一、前沿

本文代码是我在学习TensorFlow时写的CNN方面的程序，主要是根据AlexNet模型编写，但是自距此模型问世以来，深度学习领域发生了很大变化，所以其中某些方法和参数已经有所改变，目前此模型通过在GPU上训练，精确度可以达到97%左右，下面我将详细介绍Tensorflow中的实现，本教程适合TensorFlow的初学者，主要有以下特点：

- 使用Mnist数据集
- 测试精度可达97%
- 可视化输出
- 兼具卷积层、pooling（池化）层、dropout层、全连接层、softmax layer
- 提供实现此模型原始代码

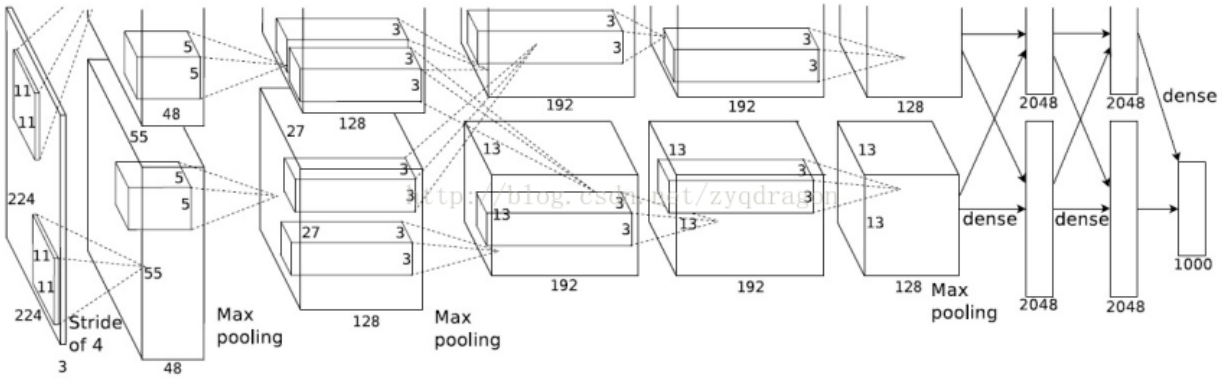
代码实现：

代码地址：https://github.com/WzsGo/AlexNet_mnist_TensorFlow.git

- **AlexNet_mnist.py**

基础版本，使用tf.nn.***函数实现，注重构造网络中的细节实现，容易理解CNN的运行原理，从细节理解CNN结构。

二、模型结构



AlexNet将LeNet的思想发扬光大，把CNN的基本原理应用到了很深很宽的网络中。AlexNet主要使用到的新技术点如下。

（1）成功使用ReLU作为CNN的激活函数，并验证其效果在较深的网络超过了Sigmoid，成功解决了Sigmoid在网络较深时的梯度弥散问题。虽然ReLU激活函数在很久之前就被提出了，但是直到AlexNet的出现才将其发扬光大。

（2）训练时使用Dropout随机忽略一部分神经元，以避免模型过拟合。Dropout虽有单独的论文论述，但是AlexNet将其实用化，通过实践证实了它的效果。在AlexNet中主要

是最后几个全连接层使用了Dropout。

（3）在CNN中使用重叠的最大池化。此前CNN中普遍使用平均池化，AlexNet全部使用最大池化，避免平均池化的模糊化效果。并且AlexNet中提出让步长比池化核的尺寸小，这样池化层的输出之间会有重叠和覆盖，提升了特征的丰富性。

（4）提出了LRN层，对局部神经元的活动创建竞争机制，使得其中响应比较大的值变得相对更大，并抑制其他反馈较小的神经元，增强了模型的泛化能力。

三、代码实现简介

(1)定义的超参数：dropout层的保留概率/每次提取图片数目/迭代次数

train_keep_prop = 0.0045/batch_size = 128/epcoh = 1800

可以更改这些参数，以改善训练结果。

(2)模型结构

input - 输入数据：Mnist

数据集被分成两部分：60000 行的训练数据集（mnist.train）和10000行的测试数据集（mnist.test），每张图片是 28*28*1，经过reshape后，维度格式为：[28,28,1]

conv1 - 卷积层：

卷积层：kenel：3*3*64，strides = 1,padding = SAME -->> [28,28,64]

