第一讲　环境搭建和图像采集

1. Windows 下环境搭建

1.1 anaconda 安装

https://www.anaconda.com/products/individual#windows
pycharm

1.2 相关扩展库安装

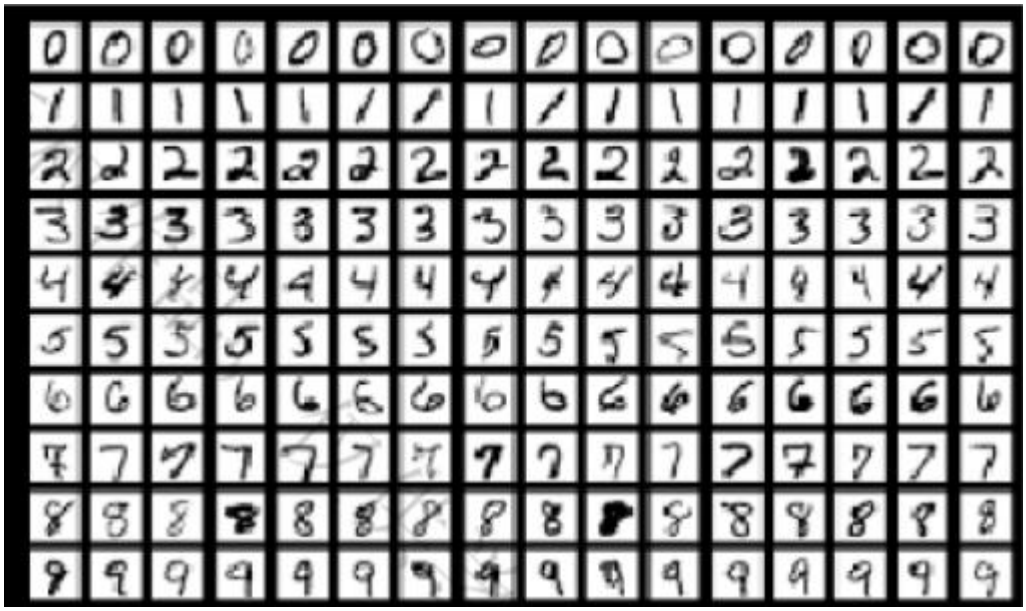
安装 opencv：pip install opencv-python
安装 labelme：pip install labelme==4.2.10 -i https://mirrors.aliyun.com/pypi/simp

2. 图像数据采集

2.1 开放图像数据集

（1）MNIST 手写数据集

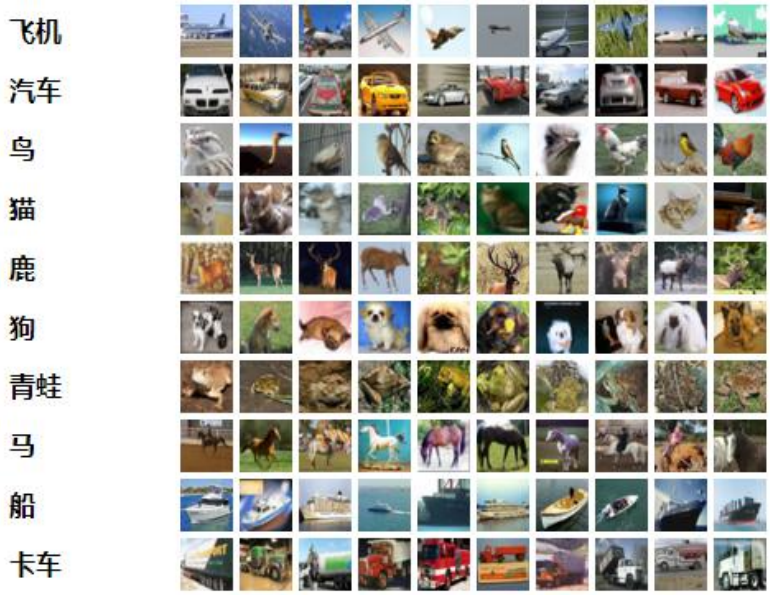
http://yann.lecun.com/exdb/mnist/index.html
数据集由 60000 个训练样本和 10000 个测试样本组成
每个样本都是一张 28 * 28 像素的灰度手写数字图片
每个像素点是一个 0-255 的整数



