目录

[GNT数据预处理 2](#_Toc485149233)

[数据读取 2](#_Toc485149234)

[输入流大小归一化 3](#_Toc485149235)

[处理结果 4](#_Toc485149236)

[机器学习模型 5](#_Toc485149237)

[Softmax回归 5](#_Toc485149238)

[简介 5](#_Toc485149239)

[代价函数 6](#_Toc485149240)

[结果 6](#_Toc485149241)

[卷积神经网络 6](#_Toc485149242)

[初始化 6](#_Toc485149243)

[第一层卷积 6](#_Toc485149244)

[第二层卷积 7](#_Toc485149245)

[密集连接 7](#_Toc485149246)

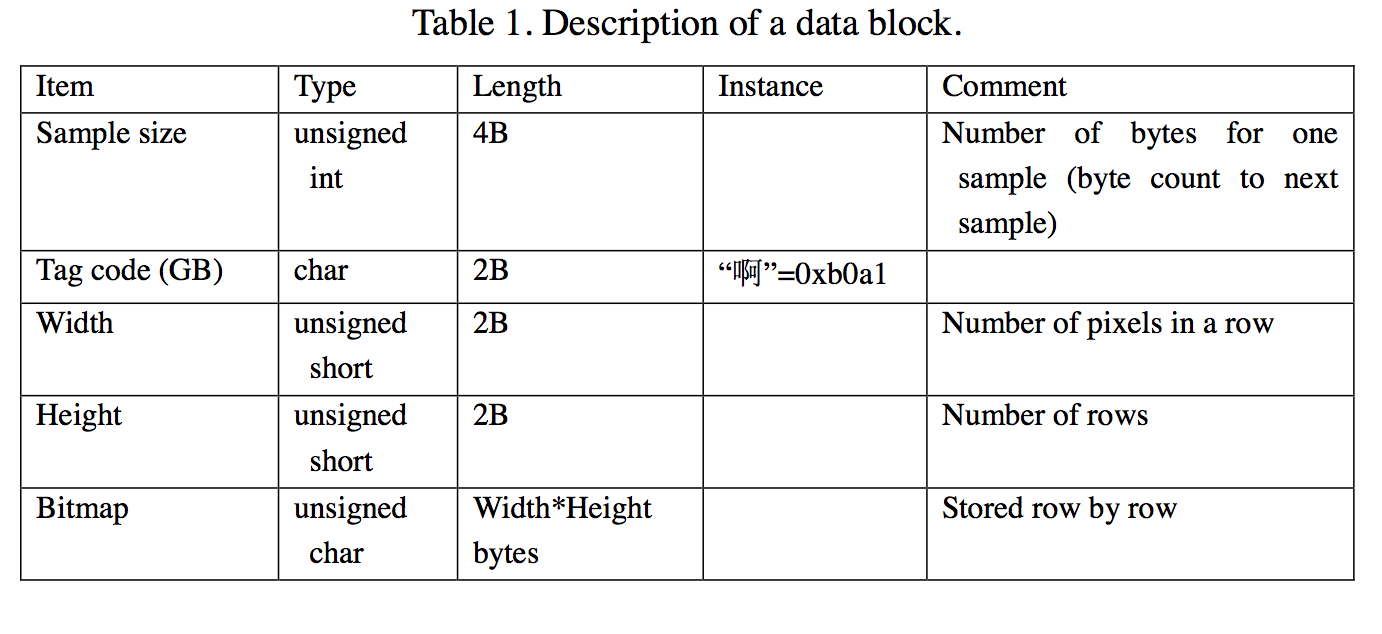
[输出层 7](#_Toc485149247)

[不同训练强度下的比较 8](#_Toc485149248)

