# 一、京东数据爬取

工具：python+bs4+urllib+requests

## 1.1 获取keyword对应的item的data-sku

确定keyword，输入：**url：**https://search.jd.com/Search?keyword='+keyword+'&enc=utf-8&qrst=1&rt=1&stop=1&vt=2&offset=5&wq='+keyword+'&page=' + str(page)

**然后分析京东网页的DOM结构，用python爬取数据：**



参考：<https://segmentfault.com/a/1190000009166312>

得到每一个页面的图片和一些数据，由于这是aiax加载的，因此前面一段的img属性是src，后面的属性是data-lazy-img

奇数页和偶数页的区别

## 1.2 爬取 item 页面

### 1.2.1 爬取item主页面图片：

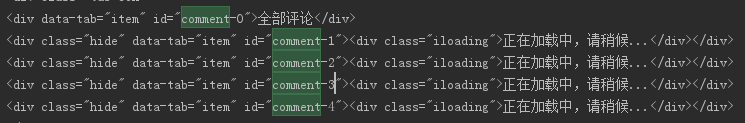
分析页面dom结构，得到每个img标签的src，利用urllib抓取存储在本地。

### 1.2.2.爬取item页面评论区内容

确定search 页面每个商品的data-sku，爬取[https://item.jd.com/**data-sku**.html](https://item.jd.com/data-sku.html)

**京东商品评论信息是由JS动态加载的，所以直接抓取商品详情页的URL并不能获得商品评论的信息。因此我们需要先找到存放商品评论信息的文件。**

比如商品页面和爬取的页面评论区内容：



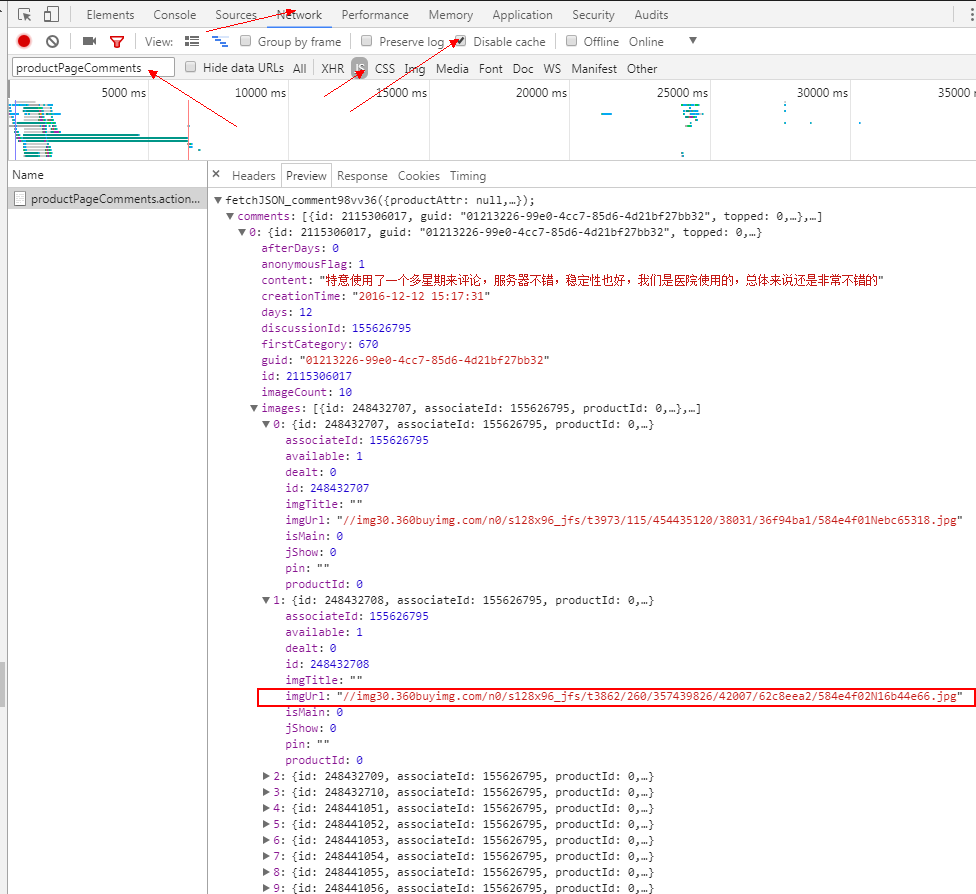


那如何抓取comments的内容呢？

网站的真实数据其实都隐藏在javascript、登陆表单和网站反抓取措施的背后。

https://sclub.jd.com/comment/productPageComments.action?callback=fetchJSON\_comment98vv36&productId=10226548046&score=0&sortType=5&page=0&pageSize=10&isShadowSku=0&fold=1

