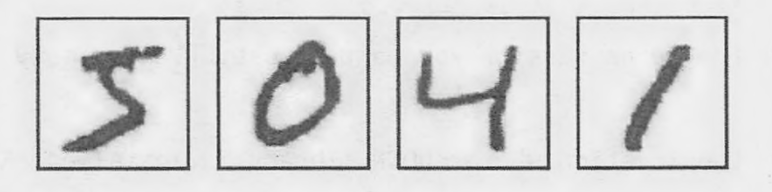
**TensorFlow｜1｜识别手写数字**

1. 基本环境

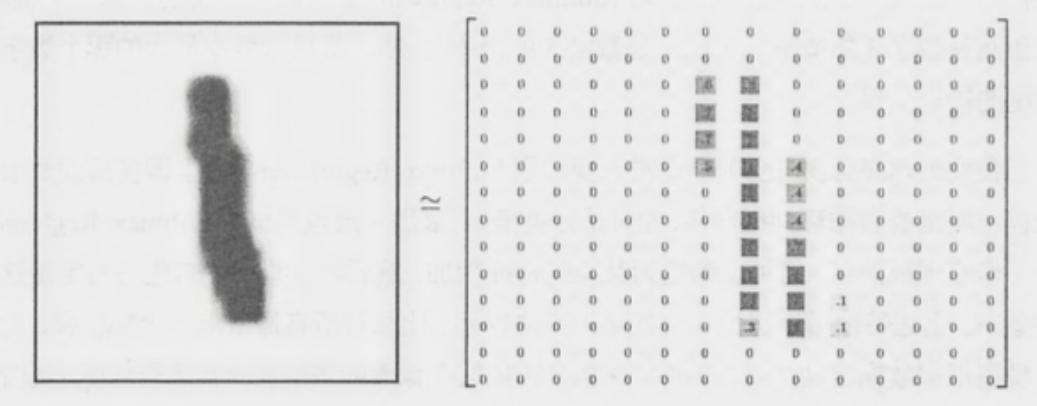
|  |  |
| --- | --- |
| 程序名称 | ex1\_1024.py |
| 硬件环境 | MacOSX |
| 软件环境 | Python3, Spyder, Anaconda |

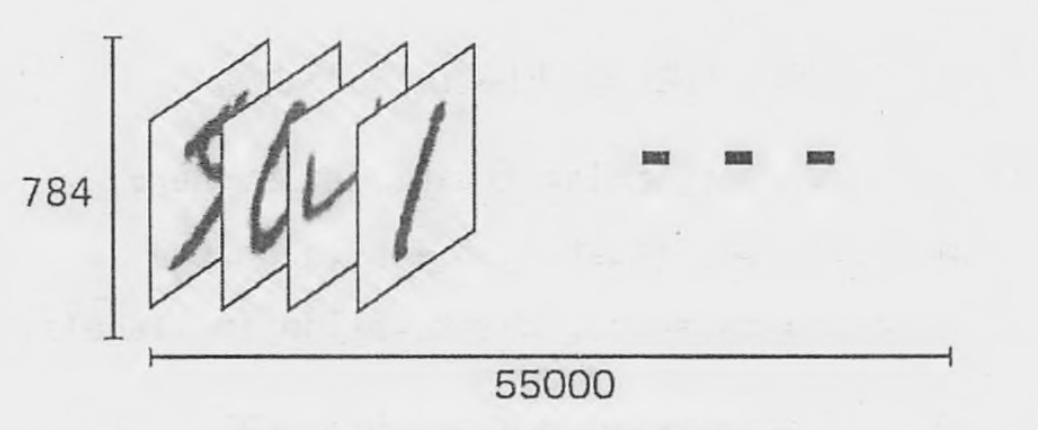
1. 核心问题：利用TensorFlow识别手写数字，基于《TensorFlow实战》

MNIST(Mixed National Institute of Standards and Technology database)是一个机器视觉数据集，如图所示，它由几万张28像素28像素的手写数字组成，这些图片只包含灰度值信息，我们的任务就是对这些图片进行分类识别，进而来探索TensorFlow。

