MNIST数据集分析和基于tensorflow的算法研究

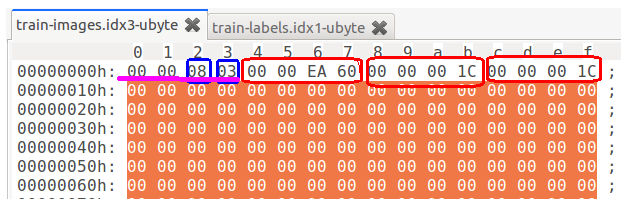
# MNIST数据格式

## 参考

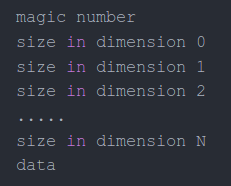
<https://www.jianshu.com/p/84f72791806f>

## 直观理解

### train-images.idx3-ubyte文件的格式



该文件中含有60000张图片，每个图片是28\*28=784个像素的灰度图。



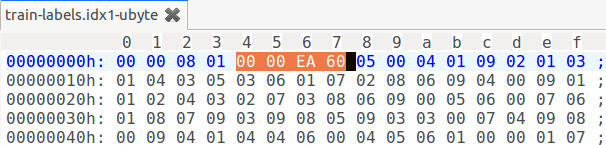
0x08表示数据都是按照无符号字节；

0x03表示数据是３维的：

* 0x0000EA60, 表示第一维的样本个数是60000;
* 0x0000001C，表示第二维的长度是28；
* 0x0000001C，表示第三维的长度是28；

最终，文件大小为16+60000\*784=47040016，刚好是文件的大小。

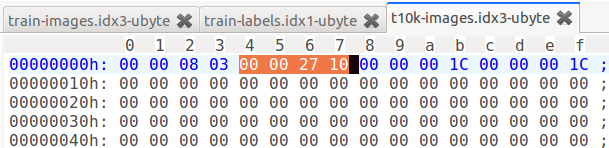
### train-labels.idx1-ubyte训练集标签文件



* 0x0000EA60, 表示第一维的样本个数是60000;

这６万个手写数字图片的分类标签只能是0~9之间的数字，因此用0x00到0x09表示。

### t10k-images.idx3-ubyte测试集文件

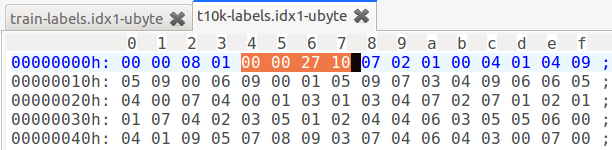


对比

* 0x00002710, 表示第一维的样本个数是10000;
* 0x0000001C，表示第二维的长度是28；
* 0x0000001C，表示第三维的长度是28；

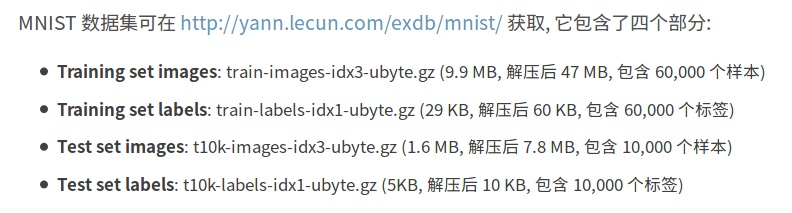
共计：16 + 10000\*784 = 7840016字节，也是文件的大小。

### t10k-labels.idx1-ubyte测试集标签文件



8 + 10000 \* 1 = 10008字节。

## 数据个数与集合拆分



<https://blog.csdn.net/business122/article/details/80940085>



## 加载dataset的代码

import tensorflow as tf

from tensorflow.examples.tutorials.mnist import input\_data

#if \*.gz is not exist, then download it, otherwise load it directly!

mnist = input\_data.read\_data\_sets("/nn/a/", one\_hot = True)

print(mnist.train.images.shape, mnist.train.labels.shape)

print(mnist.validation.images.shape, mnist.validation.labels.shape)

print(mnist.test.images.shape, mnist.test.labels.shape)

# 环境搭建

## tensorflow的CPU版本

pycharm：下载安装社区版本，安装到自己指定的目录下；

anaconda：搜索清华镜像的版本，然后通过sh xxx.sh命令来执行安装；

tensorflow：anaconda安装成功后，通过

./conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/

./conda config --set show\_channel\_urls yes

./conda install tensorflow

./conda list #查看已经安装的所有包，已经包含了numpy, scikit-learn和tensorflow等

环境变量配置：

/root/.bashrc中添加PATH=/srv/anaconda3/bin:$PATH

source /root/.bashrc

验证是否安装成功：

import tensorflow as tf

print(tf.\_\_version\_\_)

print(tf.\_\_path\_\_)

没有使用GPU版本的tensorflow时，执行时可能会输出下面的信息

2019-01-08 12:09:21.336674: W tensorflow/core/platform/cpu\_feature\_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.