参考：http://ju.outofmemory.cn/entry/293948



利用京东提供的API爬取每个item评论区的图片数据。

Showordercomments部分内容：

* 1. showOrderComment:{id: 44267280, guid: "dfe5b5b2-a839-4d1d-aeec-b191aae58124",…}
     1. content:"价格实惠，效果不打折，信号很强，走出家门好远还可以有信号，棒！<div class='uploadimgdiv'><img class='uploadimg' border='0' src='http://img30.360buyimg.com/shaidan/jfs/t1612/4/1280856363/15763/f9b50a20/55c2e6b6N372781ae.jpg' /></div><div class='uploadimgdiv'><img class='uploadimg' border='0' src='http://img30.360buyimg.com/shaidan/jfs/t1612/14/1266983513/19679/258da7c6/55c2e6b6N765f7185.jpg' /></div><div class='uploadimgdiv'><img class='uploadimg' border='0' src='http://img30.360buyimg.com/shaidan/jfs/t1402/211/1265810446/12092/7798da9/55c2e6b6N71f4fd13.jpg' /></div>"

评论区的图片链接转换：

**分析规则，编写爬虫，有些图片action触发之后才会有清晰的图片内容，**通过检查发现，每次自己获取图片的url和打开缩略图的里面清晰的图片的url不是同一个url，然后通过代码里面提供的图片信息和清晰图片的url对比，可以拼凑出新的url，所以不是下载图片的问题，而是图片本身的问题，不是爬虫下载的原图。

**<img src="//img30.360buyimg.com/n0/s48x48\_jfs/t3973/115/454435120/38031/36f94ba1/584e4f01Nebc65318.jpg" width="48" height="48" alt="孤\*\*\*里的晒单图片">**

**<img src="//img30.360buyimg.com/shaidan/s616x405\_jfs/t3973/115/454435120/38031/36f94ba1/584e4f01Nebc65318.jpg" style="transform: rotate(0deg); margin-left: 0px; margin-top: 0px;">**

//img30.360buyimg.com/n0/s128x96\_jfs/t1612/4/1280856363/15763/f9b50a20/55c2e6b6N372781ae.jpg"

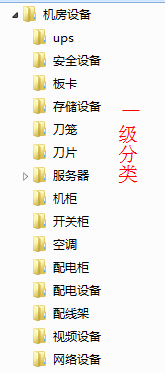
http://img30.360buyimg.com/shaidan/jfs/t1612/4/1280856363/15763/f9b50a20/55c2e6b6N372781ae.jpg'



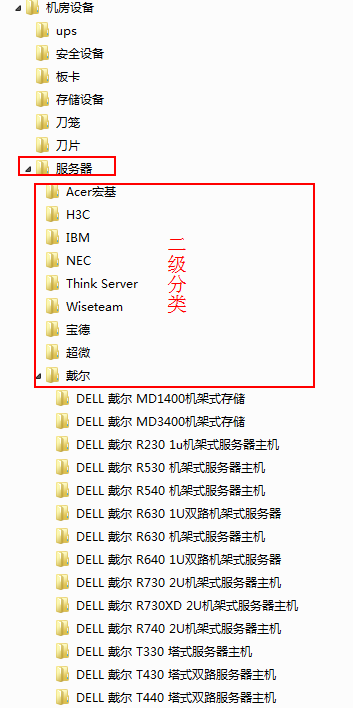
1.3 图片数据的存储

存储在本地对应的文件夹中。

一级分类：设备类型



二级分类：厂商



三级分类：



### 1.2.3 编码问题

弄了一上午

def get\_html(self):  
 try:  
 res = requests.get(self.url, headers=self.headers)  
 html = res.text  
 res.close()  
 return html  
 except:  
 time.sleep(10)  
  