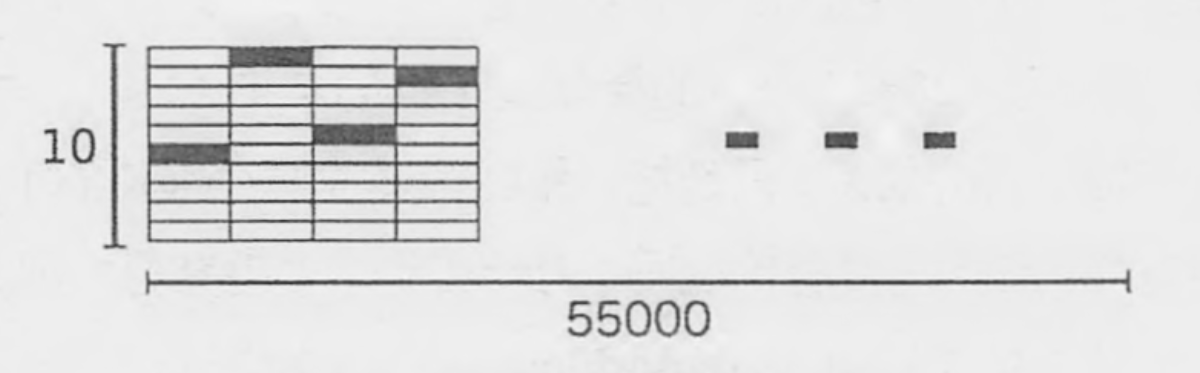


1. 解决过程：

* 首先对MNIST数据进行加载。
* 然后查看minst这个数据集的情况，可以看到训练集有55000个样本，测试集有10000个样本，同时验证集上有5000个样本。每一个样本都有它对应的标注信息，即label。我们将在训练集上训练模型，在验证集上检验效果并决定何时完成训练，最后我们在测试集评测模型的效果（可通过准确率、召回率、F1-score等评测）。
* 如下图，灰度图像空白部分全部为0，有笔迹的地方根据颜色的深浅在0到1之间取值。每个样本有2828=784维的特征，也就是784个点展开成1维的结果，每一张图片用同样的顺序展开。
* 我们训练数据的特征是一个55000784的Tensor，第一个维度时图片的编号，第二个维度是图片中像素点的编号，如下图所示。



* 如下图，这里对10个种类进行了one-hot编码，Label是一个10维的向量。只有1个值为1，其余为0。比如数字0，对应的Label就是[1,0,0,0,0,0,0,0,0,0]，数字5对应的Label就是[0,0,0,0,0,1,0,0,0,0]，数字n就代表对应位置的值为1。

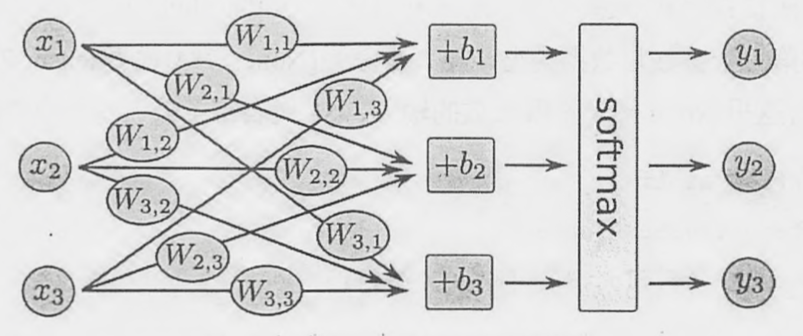


* 准备好数据后，接下来是设计算法。当我们处理多分类任务时，通常需要使用Softmax Regression模型。它的工作原理为，将可以判定为某类的特征相加，然后将这些特征转化为判定是这一类的概率。

特征featurei=I,JXJ+bI

* 接下来对所有的特征计算Softmax(x)=normalize(exp(x))，其中判定为第i类的概率就可由下面的公式得到Softmax(x)t=

把整个计算过程可视化为：



上述可以表达为：y=softmax(Wx+b)

* 接下来就是使用TensorFlow实现一个Softmax Regression。

1. 首先载入TensorFlow库，并创建一个新的InteractiveSession。

import tensorflow as tf

sess = tf.InteractiveSession()

x = tf.placeholder(tf.float32,[None,784])

