**AI**

**实 验 报 告**

**学生姓名** 吴语港

**学生学号** SA19225404

**实验日期** 2019/11/6

**实验名称** TensorFlow网络

模型压缩理论与实践

**学生学号** SA19225404

**实验地点** 思贤楼301 303实验室

1. **实验目的:**

通过量化来实现TensorFlow网络模型的压缩

1. **实验原理:**

**（1）剪枝（pruning）：**

剪枝的主要思想就是将权重矩阵中相对“不重要”的权值剔除，然后再重新fine tune 网络进行微调。

广义上的修剪包括： 正则化regularization：修改目标函数/学习问题，所以优化后可能会得到一个参数较少的神经网络，例如添加L1Norm在损失函数中。修剪pruning：面向大规模神经网络，并删除某些意义上冗余的特征或参数。增长growing:从小型网络开始，按某种增长标准逐步增加新的单元。

剪枝的基本流程 ：衡量神经元的重要程度->移除一部分不重要的神经元->对网络进行微调->返回第一步，进行下一轮剪枝

这一部分的核心问题包括： 剪枝的粒度变化——剪多深? 剪枝方法——怎么剪 ? 如何衡量权值的重要性? 如何选取去除权值的数量或比例? 什么时候剪? 在去除不重要的权值之后，如何保持网络性能? fine tune之后又会产生新的不重要权值，如何解决?

**（2）量化（Quantization）：**

一般而言，神经网络模型的参数都是用的32bit长度的浮点型数表示，实际上不需要保留那么高的精度，可以通过量化，比如用0~255表示原来32个bit所表示的精度，通过牺牲精度来降低每一个权值所需要占用的空间。此外，SGD（Stochastic Gradient Descent）所需要的精度仅为6~8bit，因此合理的量化网络也可保证精度的情况下减小模型的存储体积。

根据量化方法不同，大致可以分为二值量化，三值量化，多值量化  
对网络网络进行量化要解决三个基本问题：如何对权重进行量、如何计算二值权重的梯度、如何确保准确率。

最为典型就是二值网络、XNOR网络等。其主要原理就是采用1bit对网络的输入、权重、响应进行编码。减少模型大小的同时，原始网络的卷积操作可以被bit-wise运算代替，极大提升了模型的速度。但是，如果原始网络结果不够复杂（模型描述能力），由于二值网络会较大程度降低模型的表达能力。因此现阶段有相关的论文开始研究n-bit编码方式成为n值网络或者多值网络或者变bit、组合bit量化来克服二值网络表达能力不足的缺点。

**（3）蒸馏：**

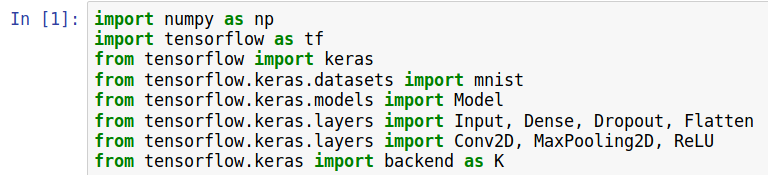
蒸馏模型采用的是迁移学习，通过采用预先训练好的复杂模型（Teacher model）的输出作为监督信号去训练另外一个简单的网络。这个简单的网络称之为student model。

总体的来说，知识蒸馏是一种迁移学习，将大模型的知识最大程度的教授给小模型，以达到压缩模型的目的。目前来看，无论是压缩比还是蒸馏后的性能都还有待提高。

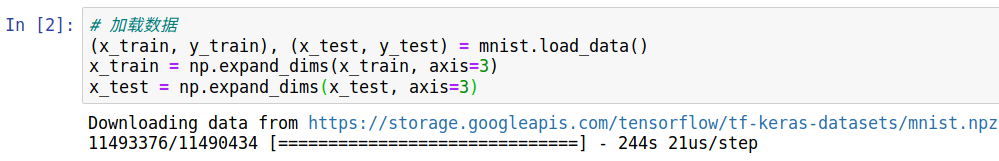
存在的问题和研究的趋势，如何处理“知识”的不同形式，去除softmax的限制，研究趋向于选用中间特征层。如何选择特征层，如何设计损失函数。训练学生模型数据集的选择，以及学生模型的设计。如何和其他压缩方法集成。

1. **实验过程:**
2. **普通训练：**

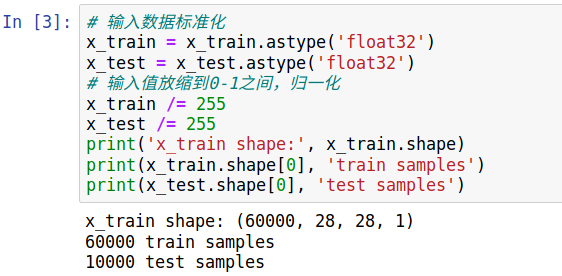
导入tensorflow，numpy使用keras来搭建模型



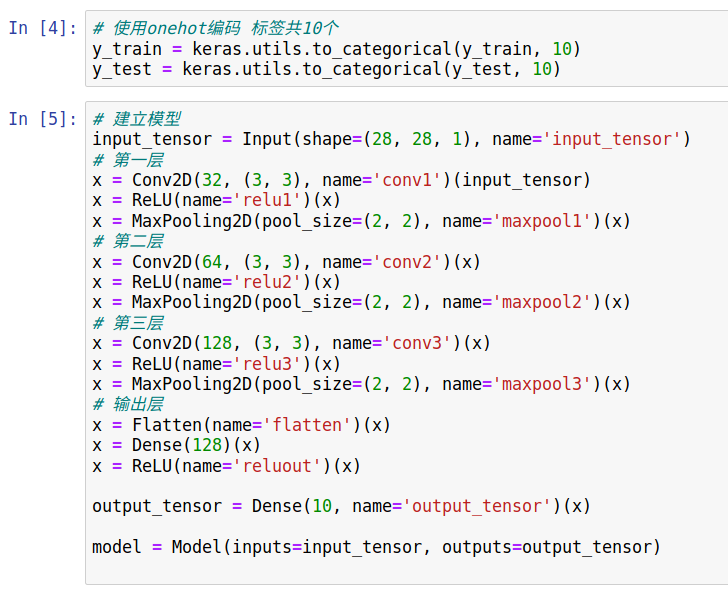
通过keras自带的mnist数据集，导入为训练集与测试集



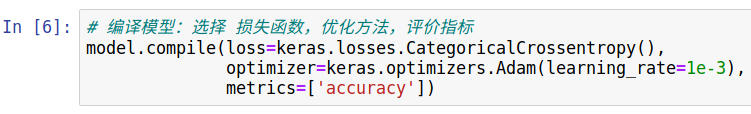
对数据进行归一化处理为0~1区间，查看数据的形状，一共有60000个训练样本，10000个测试样本，每个图片为28x28的灰度图片。