# 得到每一个页面的id，图片链接页面的id：href="//item.jd.com/10329818949.html"  
def get\_pids(self):  
 try:  
 html = self.get\_html()  
 soup = BeautifulSoup(html, 'lxml')  
 lis = soup.find\_all("li", class\_='gl-item')  
 for li in lis:  
 data\_pid = li.get("data-sku")  
 print('li')  
 #得到该pid对应的goodname  
 div =li.find("div",class\_="p-name p-name-type-2")  
 aname = div.find('a')  
 p =aname.em.text  
 font = aname.em.font.text  
 print(p)  
  
 name = p+font  
 if (data\_pid):  
 self.pids.add(data\_pid)  
 self.names[data\_pid] = name  
 item = ahref(data\_pid)  
 self.ahrefs.add(item)  
 self.tmp = data\_pid  
 print(self.names)

这种情况下，输出的em中的中文字符串是乱码的，



只需要把get\_html(self):

函数中的

html = res.text

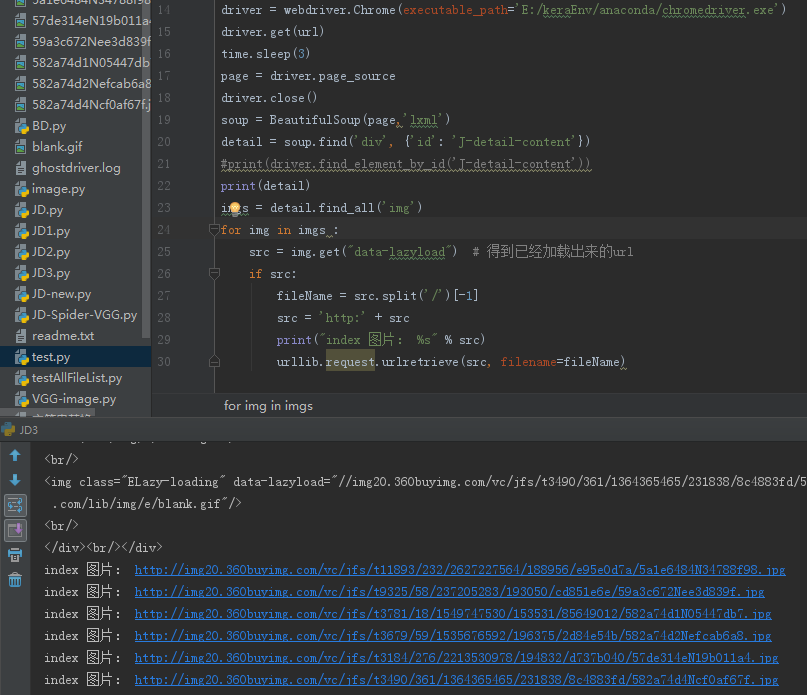
改成

html = res.content

### 1.2.4 如何解决Ajax懒加载

下载安装Selenium和chromedriver

加载出来的img不要提取src attr，需要提取 data-lazyload



## 1.3 图片数据清洗

将爬取的数据放在VGG网络中归类存储，过滤掉脏数据。

### 1.3.1 打印机数据爬取清洗

测试VGG网络对图片数据的归类，方法是将图片数据放到可分类1000classes的VGG网络里面，如果网络识别这是相关设备，则保留该图片数据，否则过滤掉。



Probability 79.76% => [modem]

Probability 2.24% => [radio]

Probability 1.63% => [toaster]

Probability 1.46% => [mouse]

Probability 1.45% => [hand\_blower]



Probability 50.01% => [modem]

Probability 20.01% => [iPod]

Probability 19.98% => [cellular\_telephone]

Probability 3.24% => [remote\_control]

Probability 0.82% => [combination\_lock]



Probability 57.04% => [radio]

Probability 15.66% => [microphone]

Probability 7.53% => [modem]

Probability 5.30% => [screwdriver]

Probability 2.06% => [remote\_control]



Probability 41.23% => [Polaroid\_camera]

Probability 13.36% => [oscilloscope]

Probability 6.14% => [joystick]

Probability 5.76% => [tape\_player]

Probability 4.08% => [modem]