池化层：ksize: 2*2 ,strides = 2 -->> [14,14,64]

Normal: -->> [14,14,64]

drop层：keep_prop = train_keep_prop -->> [14,14,64]

conv2 - 卷积层：

卷积层：kenel：3*3*128，strides = 1,padding = SAME -->> [14,14,128]

池化层：ksize: 2*2 ,strides = 2 -->> [7,7,128]

Normal: -->> [7,7,128]

drop层：keep_prop = train_keep_prop -->> [7,7,128]

conv3 - 卷积层：

卷积层：kenel：3*3*256，strides = 1,padding = SAME -->> [7,7,256]

池化层：ksize: 2*2 ,strides = 2 -->> [4,4,256]

Normal: -->> [4,4,256]

drop层：keep_prop = train_keep_prop -->> [4,4,256]

Flaten层：

将[4,4,256]矩阵形式 -->> [4*4*256]向量形式

Fucn1 - 全连接层：

权重：[4*4*256,1024]

Fucn2 - 全连接层：

权重：[1024,256]

Fucn3 - 全连接层：

权重：[256,10]

Softmax层：

输出one-hot向量，对应每一类的概率。

四、代码实现

代码地址：https://github.com/WzsGo/AlexNet_mnist_TensorFlow.git

- **AlexNet_mnist.py**

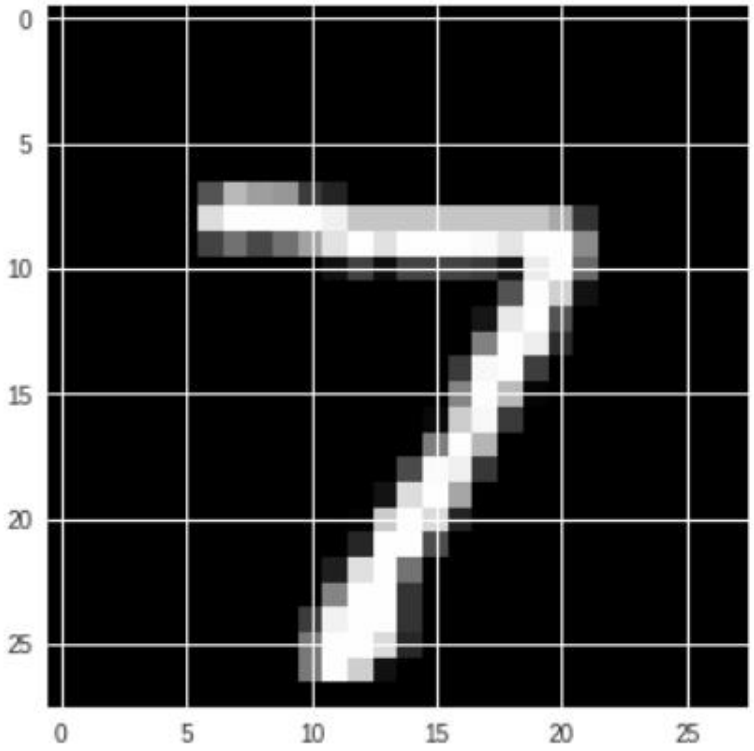
基础版本，使用tf.nn.***函数实现，注重构造网络中的细节实现，容易理解CNN的运行原理，从细节理解CNN结构。

五、执行程序

执行命令： python AlexNet_mnist.py

六、训练结果

```
Loss : 464.80823 | Train Accuracy : 0.953125 | Test Accuracy : 0.964
Loss : 365.61957 | Train Accuracy : 0.9609375 | Test Accuracy : 0.973
Loss : 149.6302 | Train Accuracy : 0.9296875 | Test Accuracy : 0.972
Loss : 388.74878 | Train Accuracy : 0.9296875 | Test Accuracy : 0.968
Loss : 437.18393 | Train Accuracy : 0.9609375 | Test Accuracy : 0.953
Loss : 151.91669 | Train Accuracy : 0.9609375 | Test Accuracy : 0.97
Loss : 334.14264 | Train Accuracy : 0.9765625 | Test Accuracy : 0.972
--------------------Compare to True and Test---------------------------
```




```
True label : [7]
Test label : [7]
```