:
F1 score: 0.656863
For index 5:
F1 score: 0.0628272
For index 6:
F1 score: 0.692661
For index 7:
F1 score: 0.727273
For index 8:
F1 score: 0.521212
For index 9:
F1 score: 0.525164
```

使用每次从训练集中随机选取 200 个样例的 mini-batch 梯度下降方法，总共进行 100 次梯度下降，学习率$\alpha = 0.1$的情况下，有：（以下是代码输出）

```
Training Time: 32514ms
For index 0:
F1 score: 0.834646
For index 1:
F1 score: 0.914405
For index 2:
F1 score: 0.81
For index 3:
F1 score: 0.795
For index 4:
F1 score: 0.768116
For index 5:
F1 score: 0.5
For index 6:
F1 score: 0.754601
For index 7:
F1 score: 0.697297
For index 8:
F1 score: 0.60198
For index 9:
F1 score: 0.670968
```

可以看出，提升训练数据的量可以在整体上提高预测的精确程度，但因为 mini-batch 方法具有一定的随机性，所以表现有时也会不稳定。如上面的对数字 5 的识别，在提升了训练量后表现反而变差了。

使用每次从训练集中随机选取 200 个样例的 mini-batch 梯度下降方法，总共进行 100 次梯度下降，学习率$\alpha = 0.2$的情况下，有：（以下是代码输出）

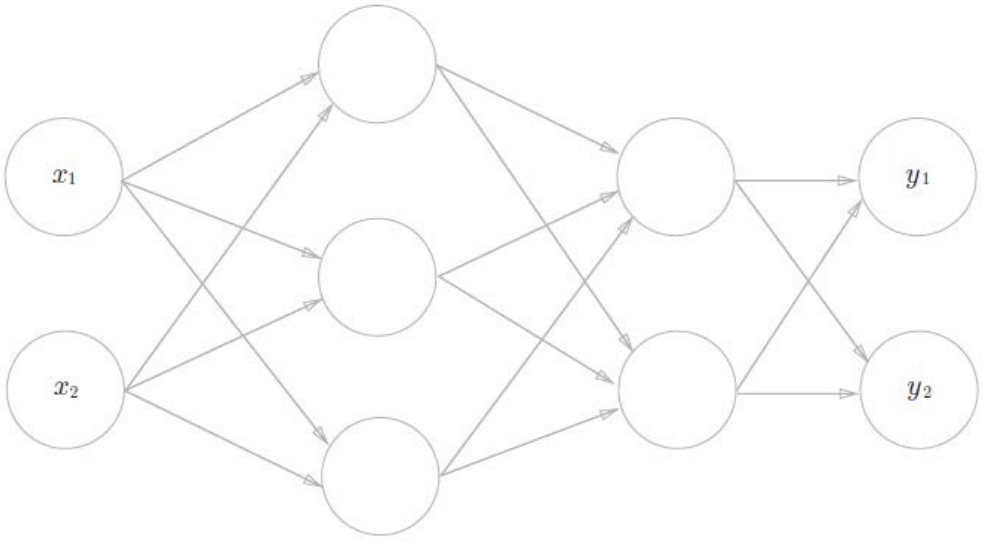
```
Training Time: 30716ms
For index 0:
F1 score: 0.885057
For index 1:
F1 score: 0.89022
For index 2:
F1 score: 0.744792
For index 3:
F1 score: 0.762332
For index 4:
F1 score: 0.77451
For index 5:
F1 score: 0.398268
For index 6:
F1 score: 0.806846
For index 7:
F1 score: 0.794118
For index 8:
F1 score: 0.618834
For index 9:
F1 score: 0.706444
```

可以看出，相较前一次而言，对数字 5 的识别精度更加的下降了，但对于其他数字都有所提升。可以推测，使用更高的学习率能够让其他数字更快地收敛到损失函数的极小值附近，而对于数字 5 则很可能会愈发偏离极小值。

四、 多层前向神经网络

(一) 理论简介

多层前向神经网络相较于单层感知机而言，能够对任意类型的函数进行更好的拟合。其对机器学习精确度提升的关键在于其的隐藏层能够获取一些单靠人类难以知晓的特征，从而实现对机器学习任务更好的刻画与判定。



如图，是一个 4 层的神经网络。它具有 2 个隐藏层。

（二） 代码实现

单层感知机对手写字符识别的 C++描述版本在项目文件夹 Multi-Layer NN 下。本次实验中，建立了一个 3 层的前向神经网络：第一层为输入层，最后一层是输出层，有 10 个神经元，分别代表数字 0-9。中间的隐藏层有 40 个神经元（未计入偏置神经元）。使用的训练方法是 mini-batch 随机梯度下降法。使用的误差函数是交叉熵（cross-entropy）误差函数。

（三） 实验结果

实验结果如下：（每次固定选取前 2000 个测试样例）

使用每次从训练集中随机选取 2000 个样例的 mini-batch 梯度下降方法，总共进行 60 次梯度下降，学习率$\alpha = 0.1$的情况下，有：（以下是代码输出）

```
Training Time: 148680ms
For index 0:
F1 score: 0.595843
For index 1:
F1 score: 0.648557
For index 2:
F1 score: 0.214925
For index 3:
F1 score: 0.413502
For index 4:
F1 score: 0.423077
For index 5:
F1 score: 0.260274
For index 6:
F1 score: 0.161538
For index 7:
F1 score: 0.428904
For index 8:
F1 score: 0.10989
For index 9:
F1 score: 0.328431
```

可以看出，虽然神经网络的训练时间是单层感知机的将近 10 倍，但准确度却难以与其抗衡。原因可能是神经网络的训练不够充分。

使用每次从训练集中随机选取 2000 个样例的 mini-batch 梯度下降方法，总共进行 300 次梯度下降，学习率$\alpha = 0.1$的情况下，有：（以下是代码输出）

```
Training Time: 718007ms
For index 0:
F1 score: 0.864407
For index 1:
```

```
F1 score: 0.860465
For index 2:
F1 score: 0.675325
For index 3:
F1 score: 0.687371
For index 4:
F1 score: 0.72
For index 5:
F1 score: 0.587838
For index 6:
F1 score: 0.756164
For index 7:
F1 score: 0.745283
For index 8:
F1 score: 0.624309
For index 9:
F1 score: 0.693976
```

可以看出，虽然花费了大量的训练时间，但是表现出的识别效果却是总体上优于单层感知机的。可以推测，如果训练的样本数目增多，那么识别效果将会进一步提升。

五、 总结

本次实验实现了两种简单的神经网络，并且对他们的效率做了部分探究和比较。

本次实验对于神经网络的一个额外的发现是：如果神经网络的初始权重全部是正数的话，那么神经网络将无法正常被训练，具体表现为神经网络会将所有样例判定为同一个数字（这个数字是在随机初始化权重时随机选择的）。笔者在实验中观察了很久才发现这个问题。

参考资料:

[1] Jacek M. Zurada, Introduction to Artificial Neural Systems, West Press, 1992.

[2] Peter Harrington, Machine Learning in Action, Manning Publications, 2012.

[3] [日]斋藤康毅 著，陆宇杰 译，深度学习入门:基于 Python 的理论和实现，人民邮电出版社, 2018.