交换机：



Probability 88.60% => [modem]

Probability 3.51% => [radio]

Probability 1.13% => [tape\_player]

Probability 0.82% => [cassette\_player]

Probability 0.58% => [CD\_player]



Probability 37.96% => [abacus]

Probability 15.72% => [modem]

Probability 8.38% => [switch]

Probability 3.74% => [computer\_keyboard]

Probability 3.36% => [radio]



Probability 99.06% => [harmonica]

Probability 0.11% => [rule]

Probability 0.10% => [buckle]

Probability 0.04% => [pencil\_sharpener]

Probability 0.04% => [cassette]



Probability 60.29% => [modem]

Probability 9.54% => [laptop]

Probability 7.43% => [notebook]

Probability 2.05% => [cleaver]

Probability 1.30% => [hard\_disc]



Probability 36.93% => [modem]

Probability 20.58% => [radio]

Probability 14.64% => [tape\_player]

Probability 6.70% => [cassette\_player]

Probability 6.63% => [switch]

### 1.3.2 遇到的问题

1.在清洗打印机图片时，遇到问题：tensorflow.python.framework.errors\_impl.ResourceExhaustedError: OOM when allocating tensor with shape[25088,4096]

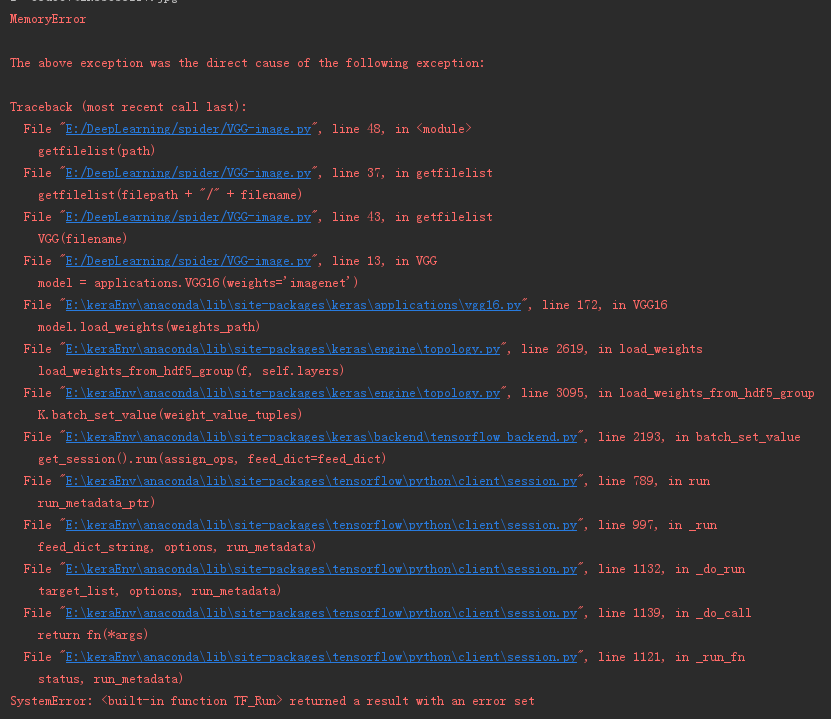
ResourceExhaustedError (see above for traceback): OOM when allocating tensor with shape[25088,4096]

[[Node: fc1\_34/kernel/Assign = Assign[T=DT\_FLOAT, \_class=["loc:@fc1\_34/kernel"], use\_locking=true, validate\_shape=true, \_device="/job:localhost/replica:0/task:0/cpu:0"](fc1\_34/kernel, fc1\_34/random\_uniform)]]

资源耗尽，这个矩阵尺寸是25088\*4096，每个元素是一个float64占用8字节，所以这单个对象就需要占用(28\*28\*32=25088) 25088\*4096\*8/1024/1024/1024=0.8G，当处理的图片变多的时候，算上其他的系统运行内存和其他变量，CPU资源就会耗尽。