## tensorflow的GPU版本(?)

需要进行性能对比测试，了解性能之间的数量级的差异。

# tensorflow研究

## 参考

<https://blog.csdn.net/xiaopihaierletian/article/list/16?t=1>

第１６页开始。

## tensorflow基本操作

https://blog.csdn.net/xiaopihaierletian/article/details/61925379

<https://blog.csdn.net/xiaopihaierletian/article/details/61927137>

<https://blog.csdn.net/xiaopihaierletian/article/details/61923808>

## 简单的神经网络例子（手写数字识别）

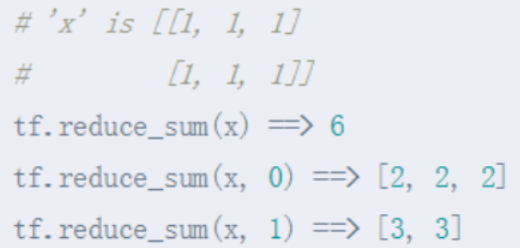
<https://blog.csdn.net/xiaopihaierletian/article/details/61924468>

### softmax函数



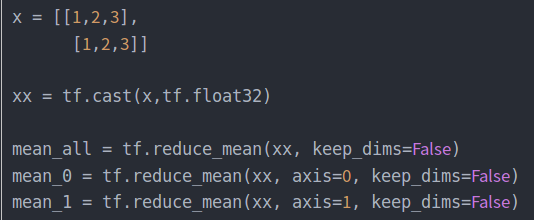
其实，就是将数值转换到(0~1)的区间内，并且让数值的差距变得更大一些，符合我们多分类的概率判断需求，所以我们选择用它，而不是二分类的sigmoid函数。。

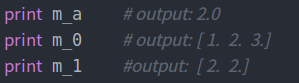
### reduce\_sum函数



reduction\_indies=1表示按行求和。

### reduce\_mean函数





表示求均值。

### argmax函数和argmin函数

argmax表示找到某行或某列中的最大元素所在的位置

第二个参数为dimension=0表示按列寻找；

第二个参数为dimension=１表示按行寻找；

### 交叉熵方法小结

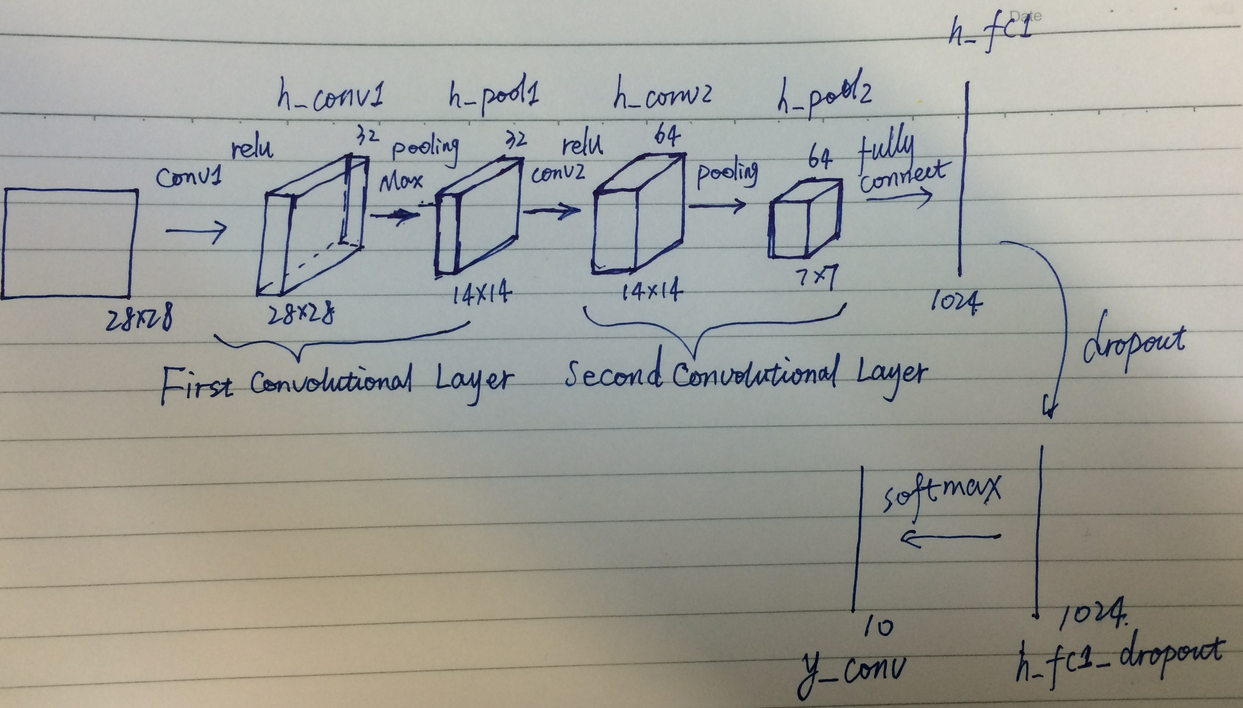
使用“**交叉熵**”来作为损失函数，准确率只有86.73%

## 使用CNN（cifar数据集的据图片识别?）

### 参考

<https://blog.csdn.net/xiaopihaierletian/article/details/61924916>

### 卷积神经网络图形



### 准确率

step 0 training accuracy 0.1

step 100 training accuracy 0.06

step 200 training accuracy 0.08

.....

......

......

step 19600 training accuracy 0.92

step 19700 training accuracy 0.94

step 19800 training accuracy 0.96

step 19900 training accuracy 0.92

test accuracy **0.9545**

Process finished with exit code 0

## 官方CNN例子（cifar数据集的据图片识别?）

### 参考

<https://en.wikipedia.org/wiki/Canadian_Institute_for_Advanced_Research>

<https://github.com/tensorflow/models/tree/master/tutorials/image/cifar10>

<http://www.cs.toronto.edu/~kriz/cifar.html>

### 多文件

python程序可以有多个main函数，作为各个模块的测试入口。