1. 接下来要给Softmax Regression模型中的weights和biases创建Variable 对象。

W = tf.Variable(tf.zeros([784,10]))

b = tf.Variable(tf.zeros([10]))

1. 接下来就是要实现Softmax Regression算法。

y = tf.nn.softmax(tf.matmul(x,w)+b)

1. 在TensorFlow中定义cross-entropy。

y\_ = tf.placeholder(tf.float32,[None,10])

cross\_entropy = tf.reduce\_mean(-tf.reduce\_sum(y\_ \* tf.log(y),reduction\_indices=[1]))

1. 下一步使用TensorFlow的全局参数初始化器tf.global\_variables\_initializer，并执行它的run方法。

tf.global\_variables\_initializer().run()

1. 最后，开始迭代地执行训练操作train\_step。

for I in range(1000)

batch\_xs, batch\_ys = mnist.train.next\_batch(100)

train\_step.run({x:batch\_xs, y\_: batch\_ys})

1. 代码：

import tensorflow as tf

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

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

sess=tf.InteractiveSession()

x=tf.placeholder(tf.float32,[None,784])

W=tf.Variable(tf.zeros([784,10]))

b=tf.Variable(tf.zeros([10]))

y=tf.nn.softmax(tf.matmul(x,W)+b)

y\_=tf.placeholder(tf.float32,[None,10])

cross\_entropy=tf.reduce\_mean(-tf.reduce\_sum(y\_ \*tf.log(y),reduction\_indices=[1]))

train\_step=tf.train.GradientDescentOptimizer(0.5).minimize(cross\_entropy)

tf.global\_variables\_initializer().run()

for i in range(1000):

batch\_xs,batch\_ys=mnist.train.next\_batch(100)

train\_step.run({x:batch\_xs,y\_:batch\_ys})

correct\_pre=tf.equal(tf.argmax(y,1),tf.argmax(y\_,1))

accu=tf.reduce\_mean(tf.cast(correct\_pre,tf.float32))

print(accu.eval({x:mnist.test.images,y\_:mnist.test.labels}))

1. 结果：

准确率为0.91左右，由于只使用了最基础的模型，故准确率不高。

通过这个例子，我们使用TensorFlow实现了一个简单的机器学习算法Softmax Regression，整个流程可以分为四部分：

1. 定义算法公式，也就是神经网络forward时的计算。
2. 定义loss，选定优化器，并制定优化器优化loss。
3. 迭代地对数据进行训练。
4. 在测试集或验证集上对准确率进行评测。
5. 报错记录

输入“mnist = input\_data.read\_data\_sets("MNIST\_data/", one\_hot=True)”

报错原因：网页找不到

解决方法：翻墙

log:

Extracting MNIST\_data/train-images-idx3-ubyte.gz

Traceback (most recent call last):

File "<ipython-input-4-6f78a56b35f2>", line 1, in <module>

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

File "/Users/xuyizhou/anaconda3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py", line 240, in read\_data\_sets

SOURCE\_URL + TRAIN\_LABELS)

File "/Users/xuyizhou/anaconda3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/base.py", line 208, in maybe\_download

temp\_file\_name, \_ = urlretrieve\_with\_retry(source\_url)

File "/Users/xuyizhou/anaconda3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/base.py", line 165, in wrapped\_fn

return fn(\*args, \*\*kwargs)

File "/Users/xuyizhou/anaconda3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/base.py", line 190, in urlretrieve\_with\_retry

return urllib.request.urlretrieve(url, filename)

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 248, in urlretrieve

with contextlib.closing(urlopen(url, data)) as fp:

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 223, in urlopen

return opener.open(url, data, timeout)

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 526, in open

response = self.\_open(req, data)

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 544, in \_open

'\_open', req)

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 504, in \_call\_chain

result = func(\*args)

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 1361, in https\_open

context=self.\_context, check\_hostname=self.\_check\_hostname)

File "/Users/xuyizhou/anaconda3/lib/python3.6/urllib/request.py", line 1320, in do\_open

raise URLError(err)

URLError: <urlopen error [Errno 60] Operation timed out>