2018-09-第一周

`CASIA`

---

1. 工作总结与安排

本周工作总结

1. 本周工作进度，主要是看了VoxelNet一部分的论文，孟老师这边安排让着手阿里竞赛链接，一个思路跟上周看过的SqueezeSeg类似，我这边在ubuntu系统上配置了环境并拷贝了数据，代码的demo.py部分已经跑通，准备结合代码看下论文；
2. 今天开周会，调整了对深度学习的学习方法把握，发现自己应该把重点放在通过对深度学习工具的使用，来完成一些实际性的工作，而不是对于工具本身的研究；
3. 基于上一条的学习方向调整，之后的工作应该集中放在对工具的了解以及使用上，各种概念性的东西了解后可以入手开始修改代码；
4. 总结：本周进度缓慢，学习效率有待提高，应该加强对论文阐述的内容做深入了解；

下周工作安排

1. 完成SqueezeSeg代码的训练部分；
2. 研究SqueezeSeg的输入输出，尝试修改部分参数；
3. 与王腾讨论合作把竞赛用数据转化成SqueezeSeg网络可用的数据，用该网络尝试训练；

2. 论文

- SqueezeSeg阅读
  原文链接，github上不支持公式显示，链接为作业部落。
- VoxelNet阅读
  原文部分翻译

3. 代码

Python开发工具：virtualenv

使用virtualenv

> Python 的第三方包成千上万，在一个 Python 环境下开发时间越久、安装依赖越多，就越容易出现依赖包冲突的问题。为了解决这个问题，开发者们开发出了 virtualenv，可以搭建虚拟且独立的 Python 环境。这样就可以使每个项目环境与其他项目独立开来，保持环境的干净，解决包冲突问题。

- 安装virtualenv

```
$ pip install virtualenv
```

- 添加参数

```
# 使用系统包
-system-site-packages
# 不包含任何包
-no-site-packages
```

- 指定版本

```
$ virtualenv -p /usr/bin/python2.7 venv
# 其中/usr/bin/python2.7是所要用到的python版本的安装路径
```

- 进入及退出虚拟环境

```
# 进入
$ source /venv/bin/activate
# 退出
$ deactivate
```

使用virtualenvwrapper

> 这是 virtualenv 的扩展工具，提供了一系列命令行命令，可以方便地创建、删除、复制、切换不同的虚拟环境。同时，使用该扩展后，所有虚拟环境都会被放置在同一个目

> 录下。

- 安装virtualenvwrapper

```
pip install virtualenvwrapper
```

- 配置环境

```
# 添加以下代码到~/.bashrc(或~/.zshrc)
if [ -f /usr/local/bin/virtualenvwrapper.sh ]; then
   export WORKON_HOME=$HOME/.virtualenvs
   source /usr/local/bin/virtualenvwrapper.sh
fi
# .virtualenvs 是可以自定义的虚拟环境管理目录
$ source ~/.bashrc
# 使用 virtualenvwrapper
```

- 使用方法
  mkvirtualenv 也可以使用 virtualenv 的参数，比如 –python 来指定 Python 版本。创建虚拟环境后，会自动切换到此虚拟环境里。虚拟环境目录都在 WORKON_HOME 里。

```
$ mkvirtualenv venv

# 列出虚拟环境
$ lsvirtualenv -b

# 切换环境
$ workon [虚拟环境名称]

# 查看环境安装的包
$ lssitepackages

# 复制虚拟环境
$ cpvirtualenv [source] [dest]

# 删除虚拟环境
$ rmvirtualenv [虚拟环境名称]

# 进入当前环境的目录(理论上用cd就可以)
$ cdvirtualenv [子目录名]
```

计算机视觉工具：OpenCV

Mac上配置OpenCV

- 下载opencv完整源码并编译
  OpenCV现使用Git作为版本管理，使用CMake来构建工程。
  网站链接：http://opencv.org
  源码地址：https://github.com/opencv/opencv
- CMake环境搭建
  OpenGL搭建下有Cmake安装
  使用Cmake编译OpenCV

Mac下OpenCV环境配置

- 安装brew

```
$ ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
# 卸载
$  ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/uninstall)"
```