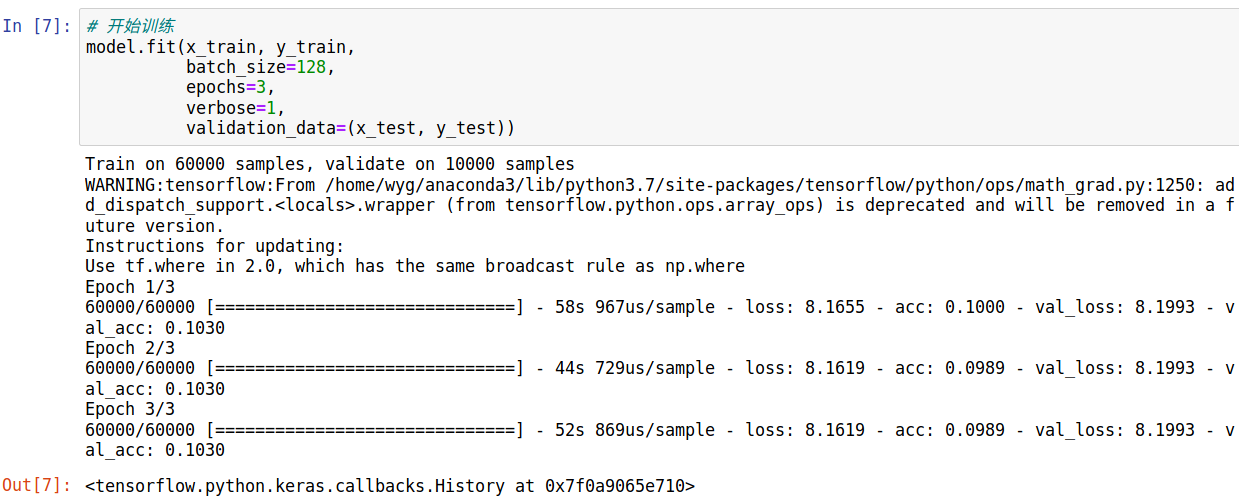
使用onehot编码对输出标签分类，建立了实验二当中使用的一个卷积神经网络



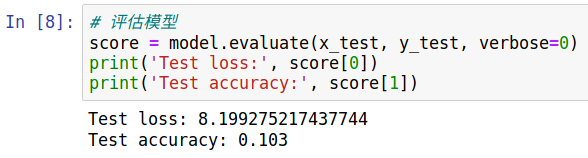
选择了分类交叉熵函数作为损失函数，Adam优化方法，并进行了编译



开始训练，3个epochs，一个epochs一共60000张图片，每个batch读入128张图片



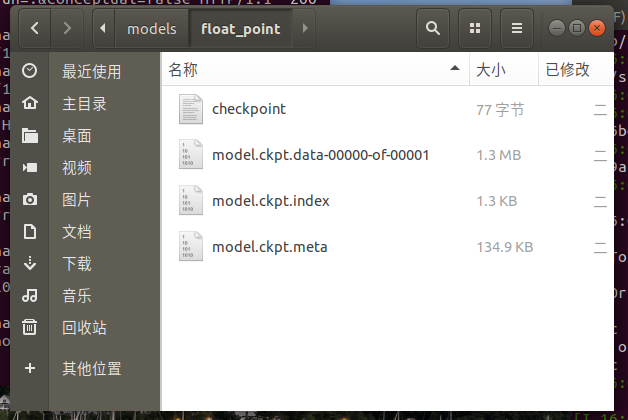
准确率不高，不过主要是给后面的量化感知训练做预训练



将训练的模型保存到当前目录下的./models/float\_point/model.ckpt



保存后的目录



其中：

**ckpt.meta**文件以 “protocol buffer”格式保存了整个模型的结构图，模型上定义的操作等信息。

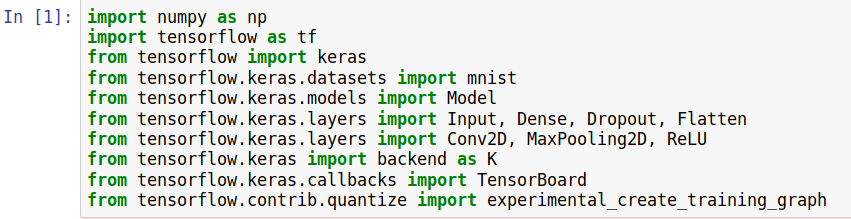
**ckpt.data-00000-of-00001**保存了网络的变量信息以及一些权重参数信息