（2）CIFAR-100 数据集

http://www.cs.utoronto.ca/~kriz/cifar.html

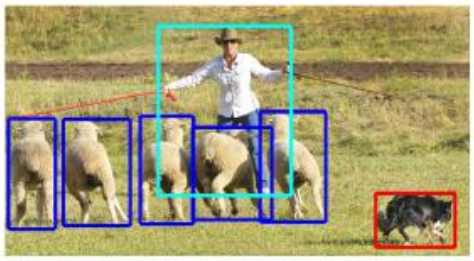
CIFAR-100 由 60000 张大小为 32*32 的三通道彩色图像组成，分为 20 个大类，每个大类又包含 5 个小类，总共 100 个小类。每个小类包含 600 张图像，其中 500 张用于训练，100 张用于测试。



（3）MS-COCO 数据集

https://cocodataset.org/

MSCOCO 数据集是微软开发维护的大型图像数据集，次数聚集的任务包括识别(recognition)，分割(segementation)，及检测(detection)



(b) Object localization

(d) This work

（3）PASCAL **VOC** 数据集

https://pjreddie.com/projects/pascal-voc-dataset-mirror/

待识别的物体有 20 类：

人、6 类动物、7 类交通工具、6 类生活用品

person
bird, cat, cow, dog, horse, sheep
aeroplane, bicycle, boat, bus, car, motorbike, train
bottle, chair, dining table, potted plant, sofa, tv/monitor

2.2 网络爬虫获取图片

（1）urllib
（2）request
（3）beautifulsoup

```
import requests
try:
    r=requests.get(“http://www.baidu.com”)
    r.raise_for_status()#如果状态不是 200，引发 httperror 异常
    r.encoding=r.apparent_encoding
    r.text
except:
    print(“产生异常”)

from bs4 import BeautifulSoup
soup=BeautifulSoup(html,"html.parser")
tdlist=soup.find("tbody").find_all("td")
```

3. 网络图像采集

在中国，每年会有不少于 4000 万只狗在流浪，通过网络交流平台（如百度贴吧）上发布关于流浪狗领养的信息，爱狗人士把救助站，或是失去家园的狗的图片信息发布出去，来供有余力的人家领养，帮助这些流浪狗获得爱与关怀。但是在交流平台上交换信息内容大部分与主题无关，本次案例就是实现通过数据爬取，快速取得百度贴吧内流浪狗领养图片，以便筛选。

3.1 网页分析

本任务是爬取百度贴吧流浪狗领养贴中的图片，所爬取网址为https://tieba.baidu.com/p/6045474546，在进行抓取之前首先获取图片地址信息标签位置。进入网址，右击鼠标，在弹出的快捷菜单中选择“查看页面源代码”选项（或通过快捷键 F12），即可看到网页源代码（以火狐浏览器为例），在弹出页面后单击其左上角的小箭头，选中页面中的元素（或按 Ctrl+Shift+C）即单击所查看的图片。

即可跳转到图片信息所在的代码行。

可以看到图片标签<img>，文档内标识属性<class>，图片 URL 属性<src>，图片大小属性

<size>、<width>和<height>，截取部分代码如下所示：

<img class="BDE_Image"
src="https://imgsa.baidu.com/forum/w%3D580/sign=bd7e89d7e91190ef01fb92d7fe1b9df7/d3dcc3cec3fdfc03aa0eba82da3f8794a4c226b1.jpg" size="40923" width="520" height="293">

3.2 读取网页内容

首先创建一个文件名为 fetch_image.py 的.py 文件，代码中首先导入 urllib.request，bs4。参考代码如下，代码前数字含义表示执行顺序和标记：

```
import requests
from bs4 import BeautifulSoup
```

