计算机科学与技术 学院

机器学习 课程实验报告

学号：	姓名：	班级：
实验题目：手势识别		
实验学时：10	实验日期：　　2018/12/15-2018/12/20	

实验目的：

1 介绍：

在这个项目中，你们需要实现手势识别。手势识别即，输入一张图片，输出这张图片所代表的手势数字。

2 数据集

手势识别的数据集来自于 Turkey Ankara Ayrancı Anadolu High School'sSignLanguageDigitsDataset，注意到这个数据集由土耳其人制作， 所以用来表示数字 3 的手势会和中国人表示 3 的手势有略微差异，按照这个数据集的手势训练与测试即可。

图像大小：

100*100 像素 颜色空间：RGB 种类：6 种(0,1,2,3,4,5) 每种图片数量：200 张

百度云链接：

https://pan.baidu.com/s/1aruOTvbm4XgJ9LKIHNIb5g 提取码：6kar

3 实现方法

任何方法均可。自选。

硬件环境：

操作系统

Windows 10 家庭中文版 64-bit

CPU

Intel Core i5 7200U @ 2.50GHz　41 ° C

Kaby Lake-U/Y 14nm 工艺

RAM

8.00GB 单个的-通道 未知 (15-15-15-35)

主板

HP 81D1 (U3E1)

图像

Generic PnP Monitor (1920x1080@60Hz)

Intel HD Graphics 620 (HP)

存储器

476GB NVMe THNSN5512GPUK TO (未知)

40GB Microsoft 虚拟磁盘 (File-backed Virtual)

光盘驱动器

没有检测到光纤磁盘驱动

音频

Conexant ISST Audio

软件环境：

Win10 + python3.6 + tensorflow1.10.0 + opencv3.4.4

实验步骤与内容：

一、数据集介绍与划分

数据集 git 链接：

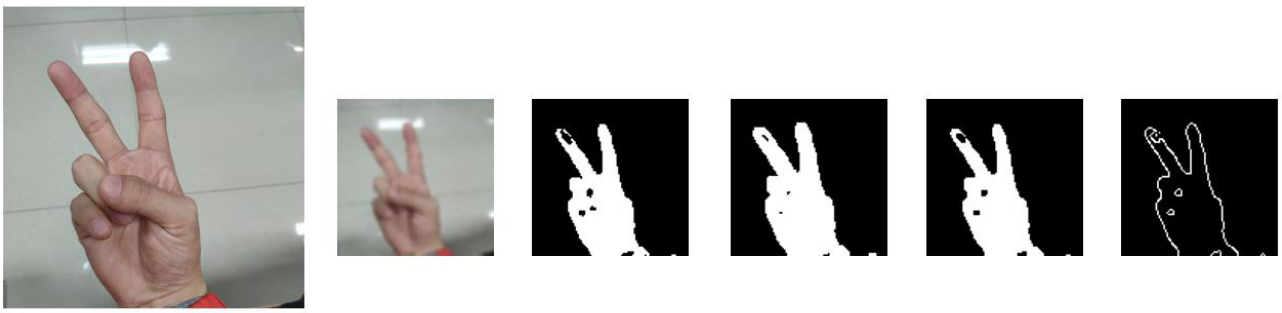
https://github.com/ardamavi/Sign-Language-Digits-Dataset/

一共 6 种手势，每种手势 200 张图片，共 1200 张图片（100x100RGB）

数据集划分：9:1（1080：120），每种手势 180 张用于训练，20 张用于测试

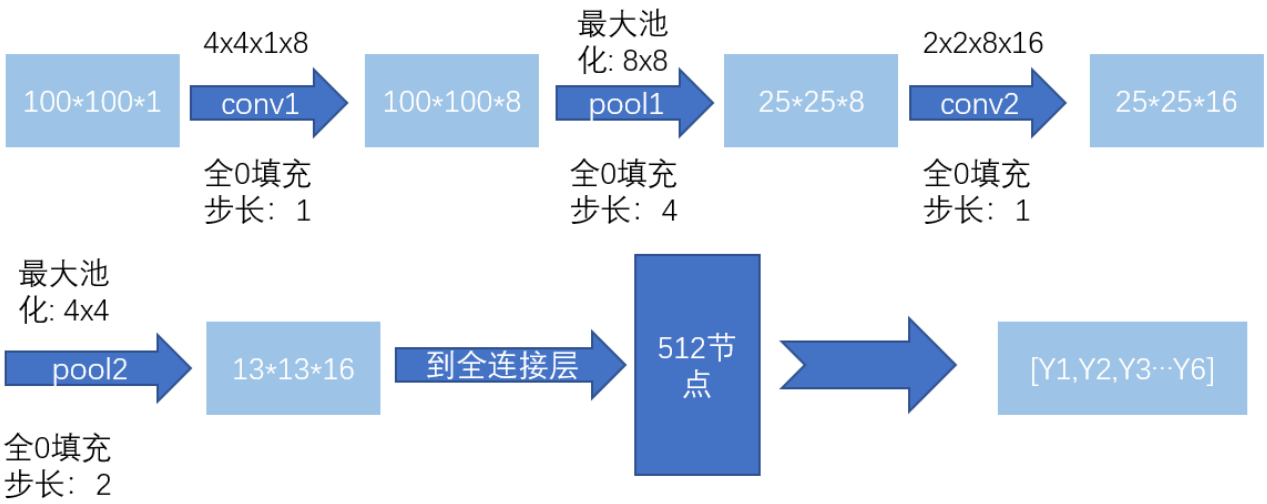
二、图片预处理及 CNN 模型(输入/输出)

