程序说明文档

项目：基于深度卷积神经网络的“慧眼识人”——勾勒图片中的人物轮廓

团队编号：309 号团队

1. 程序简介

该程序是针对“中国云·移动互联网创新大奖赛”人才技术类赛题 A 任务二——慧眼识人而设计的一个深度学习算法。该算法直接以像素作为模型输入，中间经过 3 层可训练的卷积层和 1 个隐层进行变换，最后以 logistic 回归输出每个像素的分类概率。

算法一共可以分为两部分：第一部分利用样本训练模型，第二部分利用已训练好的模型来分割图片，输出图片中人物区域的剪影。

该算法主要用到 python 中的免费开源的深度学习库——theano，这个库可以利用 GPU 极大提高计算速度。在本次任务中我们采用 GPU GTX650 进行计算，训练时间约为 4-5 天。训练后的效果为：本地 i5 四核配置下平均每张图片处理时间为 1.14s，交并比为 61.96%。

2. 运行环境

- 操作系统：Ubuntu 12.04，32-bit，内核 Linux 3.5.0
- 编程语言：python 2.7.3
- 所用到的库版本说明：
- Numpy 1.6.1
- Scipy 0.9.0 (此版本会出现一个 warning，但不影响运行）
- PIL 1.1.7
- Theano 0.6.0rc3

3. 使用方法

```
$ cd ./src/recog/
$ ./run.sh dir1 dir2
```

（dir1：待处理的图片目录， dir2：保存剪影图片的目录）

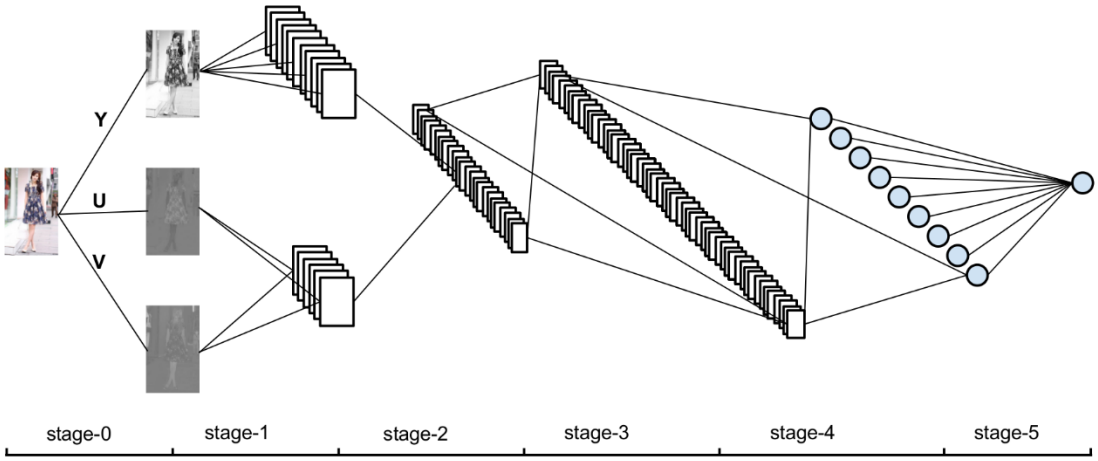
例子：$ ./run.sh /home/dailiang/train-origin-pics/ /home/dailiang/profiles-pics/

注意事项：

1. shell 脚本开启四个进程，若程序运行中断，请手动将后台的三个 python 进程关闭；
2. 产生的结果是与原图尺寸相同的 256 灰度级的单通道 jpg 图片。

4. 代码说明

该算法采用的卷积神经网络模型如下图（有关模型的详细情况请参考《系统概述》文档）：



a) 训练部分：./src/train/

第一步：preProcess.py 将原始图片进行预处理并和对应剪影图片打包存储。预处理流程为：1）将原始图片裁剪、缩小成 320*240 像素大小 → 转换到 YUV 空间 → 对每个通道的图片进行 15*15 大小的局部归一化（有利于卷积神经网络训练）→ 对对应的剪影图片进行同样的裁剪 → 将黑色区域设定值为 1，白色区域设定为 0 → 将处理后的 YUV 以及 label 图片压缩保存。

第二步：cnn_logistic_regression.py 训练卷积层参数。由上图的三层卷积神经网络 stage-1、stage-2、stage-3 后加一层 logistic regression 组合成一个网络，以 cross-entropy 作为目标函数，以一张图片为单位进行 minibatch SGD，利用 bp 算法（theano 自动求导）调整参数，其中学习率设置为 0.001，训练完毕后保存卷积层的参数。这个步骤的训练时间大概需要 4 天。

第三步：cnn_mlp.py 训练多层感知器的参数。去掉第二步的 logistic regression，在卷积层后面添加 hidden layer 和 logistic regression 组建一个如上图的完整网络。导入并固定第二步训练好的卷积层参数，用同样的方法调整 hidden layer 和 logistic regression 的参数，训练完毕保存所有模型参数。学习率设定为 0.0005，这个过程大概需要 10 个小时。

第四步：fine_tuning.py 对整个网络进行微调。与第三步设置相似，但不固定卷积层参数，设置学习率为 0.0002，这个过程大概持续 12 个小时，保存训练所得的参数作为第二部分的模型参数。

b) 识别部分：./src/recog/

第一步：preProcess.py 进行预处理。对原始图片进行预处理，流程与第一部分一样，预处理后的 YUV 通道的图像数据作为神经网络的输入。

第二步：process0.py process1.py process2.py process3.py 并行识别。预处理后的图片数据流经上图的神经网络，得到每个像素的标签。开启四个进程是为了充分利用多核计算机的计算能力，缩短识别时间。

第三步：postProces.py 去除噪点。利用五个不同大小卷积核去掉孤立的黑点与白点，可将正确率提高 2.0%。