定义目标图像 URL 地址。代码如下：

```
url = 'https://tieba.baidu.com/p/6045474546'
```

实际情况当中某些网站会采取反爬机制，采取反爬机制之后，百度等搜索引擎无法对网站的内容进行网页爬取，解决方法是修改 User Agent 来模拟浏览器访问。代码如下：

```
header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36'}
```

打开网站发起请求相应内容，以获取所需数据，打印输出结果，代码如下所示：

```
r=requests.get(url,headers=header)
txt=r.text
print('打印 html','\n',txt[500])
```

```
<!DOCTYPE html><!--STATUS OK--><html><head><meta name="keywords" content="百度贴吧,狗流浪,狗领,养站"/><meta name="description" content="流浪狗领养站，送狗贴..【领养狗狗】家里大狗下了几只小金毛，狗场300要收家里不想卖。说要送人，一共三只 我留一只养养，大狗养了四五年也蛮聪明的，不咬人。农村养的不嫌弃就好。贴吧不常看vx:lianhuahc" /><meta charset="UTF-8"><meta furl="tieba.baidu.com/f?kw=%E7%8B%97&ie=utf-8" fname="狗"><meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"><meta name="baidu-site-verification" content="jpBCrwX689" /><meta name="baidu-site-verification" content="code-ku2CCMz5
```

接下来需要解析 html，以及下载图片并重命名，这里自定义一个 get_images 函数。函数功能是取得图片 URL 并下载到本地计算机，同时打印输出"全部抓取完成"提示信息。函数调用如下：

```
get_images(txt) # 通过 get_images 取得图片
print(' 抓取完成 !')
```

get_images 函数完整定义见 1.3。

3.3 获取图片数据

编写 get_images 函数，首先创建一个 BeautifulSoup 的对象，获取的数据除了图片还有很多无用的数据，接下做筛选。

beautifulsoup4 库中主要的类是 BeautifulSoup，它的实例化对象相当于一个页面，得到的是一个树形结构，它包含 HTML 页面的每一个标签（Tag），比如<head>,<body>等，可以理解这时候 HTML 中的结构都变成了 BeautifulSoup 的一个属性，可以直接通过 Tag 属性访问。

这样就可以通过 Tag 属性获取到图片的路径。

函数内定义 soup 这个 BeautifulSoup 对象，指明采用 html 解释器，查看是否输出成功。百度贴吧页面内图片标识为'BDE_Image'，通过 find_all 函数进行筛选，并打印查看是否只有图片数据。参考代码如下：

```
def get_images(txt):
    soup = BeautifulSoup(txt,'html.parser') # 创
建 beautifulsoup 对象 soup
    lstImg = soup.find_all('img','BDE_Image') # 找到所有 img 标签
    print("打印 lstImg",lstImg)
```

可以看到变量 all_img 已经存储了筛选出的图片数据，包含图片基本信息如：height，size，src，width 等。如图所示：

使用 for 循环遍历 all_img 内容把每个图像进行重命名，通过 urllib.request.urlretrieve 或者 requests.get 下载图片保存到本地，该函数有一个必填参数即网页<img>标签 src 属性以及可选参数即下载之后的图片存放路径。其中图片存放路径可以只写一个文件名（image_name），这样会默认保存到工作目录，也可以指定路径。

参考代码如下：

```
    i = 0
    for img in lstImg: # 遍历所有 img 标签内容
        imgName = '%s.jpg'%str(i) # 图片名称规范
        imgURL = img['src'] # 取出每一张图片的 url 地址
        i = i+1
        #当前目录下新建 dogImg 文件夹，下载图片保存到该目录下
        with open('./04shijue1+x/2021cv/dogImg/'+imgName,'wb')
 as f:
            r1=requests.get(imgURL) # 爬取图片
            f.write(r1.content) # 保存图片
            print(' 成功抓取到图片 ',img['src'])
```