实际图片处理展示：resize 前先高斯模糊，提取边缘后可以根据实际需要增加一次中值滤波去噪：



三、神经网络结构及具体参数

使用 tensorflow 的框架：




```
def forward(x, train, regularizer):

    conv1_w = get_weight([CONV1_SIZE, CONV1_SIZE, NUM_CHANNELS, CONV1_KERNEL_NUM], regularizer)
    conv1_b = get_bias([CONV1_KERNEL_NUM])
    conv1 = conv2d(x, conv1_w)
    relu1 = tf.nn.relu(tf.nn.bias_add(conv1, conv1_b))
    pool1 = max_pool_8x8(relu1)

    conv2_w = get_weight([CONV2_SIZE, CONV2_SIZE, CONV1_KERNEL_NUM, CONV2_KERNEL_NUM],regularizer)
    conv2_b = get_bias([CONV2_KERNEL_NUM])
    conv2 = conv2d(pool1, conv2_w)
    relu2 = tf.nn.relu(tf.nn.bias_add(conv2, conv2_b))
    pool2 = max_pool_4x4(relu2)

    pool_shape = pool2.get_shape().as_list()
    nodes = pool_shape[1] * pool_shape[2] * pool_shape[3]
    reshaped = tf.reshape(pool2, [pool_shape[0], nodes])

    fc1_w = get_weight([nodes, FC_SIZE], regularizer)
    fc1_b = get_bias([FC_SIZE])
    fc1 = tf.nn.relu(tf.matmul(reshaped, fc1_w) + fc1_b)
    if train: fc1 = tf.nn.dropout(fc1, 0.5)

    fc2_w = get_weight([FC_SIZE, OUTPUT_NODE], regularizer)
    fc2_b = get_bias([OUTPUT_NODE])
    y = tf.matmul(fc1, fc2_w) + fc2_b
    return y
```

Dropout： 增加鲁棒性帮助正则化和避免过拟合

一个相关的早期使用这种技术的论文（（**ImageNet Classification with Deep Convolutional Neural Networks, by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton (2012).））中启发性的 dropout 解释是：这种技术减少了神经元之间复杂的共适性。因为一个神经元不能依赖其他特定的神经元。因此，不得不去学习随机子集神经元间的鲁棒性的有用连接。换句话说。想象我们的神经元作为要给预测的模型，dropout 是一种方式可以确保我们的模型在丢失一个个体线索的情况下保持健壮的模型。在这种情况下，可以说他的作用和 L1 和 L2 范式正则化是相同的。都是来减少权重连接，然后增加网络模型在缺失个体连接信息情况下的鲁棒性。在提高神经网络表现方面效果较好。

四、神经网络结构及具体参数

- 激活函数：relu

- 损失函数计算：

```
ce=tf.nn.sparse_softmax_cross_entropy_with_logits(logits=y,
labels=tf.argmax(y_, 1))
cem = tf.reduce_mean(ce)
loss = cem + tf.add_n(tf.get_collection('losses'))
```

- 动态调整学习率，采用指数衰减
- 采用梯度下降优化：tf.train.GradientDescentOptimizer
- 使用滑动平均，增加模型泛化能力
- 使用 L2 正则化，超参为：0.0001

```
tf.add_to_collection('losses',
tf.contrib.layers.l2_regularizer(regularizer)(w))
```