**ckpt.index** 保存了.data文件中数据和 .meta文件中结构图之间的映射关系

**checkpoint**是一个文本文件，记录了训练过程中在所有中间节点上保存的模型的名称，首行记录的是最后（最近）一次保存的模型名称。

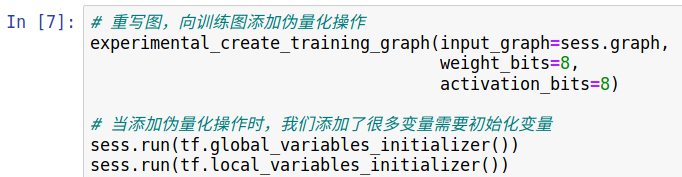
1. **感知量化训练：**

多导入了TensorBoard和Quantize模块来建训练图

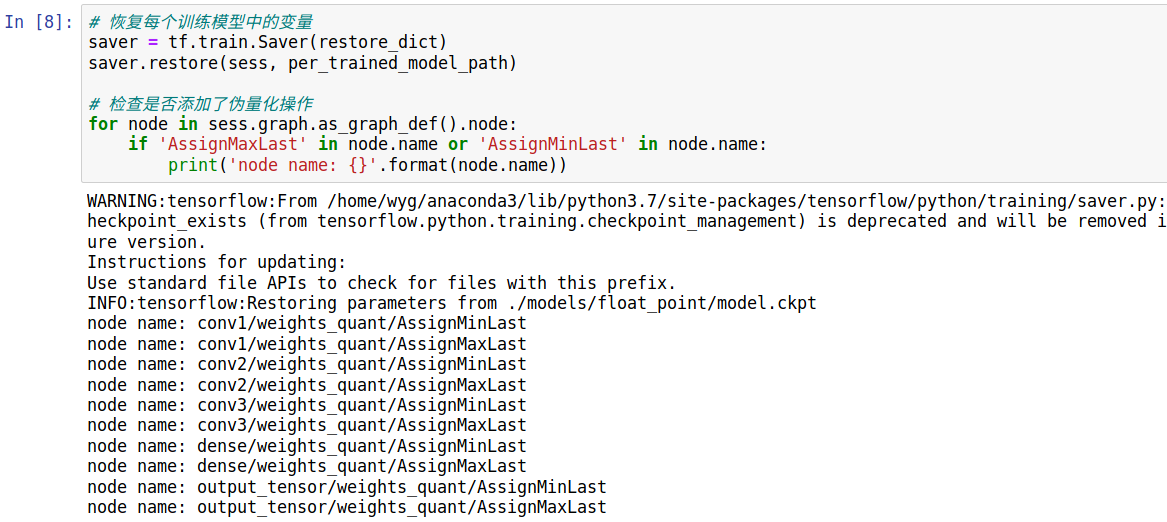


后面的一些部分与上文相似

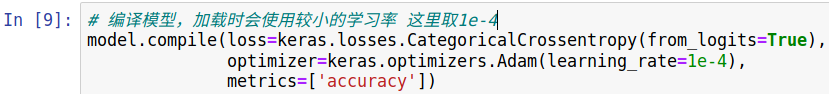
进行重写图的一些预处理



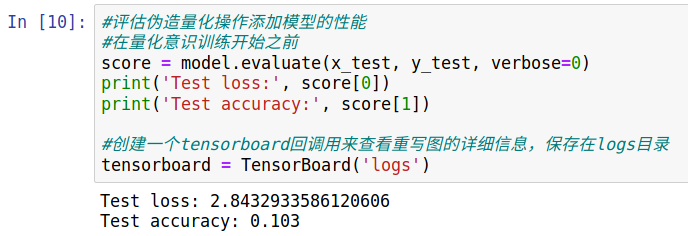
对节点的权重参数进行检测，获取最大最小值，从而为后面TOCO工具的量化操作服务



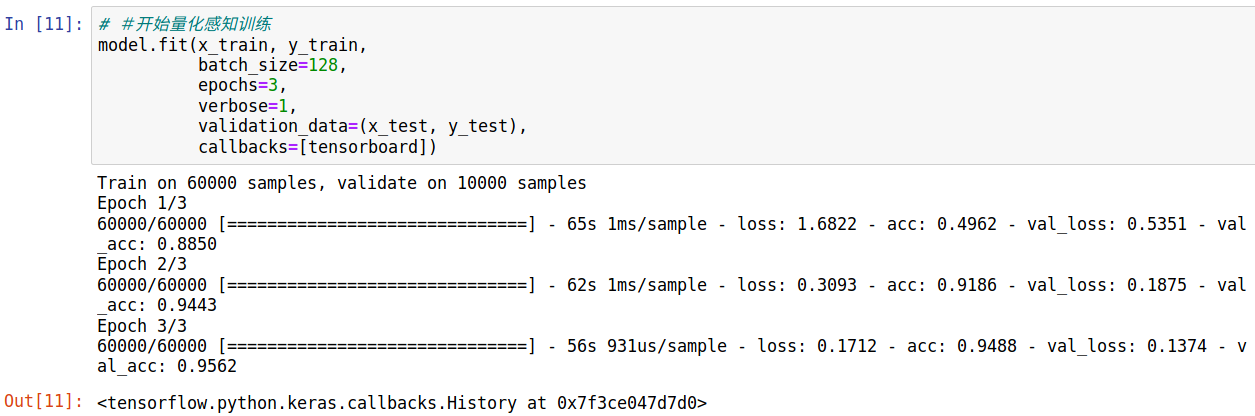
编译模型，准备训练



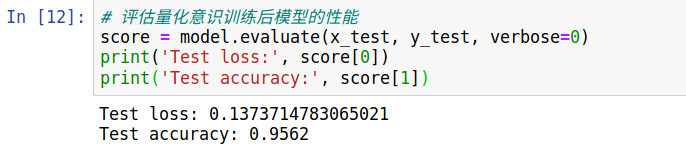
先评估一下伪量化操作后模型性能



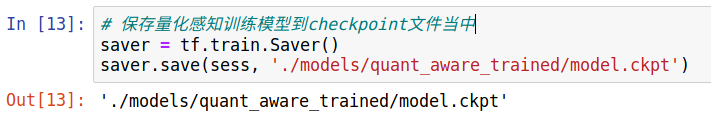
开始训练，准确率大幅上升到0.95



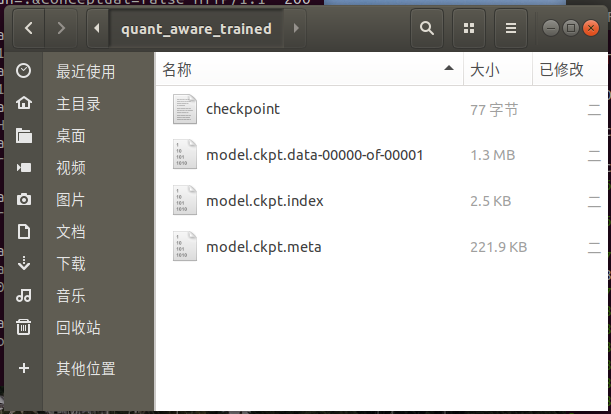
再评估一下训练后的模型



保存模型



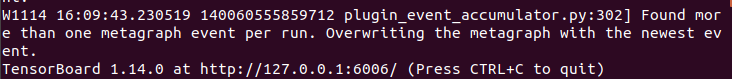
查看保存后的文件



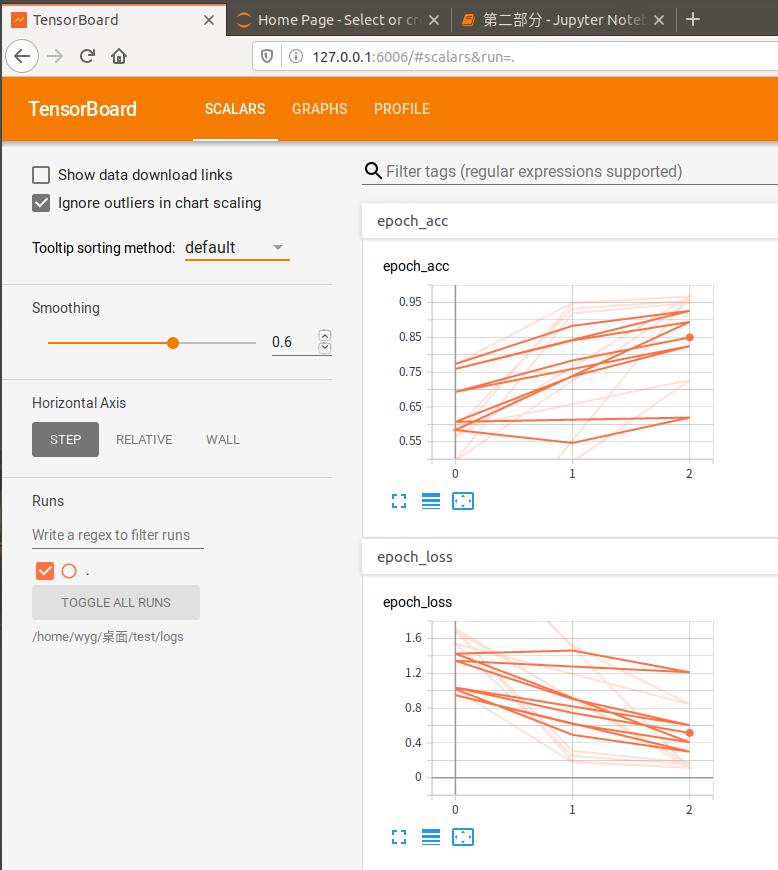
ckpt.index文件和ckpt.meta文件变大了，应该是加了伪量化节点的缘故。