- 安装cmake

```
$ brew install cmake
```

- 安装wget

```
# 官方建议安装
$ brew install wget
```

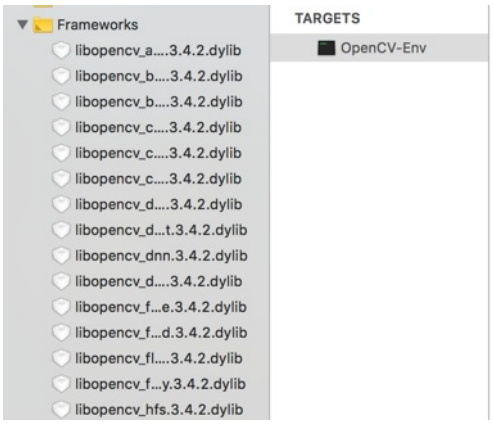
- 安装opencv

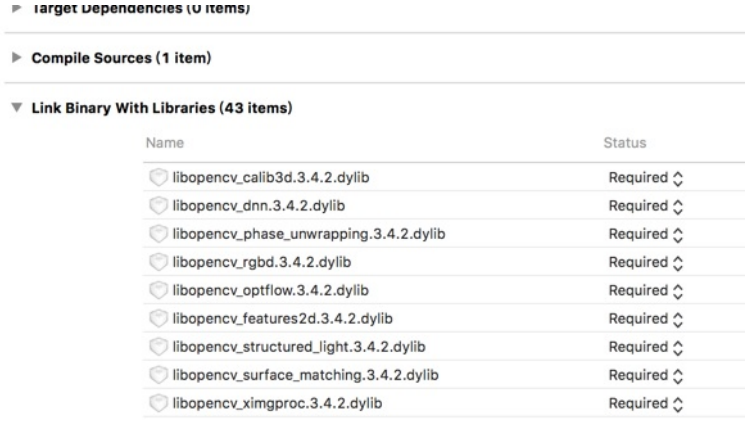
```
$ brew install opencv
```

- Xcode配置
  1. Xcode新建一个基于Mac OS的Command Line Tool工程，语言选择C++；
  2. 在Targets->Build Settings里，我们要对**Header Search Path**和**Library Search**

Path进行添加，以找到我们需要的库及头文件：

3. 添加依赖库，接下来在工程的build pases里点击link binary with librarys的加号，在add other中按Command+shift+g，输入/usr/local/Cellar/opencv，在版本目录下的lib文件夹下，添加所有dylib文件，如图：