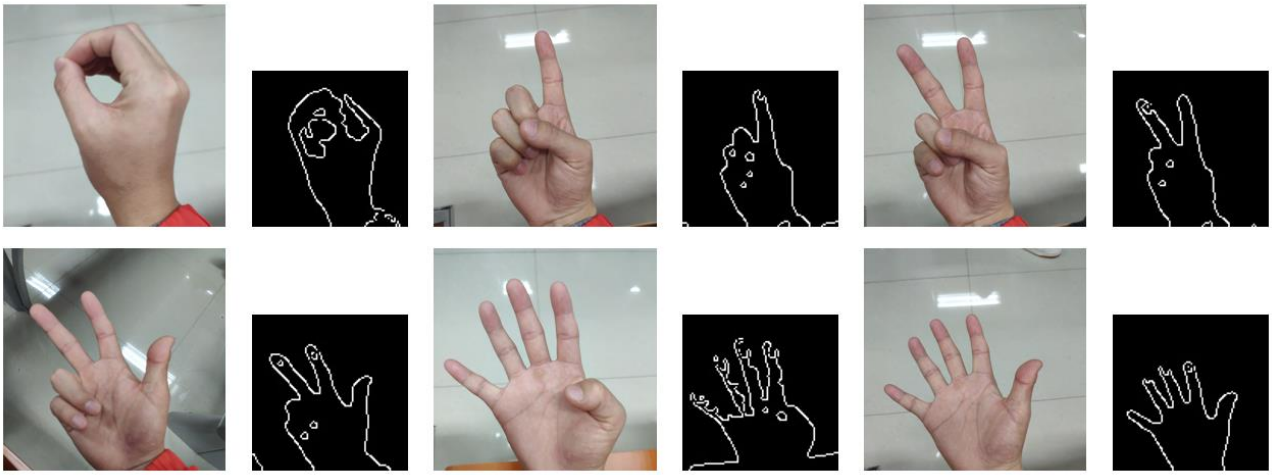
五、实验训练过程及结果

经过约 4800 轮的训练后，loss 基本收敛，在 0.6 左右，
在 120 份的测试样本上的模型准确率能够达到约 96%

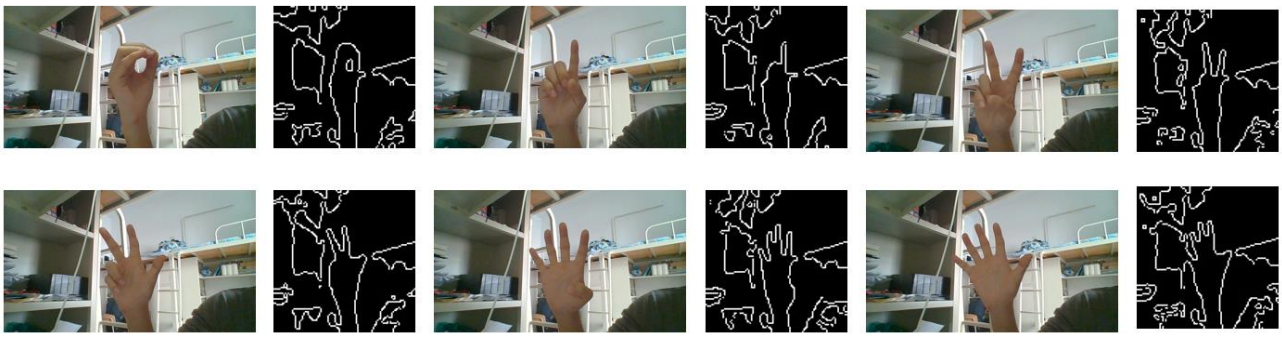
```
After 4602 training step(s), loss on training batch is 0.597605.
After 4702 training step(s), loss on training batch is 0.57159.
After 4802 training step(s), loss on training batch is 0.589368.
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
After 4802 training step(s), test accuracy = 0.958333
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
After 4802 training step(s), test accuracy = 0.958333
```

六、进行实际应用

1、对输入图片处理后进行手势识别，分别测试在简单背景下，一般复杂背景下、以及复杂背景下的模型识别效果：



2、复杂背景干扰：



```
input the number of test pictures:6

the path of test picture:0.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [0]

the path of test picture:1.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [1]

the path of test picture:2.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [2]

the path of test picture:3.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:4.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:5.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]
```

```
input the number of test pictures:6

the path of test picture:00.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:11.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:22.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:33.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:44.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]

the path of test picture:55.jpg
INFO:tensorflow:Restoring parameters from ./model/my_geature_model-4802
The prediction number is: [3]
```

分析：

对于一般手势的角度与数据集中一样比较正直的有较高的识别准确率，但是对于倾斜较大或噪声很难去除的情况会干扰识别，导致出错。

应对方法：

进一步缩小肤色的 RGB 值范围，对手势结构进行更精确的提取，目前的肤色提取范围为：

```
lowerBoundary = np.array([0,40,30],dtype="uint8")
upperBoundary = np.array([43,255,254],dtype="uint8")
```

3、对输入视频进行实时的识别：见附件 ges_pro02.mp4

结论分析与体会：

通过实验，对神经网络的实际应用有了更深入的了解，熟悉了在 tensorflow 环境下搭建网络结构的流程，对于整体框架有了更多的认识，在实验过程中对于各种参数有了更深入的了解，不同的损失函数等也有了更为直观的认识，另一方面也通过实际的应用对深度学习有了更为浓厚的兴趣，希望以后能更加深入地进行学习。

附录：程序源代码