在tensorboard查看一下准确度与损失的变化

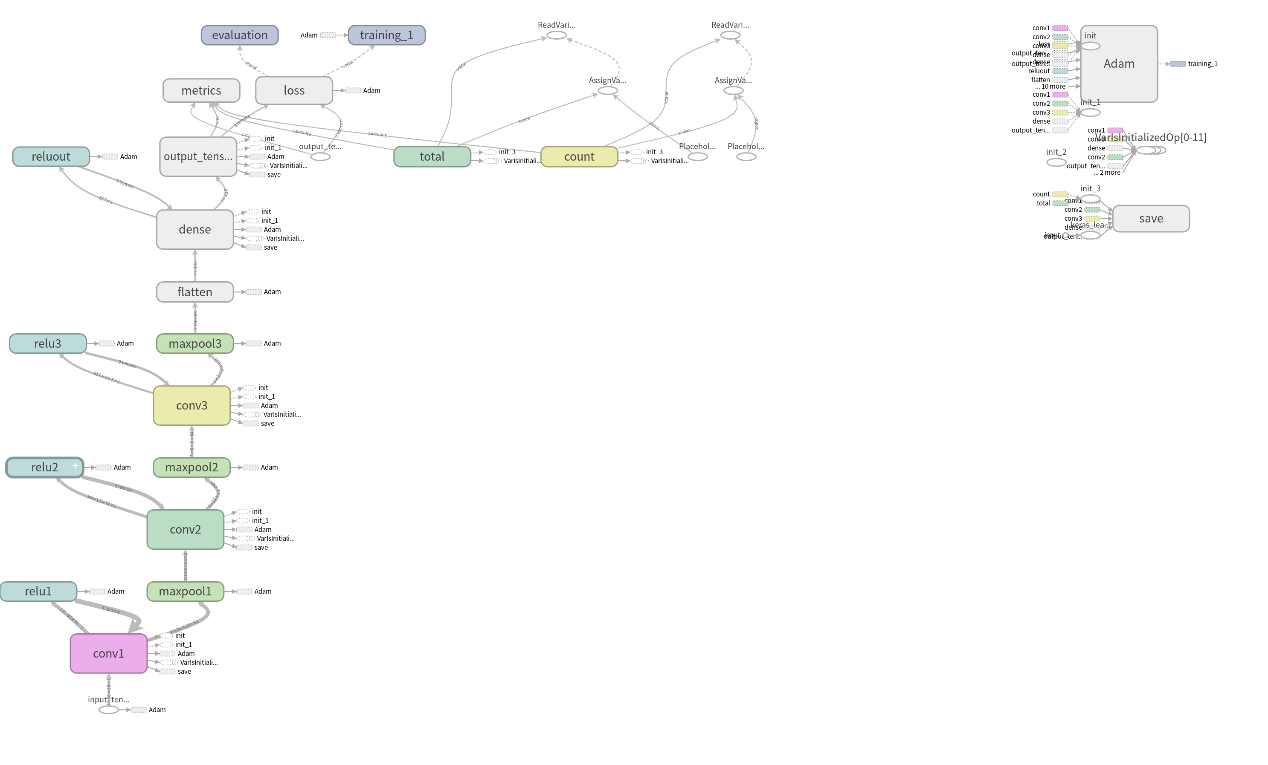




精确度在上升，损失在下降。

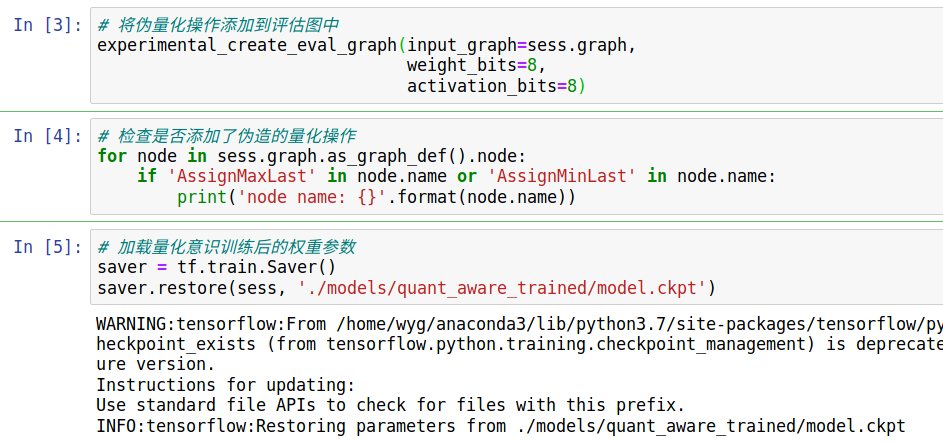


查看tensorboard保存的神经网络图示

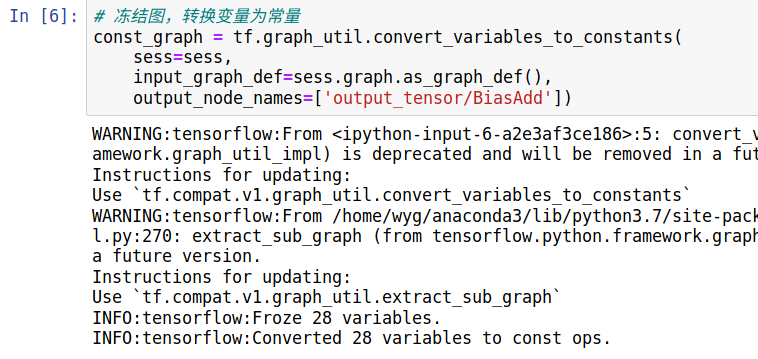
****

1. **冻结变量为常量：**

前面的操作与上文类似，就是导入模块，定义模型，然后就是将伪量化操作添加到评估图当中。



加载量化训练后的模型



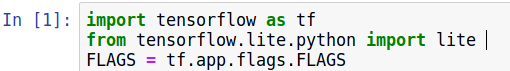
对图进行冻结操作，也就是变量转常量



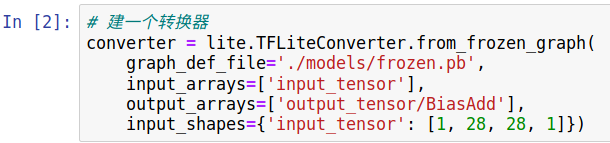
将冻结后的图存储为pb文件

1. **转化为tflite：**

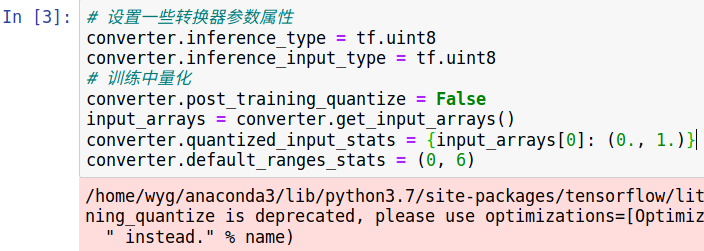
导入lite模块



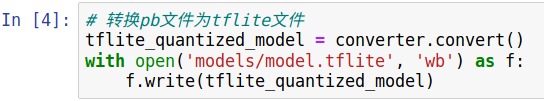
建立pb->tflite转换器



设置转换器属性

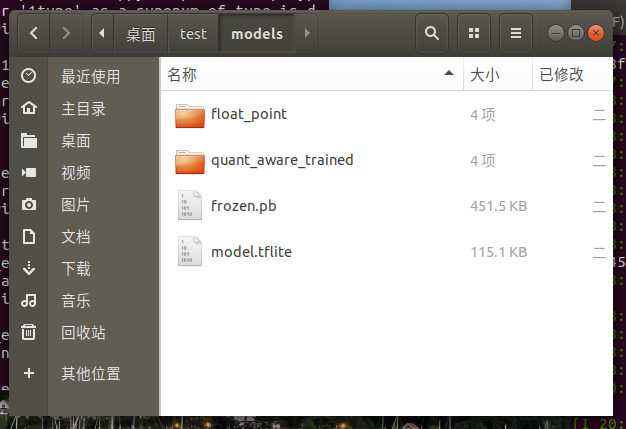


进行文件的转换



1. **实验结果：**

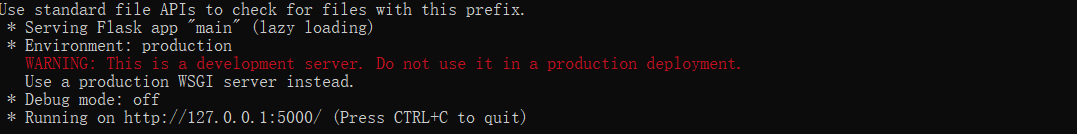
**对比各个模型文件大小：**



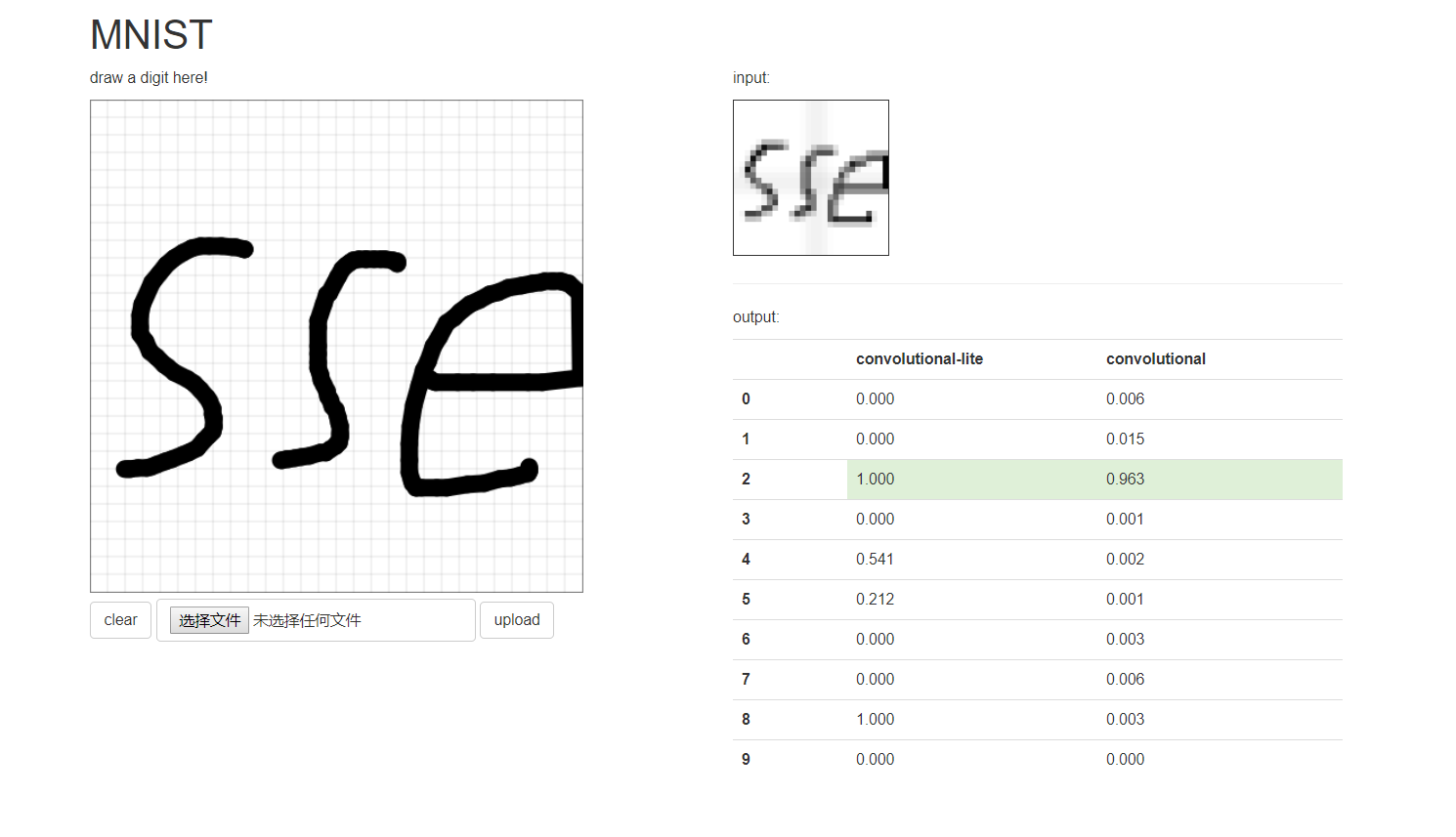
从frozen文件到model.tflite文件大概缩小了4倍，符合float32占int8空间的4倍的原理。

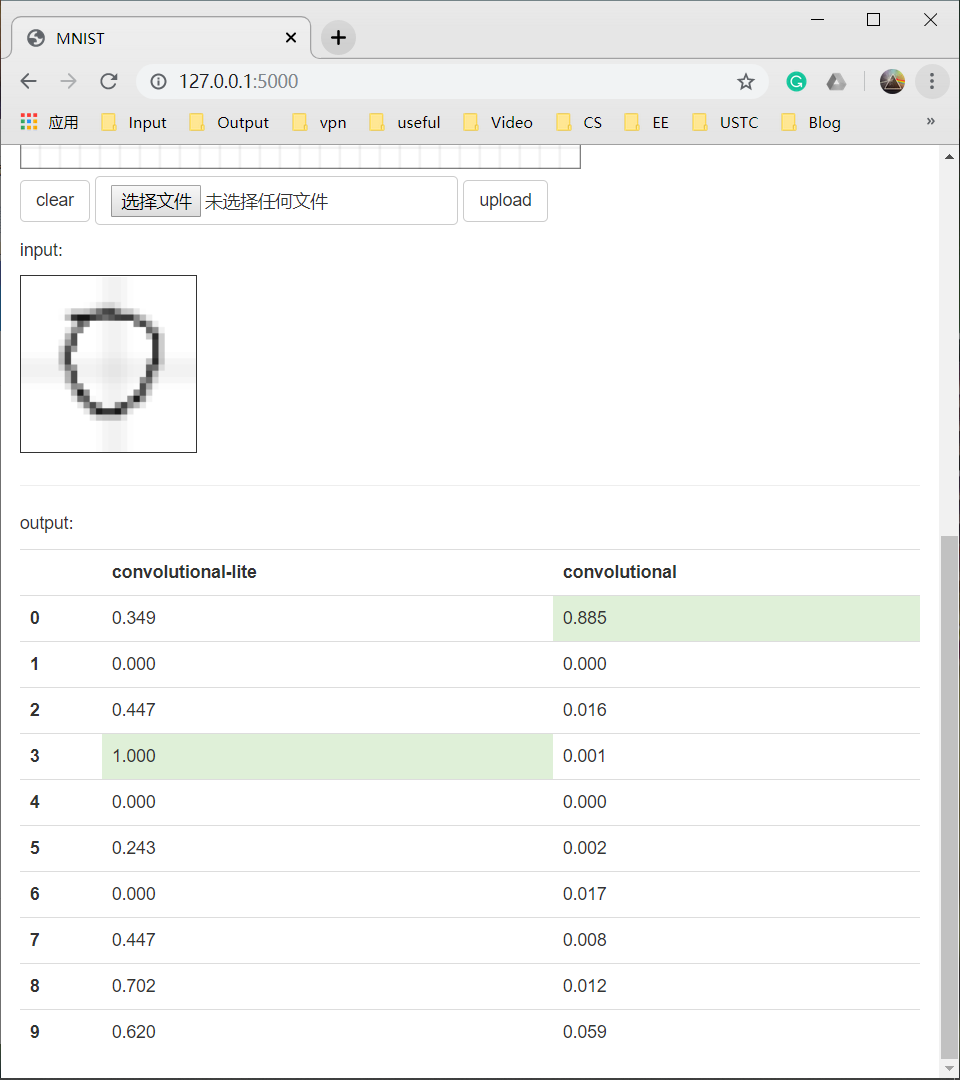
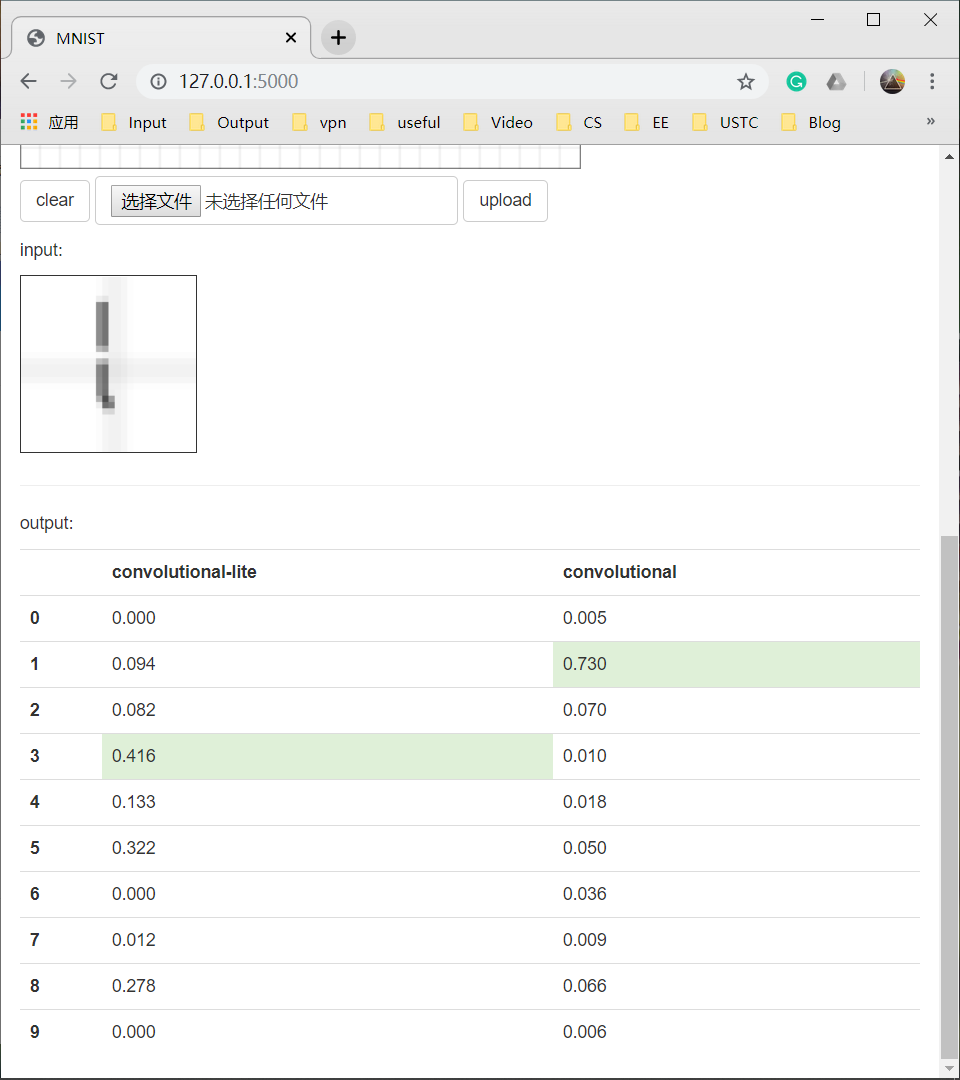
通过下一次实验将模型部署，查看模型效果