4. 接下来在工程中添加头文件：

   `#include <opencv2/opencv.hpp>`

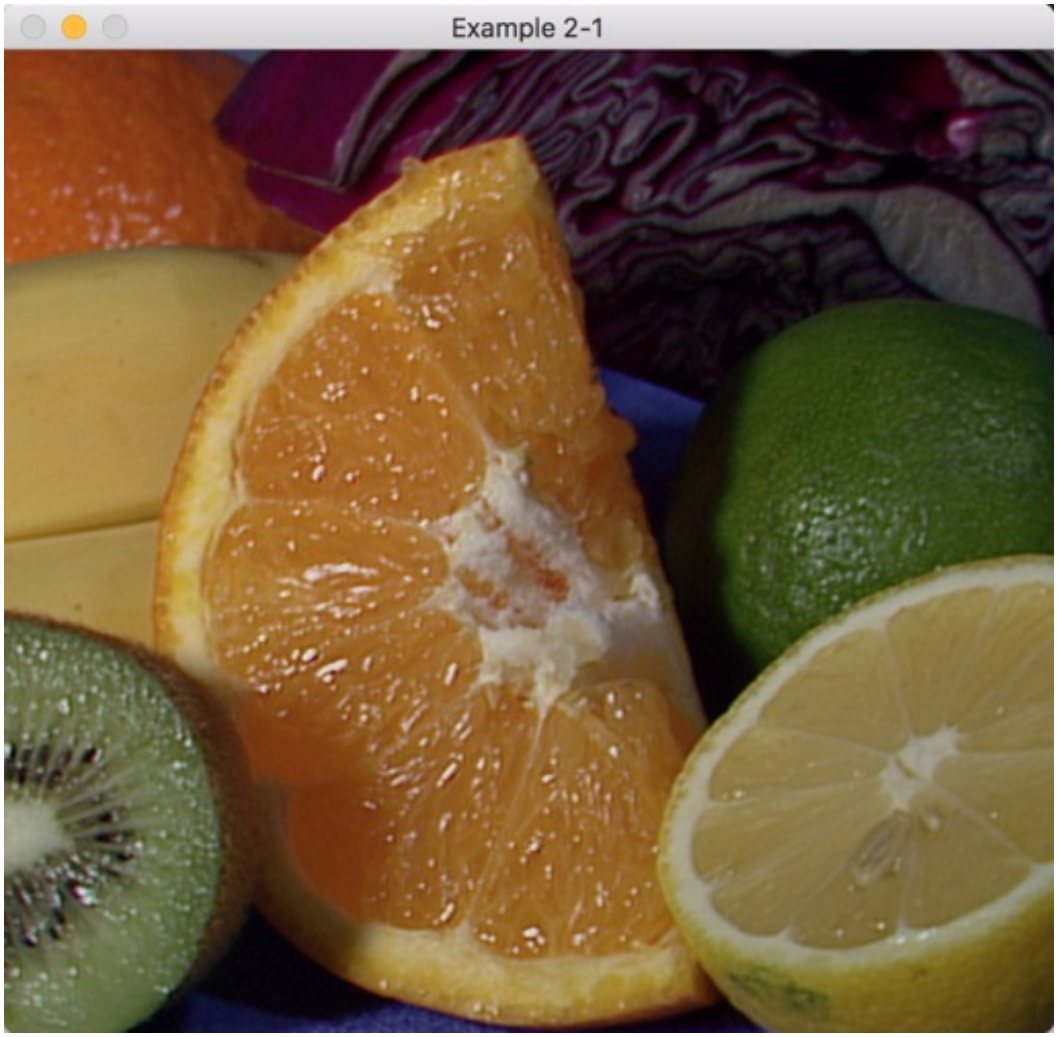
   编译通过！完成！

- 我们来显示一张图片

```
#include <iostream>
#include <opencv2/opencv.hpp>

using namespace cv;

void help(char** argv ) {
    std::cout << "\n"
    << "A simple OpenCV program that loads and displays an image from disk\n"
    << argv[0] <<" <path/filename>\n"
    << "For example:\n"
    << argv[0] << " ../fruits.jpg\n"
    << std::endl;
}


int main( int argc, char** argv ) {

    std::cout << "argc:" << argc << std::endl;
```

```
19.     std::cout << "argv:" << argv[0] << std::endl;
20. 
21. 
22.     if (argc != 2) {
23.         help(argv);
24.         return 0;
25.     }
26. 
27.     cv::Mat img = cv::imread( argv[1], -1 );
28. 
29.     if( img.empty() ) return -1;
30. 
31.     cv::namedWindow( "Example 2-1", cv::WINDOW_AUTOSIZE );
32.     cv::imshow( "Example 2-1", img);
33.     cv::waitKey( 0 );
34.     cv::destroyWindow( "Example 2-1" );
35. 
36.     return 0;
37. }
```

以上示例使用opencv从内存中加载并显示了一张图片：



4. 学习笔记

概念拾遗

- 3D point clouds
  知乎：什么是点云
- 光栅化
- 图形渲染管线

CS231n：图像分类笔记

CS231n课程笔记翻译：图像分类笔记（上

图像分类的任务，就是对于一个给定的图像，预测它属于的那个分类标签（或者给出属于一系列不同标签的可能性）。

- **数据驱动方法**
  如何写一个图像分类的算法呢？这和写个排序算法可是大不一样。怎么写一个从图像中认出猫的算法？搞不清楚。因此，与其在代码中直接写明各类物体到底看起来是什么样的，倒不如说我们采取的方法和教小孩儿看图识物类似：给计算机很多数据，然后实现学习算法，让计算机学习到每个类的外形。这种方法，就是数据驱动方法。
  运用到图片分类的数据驱动算法描述：给计算机数据，实现学习算法，使计算机学习到每个类的外形；

- 图像分类流程
  输入包含N个图像的集合，每个图像的标签是K个分类标签中的一种，使计算机基于给定的集合学习一种模型，生成的分类器用来预测未曾见过图像的分类标签，将预测标签与真实标签通过对于来评价分类器的的质量；

- 计算机存储图像原理
  首先要理解计算机存储图像原理，计算机存储图像，而用灰度图表示则为以下,图为一个4 * 4的用单颜色通道表示的像素图：

00000000	11110000	00001111	10101010
10101010	10101010	10101010	10101010
10101010	10101010	10101010	10101010
10101010	10101010	10101010	10101010

- Python函数：shape reshape
  shape:
  输入参数：类似数组（比如列表，元组）等，或是数组
  返回：一个整型数字的元组，元组中的每个元素表示相应的数组每一维的长度
  nn.shape[0]:
  

  reshape:
  函数功能：给予数组一个新的形状，而不改变它的数据
  numpy函数