邹天舒

北京交通大学 | 指导老师 魏祥

手写数字识别

机器学习第三次

**目录**

[1. 摘要 2](#_Toc39574494)

[2. 需求分析 2](#_Toc39574495)

[3. 实验过程 2](#_Toc39574496)

[3.1 模型训练 2](#_Toc39574497)

[3.2 模型优化 4](#_Toc39574498)

[3.3 识别结果 7](#_Toc39574499)

[4. 总结 8](#_Toc39574500)

# 摘要

卷积神经网络（Convolutional Neural Networks / CNNs / ConvNets）与普通神经网络非常相似，它们都由具有可学习的权重和偏置常量(biases)的神经元组成。每个神经元都接收一些输入，并做一些点积计算，输出是每个分类的分数，普通神经网络里的一些计算技巧到这里依旧适用。

所以哪里不同呢？卷积神经网络默认输入是图像，可以让我们把特定的性质编码入网络结构，使是我们的前馈函数更加有效率，并减少了大量参数。

# 需求分析

基于Tensorflow.js的在线手写数字识别

基本需求：

1.   三个js文件，分别完成：网络训练以及模型保存、模型加载及准确率测试、在线手写数字识别；

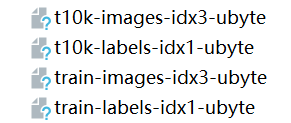
2.  模型测试准确率要高于99.3%（尽量）；

3.  在线手写数字识别需要能够通过鼠标在画布中写入0~9数字，并进行实时识别，按空格键清除（下图示例）。测试需具有一定的准确性。

# 实验过程

## 模型训练

在进行模型训练之前，我们需要加载数据集，将下载好的数据集放到项目目录中：



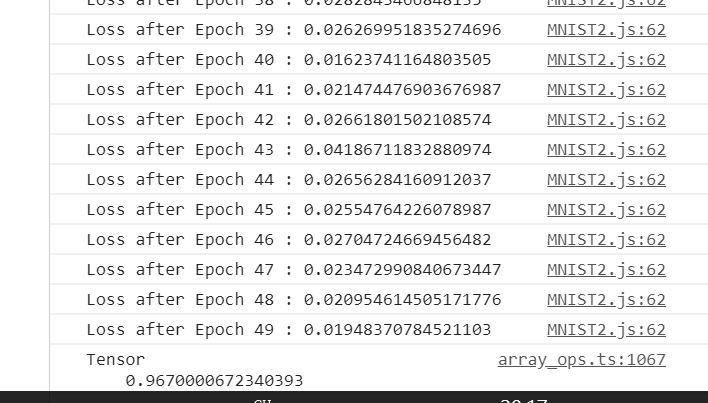
然后读取文件：

function *loadMNIST*(callback) {  
 let mnist = {};  
 let files = {  
 train\_images: 'train-images-idx3-ubyte',  
 train\_labels: 'train-labels-idx1-ubyte',  
 test\_images: 't10k-images-idx3-ubyte',  
 test\_labels: 't10k-labels-idx1-ubyte',  
 };  
 return *Promise*.all(*Object*.keys(files).map(async file => {  
 mnist[file] = await *loadFile*(files[file])  
 }))  
 .then(() => callback(mnist));  
}

我们为模型添加如下的网络进行训练：

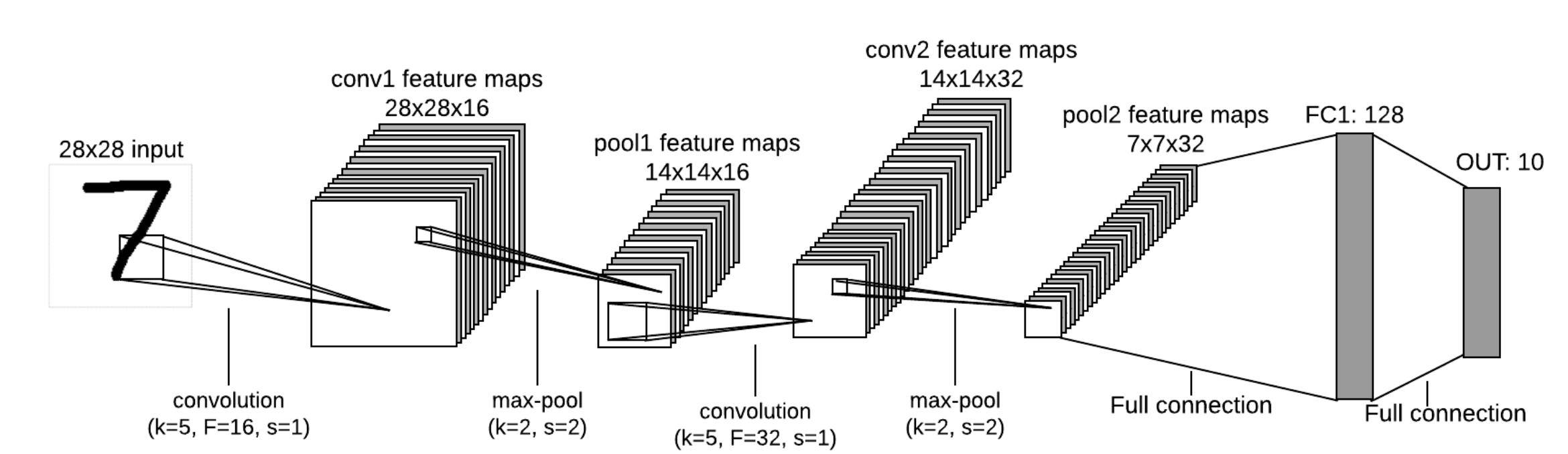
*model*.add(tf.layers.dense({  
 units: 128,  
 inputShape: [784],  
 activation: 'relu'  
 }));  
   