此时用的实验室主机，RAM是8G。

2.在实验室主机上加了2个4G的内存条，RAM是16G，通过VGG 尝试清洗数据，结果是：



内存有限，连系统盘C盘也不够用了，所以老师让扩容 RAM 的方法暂时失效。没有GPU就是跑不动。

## 1.4 每个设备单独爬取

代码：JD\_new

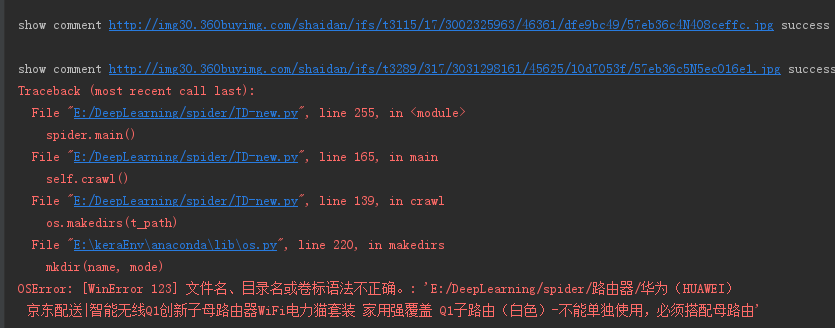
### 遇到的问题：

1.res.text和res.contents的字符编码不同，乱码问题

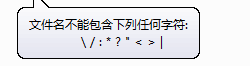
2.怎么启动懒加载：

交换机bug：路径错误





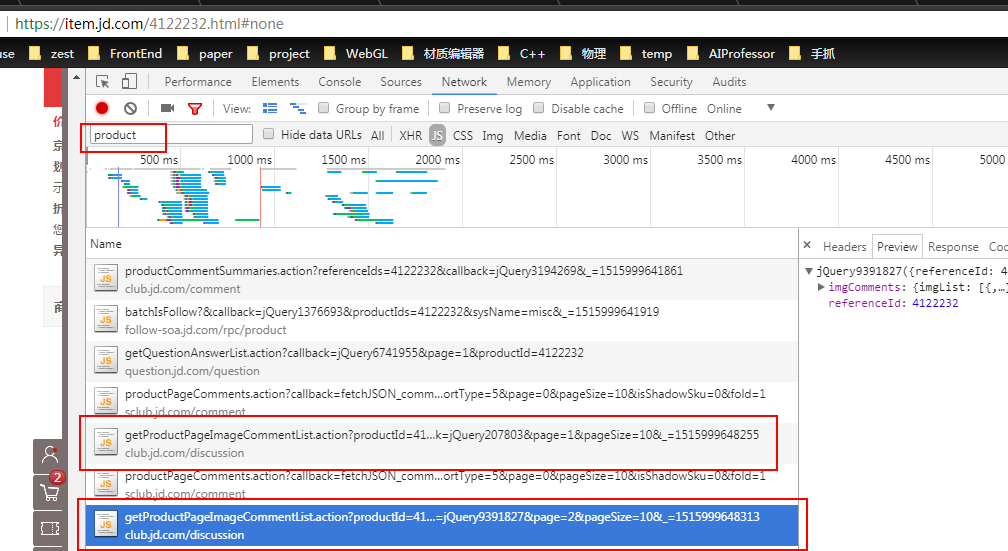
路径错误是因为怕去到的页面“title”中包含一些特殊字符：



所以只需要讲这些特殊字符替换成windows文件夹可用的名称，按照命名规则即可。

strxxx = self.names[id].replace('/',' ')#防止文件夹嵌套  
strx = strxxx.replace('\*','\_')  
strxx = strx.split(' ')#防止地址过长  
if(len(strxx)>=5):  
 join\_str = strxx[0] + ' ' + strxx[1] + ' ' + strxx[2] + ' ' + strxx[3] + ' ' + strxx[4]