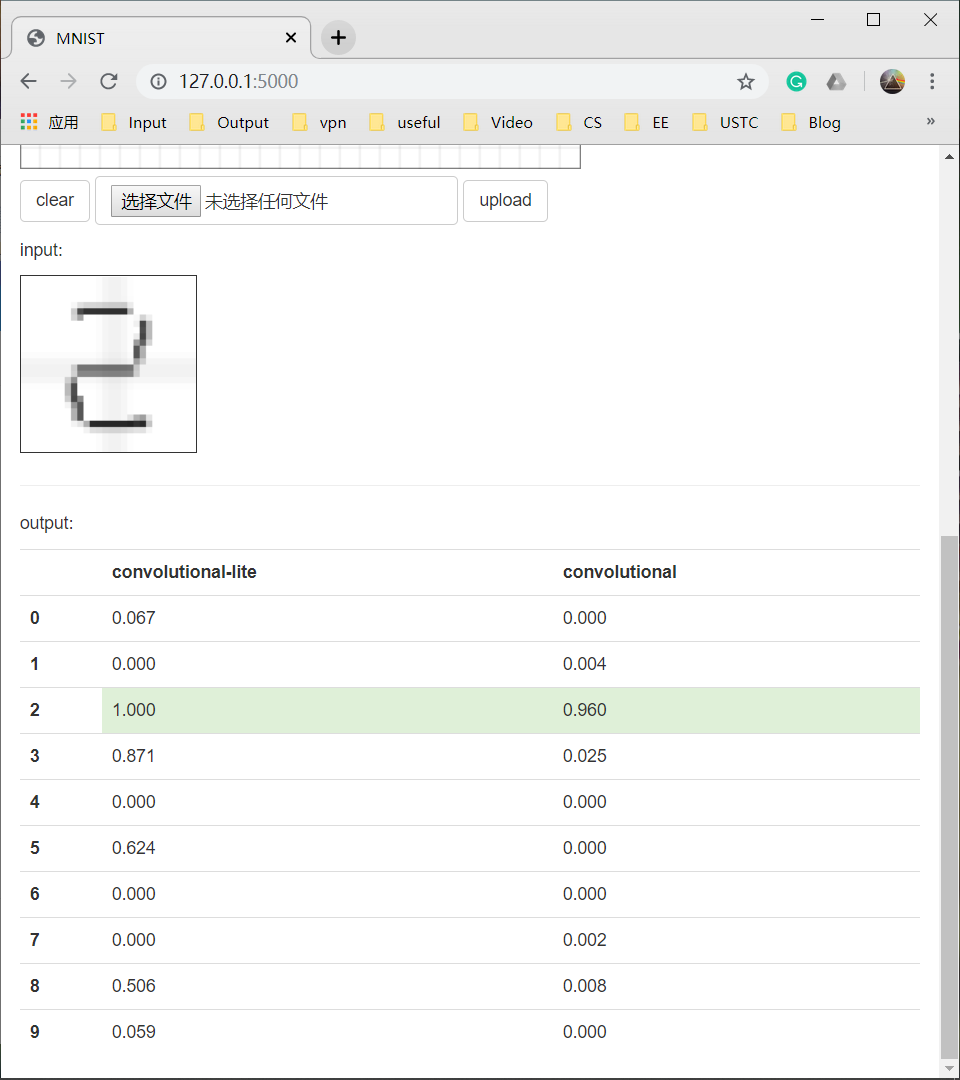
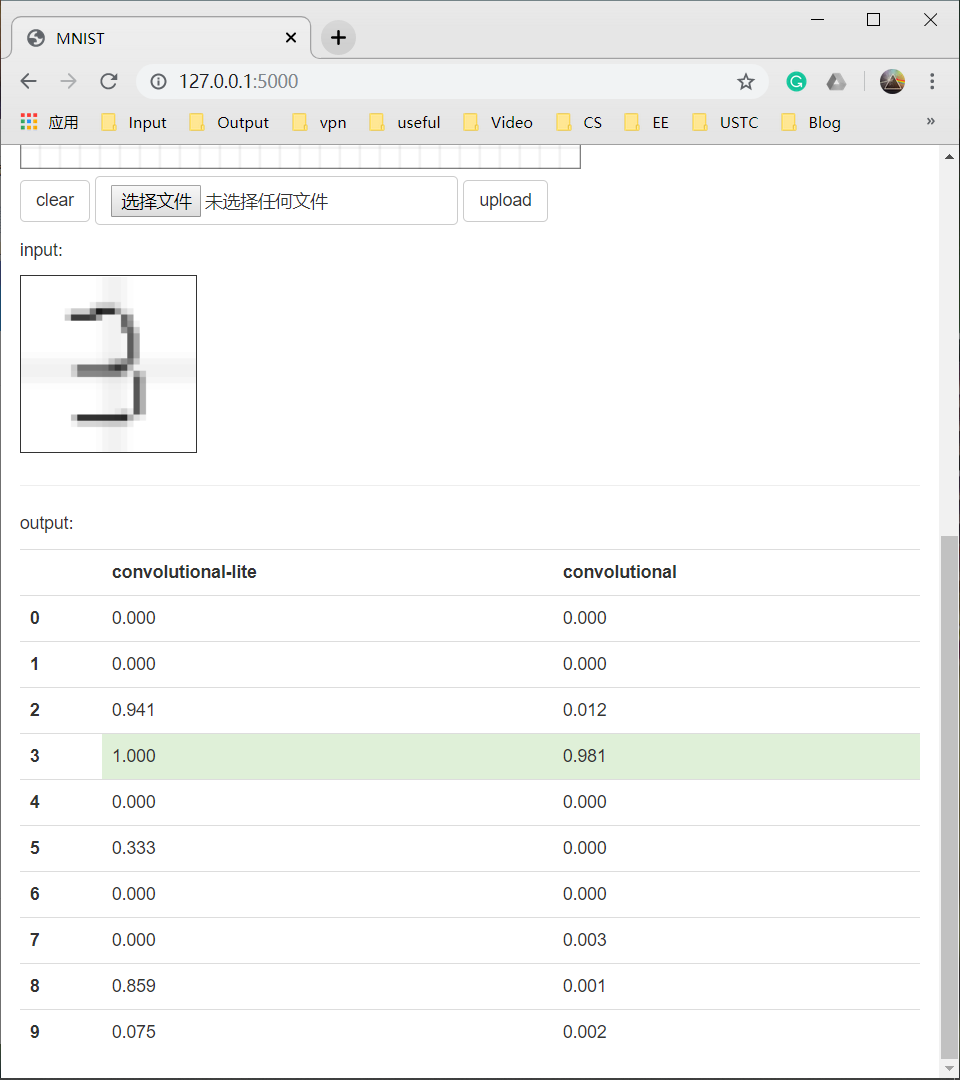