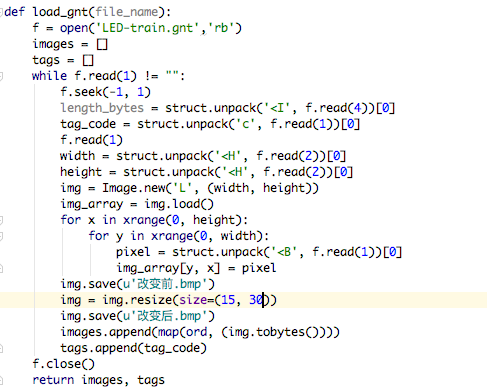
# GNT数据预处理

## 数据读取

本系统的训练集和测试集均为GNT格式的文件，GNT的文件格式如下表：



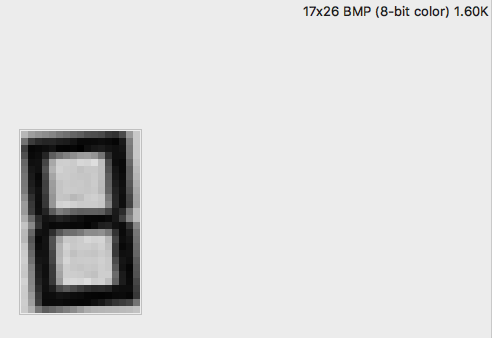
编写程序读取GNT文件：



## 输入流大小归一化

同时，由于图片大小不一样，所以需要对图像进行resize，下图为输出的demo结果：

改变前的图片为（13\*28）

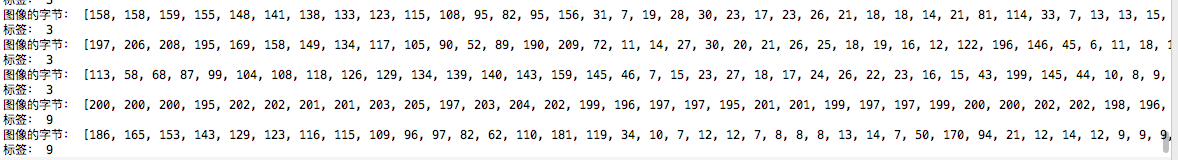


改变后的图片均为（28\*28）：



## 处理结果

经过对GNT数据的预处理，其结果如下：



# 机器学习模型

## Softmax回归

### 简介

我们首先使用最基本的单层模型对数据进行训练。我们把图像整个字节段展开，作为15\*30个特征值进行训练：

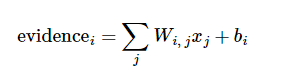
设定输入：

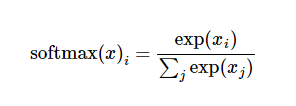
设定w与b：

设定tags与预测输出：



使用到的数学表达式为：

和

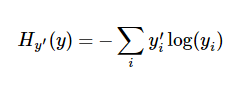


### 代价函数

这里使用交叉熵作为代价函数，使用梯度下降进行训练：



数学公式为：



### 结果

经过预测，结果的正确率为：

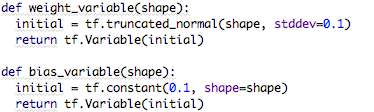


预测结果并不是很理想，下面我们使用卷积神经网络进行训练。

## 卷积神经网络

### 初始化

为了创建这个模型，我们需要创建大量的权重和偏置项。这个模型中的权重在初始化时应该加入少量的噪声来打破对称性以及避免0梯度。由于我们使用的是ReLU神经元，因此比较好的做法是用一个较小的正数来初始化偏置项，以避免神经元节点输出恒为0的问题。为了不在建立模型的时候反复做初始化操作，我们定义两个函数用于初始化。



### 第一层卷积

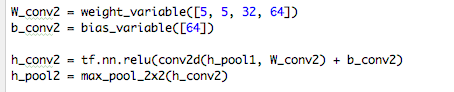
我们把一层卷积配置为5\*5的patch，通道数为1，可以输出32个特征值。



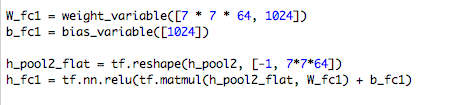
对图像与权值向量进行卷积再加上偏置项，再使用RELU激活函数，然后使用max pooling：



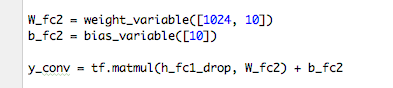
### 第二层卷积



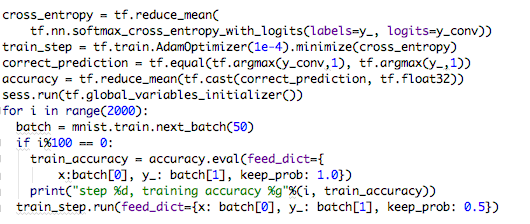
### 密集连接



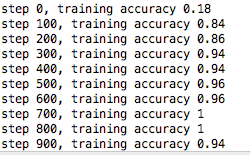
### 输出层

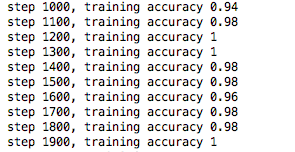


### 不同训练强度下的比较



我们使用交叉熵作为代价函数，然后使用随机梯度下降算法，然后进行训练。总共要迭代2000次，我们可以查看每100次迭代下准确率：





最终的准确率为。

我们可以看出，利用部分训练数据进行测试的情况下，并非训练地越多，准确率就越高。