## 1.5只爬取commentsImage



修改了get的headers和cookies

代码见JD\_new 的crawlCommentsImage（）

# 二、keras 框架的研究与应用

## 2.1 基础研究

### 2.1.1 ILSVRC竞赛模型调研

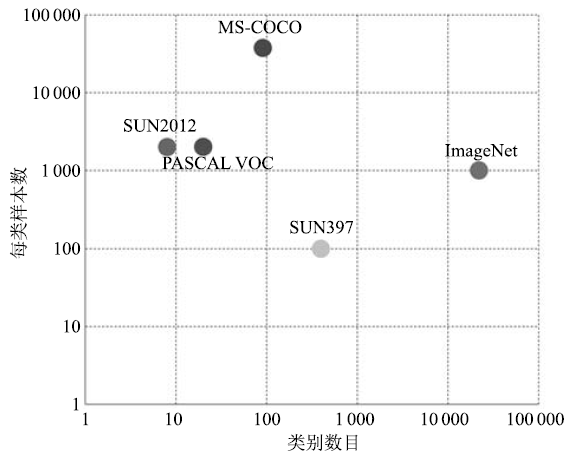
Imagenet账号：[zhangyaru@bupt.edu.cn，CatherineZYR](mailto:zhangyaru@bupt.edu.cn，CatherineZYR)，\_1994313

|  |  |  |  |
| --- | --- | --- | --- |
| **CNN模型** | **Top-5 错误率(%)** | **年份** | **比赛** |
| **AlexNet** | 16.4 | 2012 | ILSVRC |
| **ZFNet** | 14.8 | 2013 | ILSVRC |
| **VGG** | 7.3 | 2014 | ILSVRC |
| **GooleNet（Inception-v1）** | 6.7 | 2014 | ILSVRC |
| **ResNet** | 3.57 | 2015 | ILSVRC |
| **Inception-v4, Inception-ResNet** | 3.08 | 2016 | ILSVRC |
| **SENet** | 2.251 | 2017 | ILSVRC |

ILSVRC(ImageNet Large Scale Visual Recognition Challenge)竞赛是基于ImageNet图像数据库的国际计算机视觉识别竞赛。2010年开始举办，逐渐发展为国际计算机视觉领域受关注度最大、水平最高、竞争最激烈的竞赛。2016 年 ILSVRC 的图像识别错误率已经达到约 2.9%，不仅远远超越人类（5.1%），今后再进行这类竞赛意义也不大了。

Imgnet中有14,197,122 images, 21841 synsets indexed

几种公共数据集比较



### 2.1.2 keras框架的基础研究

安装的时候，直接安装anconda，并将anconda的python编辑器加入到pycharm中即可。

Virtualenv的应用：

Virtualenv可以创建一个隔绝的python虚拟环境。

Keras提供了五种开箱即用型的CNN：

1.VGG16

2.VGG19

3.ResNet50

4.InceptionV3

5.Xception

官方示例：<https://keras-cn.readthedocs.io/en/latest/other/application/>

| **模型** | **大小** | **Top1准确率** | **Top5准确率** | **参数数目** | **深度** |
| --- | --- | --- | --- | --- | --- |
| Xception | 88MB | 0.790 | 0.945 | 22,910,480 | 126 |
| VGG16 | 528MB | 0.715 | 0.901 | 138,357,544 | 23 |
| VGG19 | 549MB | 0.727 | 0.910 | 143,667,240 | 26 |
| ResNet50 | 99MB | 0.759 | 0.929 | 25,636,712 | 168 |
| InceptionV3 | 92MB | 0.788 | 0.944 | 23,851,784 | 159 |
| IncetionResNetV2 | 215MB | 0.804 | 0.953 | 55,873,736 | 572 |
| MobileNet | 17MB | 0.665 | 0.871 | 4,253,864 | 88 |

VGG网络架构于2014年出现在Simonyan和Zisserman中的论文中,《VeryDeepConvolutionalNetworksforLargeScaleImageRecognition》。ResNet于2015年出现在Heetal的论文《DeepResidualLearningforImageRecognition》中，它的出现很有开创性意义，证明极深的网络也可以通过标准SGD（以及一个合理的初始化函数）来训练。Keras中的InceptionV3架构来自于Szegedyetal.的后续论文，《RethinkingtheInceptionArchitectureforComputerVision(2015)》，该论文打算通过更新inception模组来提高ImageNet分类的准确度。关于Xception的出版物《DeepLearningwithDepthwiseSeparableConvolutions》可以在这里找到。