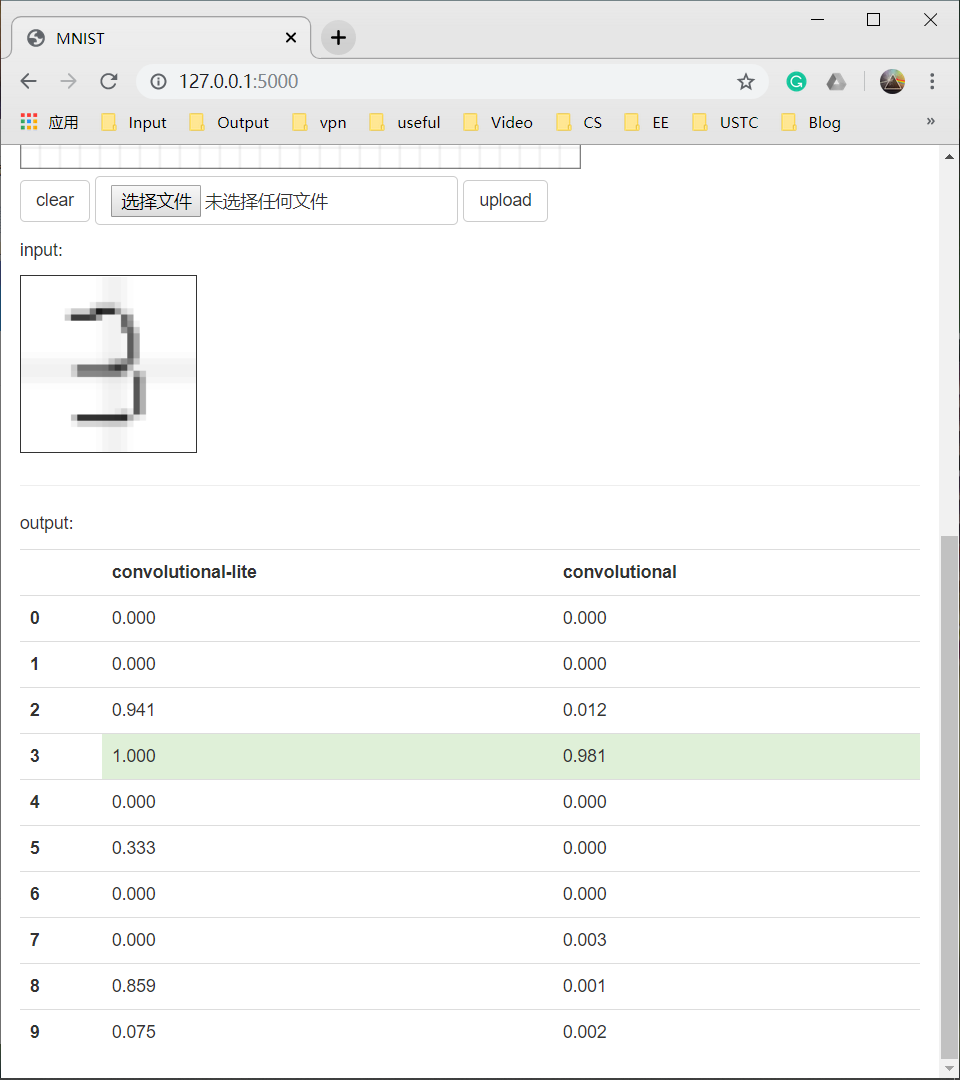
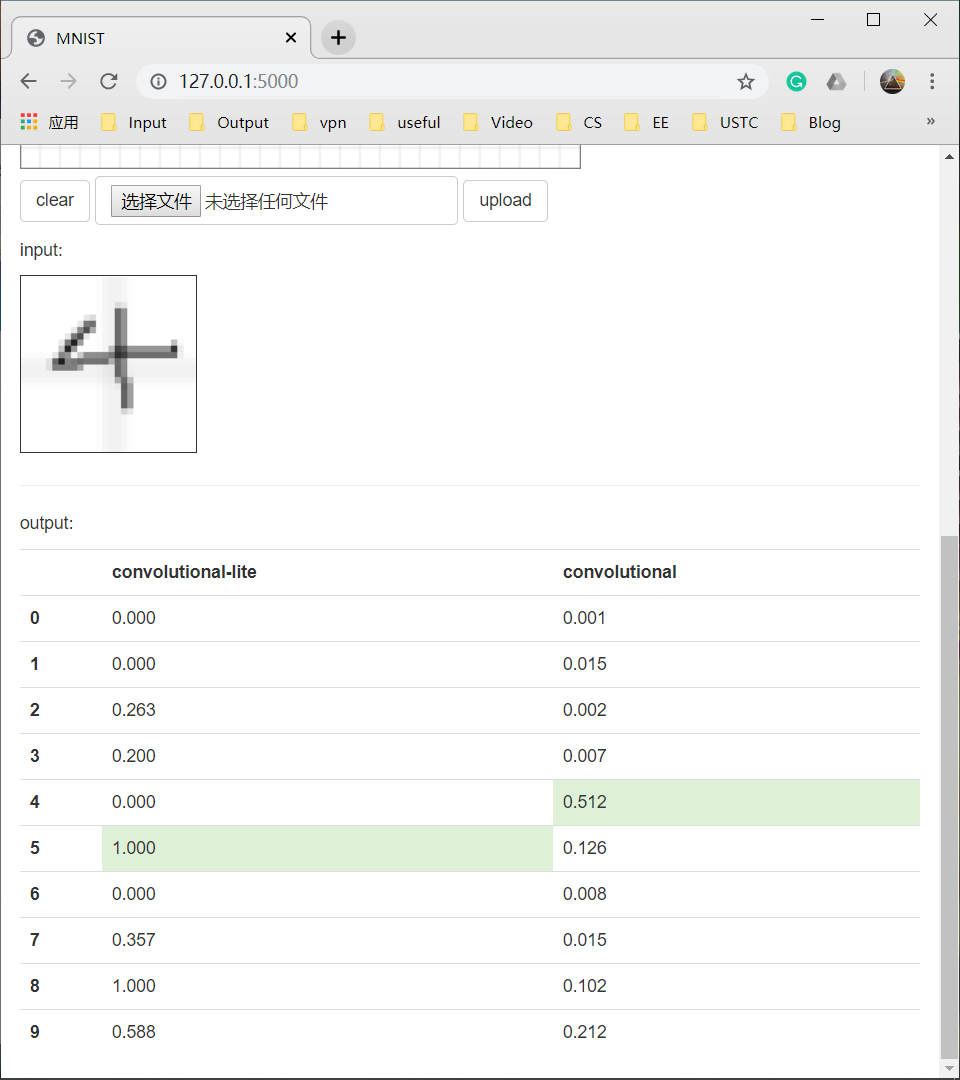


