TensorFlow 实现 Softmax Regression 识别手写体

1 MNIST 数据集

MNIST(Mixed National Institute of Standards and Technology database) 由几万张 28×28 像素的手写数字组成，这些图片只包含灰度值信息。MNIST 数据集样张如下：

其中空白部分为 0，有笔记的地方根据颜色深浅为 0~1 之间的取值。

2 独热码 (One-Hot Code)

One-Hot 编码，又称为一位有效编码，主要是采用 N 位状态寄存器来对N个状态进行编码，每个状态都由他独立的寄存器位，并且在任意时候只有一位有效。

One-Hot 编码是分类变量作为二进制向量的表示。这首先要求将分类值映射到整数值。然后，每个整数值被表示为二进制向量，除了整数的索引之外，它都是零值，它被标记为 1。

独热编码的优点为：

- 能够处理非连续型数值特征。
- 在一定程度上也扩充了特征。比如性别本身是一个特征，经过one hot 编码以后，就变成了男或女两个特征。

当然，当特征类别较多时，数据经过独热编码可能会变得过于稀疏。

3 Softmax Regression 和训练过程

softmax 回归用于多分类问题，它会对每一种类别估算一个概率，最后取概率最大的那个作为输出的结果。模型具有 k 组参数，$(\theta_1,\theta_2,...\theta_k)$，其中$\theta_i = [\theta_i^0, \theta_i^1, \ldots, \theta_i^n]^T$

softmax 回归假定给定样本$x^{(i)}$，样本属于类别 k 的概率 $P(y^{(i)} = k|x^{(i)}, \theta) = \frac{e^{\theta_k^T x^{(i)}}}{\sum_{j=1}^{n} e^{\theta_j^T x^{(i)}}}$,写成矩阵形式为：

$$\begin{bmatrix} P(y^{(i)}=1|x^{(i)},\theta) \\ P(y^{(i)}=2|x^{(i)},\theta) \\ \vdots \\ P(y^{(i)}=k|x^{(i)},\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{n} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} \theta_1^T x^{(i)} \\ \theta_2^T x^{(i)} \\ \vdots \\ \theta_k^T x^{(i)} \end{bmatrix}$$

将θ写到一个矩阵里面，有：

$$\theta = \begin{bmatrix} \theta_1^T \\ \theta_2^T \\ \vdots \\ \theta_k^T \end{bmatrix} = \begin{bmatrix} \theta_1^0 & \theta_1^1 & \cdots & \theta_1^n \\ \theta_2^0 & \theta_2^1 & \cdots & \theta_2^n \\ \vdots & \vdots & \ddots & \vdots \\ \theta_k^0 & \theta_k^1 & \cdots & \theta_k^n \end{bmatrix}$$

可以将特征写成如下公式，i 代表第 i 类，j 代表一张图片的第 j 个像素。bi 是bias，是数据本身的一些倾向，则会有：

$$feature_i = \sum_j W_{i,j} x_j + b_i$$

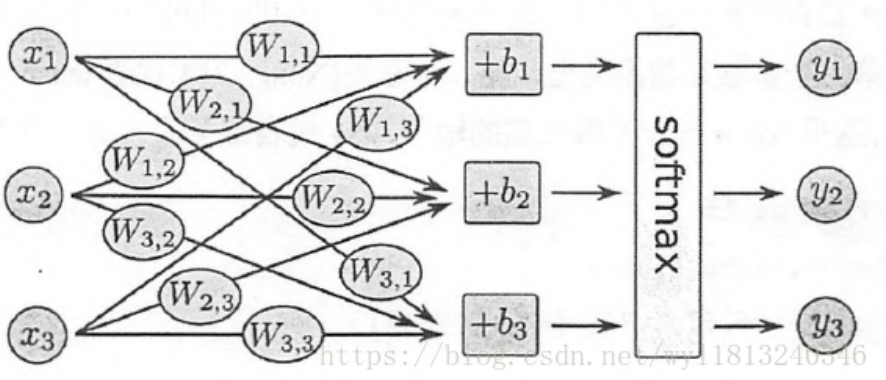
如上方公式，softmax 就是计算一个 exp 函数，然后再进行标准化（让所有类别输的概率值和为 1）。

$$softmax\left(x\right) = normalize\left(exp(x)\right)$$

然后第 i 类的判定公式为：

$$softmax(x)_i = \frac{exp(x_i)}{\sum_j exp(x_j)}$$

计算过程可视化如图



即上文的矩阵方程，可以简写成。

$$y = softmax(Wx + b)$$

TensorFlow 能够在训练时自动求导并进行梯度下降，这是很方便的，所以我们只需要再定义个损失函数（Loss Function）即可开始训练了。

关于 softmax 的推导可以看看这个链接：softmax 梯度公式推导

损失函数我们选择交叉熵（Cross Entropy），其定义如下，其中y是预测的概率分布，y'是真实概率分布，通常可以用它来判断模型对真实概率分布估计的准确程度。

ps：为什么不用均方误差?

当输入值与目标值差距很大的时候，二次代价函数就不是很恰当了，**因为学习速率会变得很慢。** 可以看看这篇博客交叉熵与 softmax

交叉熵的定义如下：

$$H_{y'}(y) = -\sum_i y_i' log(y_i)$$

训练时，我们使用最常见的随机梯度下降（Stochastic Gradient Descent）。每次随机从训练集抽取100条样本构成一个 batch，进行训练。使用随机梯度下降优点是收敛速度会快很多，计算量也小，容易跳出局部最优。

测试集上验证的话就很简单了，找出概率最大的那个类别作为分类类别和本身的标签对比。在求均值，最后我们可以得到这个分类器的精度。

4 TensorFlow 一般流程

1. 定义算法公式，就是神经网络 forward 时的计算。
2. 定义 loss function，选定优化器，指定优化器的 loss。
3. 迭代地对数据进行训练。
4. 在测试集上对准确率进行评测。

5 使用 iris 数据集练习

Iris数据集是常用的分类实验数据集，由Fisher, 1936收集整理。Iris也称鸢尾花卉数据集，是一类多重变量分析的数据集。数据集包含150个数据集，分为3类，每类50个数据，每个数据包含4个属性。可通过花萼长度，花萼宽度，花瓣长度，花瓣宽度4个属性预测鸢尾花卉属于（Setosa，Versicolour，Virginica）三个种类中的哪一类。

Sepal.Length(花萼长度)	Sepal.Width(花萼宽度)	Petal.Length(花瓣长度)	Petal.Width(花瓣宽度)	Class(种类)
数值：cm	数值：cm	数值：cm	数值：cm	三种：Setosa、Versiclour、Virginica

- 数据预处理

读入数据后先将种类字符串部分转换成数字。这块使用的是 pandas 中的 codes 方法。

```
iris[5] = pd.Categorical(iris[5]).codes
```

之后对数据做独热编码，用的是 sklearn 中的编码工具。

```
from sklearn.preprocessing import OneHotEncoder
ohe = OneHotEncoder()
ohe.fit(iris_y)
iris_y = ohe.transform(iris_y).toarray()
```

需要注意的是必须是二维矩阵才能进行编码。

然后将数据分成测试集和训练集。

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(iris_x,iris_y,test_size=0.25)
```

- 使用 softmax 进行分类

这部分基本和手写体识别相同。因为样本较少，我选择了每次选其中 5 个做 SGD 训练。

- 最后的结果

在测试集上的识别率在 95 %左右。