Xception最小仅有91MB。

1.在keras框架下应用VGG-16

项目地址：E:\DeepLearning\workSpace\venv\VGG

参考链接：

<https://www.jianshu.com/p/20585e3b6d02>：如何应用VGG识别归类？

<http://blog.csdn.net/sinat_26917383/article/details/72859145>：解析keras的VGG

结果：

**Probability 99.20% => [printer]**

**Probability 0.65% => [photocopier]**

**Probability 0.04% => [toilet\_seat]**

**Probability 0.04% => [projector]**

**Probability 0.02% => [iron]**

**VGG-16**

输入：一张图片

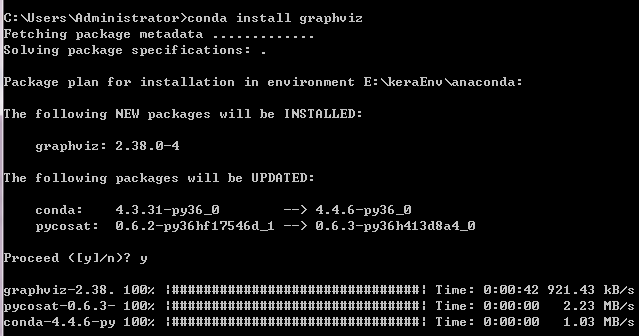
输出：前五个最高概率的分类结果

因为ILSVRC使用imagenmet数据集中的1000个类，每个类中有1000个图片。所以训练结果中也只有1000种result。

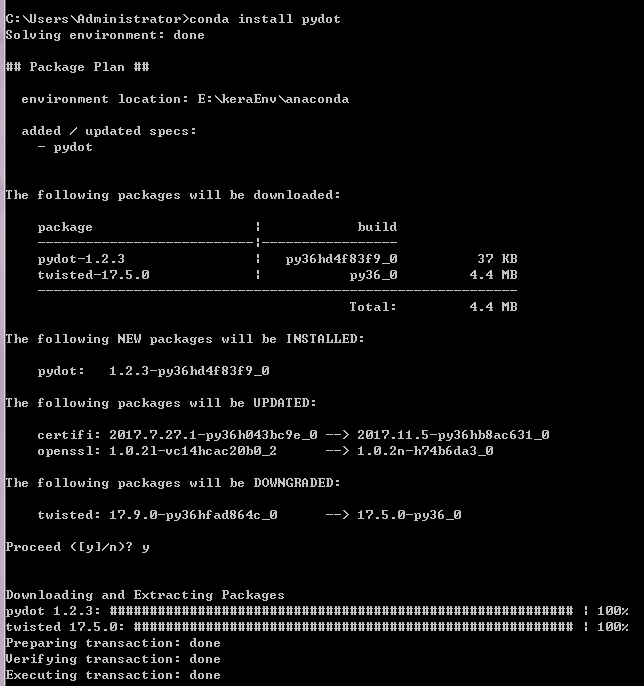
2.1.3 keras框架下模型可视化

官方链接：<http://keras-cn.readthedocs.io/en/latest/other/visualization/>

安装环境：先安装 graphviz：



再安装pydot-ng：



### 2.1.3 keras框架的配置 & CNN模型训练环境

安装anaconda

安装tensorflow

安装keras

2.2 项目阻碍

训练数据不足

计算资源有限

梯度消失

梯度溢出

# 三、模型训练

## 3.1 图像分类

### 3.1.1单标签分类

单标签输入，单标签输出。

单标签单分类

单标签多分类

### 3.1.2 多标签分类

### 多标签单分类

### 多标签多分类

# 四、深度学习相关知识学习

## 4.1 相关资料

### 4.1.1相关博客文章（必须找原文）

【Valse首发】CNN的近期进展与实用技巧（上）

[https://mp.weixin.qq.com/s?\_\_biz=MzI1NTE4NTUwOQ==&mid=502840969&idx=1&sn=05b8d27dccf6efc3f85f47a74669e1c9#rd](https://mp.weixin.qq.com/s?__biz=MzI1NTE4NTUwOQ==&mid=502840969&idx=1&sn=05b8d27dccf6efc3f85f47a74669e1c9" \l "rd)

[经典卷积神经网络总结:LeNet-5、AlexNet、ZFNet、VGG、GoogleNet、ResNet](http://blog.csdn.net/mr_kktian/article/details/53144119)

：<http://blog.csdn.net/Mr_KkTian/article/details/53144119>

### 4.1.2 相关论文

### 4.1.3 相关书籍

### 4.1.4 相关视频