*model*.add(tf.layers.dense({  
 units: 64,  
 activation: 'relu'  
 }));  
   
*model*.add(tf.layers.dense({  
 units: 64,  
 activation: 'relu'  
 }));  
   
*model*.add(tf.layers.dense({  
 units: 10,  
 }));

运行出来的准确度约是0.967，运行截图如下：



## 模型优化

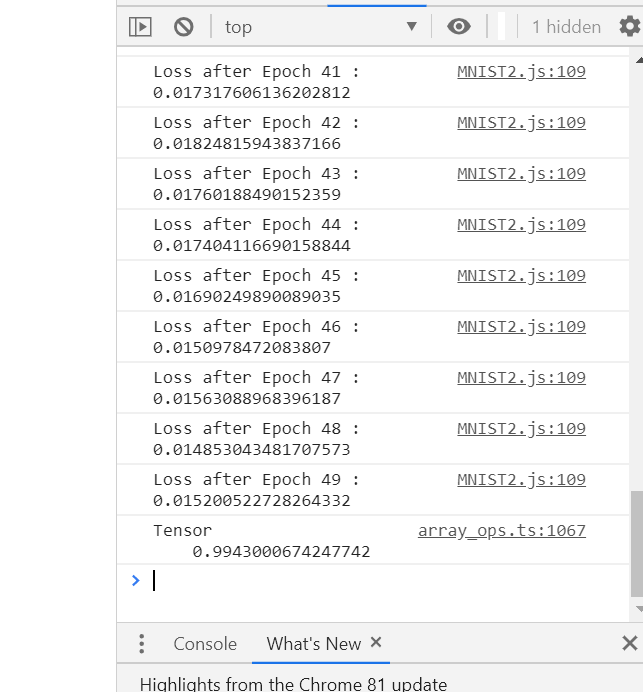
那么为了提高模型训练的精确度，我们可以采用CNN神经网络，如下图所示：

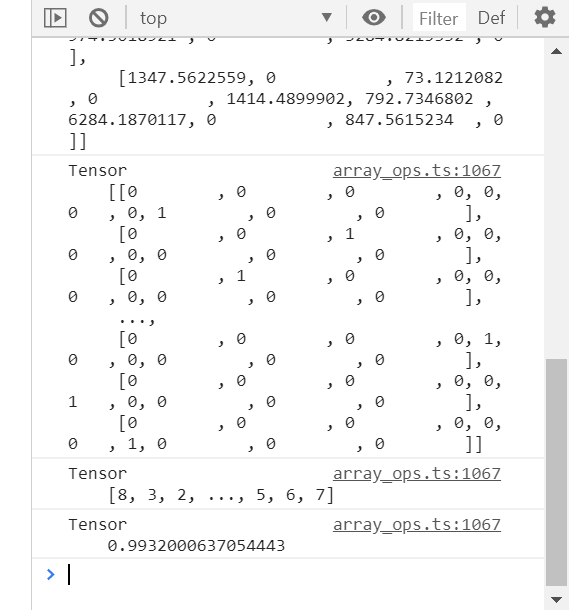


为了更加提高训练的准确度，可以添加dropout来进行正则化。

*model*.add(tf.layers.conv2d({  
 inputShape: [28, 28, 1],  
 kernelSize: 5,  
 filters: 16,  
 strides: 1,  
 activation: 'relu',  
 kernelInitializer: 'varianceScaling'  
}));  
*model*.add(tf.layers.maxPooling2d({  
 poolSize: [2, 2],  
 strides: [2, 2]  
}));  
*model*.add(tf.layers.conv2d({  
 kernelSize: 5,  
 filters: 32,  
 strides: 1,  
 activation: 'relu',  
 kernelInitializer: 'varianceScaling'  
}));  
*model*.add(tf.layers.maxPooling2d({  
 poolSize: [2, 2],  
 strides: [2, 2]  
}));  
*model*.add(tf.layers.dropout({  
 rate:0.5,  
}));  
*model*.add(tf.layers.flatten());  
*model*.add(tf.layers.dense({  
 units: 128,  
 activation: 'relu'  
}));  
*model*.add(tf.layers.dense({  
 units: 10,  
 activation: 'relu'  
}));

通过修改网络模型，我们将训练的准确率提高到了99.3%以上，运行截图如下：





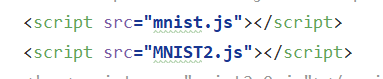
## 识别结果



## 目录结构



首先有两个html文件，index.html是用来进行模型训练保存的，load.html是用来加载模型然后根据训练的模型进行识别用户的手写数字的。



Index.html中导入图上两个js文件，一个用来读取数据集一个用来训练模型



load.html中导入如上图两个文件，除了读取数据集的js文件，result是用来加载训练模型然后进行手写数字识别的。

# 总结

经过这次实验，尤其是在进行模型的优化实验的时候，深刻的体会到了，CNN神经网络运用，以及dropout的应用，如何运用适合的模型对具体的数据集进行训练一步步提高模型的准确率。