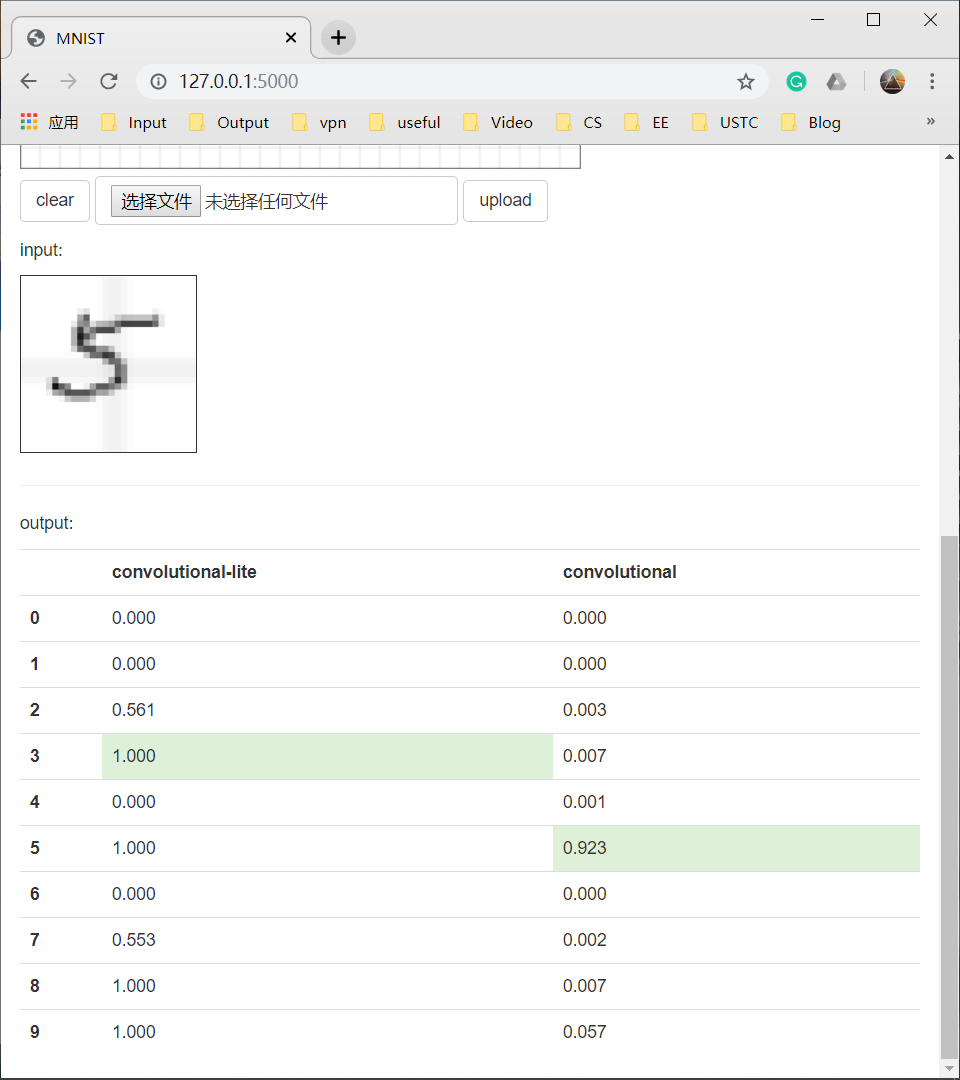
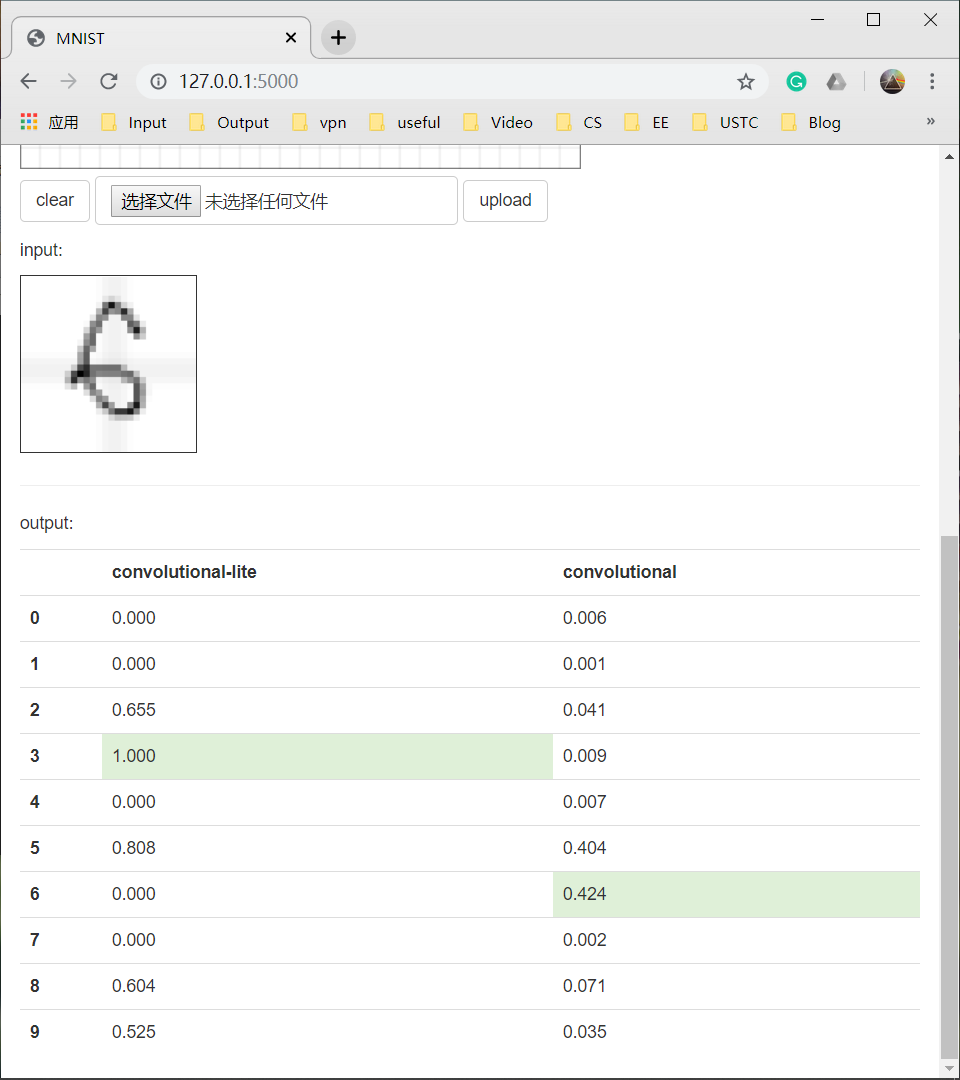
先试一试书写，然后输入0-9，看看压缩前后的模型的表现情况

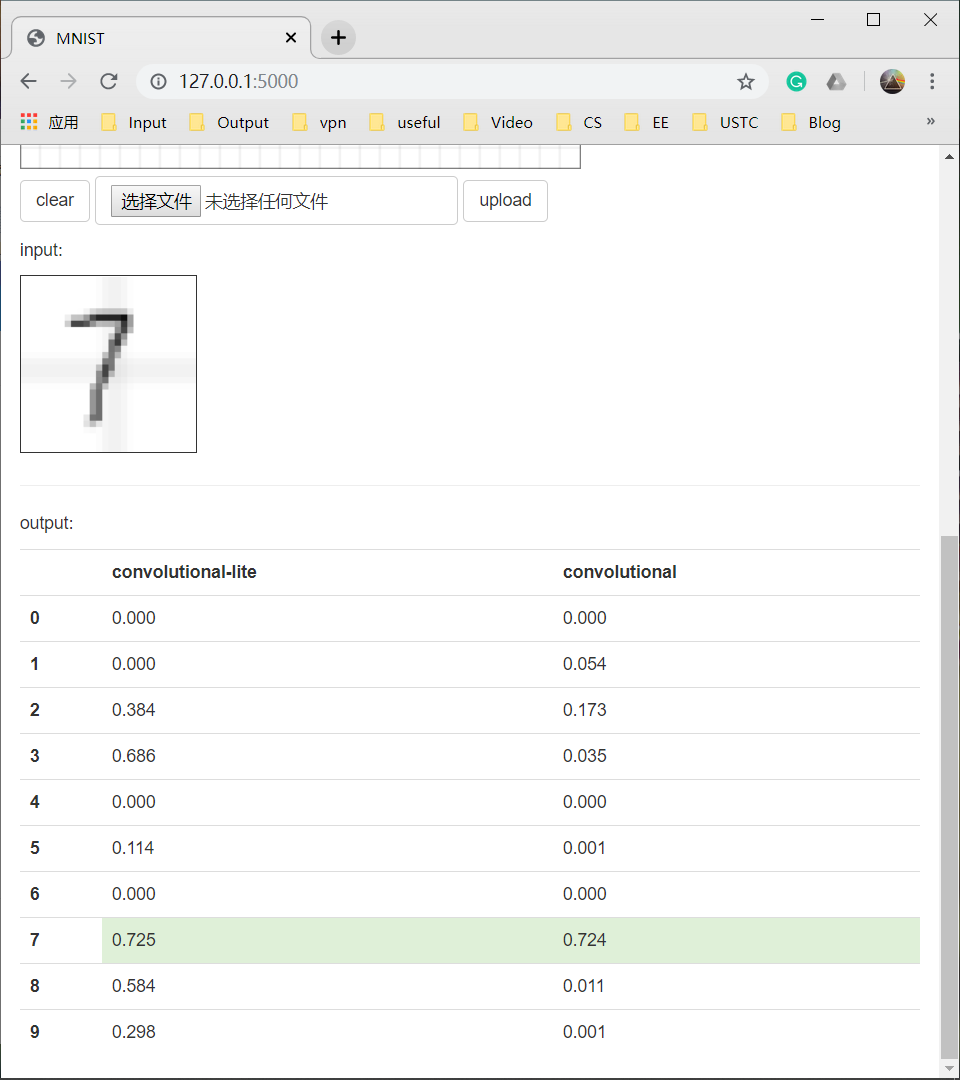
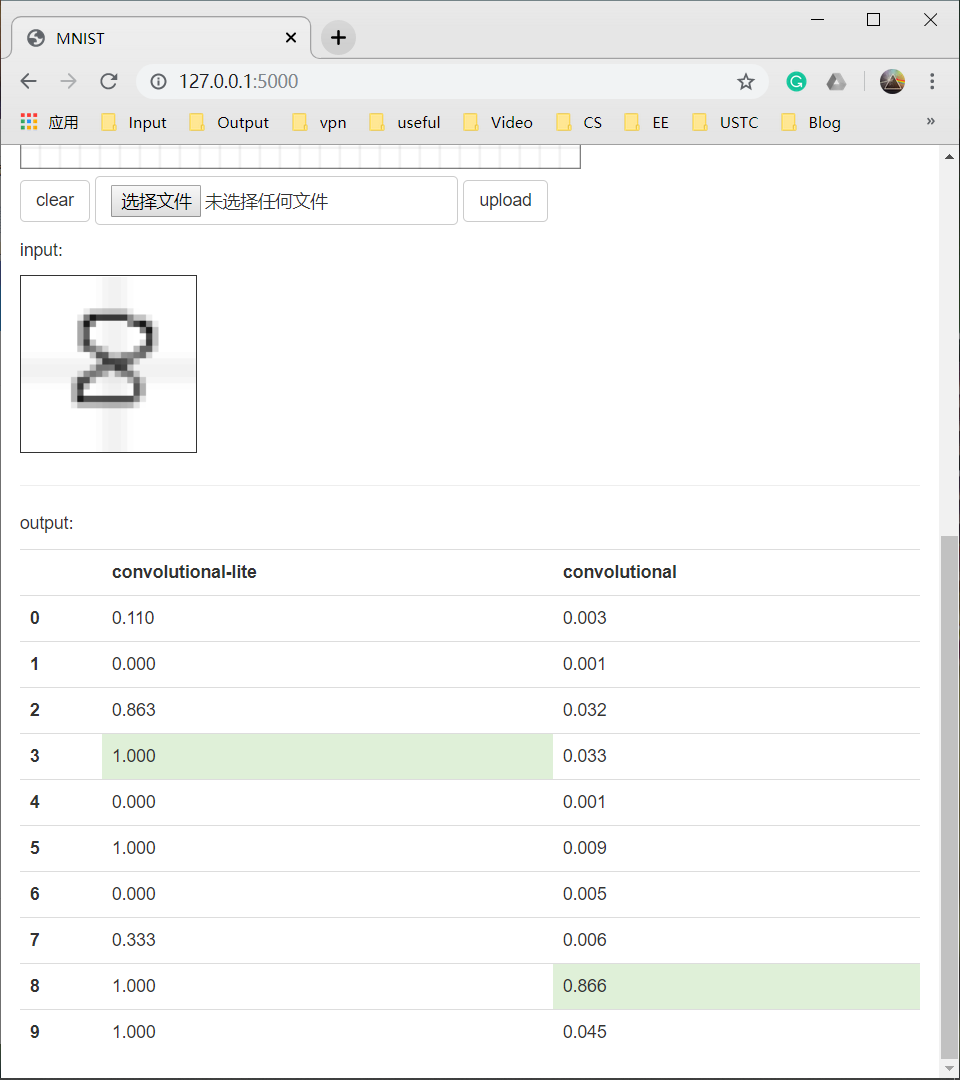


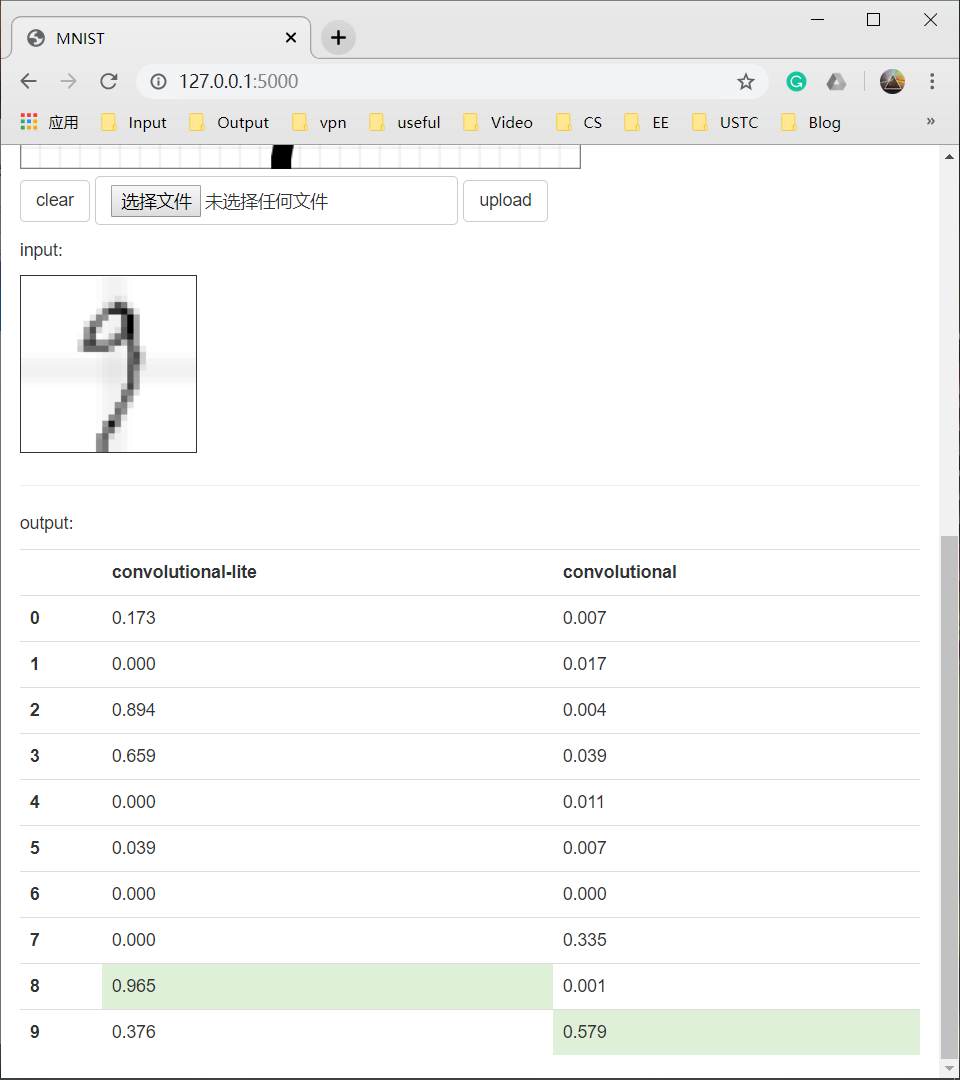
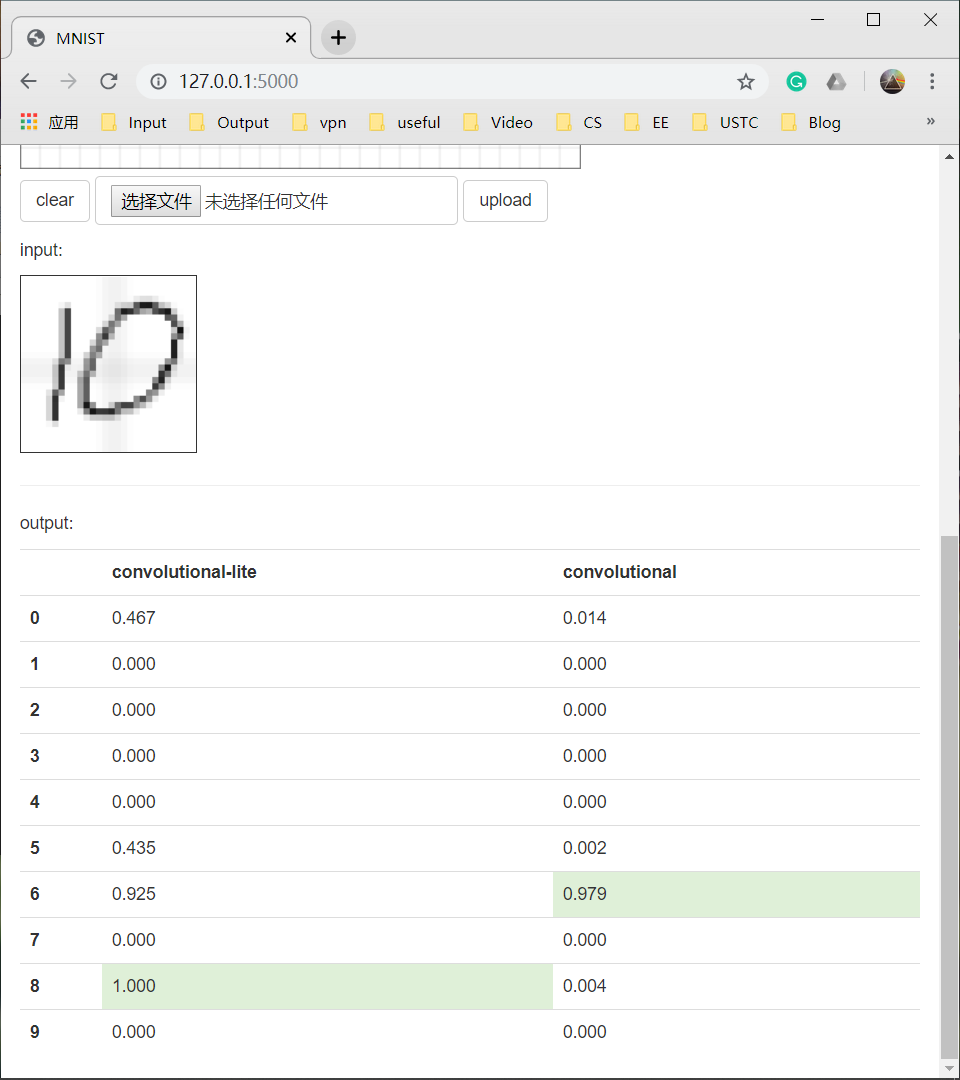
 

压缩后模型准确率略有下降，压缩模型带来的准确率下降是下一个值得研究的问题